The Anatomy of a Large-Scale Collective Decision Making System

Marko A. Rodriguez*
CCS-3: Modeling, Algorithms & Informatics
Los Alamos National Laboratory

Daniel J. Steinbock
Learning Sciences & Technology Design
Stanford University

March 13, 2006

Abstract

This paper presents a general purpose framework for the design of a large-scale collective decision making system. The single, unifying construct is a weighted semantic network, generated by individual choices, that connects humans, domains of expertise, problems and solutions. Swarms of simple particles traverse this complex knowledge network, aggregating individual choices into collective choices, and ranking solutions to the problems facing the group.

1 Introduction

As information technology continues to play an ever more mediative role in the coordination and communication activities of social groups, certain software models have developed as analogues for basic social processes. Email and instant messaging stand in for personal communication; chat rooms and threaded discussion boards stand in for group conversation. Even historically complex modes of social discourse now have online equivalents in multivaried forms like weblogs and Wikis.

Communication, discussion, collaboration – these processes are fundamental to all social groups, whether embodied or distributed. Though software analogues may bear only vague resemblance to the embodied practices they stand in for, digitally-mediated collaboration achieves a scale of interaction that was not possible before the globally shared medium that is the Internet.

In recent years we have witnessed an explosion in so-called 'social software' whose explicit purpose is to mediate and extend social processes (Tepper, 2003). Many of these are built around generative collaboration, systems where content is contributed by the users and the system mediates many-to-many interactions for the sharing of knowledge.

What is striking about these systems is the fact that their value increases in proportion to the size of their userbases (O'Reilly, 2005) – a knowledge economy of scale. Wikipedia[1], the

*corresponding author: mrodrig2@vub.ac.be, http://www.soe.ucsc.edu/ okram
[1]Wikipedia available at: http://en.wikipedia.org/wiki/Main_Page

user editable encyclopedia, is an exemplar of this effect: useful articles draw more users to participate in writing and editing; more participation makes for even more useful articles, closing a positive feedback loop. Another example is the Digg[2] web service, where individual users contribute news stories and vote for stories they think highly of, producing a collective ranking of news. Again, there is a feedback loop of scale: interesting stories draw more readers to participate in the ranking of stories that reflect collective interest.

For this class of social information system, scale is everything. Individual contributions and rankings aggregate into collective knowledge structures that increase in breadth, depth and quality as more able-minded participants join in the collaborative process. These systems are harnessing the collective intelligence of the millions who participate (O'Reilly, 2005), both in their generative ability to contribute content and their filtering/aggregative ability to rank content. This pattern of individual contribution and collective aggregation is repeated in many forms throughout the social software sphere and more forms are constantly emerging in this period of innovation.

As researchers, we are interested to know if there are generalizable processes that these information systems have in common? If so, to identify the underlying processes would enable a new generation of researchers to model the novel social activities that are becoming more and more prominent in the lives of everyday people. Furthermore, a general model would help the architects of social information systems to better comprehend what it is they are creating and thereby drive future innovation.

To address this emerging need, we have developed a general model of how individual contributions and rankings in a digitally-mediated environment can be aggregated into collective knowledge structures. To apply the model in a demonstrative context, we present it here as a framework for large-scale collective decision making. The general model has far wider applicability to social information systems generally, and future work will further elaborate on both its practical usage and its mathematical underpinnings.

To give a preview of what's to follow in this paper, we will introduce a framework for the design of a large-scale collective decision making system that has two major components. The first is a directed, weighted semantic network, generated by individuals' contributions of content (problems and solutions) and their rankings of both solutions and the expertise of others. In its fully-elaborated form, the network structure also captures categories of knowledge that describe problems, and the corresponding domains of expertise that describe individuals.

The second component of the framework is the use of algorithmic particle swarms that traverse the semantic network and aggregate collective rankings from the network structure. To use an explanatory analogy: just as we could infer the relative sizes of rooms in a house by counting the number of air molecules in each (at equilibrium), we can infer the weighted ranks of nodes in a network by releasing a swarm of randomly walking particles and measuring their distribution over the nodes once the swarm reaches equilibrium.

These two components of the general model are applied in this framework to model the structure and process of collective decision making: problem space organization, delegation of power, parallel problem solving, integration of individual perspectives, outcome selection, et cetera. In addition, the semantic network architecture of our system is decoupled from any particular rank aggregation algorithm, making it a suitable testbed for different algorithms, with a metadata schema for exposing their results. A principal aim in our design of this framework is to enable research that can make substantive recommendations about which algorithmic methods of aggregation are most appropriate for which classes of decision making problems.

[2]Digg available at: http://digg.com/

We begin with a review of relevant literature before introducing our three stage process of decision making and the semantic network data structure. This is followed by the formal elaboration of our semantic particle network paradigm which includes the relations among people, their domains of expertise, problems they are trying to solve, and the solutions they are considering. We end with a discussion of directions for future research and extensions of the present framework.

1.1 Challenges for Large-Scale Decision Support

Despite the recent surge of innovation in social software engineering, there are no systems that are designed to support generalized distributed decision making on a large-scale. Group decision support system research has produced numerous systems to facilitate group decision making, but these systems tend to be specific to a particular problem domain and, for the most part, do not scale to the size of societal populations (Heylighen, 1999; Turrof, Hiltz, Cho, Li, & Wang, 2002).

As the size of a decision making collective increases, so do the number and diversity of problems and solutions, competing for the attention of decision makers. In order to handle this overload problem (Rodriguez & Steinbock, 2004b), a division of labor is necessary so that problem solving is parallelized. With parallelization comes fluctuating levels of participation in any one decision process. Not every member of the collective can actively participate in every decision and therefore the necessity of representation arises, where non-participating individuals delegate power to participants to make decisions on their behalf. However, not all delegates' preferences are necessarily representative of non-participants, suggesting that the distribution of decision making power among participants be differentially weighted in order to better reflect the beliefs, opinions and values of the individuals being represented. Finally, there is the critical question of how to aggregate the diverse perspectives of many differentially empowered decision makers into a single decision outcome.

Regardless of the collective's size, the most difficult problem in collective decision making is knowing how to integrate the many individual perspectives to produce a single collective outcome. One option is to import the methods long employed by self-governing societies and organizations. Procedures like voting and majority rule are widely used today due to their ease of understanding and deterministic mapping from individual to collective preference, i.e. vote-counting.

Yet the simple mechanics of voting are reductionistic, treating heterogeneous individuals as equivalently knowledgeable decision makers and discounting deliberative social interaction as a key part of decision analysis. Human expertise, wisdom and insight are highly variable within a population, often transient and context-specific. A single individual's abilities vary over time, with respect to different domains and to the particulars of a given problem. One person/one vote may be simple, but it reduces the diversity and dynamics of humanity to the homogeneity of a mass.

Of course, this is by design. One person/one vote is meant to operationalize the democratic notion of equality, where no person has more power than any other, regardless of the problem context. While this may be appropriate when individual equity as a cultural value is critical to the health of a society beyond the realm of decision making, it can be limiting in situations where the recognition of differential expertise would result in a better collective outcome. This is a recurring tension between two goals of democracy: individual equality and the collective good (Held, 1984). In addition, social choice theory reveals that mathematical paradoxes lurk in the underlying assumptions of seemingly straightforward voting procedures (Arrow, 1963).

1.2 From Voting to Emergent Decision

An alternative approach to the aggregation problem recognizes the complex and dynamic *interactional* contexts in which decision making usually happens in everyday situations. Few decisions are as well-specified as an annual national election. More often, decision outcomes are the final, emergent result of a bottom-up process that includes the formulation of problems and potential solutions and the collaborative interaction between decision makers.

Today, software systems based on bottom-up emergence are ubiquitous. Though not conceived as decision making systems *per se*, they do solve the problem of aggregating individual contributions into collective mental maps (Heylighen, 1999). These include collaborative filtering recommender systems (Herlocker, Konstan, Terveen, & Riedl, 2004), adaptive hypertext systems (Bollen & Heylighen, 1996, 1998) and search engines that exploit the structure of the world wide web (Brin & Page, 1998; Brin, Motwani, Page, & Winograd, 1998). In all these systems, users acting on the basis of individual preferences together modify a shared environment, resulting in a co-construction of collective preference representations. While different applications utilize different aggregation algorithms, the key insight is that the computation of the system – one might even say, the intelligence – is distributed in the mediating environment (Simon, 1996; Pea, 1993).

Such environmentally-mediated coordination is widespread in natural complex systems, most notably in connection with eusocial insects like termites (Grasse, 1959), ants (Doringo, Maniezzo, & Colorni, 1996; Bonabeau, Dorigo, & Theraulaz, 1999) and honey bees (Cazamine & Sneyd, 1991; Tereshko & Lee, 2002). Through a combination of biological and mathematical techniques, researchers studying these insects have inferred simple rule-based algorithms that govern collective behaviors like food foraging and nest building. For example, ants lay down pheromone trails when foraging for food in the vicinity of their colony. A trail can be thought of as an ant's 'vote' for a promising path. When an ant intersects another ant's pheromone trail, it will follow it with probability proportional to the magnitude of the pheromone deposit, otherwise it will continue on its own trail-breaking path. If it does follow, it will continue to lay down pheromone, adding to the strength of this particular path and thus, through positive feedback, making it more likely that even more ants will follow it. In the event that an ant finds food, it doubles back on its own trail, immediately creating a more enticing path.

This form of indirect communication mediated by a modifiable environment is called *stigmergy* (Grasse, 1959), and the resulting network of pheromone trails is an aggregate of individual 'votes' that provide a quantifiable ranking of shortest paths to food sources. Though the underlying rules of ant foraging are simple, their interaction with a complex environment produces a highly sophisticated problem solving system that has directly inspired computer scientists in the design of multi-agent optimization algorithms for problems in graph theory, routing, load balancing and other domains (Schoonderwoerd, Holland, Bruten, & Rothkrantz, 1996; Dorigo & Gambardella, 1997; Johnson, 1998).

It is illuminating to keep these natural, emergent systems in mind when designing and evaluating software for the aggregation of individual choices into collective representations. In the case of Google's PageRank algorithm (Brin & Page, 1998), the choice of a web author to create a hypertext link from one page to another is analogous to a 'vote of confidence' in the linked-to page. We can then count the votes for every page on the Internet to produce a single collective ranking that aggregates the choices of every web author. This is like a direct democracy where anyone can vote for anyone else and the 'winner' is the site with the most votes. But clearly, not all web pages are equally authoritative in assigning links. A link from the New York Times to one's personal website carries more weight than a link from someone else's personal website

and so this link should be worth more. In Google's case, the importance, or authority, of a web page depends on the authority of the pages that link to it – a recursive definition – plus the authority of the pages that link to those pages, ad infinitum. This is an example of an aggregation process that traverses a collectively co-constructed network, infers authority from the network structure, and differentially weights certain individuals (web pages, in this case) to produce a single collective ranking for all nodes in the network.

A similar example comes from the problem of ranking journals in the scientific community (Ball, 2005, 2006), where the ISI impact factor has been the *de facto* standard in quantifying the status of a journal. The impact factor, in short, states that the more citations a journal receives, the higher the journal status; a citation is considered a vote of confidence in the material of the cited journal. Other methods for aggregating individual citation preferences have been used, such as the use of the PageRank algorithm within a journal citation graph (Bollen, Rodriguez, & Sompel, 2006). What is striking is that, for certain scientific domains, ISI impact factor and the PageRank method have been shown to have widely divergent rankings of journals.

When is one collective aggregation more appropriate than another? This question is central to our inquiry and has guided the development of the decision making framework to be described in succeeding sections. It has been our goal to model a large family of aggregation algorithms using a single a parameterized network construct and we have succeeded in generalizing emergent, stigmergetic, Page-Rank style aggregation as well as direct-, representative- and novel forms of democracy within our unifying framework of a semantic particle swarm. In the next section, we'll briefly describe our three stage model of collective decision making which will serve to structure the rest of the paper.

2 Three Stages of Collective Decision Making

Decision making is fundamental to the collaborative activities of social groups. Broadly defined, decision making includes the formulation of problems, generation of solutions, ranking of these solutions and selection of an outcome. With the exception of the last step, these are all activities performed by the individual members of a decision making collective. In this paper, we are primarily concerned with aggregating individual rankings into collective rankings to reveal a preferred outcome. We'll use a problem/solution terminology and define a three stage model for collective decision making: *individual solution ranking*, *collective solution ranking*, and *selection*.

Figure 1 depicts the three stage model of collective decision making. In stage one, individuals review solutions to a problem and rank them according to their subjective opinions of each. In our model, to rank solutions is to weight them as fractions of a single unit of decision making power. This allows the greatest range of possible preference expressions, beyond just rank ordering or a single vote. However, a weight of 1.0 is equivalent to a single vote, and a rank weighting implies a rank ordering (with the possibility of ties).

To go from individual rankings to a collective ranking, some algorithmic procedure must be used to integrate these numeric expressions of opinion into a single ranking. One such algorithm is where participating individuals have equal power and the weights they assign to solutions are averaged. This is represented in Figure 1 where the weights assigned to each solution in the collective ranking stage are the averages of weights assigned in the individual ranking stage.

Finally, given a single collective solution set ranking, a selection function is used to output a single final solution. For problems with numeric solutions such as the price to set for a

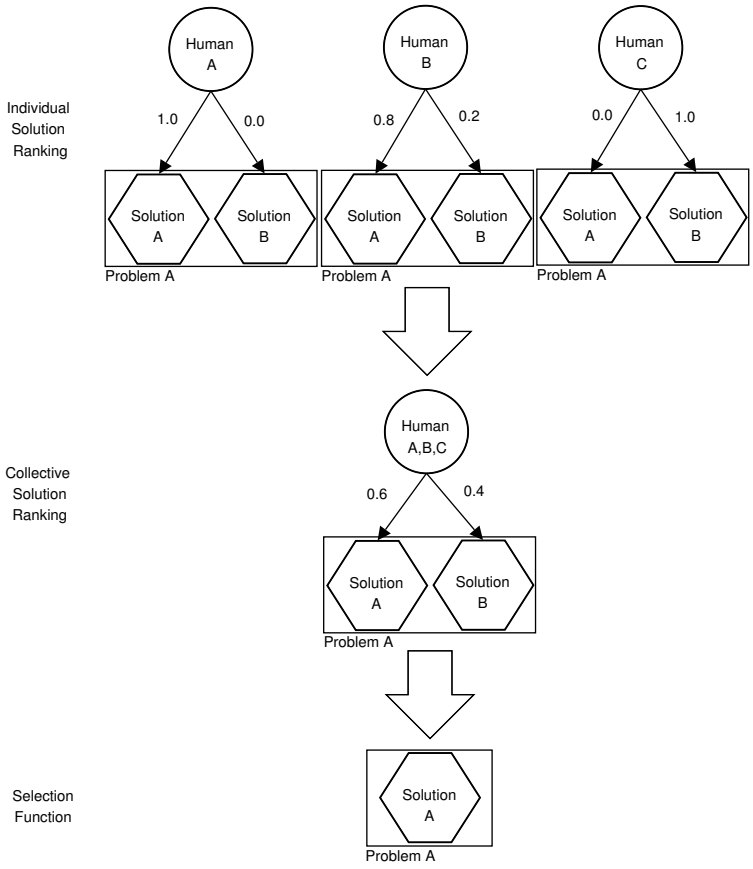

Figure 1: Three stage model of collective decision making: individual weighted ranking of solutions, aggregation into a collective weighted ranking, selection of decision outcome.

product or the estimated size of a population, a final solution could be selected by averaging the different solutions together weighted by their respective collective ranking. For problems with nominal or categorical solutions such as the choice of whether or not to buy a product or the choice of an elected official, solutions cannot be averaged together. In most situations, the best solution is considered the one with the highest collective ranking.

In the following sections, we'll formalize a semantic network construct that connects individuals, their domains of expertise, problems and solutions. Following this we introduce a family of particle swarm algorithms for aggregating over the semantic network structure.

3 The Semantic Network Ontology

A semantic network is used to represent a heterogeneous set of entities (nodes) and a heterogeneous set of relationships (edges) (Sowa, 1991). Therefore, there exists not only multiple entities of varying type, but also different ways, or semantics, by which these entities are connected. The *ontology* for a particular semantic network defines the type of entities and relationships that can be instantiated in the network. This section will describe the human collective decision making ontology used throughout the remainder of this paper. This described network ontology is independent of the particle swarm algorithms presented later in the paper; so ex-

tensions can be made to the basic ontology without necessarily reducing the usefulness of the algorithms. The ontology we propose contains a number of basic entities for implementing a human collective decision making system: *humans*, *domains*, *problems*, *solutions*, and their various semantic relations.

The entities and their relations are represented by a directed weighted semantic network. Formally, this network is defined by the tuple $G = (N, W)$. For the semantic graph G, N is the set of nodes in the network and W is the set of weighted labeled edges such that $W \subseteq N \times N$, $w^{\lambda}_{n_i,n_j}$ connects node n_i and n_j according to the semantic label λ, and $w^{\lambda}_{n_i,n_j} \in \mathbb{R}$. Furthermore, $\lambda \in \Sigma^*$ where Σ is some alphabet. The set N is composed of four non-intersecting sets of heterogeneously typed nodes. The four sets are human nodes H, the domain nodes D, the problem nodes P, and the solution nodes S. Note that $H \cup D \cup P \cup S = N$ and $H \cap D \cap P \cap S = \varnothing$. These four sets and the description of their nodes are as follows:

- $H \subseteq N$: the set of all humans participating in the system
- $D \subseteq N$: the set of all domains used by humans to categorize entities
- $P \subseteq N$: the set of all problems in the system
- $S \subseteq N$: the set of all solutions proposed for the problems in P

Each human node, $h_i \in H$, is associated with a personal collection of domain nodes used to categorize their social relationships and to categorize their problems. The set of domain nodes for human h_i are denoted $h_i(D)$ such that $D = \bigcup_{\forall i} h_i(D)$. Any particular domain d_l of human h_i is indexed as $h_i(d_l)$. Furthermore, $h_i(D) \cap h_j(D) = \varnothing \ : \ i \neq j$. Any problem p_j has a collection of solution nodes associated with it, $p_j(S)$, such that $S = \bigcup_{\forall j} p_j(S)$. Furthermore, $p_i(S) \cap p_j(S) = \varnothing \ : \ i \neq j$. The remainder of this section will describe the graph-theoretic representations of humans and domains (the social space) and problems and solutions (the problem space).

3.1 The Social Space

The social space is the subset of G that contains all humans, their domains and their relationships to other humans. In their most elaborated form, domains are related to each other by similarity (see 3.1.4). Before we look at the multiple ways of representing domains, we introduce the most fundamental construct of the social space: human trust-based social networks.

3.1.1 Social Trust Networks

The relationships that people create between one another in the social space are trust relations. For example, human h_0 can make explicit his or her trust in human h_1. The meaning of the trust relation will depend on the domain context and the individuals being connected. For example, trust could be a measure of similarity (Ziegler & Golbeck, 2005): human h_0 believes that human h_1 will make decisions that are in accord with their value system or h_0 perceives h_1 to possess comparable expertise. In other contexts, trust is based on differences in expertise, as in the case that h_0 lacks expertise, h_0 believes that h_1 will make a better decision – that is, h_0 sees h_1 as being more expert in the domain. The degree of trust that h_i has for h_j can be represented by the conditional probability, $\mathsf{P}(A|B)$,

$$w^{trusts}_{h_i,h_j}(h_j \text{ is trustworthy} \mid h_i\text{'s knowledge of } h_j) \tag{1}$$

Furthermore, given that humans are highly diversified entities (i.e. multi-faceted, multi-skilled), trust relations in reality are domain specific, or context sensitive. This is the role of the

set of domain nodes connected to an individual, $h_i(D)$. Human h_0 can trust that h_1 will make 'good' decisions in the domain of d_0. Generally,

$$w^{trusts}_{h_i(d_l),h_j}(h_j \text{ is trustworthy in } d_l \mid h_i\text{'s knowledge of } h_j \text{ in } d_l) \quad (2)$$

There are multiple ways to contextualize trust in the social space of G. We will now describe three different ones, each progressively more complex, but each subsuming the previous. The last model generalizes the first two.

- *single domain model*: trust has a single, undifferentiated context
- *multiple unrelated domains model*: trust is context-sensitive
- *multiple related domains model*: trust is context-sensitive and contexts are related to each by similarity

Depending on the requirements of a particular system implementation, one model may be more appropriate than another. Following our presentation of the three models for trust context. The end of this section will discuss various methods for automatically generating domains and their relationships.

3.1.2 Single Domain Model

The simplest model, the single domain model, represents all trust relations and problem categories under a single domain context. Therefore a semantic network using this model does not actually require the domain construct. An example of a trust relation in the single domain model is $w^{\text{trusts}}_{h_i,h_j}$. This weighted edge states that human h_i trusts human h_j to some degree, irrespective of the context. A context-insensitive social trust network is shown in Figure 2. Note that nodes h_0 and h_1 have associated metadata (e.g. Brad and Victor). As will be used throughout the remainder of this paper, the function $\text{key}(n_i, \Omega) \in \Sigma^*$ returns the data associated with key Ω. For instance, in the following example, $\text{key}(h_0, \text{type}) \rightarrow \text{human}$ and $\text{key}(h_0, \text{name}) \rightarrow \text{Brad}$.

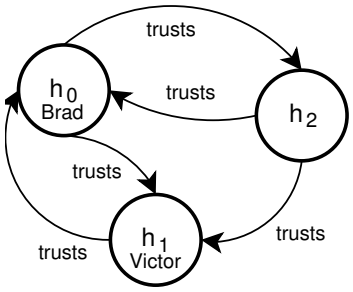

Figure 2: A trust-based social network where trust is irrespective of context

For collective decision making systems with a focused problem space (i.e. a single domain), the single domain model would be the most appropriate since the complexity of the implementation is minimal. However, for large-scale collective decision making systems, the problem space will tend to be composed of problems from various domains and therefore the trust that one delegates to another may need to be contextualized. For example, h_0 may trust human h_1 in domain d_0, but not in domain d_1. Therefore, in such situations, the single domain model is insufficient to represent the complexity of the social space. The next section introduces context-sensitive trust networks.

3.1.3 Multiple Unrelated Domains Model

The multiple unrelated domains model allows the individual to specify the context for which trust is given. Any human $h_i \in H$ is associated with a collection of unique domain nodes, $h_i(D)$. The nodes $h_i(D)$ and their relations compose an individual's *personal domain network*. The purpose of domains is to allow individuals to categorize the type of trust relations they have with one another and to categorize the types of problems contained within the problem space (Section 3.2). Therefore, in this model, it is possible to for h_0 to project a trust-based relation to h_1 in the domain d_0. Domain specific trust can be represented in one of three ways. The purpose of the three representations is to build up to the last model which will be used throughout the remainder of this paper. The first two representations are to help the reader to better understand the model as a transition from the single domain model presented previous.

For the first representation, it is possible to allow the group to create individual trust-based social networks for each domain of the problem space. This idea is depicted in Figure 3 where there exists two domains, d_0 (societal-scale decision making systems) and d_1 (group decision support systems), and therefore, there exists two independent trust-based social networks.

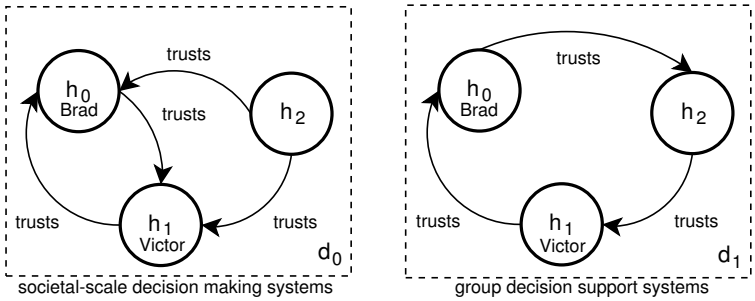

Figure 3: A representation of the multiple related domains model

In this representation, a domain is a container for the trust-based social network within. It is possible to invert this representation and still preserve all the same information. Figure 4 depicts domain-specific trust where the human is the container for domains.

Figure 5 provides a final representation of the same information. Figure 5 explicitly represents a human node. A trust-based edge is represented as $w^{\text{trusts}}_{h_i(d_l),h_j}$. This notation states that human h_i trusts human h_j within the domain d_l. Furthermore, any human node h_i connects to every node in their personal domain network by the edge $w^{\text{uses}}_{h_i,h_i(d_l)}$. It is important to emphasize that this final representation subsumes the previous two but is not more general because it adds human nodes and relations to the domains they use. However, any of these representations can be collapsed (i.e. domains removed) to represent the single domain model presented in Section 3.1.2.

The allowed edge types allowed by the multiple unrelated domain model is as follows:

- $w^{\text{uses}}_{h_i,h_i(d_l)}$: human h_i uses domain $h_i(d_l)$ in their personal domain network
- $w^{\text{trusts}}_{h_i(d_l),h_j}$: human h_i trusts human h_j in domain d_l

It happens to be the case that, in Figure 5, d_0 and d_1 are similar domains because much of the literature on societal-scale decision making systems refers to group decision support system research. Therefore it is likely that, because h_0 trusts h_1 in domain d_0, h_0 would also trust h_1

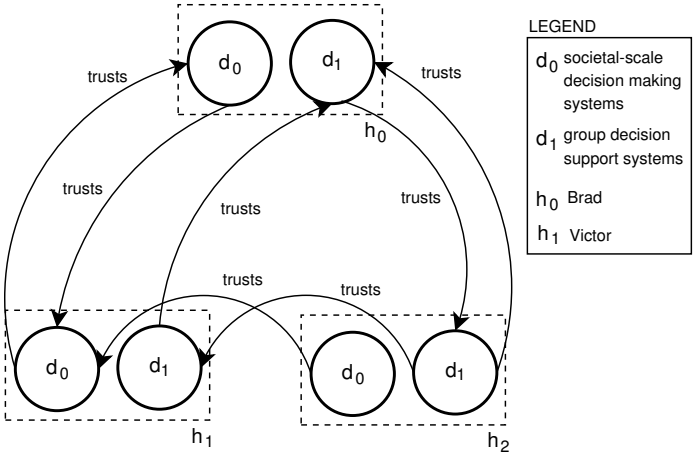

Figure 4: Another representation of the multiple related domains model

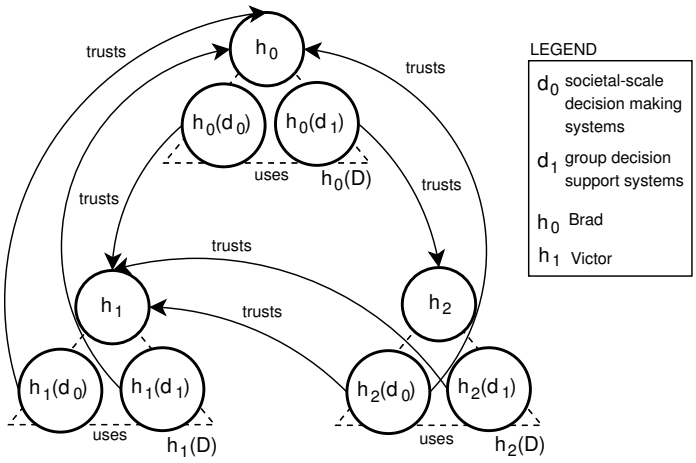

Figure 5: Yet another representation of the multiple unrelated domains model

in domain d_1, to some degree. The idea of inferred trust through domain similarity cannot be made explicit in the multiple unrelated domains model because domains are not related by any measure of similarity. Therefore, in this model, it is necessary to create a trust edge for each domain of the problem space. For smaller decision making systems with only a few problem domains, this architecture may suffice, but as the system scales to encompass many domain generated in an ad hoc manner, requiring every individual to specify who they trust in every domain may become an undue burden. As a remedy, the final domain model introduces domain similarity as a way to infer trust across context boundaries.

3.1.4 Multiple Related Domains Model

In this final social space model, individuals are not only able to explicitly represent the domain for which trust is given, but also to specify the degree of similarity between various domains (Rodriguez, 2004). The domain network is thus a map representing an individual's mental model of related concepts (Heylighen, 1999; Bollen, 2001). There are three types of edges

used to represent the multiple related domain model.

- $w^{\text{uses}}_{h_i,h_i(d_l)}$: human h_i uses domain $h_i(d_l)$ in their personal domain network
- $w^{\text{trusts}}_{h_i(d_l),h_j}$: human h_i trusts human h_j in domain d_l
- $w^{\text{similarTo}}_{h_i(d_l),h_i(d_m)}$: human h_i believes that domain d_l is similar to domain d_m

The set of h_i's personal domain nodes, $h_i(D)$, are related by the property of $\lambda = \text{similarTo}$. For instance, $w^{\text{similarTo}}_{h_0(d_0),h_0(d_1)}$ states that, human h_0 believes that domain d_0 is similar to d_1 by the amount dictated by the edge weight. By allowing for the explicit representation of domain similarity, the amount of trust edges between individuals can be reduced without losing necessary trust inference information. For instance, if human h_0 believes that domain d_3 is similar to domain d_0, then if h_0 trusts h_1 within domain d_0, then it can be inferred that h_0 trusts h_1 within d_3 by an amount that is a function of $w^{\text{similarTo}}_{h_0(d_3),h_0(d_0)}$. The domain similarity and human trust edges are depicted in Figure 6.

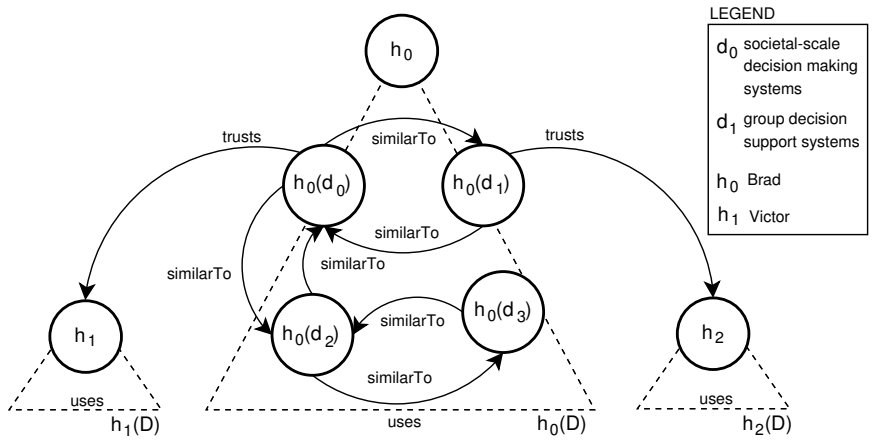

Figure 6: A representation of the multiple related domains model

The remainder of this paper will focus specifically on the multiple related domains model. It is trivial to modify the forthcoming algorithms to support the other two domain models.

3.1.5 Generating Domains and Determining Their Relationships

Each individual's personal domain network contains the same number of nodes, $|h_i(D)| = |h_j(D)|$ where $i \neq j$, $h_i(D) \cap h_j(D) = \varnothing$ where $i \neq j$, and finally, $\text{key}(h_i(d_0), \text{name}) \equiv \text{key}(h_j(d_0), \text{name})$. Therefore, there exists a abstract collection of domains available to the user for contextualizing trust relations. The question that remains is how domains are instantiated in the system. There are two extreme examples of domain generation. The first example is a *top-down approach* where by system administrators/designers prescribe the set of domains allowed for the human users. If the system is being used for focused decision making with well defined domains, then this approach may be the most appropriate as the amount of domains can be contained to a necessary stable set. If the system is open to general purpose collective decision making, where the problem space is a priori unknown, then a more *bottom-up approach* may be more appropriate. The bottom-up approach allows users to define domains as they

see fit. This approach is similar to the folksonomy concept of ad hoc classification schemes (Mathes, 2004). The main issue with the folksonomy tagging approach is that the amount of domains can become very large, very fast. One *hybrid approach* is to pose the problem of domain generation to the group: What domain should we add next?. By using the system to generate its own domains, the individuals are, in a bottom-up fashion proposing new domains, while at the same time, the collective is, in a top-down fashion, regulating the influx of new domains.

It is possible for each individual's internal domain network to contain a different set of edges. This means that the subjective realization of the similarity between two domains can differ amongst individuals. When determining domain similarity, like domain generation, there are two extreme scenarios. The manual, bottom-up approach, is such that every individual determines the relationship between each and every domain within their domain network. Algorithms to facilitate this manual process can be found in the literature on self-organizing, or adaptive, hypertext networks (Bollen & Heylighen, 1998; Bollen, 2001). For a relatively small amount of domains, this approach may be sufficient. On the other hand, in a fully open folksonomy domain generation environment where the potential for a large amount of domains exists, domain similarity can be generated automatically via various statistical analysis techniques of documents external and internal to the system. For example, It is possible to generate co-occurrence networks (domain similarity networks) of terms by mining web or manuscript resources. In relation to Figure 6, the concept societal-scale decision making systems will tend to co-occur with group decision support systems, therefore there will exist a $\mathrm{similarTo}$ edge between these two domains in every individual's domain network. Also, it is possible to analyze how domain edges such as trusts, $\mathrm{similarTo}$, and $\mathrm{categorizedAs}$ (Section 3.2) co-occur in the system's semantic network. In this way, the system can use itself to generate more associations between human domain nodes. Finally, a hybrid approach could automatically generate a collectively defined domain similarity network that can then be fine tuned by the individual human. The various approaches to domain node and domain edge generation are presented in Table 1. It is not assumed that Table 1 encompasses all possible methods.

	domain node generation	**domain edge generation**
top-down	administered	data mining
bottom-up	folksonomy approach	manual method
hybrid	collectively decided	combination

Table 1: Multiple methods for node and edge generation in domain networks

This section has outlined the social space where by individuals can create trust relations amongst themselves. The novelty of this social space model is in the inclusion of domains which allows an individual to contextualize the type of trust one individual has for another. Furthermore, methods for the generation of domain nodes and domain similarity edges has been presented to provide insight into how personal domain networks grow within the system.

3.2 The Problem Space

The problem space is the set of all problems created and solutions proposed by the collective, $P \cup S$. In the problem space, individuals can create and categorize problems and propose and vote on solutions. The solution nodes, S, represent the set of all proposed solutions to the problems facing the group. The set $p_j(S)$ is the set of all solutions proposed for problem p_j such that $S = \bigcup_{\forall j} p_j(S)$. The unification of the problem space and the social space is depicted in Figure 7.

- $w^{\text{hasProposed}}_{p_j,p_j(s_m)}$: problem p_j has proposed solution s_m
- $w^{\text{created}}_{h_i,p_j}$: human h_i created problem p_j
- $w^{\text{categorzied}}_{h_i(d_l),p_j}$: human h_i categorized problem p_j as being in domain d_l
- $w^{\text{proposed}}_{h_i,p_j(s_m)}$: human h_i proposed solution s_m for problem p_j
- $w^{\text{votedOn}}_{h_i,p_j(s_m)}$: human h_i voted on solution s_m as the solution for p_j

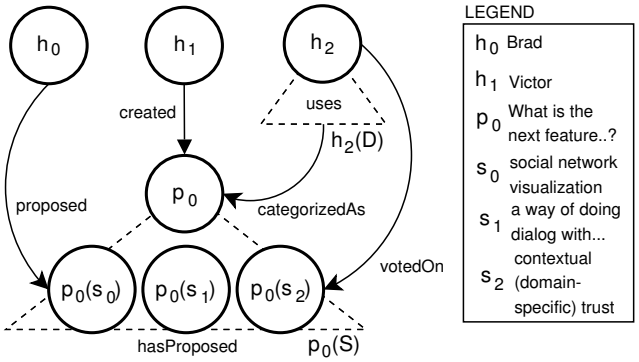

Figure 7: The unification of the social and problem spaces

It is important to note that the votedOn edges are the explicit representation of the results of the individual's solution ranking algorithm. Remember that individual solution ranking is the first stage in collective decision making as originally presented in Figure 1. Individual solution ranking is a process that occurs internal to the individual's cognition. Therefore, in order to retrieve the results of their internal algorithm, a means of querying the individual is necessary. This is possible through a web-based representation of the semantic network whereby individuals can determine which problems exist, which solutions have been proposed, and finally, to submit their perspective on what is the most optimal solution ranking for a particular problem. Note that no two solution sets for any problem intersect, $p_i(S) \cap p_j(S) = \varnothing$, since, given the framework so far, it would be impossible to determine for which problem any individual was ranking the solution set for.

The problem space model presented thus far is only what is required to implement collective decision making. Other problem space models do exist and have been reported in the literature. For instance, the argumentation problem space model of the SDSS (social decision support system) described in (Turrof et al., 2002) not only has problems (issues) and solutions (options), but also comments. Comment nodes allow individuals to remark on the proposed solutions to a problem. Comments have edges to solutions by way of the semantic projections $\lambda \in \{\text{pro}, \text{con}, \text{neutral}, \text{inquiry}\}$. Furthermore, comments can also have edges to other comments by means of the semantics of $\lambda \in \{\text{opposing}, \text{complementary}\}$. This argumentation problem space model allows users to communicate (argue) with one another for the purposes of creating consensus prior to collective decision making (Turoff, Hiltz, Bieber, & Rana, 1999).

3.3 Extensibility and Scalability

The rational for using the aforementioned semantic network data structure is *extensibility* and *scalability*. The particle swarm algorithms to be described in Section 4 are grammar-based algorithms that calculate functions only on certain subsets of the full semantic network. This means that particle behavior is dictated by the λ edges of the network. Therefore, the inclusion

of more edge and node types will not affect the calculations these algorithms. For this reason, it is possible to extend the proposed problem space model with the argumentation entities presented in (Turrof et al., 2002). Also, it is possible to extend the system by including more particle swarm algorithms to take advantage of more node and edge types.

In terms of scalability, given a large population with a large amount of problems, representing the entire graph in a single computation is inefficient, and for large systems, potentially impossible. Given that the algorithms described next process only a small subset of the graph, selective harvesting from the main semantic network data structure will tend to be the more scalable solution (Lee, 2004). For example, in collective decision making, for a particular problem, only that problem p_j and its solutions $p_j(S)$ and those participating individuals H and their domain networks $H(D)$ are harvested from the main semantic network.

4 Collective Ranking with Particle Swarms

Thus far, what has been presented, is a collective decision making ontology that determines the entities and relations in a particular semantic network data structure instantiation. Contained within any instantiation is a set of humans, their domains, their problems, and their solutions. It is provided, that via some web-based interface, individual are able to create problems, propose solutions, vote on solutions, create trust relations, instantiate domains, and modify domain similarities. That is, the individuals in the collective are able to directly affect the evolution of the semantic network data structure. By computing various algorithms on this evolving data structure, it is possible to yield emergent collective preferences.

Of particular importance to collective decision making is determining the collective solution ranking for a particular problem. The results of the the individual solution ranking algorithms of the humans are made explicit by means of the $\mathrm{votedOn}$ edges. In order to move from an individual preference to a collective preference, it is necessary to compute a collective solution ranking algorithm on the network. To do this, a general framework for aggregating perspectives is presented. In this framework, a particle swarm traverses the network, identifying strong solutions as being those which have the most and shortest paths from the human nodes. By varying the parameters of the particle swarm, different collective solution ranking algorithms can be instantiated. From this single framework it is possible to go from a dictator scenario to a social network based representative democracy.

Section 4.1 will focus on the the generic particle swarm shell which all particle swarm instantiations rely upon before discussing the particulars of the various collective solution ranking algorithms. Methods for automatically determining which algorithm is most appropriate given a particular problem context is presented in Section 4.2.4. Finally, Section 4.3 will demonstrate how a particle swarm can be used to determine the domain of a problem.

4.1 The Generic Particle Swarm

A particle swarm is a collection of indivisible entities that propagate through a network in order to compute a distributed function (White & Pagurek, 1998; Rodriguez & Bollen, 2005). The purpose of the particle swarm algorithms presented in this paper is to calculate a node rank distribution. Given a network G, a particle swarm can be used to determine the rank of a collection of nodes relative to another group of nodes. These algorithms are sometimes called relative rank, or network influence, algorithms (White & Smyth, 2003; Rodriguez & Bollen, 2005). Within the context of collective decision making, for a particular problem p_j, it is desirable to rank each solution, $p_j(S)$, relative to a group of humans, H. In solution ranking,

the humans compose the input node set, $I \subseteq H$, and the solutions are the output node set, $O = p_j(S)$. Furthermore, beyond solution ranking, other node rankings can be calculated. For example, it may be desirable to rank each domain, $I = D$, relative to a particular problem, $O = p_j$. This is called domain ranking and is used to determine problem categorization.

There exists a few constructs that all different particle swarm ranking algorithms presented in this papers rely upon. First, any particle $q_i \in Q$, is composed of two variables: a reference to its current node $c_i \in N$ and its current energy value $\epsilon_i \in \mathbb{R}$. These variables are itemized below for ease of reference:

- $c_i \in N$: the current location of particle q_i
- $\epsilon_i \in \mathbb{R}$: the current energy content of particle q_i

A particle q_i starts at its home node $c_i(t = 0)$ and traverses an outgoing edge from its current node c_i at each time step t. Graph traversal is a stochastic process that requires c_i's outgoing edge weights to form a probability distribution. Therefore, the weights must be normalized to 1.0. It is important to note that, depending on the particle swarm grammar, only certain edges can be traversed. For example, in Figure 8, the set R is the set of edges that the particle can traverse if the grammar states that only aaa edges are allowed. The function $\Theta(R) \rightarrow c_i(t+1)$ is a stochastic node selection function that returns the next node destination of particle q_i given the probability distribution formed over the weight set R. In Figure 8, if the particle is currently at node n_0, then node n_1 is the value returned by $\Theta(R)$ at $t+1$, $c_i(t) = n_0$ and $c_i(t+1) = n_1$.

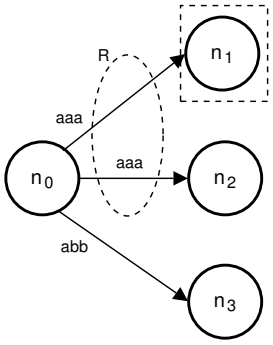

Figure 8: Edge traversal is dependent upon a subset of all outgoing edges of c_i

Every node in the graph has an associated energy, or activation, value. This is denoted by the vector $\vec{e} \in \mathbb{R}^{|N|}$. This vector is indexed by the node name. Therefore, human h_i's activation value is $\vec{e}_{h_i}$ and the current node of particle q_i's energy value is $\vec{e}_{c_i}$. Each time a particle q_i arrives at a node, it increments the node's activation value with its current energy value, ϵ_i. This is represented in Eq. 3.

$$\vec{e}_{c_i}(t+1) = \vec{e}_{c_i}(t) + \epsilon_i \tag{3}$$

Every time a particle leaves its current node, the particle decays its energy value by a global decay value $\delta \in \mathbb{R}$ as expressed in Eq. 4 and 5.

$$\epsilon_i(t+1) = (1-\delta)\epsilon_i(t) \tag{4}$$

such that,

$$\epsilon_i(t) = (1-\delta)^t \epsilon_i(0) \tag{5}$$

Once a certain k step has been reached or all nodal energy has decay to near zero, $\sum_{i=0}^{|Q|} \epsilon_i \approx 0.0$, the algorithm is complete and the desired output set's activation values can be normalized to 1.0 to create a ranking over O as demonstrated in Eq. 6. The values in $\vec{e}(k+1)$ over the set O, denoted $\vec{e}_O$, is the output ranking.

$$\vec{e}_{n_i}(k+1) = \frac{\vec{e}_{n_i}(k)}{\sum_{j=0}^{j<|O|} \vec{e}_{n_j}(k)} \ : \ n_i \in O, n_j \in O \tag{6}$$

Algorithm 1 provides the pseudo-code for ranking the entire network relative to itself ($I = O = N$). Algorithm 1 is a discrete form of the iterative method for calculating the primary eigenvector of the full semantic network (Rodriguez & Bollen, 2005). This algorithm and its constructs form the foundation for the different collective ranking algorithms described next.

```
1  δ = 0.0;
2  #distribute particles to the input group;
3  foreach (n_i ∈ N) do
4  |    c_i = h_n, ε_i = 1.0, q_i ≠ ∅;
5  end
6  #iterate until some desired k-step is reached;
7  while (t ≤ k) do
8  |    foreach (q_i ∈ Q : q ≠ ∅) do
9  |    |    ē_{c_i} = ē_{c_i} + ε_i;
10 |    |    ε_i = (1 − δ) × ε_i;
11 |    |    R = ⋃_{∀j,λ} w^λ_{c_i,n_j};
12 |    |    c_i = Θ(R);
13 |    |    if (ε_i ≈ 0.0) then
14 |    |    |    q_i = ∅;
15 |    |    end
16 |    end
17 end
```

Algorithm 1: Generic particle swarm rank algorithm for all λ edges

4.2 Collective Solution Ranking Algorithms

After individual solution ranking, which is made explicit via the votedOn edges defined in the problem space, the second component of collective decision making is collective solution ranking. A collective solution ranking algorithm is the process that aggregates the different individual solution rankings into a single collective ranking. As will be demonstrated, there are many ways to aggregate individual solutions. The following sections will formalize three collective solution ranking algorithms: direct democracy (DD), representative democracy (RD), and dynamically distributed democracy (DDD). Table 2 outlines a few scenarios in which the various algorithms are most appropriate. A double arrow is intended to signify a exaggeration of the appropriateness, e.g. $\Uparrow$ signifies 'most appropriate'. The reasons for the provided arrows are explained in the sections describing each algorithm.

	large problem space	expert-based domains	value-based domains
DD	$\Downarrow$	$\Downarrow$	$\uparrow$
RD	$\uparrow$	$\uparrow$	$\uparrow$
DDD	$\Uparrow$	$\Uparrow$	$\Uparrow$

Table 2: Scenarios for various collective solution ranking algorithms

4.2.1 Direct Democracy Swarm

Direct democracy embodies the idea of one man/one vote (Lijphart, 1984). Every individual in the group is allowed to rank the solutions to a particular problem p_j and the influence of all individual solution rankings is equal. If an individual h_i does not vote, then h_i's preference is left out of the collectively derived solution ranking of $p_j(S)$. The particle swarm grammar, or rule set, to determine the collective solution ranking for a particular problem p_j is the most basic of the three presented collective solution ranking algorithms. The direct democracy swarm algorithm is presented in Algorithm 2.

```
1  harvestSubnetwork(I ∪ I(D) ∪ p_j ∪ p_j(S));
2  while (t ≤ k) do
3      #distribute a single particle to all individual humans;
4      foreach (h_n ∈ I) do
5          c_i = h_n, ε_i = 1.0, q_i ≠ ∅;
6      end
7      #iterate until some desired k-step is reached;
8      foreach (q_i ∈ Q : q_i ≠ ∅) do
9          e⃗_{c_i} = e⃗_{c_i} + ε_i;
10         #if at a human node, choose a solution or kill
           particle;
11         if (key(c_i, type) == human) then
12             R = ⋃_{∀m} w^{voteOn}_{c_i,p_j(s_m)};
13             if (|R| == 0) then
14                 q_i = ∅;
15             end
16             else
17                 c_i = Θ(R);
18             end
19         end
20         #if at a solution node, return home;
21         else if (key(c_i, type) == solution) then
22             c_i = ∅;
23         end
24     end
25 end
26 updateMetadata(e⃗_{p_j(S)});
```

Algorithm 2: Direct democracy particle swarm for ranking the solution set of problem p_j

The function $\text{harvestSubnetwork}(I \cup I(D) \cup p_j \cup p_j(S))$ (line 1) harvests a collection of human nodes, $I \subseteq H$, their respective domains $I(D)$, the problem node p_j, and p_j's proposed solutions $p_j(S)$. First, every individual in the group is provided a single particle (line 4). Second, every particle must take a $w^{\text{votedOn}}_{h_i,p_j(s_m)}$ edge (line 12). If no such edge exists, then the particle kills itself (line 14). Finally, if a particle is at a solution node, then, the particle destroys itself (line 22). This process is repeated until the solution ranking stabilizes. This is represented in Eq. 7 by the cosine similarity calculation of the solution set energy vector, $O = p_j(S)$, at

different time steps.

$$\frac{\vec{e}_O(t) \cdot \vec{e}_O(t+1)}{\|\vec{e}_O(t)\| \cdot \|\vec{e}_O(t+1)\|} \approx 1.0 \tag{7}$$

For simplification of notation, when this stabilization occurs, the algorithm is assumed to be at its k-th step. A finite state machine representing the particle grammar is presented in Figure 9.

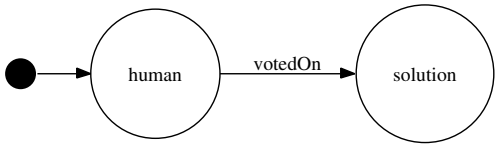

Figure 9: Finite state machine describing the direct democracy solution ranking grammar

It is possible to allow the set R to be greater than 1 (line 12). In such cases the human has voted on multiple solutions. With multiple votes, the repeated cycling from human to solution as $t \rightarrow \infty$ will ensure that the particle traverses each human's voted on solutions a number of times proportionate to the weight value $w^{\mathrm{votedOn}}_{h_i,p_j(s_m)}$. For example, in Figure 10, human h_1 has voted on two solutions with varying degrees of confidence. Each new time step t, it has a probability of taking one of the two $\mathrm{votedOn}$ edges as defined by the probability distribution over the edge weights. At $t = 100$ the particle will have traversed solution $p_0(s_1)$ approximately 60 times and $p_0(s_2)$ approximately 40 times. If each individual is only allowed to choose one solution, then after $t = 1$ the algorithm will have arrived at its stable solution set ranking.

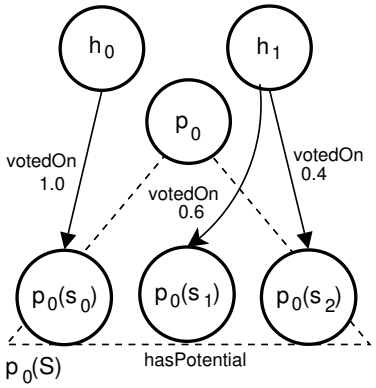

Figure 10: Repeated cycles ensures a proportionate energy over an individual's solution rankings

At the end of the k-th step (line 2), the function $\mathrm{updateMetadata}(\vec{e}_{p_j(S)})$ updates the solution metadata with its appropriate ranking (line 26). The resulting direct democracy ranking is used to update the metadata associated with each solution node in the semantic network.

For value based problems, the one man/one vote equality strategy of direct democracy may prove beneficial but, as the problem space grows in size, not all members of the collective will be able to participate in all decision making processes. Therefore, direct democracy is problematic in that those individuals who have actively participated (voted) have no influence in the

collectively derived solution ranking. Moreover the direct democracy strategy can not identify expertise in a problem domain. Without the trust-based social network, biasing the influence of different individuals is not possible. It can not be known which individuals, for a particular domain, are the most trusted individuals.

4.2.2 Representative Democracy Swarm

The direct democracy strategy reduces the flexibility of the collective in that decision making processes cannot easily be parallelized. If each individual wishes to participate in the decision making of the group, then each individual must rank the solution set for each problem. Such serialization is not practical as the amount of problems facing the group increases. With a large problem pool, not all issues can be addressed by each and every individual. Therefore, in such situations, human collective decision making systems can rely on representation. Representation makes use of the trusts edges in the social space. Individuals that do not provide an individual solution ranking for a particular problem can delegate their decision making influence (their particle) to a trusted representative.

The representative democracy swarm takes advantage of domain specific trust. Domain specific, or contextualized, representation refers to the notion that an individual can trust another individual in one domain, but not in another. For this reason, it is important to delegate an individual's decision making influence (particle) to a representative with respects to the domain of the problem. Therefore, before delegation can occur, it is necessary to determine the problem's domain. Section 4.3 will provide a collection of particle swarm domain ranking algorithms for ranking a domain relative to a particular problem via the use of the $\mathrm{categorizedAs}$ edges. For now, a simple method is provided.

Problem domain categorization, or ranking, at the individual level is made possible via the use of the $w^{\mathrm{categorziedAs}}_{h_i(d_l),p_j}$ edges. These edges states that human h_i believes that problem p_j is in the domain of d_l. Furthermore, it is possible to categorize a problem over multiple domains with multiple weighted $\mathrm{categorizedAs}$ edges such that normalized set of all $\mathrm{categorizedAs}$ edges forms the individuals representation of problem p_j's domain. The normalized $\mathrm{categorizedAs}$ provides a probability distribution which is reflected, for that particular decision making scenario, in the individual's uses edges. For instance, if human h_0 tagged problem p_0 with a 0.60 $\mathrm{categorizedAs}$ edge from domain $h_0(d_1)$ and a 0.40 edge from domain $h_0(d_2)$ then the particle has a 60% chance of jumping to $h_0(d_1)$ and a 40% chance of jumping to $h_0(d_2)$. Therefore, the weight $w^{\mathrm{uses}}_{h_0,h_0(d_2)}$ is 0.60 and $w^{\mathrm{uses}}_{h_0,h_0(d_1)}$ is 0.40. In general, for those who have provided a categorization of the problem, $\forall l\ w^{\mathrm{uses}}_{h_i,h_i(d_l)} = w^{\mathrm{categorizedAs}}_{h_i(d_l),p_j}$. For those individuals that have not defined the domain of the problem, they can rely on a collectively generated domain. The particle swarm method for determining the collectively derived domain of a problem is presented in Section 4.3. For now, it should be understood that, for a particular human h_i, $\bigcup_{\forall l} w^{\mathrm{uses}}_{h_i,h_i(d_l)}$ is a direct reflection of the problem domain, self or collectively generated. The uses weight values are determind at the collective solution ranking algorithm's runtime.

A representative democracy particle swarm is an extension to the direct democracy swarm and is represented in Algorithm 3. First, every individual is supplied with a single particle (line 4). A particle, if it can, will take a $w^{\mathrm{votedOn}}_{h_i,p_j(s_m)}$ edge to an individual's chosen solution (line 12). If no such edge exists, then the particle will jump to a domain in the individual's domain network as specified by the probability distribution over problem p_j's problem domain (line 18). If there exists a $w^{\mathrm{trusts}}_{h_i(d_l),h_j}$ edge and the human h_j has a $\mathrm{votedOn}$ edge (i.e. h_j is a voting representative), then the particle will take that edge (line 24). If not, then the particle will take a $w^{\mathrm{similarTo}}_{h_i(d_l),h_j(d_n)}$ edge (line 29). If no such edges exist, then the particle kills itself (line 34). Upon reaching a solution node, the particle will kill itself (line 40). This process of distributing

particles and propagating them from a human to a solution continues until some desired k-step is reached that will ensure a solution ranking that has converged to a stable set of values. The finite state machine of this grammar is presented in Figure 11.

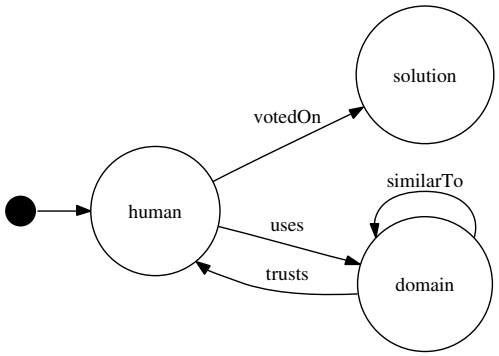

Figure 11: Finite state machine describing the representative democracy solution ranking grammar

It is important to note the energy decay parameter, δ, in the representative democracy swarm (line 2). Intuitively, the further a particle must travel before it finds a solution node, the less that individual's vote is reflective on the individual's intentions. There are two extremes to this argument. First, if the particle goes from the human strait to a solution via a $\mathrm{votedOn}$ edge, then the intent of the individual is known and the effects of decay are less pronounced as only a single step is taken from source (human) to sink (solution). The human made explicit their desired solution to the problem. If the particle must move through the individual's network of domains to a representative, the more muddled the individual's perspective becomes. The further the particle travels from source to sink, the less explicit the perspective of the individual. The other extreme of this argument is when the particle can find no solution to a problem. In such cases, the individual has no model of that aspect of the problem domain (i.e. is not connected to the appropriate solutions or representatives) and therefore there is no way of knowing the individual's preference. To reflect this lack of information, the particle dies when too many steps are taken, $\epsilon_i \approx 0.0$.

If trust is a measure of similarity of thought between two individuals within a particular domain, then one individual voting for two will reflect both individual perspectives – the participating individual's solution ranking is weighted twice. In such cases, representation is a form of social compression (Rodriguez & Steinbock, 2004b, 2004a; Rodriguez, 2004) and it is this idea that allows for parallelization of the decision making efforts of the collective. If every individual in the group has provided a solution ranking for the problem, then representative democracy is identical to direct democracy as the particle is able to take a $\mathrm{votedOn}$ edge to a solution.

On the other hand, if the domain of the problem is more expert based, individuals are weighted according to their perceived level of expertise. Individuals that abstain from participation are reducing the noise in the system (reducing poor individual solution rankings), and are, at the same time, ensuring that those individuals who are competent in the domain are weighted appropriately. In this sense, non-participants are creating an individual human ranking via their trusts edges. Representative democracy is important in expert-based problem domains where non-expert participation decreases. The trust in-degree of the representative is a expression of the individuals expertise (Wasserman & Faust, 1994).

The drawback of the representative democracy algorithm is that individual decision making

```
1  harvestSubnetwork(I ∪ I(D) ∪ p_j ∪ p_j(S));
2  δ = 0.15;
3  while (t ≤ k) do
4      foreach (h_n ∈ I) do
5          c_i = h_n, ε_i = 1.0, q_i ≠ ∅;
6      end
7      while (∀_i q_i ≠ ∅) do
8          foreach (q_i ∈ Q : q_i ≠ ∅) do
9              e⃗_{c_i} = e⃗_{c_i} + ε_i;
10             #if the human has voted, choose a solution;
11             if (key(c_i, type) == human) then
12                 R = ⋃_{∀m} w^{voteOn}_{c_i,p_j(s_m)};
13                 if (R ≥ 1) then
14                     c_i = Θ(R);
15                 end
16                 #else enter the individual's domain
                   network according to the problem domain;
17                 else
18                     R = ⋃_{∀l} w^{uses}_{c_i,h_i(d_l)};
19                     c_i = Θ(R);
20                 end
21             end
22             #find a representative or a related domain;
23             if (key(c_i, type) == domain) then
24                 R = ⋃_{∀l} w^{trusts}_{c_i,h_l} : ∃w^{voteOn}_{h_l,p_j(s_m)};
25                 if (R ≥ 1) then
26                     c_i = Θ(R);
27                 end
28                 else
29                     R = ⋃_{∀l} w^{similarTo}_{c_i,h_i(d_l)};
30                     if (R ≥ 1) then
31                         c_i = Θ(R);
32                     end
33                     else
34                         c_i = ∅;
35                     end
36                 end
37             end
38             #if at a solution node, return home;
39             else if (key(c_i, type) == solution) then
40                 q_i = ∅;
41             end
42             ε_i = (1 − δ) × ε_i;
43             if (ε_i ≅ 0.0) then
44                 q_i = ∅;
45             end
46         end
47     end
48 end
49 updateMetadata(e⃗_{p_j(S)});
```

Algorithm 3: Representative democracy particle swarm for ranking the solution set of problem p_j

influence can only go to a solution or a voting representative. The algorithm is limited in that it is unable to handle situations where a non-voting individual has not created a trust edge to

a voting, or representative, individual. This means that the potential for particles not finding their way to a solution is greater. Figure 12 demonstrates the tri-partite model of representative democracy.

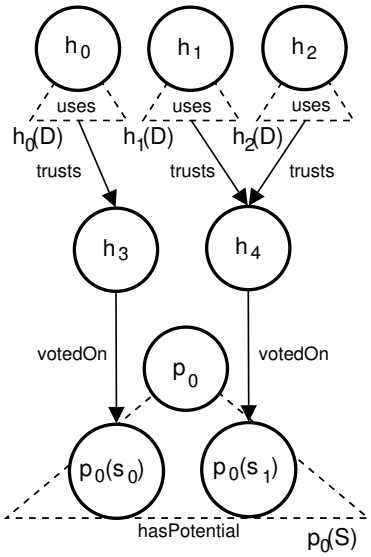

Figure 12: Tri-partite model of representative democracy

4.2.3 Dynamically Distributed Democracy Swarm

Dynamically distributed democracy (DDD) was first introduced in (Rodriguez & Steinbock, 2004b) and is a slightly modified form of the representative democracy swarm. Dynamically distributed democracy is able to capitalize on the benefits of the previous two algorithms. If every individual participates in the decision making process by providing a solution ranking, then DDD is identical to direct democracy. However, as individuals abstain from voting, DDD models the representative form of democracy in which an individual can delegate their decision making influence to a representative. Unlike representative democracy, that representative is not required to be a participating, or voting, individual.

The rule set for the DDD particle swarm is as follows. The particle begins at its human node h_i. If it can, it will take a $\mathrm{votedOn}$ edge to a solution to the problem p_j. If no such edge exists then the particle will enter the individual's domain network with a probability dictated by the problem domain of p_j, $\bigcup_{\forall l} w^{\mathrm{uses}}_{c_i,h_i(d_l)}$. If the particle can find a trusts relationship, the the particle will take that edge first before exploring $\mathrm{similarTo}$ relations. The only distinction between representative democracy and DDD is that DDD does not require one's trusted neighbor to have voted on a solution to problem p_j. Therefore, representation is recursive. The DDD pseudo-code differs from representative democracy in that at line 24 of Algorithm 3,

$$R = \bigcup_{\forall l} w^{\mathrm{trusts}}_{c_i,h_l} \ : \ \exists w^{\mathrm{voteOn}}_{h_l,p_j(s_m)} \tag{8}$$

is changed to

$$R = \bigcup_{\forall l} w_{c_i,h_l}^{\text{trusts}} \tag{9}$$

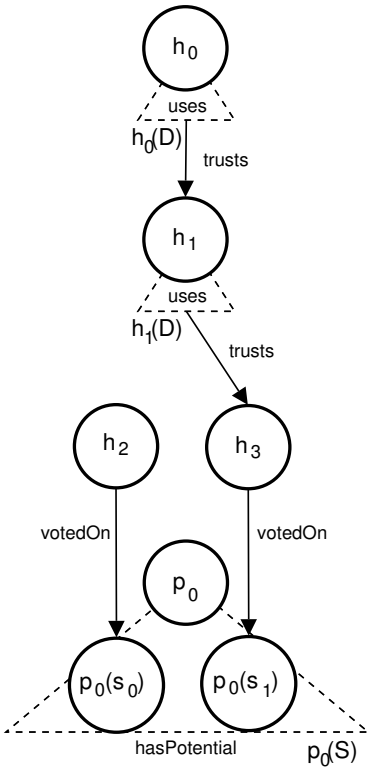

Figure 13: Social network model of dynamically distributed democracy

If individual h_i has not voted on a particular solution to a problem, then the h_i's particle is distributed to some trusted neighbor, h_j, within the domain of the problem. If that individual has not voted, then the particle is distributed to h_j's trusted neighbor. So on and so forth until the particle can find an individual that has voted on a solution to the problem. DDD is a recursive version of the representative democracy swarm where one's representative may be many step away in the trust-based social network. In representative democracy, the propagation of decision making influence takes a single step therefore, trust states how much one individual believes another individual to be a good decision maker in the domain. In DDD, since decision making influence propagates in a recursive fashion, trust states how much one individual trusts another individual in their evaluation of yet another individual (Golbeck, Parsia, & Hendler, 2003). Therefore, DDD is well suited for trust-based networks that are not bi-partite and are recurrent.

In expert based domains, h_i's expertise is identified not solely by h_i's in-degree, but also by the in-degree of the individuals that trust h_i. This recursive definition of expertise is similar to the PageRank algorithm's calculation of web-page prestige (Brin & Page, 1998) and may be a better measure of individual expertise (Wasserman & Faust, 1994). DDD has been shown, in simulation, for value-based problems, to be able to weight the active voters appropriately such that any set of voting individuals rank the solution set $p_j(S)$ as if every member of the collective had participated. The weighted voters are said to form a holographic model of the whole population (Rodriguez & Steinbock, 2004a; Rodriguez, 2004).

DDD strikes a balance between the benefits of expert identification and value-based modeling. If h_i is a voting individual, then the decision making influence propagated to h_i stops at h_i and therefore is not propagated to h_i's trusted neighbors. Experts can be identified by abstaining non-experts and value-based modeling is more accurate due to the lack of continuous propagation and muddled identification of intentions. A modification of the DDD algorithm, proxy-vote (deGraf & Grey, 2006), is a weighted version of the DDD algorithm that is more specific to expert-based domains. In proxy-vote, each individual is initially supplied an amount of particles equal their trust-based in-degree plus 1.

4.2.4 Deciding How to Decide

The different collective solution ranking algorithms presented thus far are represented by the same particle swarm framework. Therefore, the distinction between any two algorithms is a function of particular parameters. Direct democracy can be transformed into representative democracy via a boolean parameter that allows the use of uses and trusts edges. By allowing trusts edges to be taken irrespective of the trustee being a voting individual, representative democracy can be transformed into dynamically distributed democracy. Furthermore, different initial particle distributions, as in proxy-vote (deGraf & Grey, 2006), and variation of the δ decay parameter are other dimensions of the parameter space.

Given a problem with known optimal solutions, it is possible to use parameter optimization algorithms (Engelbrecht, 2005) to learn which parameters, and therefore which algorithms, are most appropriate given particular problem types. The benefit of this is that by *a prior* knowing what parameters, for particular problem types, lead to the most optimal final solutions, the recursive paradox of voting on how to vote can be remedied. Furthermore, the system can be used as a collective intelligence system tuned towards the requirements of the problem space. Parameter optimization is only possible when a fitness function for the collective solution ranking can be determined. For value-based problems or extremely complex issues where there is no known optimal solution, deriving the most appropriate particle swarm parameters is not as strait forward.

So far, this section has presented a suite of collective solution ranking algorithms that can be used in a human collective decision making system. The collection presented thus far is not exhaustive as other ranking swarms have been developed: *despotic swarm* and *random democracy swarm*. In despotic swarm, only the individual with the highest ranking in the domain of the problem – as determined by the primary eigenvector of the trust-based social network (Wasserman & Faust, 1994) – has their vote counted. All other $\mathrm{votedOn}$ edges are ignored. In the random democracy swarm, a random subset of the full population is allowed to vote. The next section, Section 4.3, will demonstrate how to use the same particle swarm framework to determine the collective's realization of a problem's domain. This will affect how uses edge weights are determined.

4.3 Categorizing a Problem

What has been presented thus far is a model of ranking solutions to a problem relative to a collection of humans. It is also possible for particle swarms to generate collective rankings of other entities in the network. In this section a swarm grammar is introduced that is able rank the collection of all domains, D, relative to a particular problem p_j. The purpose of collective domain ranking algorithm is to determine the weights for the uses edges from an individual to their domain network. The collectively derived problem domain ranking provides the probability that a particle will enter an individual's domain network at a particular domain. When uses edges are weighted with respects to the problem domain, particles are able to propagate along

trusts edges that are particular to the domain of the problem.

In collective domain ranking the $\mathrm{categorizedAs}$ edge is analogous to the $\mathrm{votedOn}$ edge of the solution ranking algorithms. Also, the $\mathrm{similarTo}$ edges are synonymous with the trusts edges of the representative forms of the solution ranking swarms. The input set, I, receives the initial distribution of particles and the output set, O, is the set being ranked. The input set for collective domain ranking is the set of all domains, $I = D$. The output set is the set of all domains that have a $\mathrm{categorizedAs}$ projection to problem p_j, $O = \bigcup_{\forall i,l} h_i(d_l) \ : \exists w^{\mathrm{categorizedAs}}_{h_i(d_l),p_j}$. The general rule is that a particle can only take a $\mathrm{categorizedAs}$ to problem p_j. In the representative rank form, if no such edge exists then the particle must take a $\mathrm{similarTo}$ edge. Anytime a particle becomes stuck, it kills itself, $q_i = \varnothing$. The direct democracy domain ranking grammar is presented in Figure 14 and the representative democracy and DDD domain ranking grammar is presented in Figure 15. A visualization of a representative democracy swarm ranking of problem p_0 is presented in Figure 16. The direct democracy form would not allow the the particle to take $\mathrm{similarTo}$ edges.

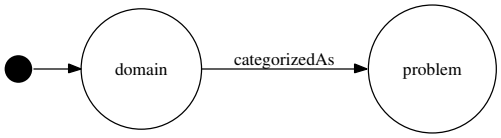

Figure 14: Finite state machine describing the direct democracy domain ranking grammar

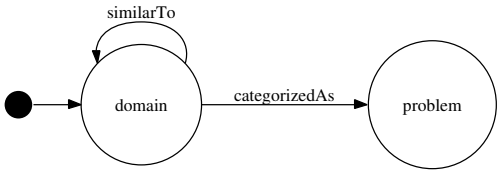

Figure 15: Finite state machine describing a representative democracy domain ranking grammar

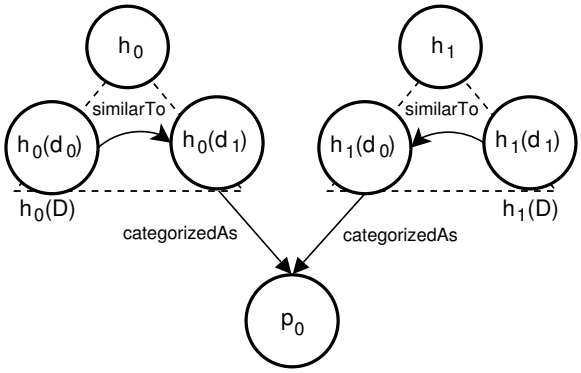

Figure 16: Representative democracy model of domain ranking

The normalization function to generate the uses weights set for particle entry into human h_i's domain network is defined in Eq. 10.

$$w^{\text{uses}}_{h_i,h_i(d_l)} = \begin{cases} \frac{\sum_{x=0}^{x<|H|} \vec{e}_{h_x(d_l)}(k)}{\sum_{x=0}^{x<|H|} \sum_{y=0}^{y<|h_x(D)|} \vec{e}_{h_x(d_y)}(k)} & \text{if } \exists w^{\text{categorizedAs}}_{h_i(d_l),p_j} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

5 Selecting a Final Outcome

As originally presented in Figure 1, the final stage of collective decision making is the selection function. The selection function takes the collective solution ranking, $\vec{e}_O$, and generates a single collective decision outcome. Depending on the nature of the problem, different selection functions are more appropriate than others. For continuous numeric solution sets, averaging can be used. In experiments where participants guess how many beans are in a large jar, averaging individual guesses turns out to be a near-optimal strategy (Treynor, 1987; Surowiecki, 2004; Watkins, 2005). One refinement to simple averaging is weighted averaging, where numeric solutions are weighted by their collective rank weighting. This has also proven successful in many numerical guessing experiments (Surowiecki, 2004). The weighted average for problem p_j is defined by Eq. 11 if the $\sum_{\forall m} \text{key}(p_j(s_m), x{-}\text{rank}) = 1.0$. That is, if the rankings are normalized to 1.0. It is important to note that the selection function requires a particular solution ranking (x-rank) for its calculation.

$$\text{weightedAverage}(p_j, x{-}\text{rank}) = \sum_{\forall m} [\text{key}(p_j(s_m), \text{title}) \times \text{key}(p_j(s_m), x{-}\text{rank})] \quad (11)$$

For a nominal, or categorical, solution set, the highest-ranked-wins rule can be used, defined by eq. 12 where $\max()$ returns the solution with the highest rank.

$$\text{highestRankedWins}(p_j, x{-}\text{rank}) = \max(p_j(S), x{-}\text{rank}) \quad (12)$$

6 The General Collective Decision Making Framework

The idea of individual and collective *solution ranking* and *domain ranking* can be generalized to simply *ranking*. The general paradigm states that the semantic network is an associative network of artifacts connected according to some homophilic property (McPherson, Smith-Lovin, & Cook, 2001). That is, artifacts are connected to each other according to some property, or aspect, of similarity. This property of similarity is made explicit by the type of nodes being connected and the λ labeled semantic edge connecting them. The degree of similarity is defined by the weight of that edge. The semantic network ontology defined by this paper can be extended to handle other similarity relationships. For instance, problems can be related to problems. Solutions to solutions. Solutions to other problems. Furthermore, other artifacts, such as manuscripts, can be included in the system to aid in other forms of collective decision making (Rodriguez, 2006). For example, a solution $p_j(s_m)$ may want to make reference to a scholarly manuscript. That manuscript is associated with authors, citations, a publishing journal, keywords, etc. Those authors may be individuals in the system, H. Those keywords can be associated with certain domains, D. What emerges, when the system is generalized to the extent that any entity n_i can be related to any other entity via some weighted semantic relation, is a multi-dimensional model of n_i. From these individual models, depending on the input set I, the output set O, and the grammar of the particle swarm algorithm $\Theta(R)(t)$, various subsets of the network can make a model of themselves with respects to other areas of the network. These subsets can be the same subset, $I = O$, or different subsets, $I \neq O$. The more artifacts and semantic relations, the greater the potential that one aspect of the network can be modeled

by another.

To collectively decide on the relative prestige of different scholarly journals (Bollen et al., 2006), the input set is the set of all journals, the output set is the set of all journals ($I = O$), and the particle grammar is with respects to $\mathrm{references}$ edges in a journal citation network. To determine the collective decision as to the most important web pages on the web (Brin & Page, 1998), the input set is the set of all web pages, the output set is the set of all web page nodes ($I = O$), and the particle grammar is with respects to href edges of the world wide web. In order to collectively determine the most appropriate set of peer-reviewers for a submitted manuscript (Rodriguez, Bollen, & Sompel, 2006b, 2006a), the input set is the manuscript needing review and the output set is the set all individuals in the scientific community ($I \neq O$). In the peer-review scientist ranking, a more complex particle grammar is required. First the particle takes a $\mathrm{citedBy}$ edge to a manuscript in the submitted manuscript's bibliographic references. Then the particles takes $\mathrm{writtenBy}$ edge to an author of the cited manuscript. Finally, the particles traverses a $\mathrm{coauthoredWith}$ edge to other authors in the greater scientific community. Once in the co-authorship network, the particles loop until all energy has decayed. The ranked scientists, O, are a model of the scientific domain of the submitted manuscript, I.

These models, or relative rankings, define the collective perspective according to some dimension of analysis. Individuals can utilize this perspective to aid in their local decision making processes. In general, what is the best solution to my problem? More specifically, who are the best scientists for me to co-author with? What will the closing price of a particular stock be today (Kaplan, 2001)? What journal should I publish my manuscript in? With the appropriate particle grammar, there is nothing that restricts the inclusion of all worldly artifacts into one gigantic collective decision making semantic network.

7 Conclusion

The proposed collective decision making framework presented in this paper is useful in decision making scenarios that can be represented by the problem-solution model. That is, the model which allows individuals to pose problems to the group, to articulate solutions to the posed problems, and to ultimately vote and collectively derive a final decision. This framework may prove useful for nation-states wishing to decentralize the means by which their decisions are executed, their governing laws created, and their justice served.

A more short term application is the system instantiation that can be used to elect governing officials. For the United States, this means, restricting the system to problems of candidate selection with the collective solution ranking algorithm being direct democracy and selecting the highest ranked candidate.

Before a social decision making application can even be considered for full-scale governance, it is important to better understand the nature of the various presented ranking algorithms. Can the *ad hoc* representative structures of DDD best reflect the values of the group and at the same time identify the expertise of the domain? In what context is one algorithm better than another? What is the definition of better? Does the common good define the fitness function driving the collective's decision making parameters? What is the common good? Is the self-governing body that can represent its 'nature' with clearest fidelity the most optimal system (Bird, 2000)? Are these questions best posed to the collective and then collectively decided upon? If so, then into the infinite regression we go – voting on how to vote.

To reach even further into the future's abyss, where a full semantic network of all worldly artifacts may exist, can this self-modeling system be used to drive the operations of the collec-

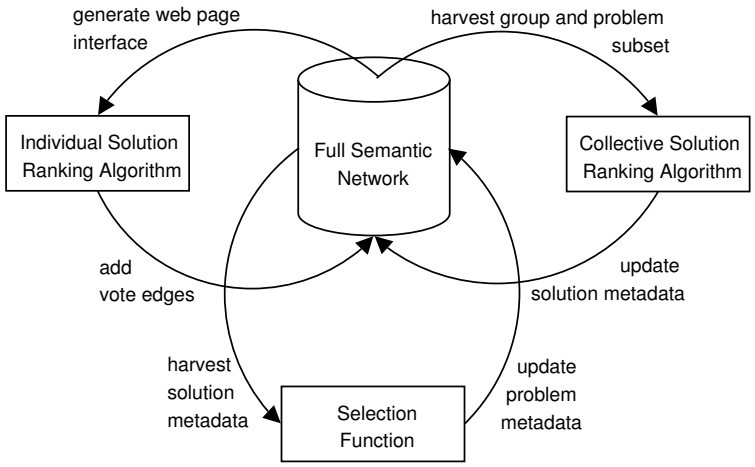

Figure 17: Collective Decision Making and the Semantic Network

tive? This ultimate theoretical system, which can represent one aspect of itself in another, may one day be the eyes as well as the body used to control itself; itself in its entirety.

8 Acknowledgments

Carlos Gershenson, Johan Bollen, Francis Heylighen, and Jennifer H. Watkins have all contributed via extended discussions on the issues concerning this topic. Funding was made possible thanks to a GAANN Fellowship from the U.S. Department of Education, the Fonds voor Wetenschappelijk Onderzoek - Vlaanderen, Los Alamos National Laboratory, the Threshold Foundation, and the Wallace Global Fund.

References

Arrow, K. (1963). *Social choice and individual values.* New York: Wiley.

Ball, P. (2005). Index aims for fair ranking of scientists. *Nature*, *436*(900).

Ball, P. (2006). Prestige is factored into journal ratings. *Nature*, *439*(16).

Bird, C. (2000). The possibility of self-government. *The American Political Science Review*, *94*(3), 563–577.

Bollen, J. (2001). *A cognitive model of adaptive web design and navigation.* Unpublished doctoral dissertation, Vrije Universiteit Brussel, Brussels, Belgium.

Bollen, J., & Heylighen, F. (1996). Algorithms for the self-organization of distributed, multi-user networks. In R. Trappl (Ed.), *Proceedings of the 13th European Meeting on Cybernetics and Systems Research* (pp. 911–917). Vienna, Austria: Austrian Society for Cybernetic Studies.

Bollen, J., & Heylighen, F. (1998). A system to restructure hypertext networks into valid user models. *The New Review of Hypermedia and Multimedia*, *4*, 189–213.

Bollen, J., Rodriguez, M. A., & Sompel, H. V. de. (2006). Journal status. *[submitted]*.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems.* New York, NY, USA: Oxford University Press, Inc.

Brin, S., Motwani, R., Page, L., & Winograd, T. (1998). What can you do with a web in your pocket? *Data Engineering Bulletin*, *21*(2), 37–47.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, *30*, 107–117.

Cazamine, S., & Sneyd, J. (1991). A model of collective nectar source selection by honey bees. *Journal of Theoretical Biology*, *149*, 547.

deGraf, B., & Grey, V. (2006). *Smartocracy: Social networks for collective decision-making.* implemented web service. (http://www.smartocracy.net)

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, *1*(1), 53–66.

Doringo, M., Maniezzo, V., & Colorni, A. (1996). The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics-Part B*, *26*(1), 1–13.

Engelbrecht, A. P. (2005). *Fundamentals of computational swarm intelligence.* West Sussex, UK: John Wiley & Sons.

Golbeck, J., Parsia, B., & Hendler, J. (2003). Trust networks on the semantic web. In *Proceedings of Cooperative Intelligent Agents 2003.* Helsinki, Finland.

Grasse, P. (1959). La reconstruction du nid et les coordinations inter-individuelles chez bellicositermes natalis et cubitermes sp. la theorie de la stigmergie. *Insectes Sociaux*, *6*, 41–83.

Held, D. (1984). *Political theory and the modern state.* Stanford University Press.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, *22*(1), 5–53.

Heylighen, F. (1999). Collective intelligence and its implementation on the web: Algorithms to develop a collective mental map. *Computational & Mathematical Organization Theory*, *5*(3), 253–280.

Johnson, N. (1998). *Collective problem-solving: functionality beyond the individual* (Tech. Rep.). Los Alamos National Laboratory Technical Report.

Kaplan, C. A. (2001). Collective intelligence: A new approach to stock price forecasting. In *Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference.* New York, USA.

Lee, R. (2004). *Scalability report on triple store applications* (Tech. Rep.). Massachusetts Institute of Technology.

Lijphart, A. (1984). *Democracies: Patterns of majoritarian and consensus government in twenty-one countries.* Yale University Press.

Mathes, A. (2004, December). *Folksonomies - cooperative classification and communication through shared metadata.* Computer Mediated Communication - LIS590CMC (graduate course).

McPherson, M., Smith-Lovin, L., & Cook, J. (2001). Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, *27*, 415–444.

O'Reilly, T. (2005, September). *What is web 2.0: Design patterns and business models for the next generation of software.* O'Reilly Network.

Pea, R. D. (1993). Practices of distributed intelligence and designs for education. In G. Salomon (Ed.), *Distributed cognitions* (pp. 47–87). New York, NY: Cambridge University Press.

Rodriguez, M. A. (2004). *Advances towards a societal-scale decision support system.* Unpublished master's thesis, University of California at Santa Cruz.

Rodriguez, M. A. (2006). A multi-graph to support the scholarly communication process. *[submitted]*.

Rodriguez, M. A., & Bollen, J. (2005). Simulating network influence algorithms using particle-swarms: Pagerank and pagerank-priors. *[submitted]*.

Rodriguez, M. A., Bollen, J., & Sompel, H. V. de. (2006a). An algorithm to determine peer-reviewers. *[submitted]*.

Rodriguez, M. A., Bollen, J., & Sompel, H. V. de. (2006b). The convergence of digital-libraries and the peer-review process. *Journal of Information Science*, *32*(2), 151-161.

Rodriguez, M. A., & Steinbock, D. (2004a). Group holographic modeling for societal-scale decision-making systems. In *NAACSOS '04: Proceedingss of the North American Association for Computational Social and Organizational Science Conference.* Pittsburgh, PA, USA.

Rodriguez, M. A., & Steinbock, D. (2004b). A social network for societal-scale decision-making systems. In *NAACSOS '04: Proceedingss of the North American Association for Computational Social and Organizational Science Conference.* Pittsburgh, PA, USA.

Schoonderwoerd, R., Holland, O., Bruten, J., & Rothkrantz, L. (1996). Ant-based load balancing in telecommunications networks. *Adaptive Behavior*, *5*(2), 169–207.

Simon, H. A. (1996). *The sciences of the artificial.* Cambridge, MA: MIT Press.

Sowa, J. F. (1991). *Principles of semantic networks: Explorations in the representation of knowledge.* San Mateo, CA: Morgan Kaufmann Publishers.

Surowiecki, J. (2004). *The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business, economies, societies and nations.* Doubleday.

Tepper, M. (2003). The rise of social software. *netWorker*, *7*(3), 18–23.

Tereshko, V., & Lee, T. (2002). How information-mapping patterns determine foraging behaviour of a honey bee colony. *Open Systems and Information Dynamics*, *9*, 181–193.

Treynor, J. L. (1987). Market efficiency and the bean jar experiment. *Financial Analysis Journal*, *43*(3), 50–53.

Turoff, M., Hiltz, S. R., Bieber, M., & Rana, A. (1999). Collaborative discourse structures in computer mediated group communications. In *32th Annual Hawaii International Conference on Systems Science (HICSS'02)* (Vol. 1). Hawaii, USA.

Turrof, M., Hiltz, S., Cho, H., Li, Z., & Wang, Y. (2002). Social decision support system (sdss). In *35th Annual Hawaii International Conference on Systems Science (HICSS'02)* (Vol. 1, p. 11). Hawaii, USA.

Wasserman, S., & Faust, K. (1994). *Social network analysis.* Cambridge: Cambridge University Press.

Watkins, J. H. (2005). *Individuals unite!: Generating synergistic understanding through online predictive markets* (Tech. Rep.). Kalamazoo, MI: Kalamazoo College: Physics Department.

White, S., & Smyth, P. (2003). Algorithms for estimating relative importance in networks. In *Kdd '03: Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 266–275). New York, NY, USA: ACM Press.

White, T., & Pagurek, B. (1998). Towards multi-swarm problem solving in networks. In Y. Demazeau (Ed.), *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS'98).* IEEE Press.

Ziegler, C.-N., & Golbeck, J. (2005). Investigating correlations of trust and interest similarity - do birds of a feather really flock together? *Decision Support Systems*, *[in press]*.