An Evolutionary Approach to Group Decision Making

Jackie Rees • Gary J. Koehler
Krannert Graduate School of Management, Purdue University,
West Lafayette, Indiana 47907-1310, USA
Department of Decision and Information Sciences, Warrington College of Business
University of Florida, P.O. Box 117169, Gainesville, Florida 32611, USA
jrees@mgmt.purdue.edu • koehler@ufl.edu

We propose modeling Group Support System (GSS) search tasks with Genetic Algorithms. Using explicit mathematical models for Genetic Algorithms (GAs), we show how to estimate the underlying GA parameters from an observed GSS solution path. Once these parameters are estimated, they may be related to GSS variables such as group composition and membership, leadership presence, the specific GSS tools available, incentive structure, and organizational culture. The estimated Genetic Algorithm parameters can be used with the mathematical models for GAs to compute or simulate expected GSS process outcomes.
(*Information Systems: Decision Support Systems; Artificial Intelligence; Probability: Markov Processes*)

1. Introduction

Work performed by groups play an important role in the success or failure of today's organizations. Technology is often used to support the tasks undertaken by these groups. Group Support Systems (GSS) encompassing Group Decision Support Systems (GDSS) and Electronic Meeting Systems (EMS) are of particular interest. Much attention has been focused on group task outcomes and the impact of technology. However, there is still much to be learned about how groups will perform given certain environments, tasks, and group membership.

Organizations utilize GSS in order to improve the eventual outcomes of group meetings. Given the interest in the eventual outcome of GSS use, the processes leading to the set of outcomes need to be carefully examined. Groups using these tools most often have a specific task or problem to address. In order to address such problems, groups must explore a space of possible solutions. The movement from solution to solution in this solution space is achieved through creative insights, negotiation, and group learning. As group members exchange information, new solutions are discovered, potentially better than previous solutions. The group adapts its search according to several factors including, but not limited to, the group composition and membership, leadership presence, the specific GSS tools available, incentive structure, organizational culture, and most importantly input and feedback from group members as the search progresses. Occasionally, a completely new line of thinking is undertaken or a random idea is inserted as a potential solution in the search space. We suggest that proposed solutions from groups using GSS can be viewed as strings of genes in an evolving, adaptive environment. These proposed solutions of the group evolve until a final solution or set of solutions is reached. This is not a new idea—Hirokawa and Johnson (1989) argued similarly. Even the "random idea out of left field" fits this view as merely a manifestation of punctuated equilibria—a hallmark of evolutionary methods. However, we build on this

0899-1499/02/1403/0278$5.00
1526-5528 electronic ISSN

observation by proposing an explicit model of evolutionary behavior—the process captured by the simple Genetic Algorithm.

There are several useful models of GSS available that provide insight into particular GSS processes and potential outcomes. Several of these models are descriptive (Poole and DeSanctis 1990; Nunamaker et al. 1991; Hiltz 1988; Rao and Jarvenpaa 1991), and others are more analytical (Gavish and Kalvenes 1996; Valacich and Dennis 1994).

The analytical model we propose for GSS departs from other GSS models in that it provides, through analogy, a computational view of GSS processes and expected outcomes. The analogy utilized is the simple Genetic Algorithm (GA). As stated above, groups using GSS for decision-making engage in a search process. It is assumed that this search is not a random search, especially with the addition of decision aids and other tools present in GSS software. This search is also influenced by the environment and interactions among group members. If a useful analogy for this search could be identified, much progress could be made in better understanding GSS processes. GAs are targeted as a possible analogy due to several features. GAs are adaptive, mirroring the evolutionary nature of GSS groups. GAs use populations of agents to search a solution space, as do GSS groups. GA search is partially guided by a fitness function that provides a measure of the potential viability of a string. In GSS processes, the potential viability of an idea influences its continued consideration. GAs are a form of heuristic search and as such are not guaranteed to find the optimal solution (if one exists) in a finite amount of time. As the same could be said about GSS groups, GAs appeared to have much potential to become the basis for an analogy for GSS search processes. The use of genetic algorithms as the basis of a model for GSS has several advantages and implications. For example, over the last several years, a sound mathematical theory has been developed that describes the exact expected behavior of GAs. This theory, in principle, could be used to determine many GSS characteristics, such as expected time till an optimal solution is generated. Also, there is a wide body of practitioner rules of thumb for GAs that could prove useful in designing better GSS processes.

The remainder of this paper is organized as follows. Section 2 presents relevant background on Group Support Systems and Genetic Algorithms. In Section 3 we summarize our arguments for representing GSS search as a genetic algorithm. In Section 4 we discuss various specifics of GA implementations. In Section 5 we show how to estimate the underlying GA parameters from GSS solution paths. This requires a presentation of the underlying mathematical models for GAs. In Section 6 we discuss specific issues pertaining to modeling a GSS as a GA. In Section 7 we present results found when estimating GA parameters from actual GSS data using the estimation procedure of Section 5. Section 8 shows how a GA model can be used to derive process values such as the expected time to see an optimal solution. Finally, in Section 9 we provide our conclusions and future directions.

2. Background

The following subsections provide relevant background on the various research areas that form the basis for this research. Section 2.1 provides a brief survey of theoretical and computational developments in GSS. This section highlights the need for additional investigation into computational and analytical models for GSS. Section 2.2 provides a brief background on genetic algorithms serving as the foundation of the model.

2.1. Group Support Systems

Group support systems are designed to support group decision-making through specialized software, hardware and decision support tools. DeSanctis and Gallupe (1987) defined GDSS, often considered the precursor term for GSS, as a combination of computer, communications and decision technologies working in tandem to provide support for problem identification, formulation, and solution generation during group meetings. Broadly stated, the fundamental goal of GSS is "... to support the exchange of ideas, opinions, and preferences within the group" (Gallupe and DeSanctis 1988, p. 278). As GSS research increased, the described goals of GSS became more narrowly proscribed. According to Watson et al. (1988) the primary goal of GSS is to reduce "process loss" attributed to disorganization within the group, social issues such as member dominance, inhibition, peer pressure, and other recognized difficulties of group interaction and to improve overall decision quality (Watson et al. 1988).

One of the first attempts to bring the empirical research into a cohesive whole in the early days of GSS research is the conceptual work of DeSanctis and Gallupe (1987). They proposed a multidimensional taxonomy of research variables in GSS research. This taxonomy was driven by three factors: group size, communication channel (face-to-face vs. computer-mediated), and task type.

At about the same time, Hiltz (1988) proposed the systems-contingency approach. Like many others at the time, this framework focused on identifying how technology was accepted and implemented and the resultant effects on productivity and performance. According to systems contingency theory, "Productivity impacts are hypothesized to be contingent upon the characteristics of the higher-level systems within which the technology is used" (Hiltz 1988, p. 1440). The implication of this theoretical approach is that there is no correct or universal method on group systems design due to the fact that the needs of each organization are complex and vary among the subgroups within each organization (Hiltz 1988). Rao and Jarvenpaa (1991) also advocated a contingency approach while aggregating other social theories thought to affect group decision-making via group support systems.

While systems contingency theory attempted to address the impact of GSS upon the users of the technology and the decision quality following from such use, another theory evolved to address the issue of the process of GSS usage within the organization. Poole and DeSanctis (1990) proposed a new theory intended to resolve some of the conflicting results gathered from empirical research. Adaptive Structuration Theory (AST) considers GSS use as an input-process-output model. The basis of the model comes from the structure concept. Structures are defined as "... the rules and resources actors use to generate and support this system," (Poole and DeSanctis 1990, p. 179) where system refers to the GSS under consideration. AST has several favorable characteristics including flexibility and generality and has been adopted as the framework for several empirical studies including but not limited to Gopal et al. (1993) and Chidambaram et al. (1991).

Nunamaker et al. (1991) presented a high-level research model that describes the major influences upon GSS processes and outcomes. The factors influencing GSS outcomes are group characteristics, task characteristics, context characteristics, and the specific technologies in use. GSS are believed to improve the quality of group decisions by minimizing "process losses" and maximizing "process gains." Process gains occur when certain aspects of the meeting improve the eventual outcome or result and process losses hinder or reduce the eventual outcome. Thus, the overall meeting outcome is dependent upon the process gains versus the process losses (Nunamaker et al. 1991).

Valacich and Dennis (1994) presented a simple regression model of electronic brainstorming using GSS. Their model presents GSS brainstorming as the ideas generated by a group of individuals, each working alone, accounting for process losses and process gains. In other words, "... group performance is a function of individual performance minus process losses plus process gains" (Valacich and Dennis 1994, p. 64). This research is based in part on the earlier work of Steiner in the group literature (Steiner 1966, 1972).

Perhaps the most closely related GSS research to ours is the economic analysis of GSS (Gavish and Kalvenes 1996). One of the important features of the economic model is that it considers GSS use by groups to be in the format of a search problem with a very large search space. This statement lays the groundwork for our central argument that groups using GSS act like a GA, as GAs are search tools that share many GSS characteristics. According to Gavish and Kalvenes' model, every feasible solution has a payoff, which must be balanced with the cost of performing the search. Another aspect of interest in their model is the discussion of a "trigger phenomenon"

(Gavish and Kalvenes 1996). This is the case when an original idea "triggers" a new line of reasoning or discussion. The model also addressed the probability of finding a solution, the expected net benefit of finding a particular solution, stopping criteria, and the marginal value of group size (Gavish and Kalvenes 1996). Many of the features inherent in the Gavish and Kalvenes (1996) model can be incorporated into the GA evolutionary model proposed later in this research and the features mentioned above are certainly complementary to the proposed model. One shared feature of both models is the assumption of GSS existence outside of the traditional decision room. However, the economic model, while complementary to the proposed model, does not provide the high-level modeling capabilities of the genetic algorithm-based model. The economic model does not take into account systemic factors such as the rate of solution exchange between group members, the diversity of solutions within groups or the impact of reward upon decision quality among other factors. Therefore, while the economic model and the proposed GA evolutionary model are highly compatible, the proposed GA evolutionary model potentially goes further in providing a system-wide level of analysis.

2.2. Genetic Algorithms

In order to make clear the analogy between genetic algorithms and GSS, we will provide a brief description of the basic or "simple" GA. GAs are a search tool loosely based on the principles of natural selection and evolution. The genetic algorithm operates on strings of "genes." These strings are often composed of a series of binary digits, representing a proposed solution. (The composition of the strings is not limited to the binary alphabet.) In a GSS context, these strings can represent possible solutions provided by group members. For example, if the group is attempting to address a production-planning type problem, the string might be made up of the possible set of customer orders. A "1" indicates that the order corresponding to the position on the string should be filled and a "0" indicates otherwise. A starting collection of strings forms an initial population (generation zero) and the operators of genetic algorithms are applied iteratively until some stopping criterion is met.

The mechanics of the genetic algorithm, while often varied and complex for given applications, are based on three fundamental operations. These operators are the selection operator, the crossover operator and the mutation operator. The selection operator implements the "survival of the fittest" principle as the better or more fit strings have a statistically better chance of survival than do less fit strings. In a GSS context, the group members determine which strings or proposed solutions are fit. This is achieved by evaluating each string according to the fitness function employed. The fitness function is related to an objective function or utility function. In GSS, the analogy might be "maximize organizational profit" or "minimize departmental costs." Selected strings have a higher probability of surviving or passing genetic material to future populations. There are several different types of selection mechanisms used in practice and these are discussed below. Once the strings have been identified or "selected" to appear in the next generation, a genetic mixing takes place. This is akin to reproduction and mutation. More specifically, a crossover operator is applied followed by possible mutation.

Strings are paired and crossed with probability χ (a crossover rate). The crossover operator acts as a "focusing" effect upon the search. As the selection operator has seeded the next generation with fitter strings (probabilistically speaking), exchanging of parts of fitter strings should, it is hoped, result in more quickly reaching a target or optimal solution than would random search. In a GSS context, crossover would be analogous to group members incorporating parts of other group member solutions into their own solutions, in essence "combining" pieces of various solutions into one. Finally, the strings undergo a mutation operation, where bits of information along the strings are randomly altered at a predefined mutation rate, μ. The mutation operation adds diversity to the search by adding random information back to the strings. Computationally, the role of the mutation operator is to shift the search away from local optima (Goldberg 1989). Within the GSS analogy, mutation would operationalize the "trigger phenomenon" described by Gavish and Kalvenes (1996). Figure 1 summarizes a typical GA implementation.

```
Algorithm: (Genetic Algorithm)
    Given:
        String length $\ell$, fitness function $f()$, mutation rate $\mu \in [0,0.5]$,
        crossover rate $\chi \in [0,1]$ and population size $n \geq 1$.

    Initialization: Generate an initial population, population 0. This is usually done by
        randomly drawing n strings from $\Omega = \{0,1,\ldots,2^{\ell}-1\}$ with replacement.

    Step 1: Form a new population as follows. Repeat the following steps until the new
        population has n members.
        (A) Randomly choose two (or more) members from the old population according
            to a selection process. These are called parent strings.
        (B) Form one or more children through a mixing process consisting of crossover
            and mutation operations.

    Step 2: If stopping conditions are not met, return to Step 1.
```

Figure 1 The Simple Genetic Algorithm

3. An Evolutionary Approach to GSS

We propose that the search of the solution space in group problem solving, when supported by GSS, can be modeled by a genetic algorithm, utilizing selection, crossover, and mutation. A population of strings represents the solutions generated by a group. Each string in the population at time step t represents the solution proposed by group members at time step t.

The genetic algorithm was chosen as the basis of this model for several reasons. First, GAs are adaptive, meaning there is change over time in response to the environment that includes the current solution set, the fitness function, various operators, and other constraints. This adaptive capability captures one of the basic principles of group decision making put forth by Hirokawa and Johnson (1989, p. 503) that "group decision making is an evolutionary process."

Second, the operations of the simple genetic algorithm resemble basic processes of groups using GSS. Proposed solutions are discussed and evaluated by the group, analogous to the selection process of the simple genetic algorithm. The better ("fitter") proposed solutions are combined with other fit proposed solutions to yield improved solutions, similar to the crossover operation. Random changes occur to these proposed solutions along the way, much like the mutation operation. Other, more complex, GA operators exist that could be incorporated into the model, including dominance and niching behaviors; however, these behaviors fall beyond the scope of this research.

Third, groups often propose good solutions that may be very different from current solutions. Gavish and Kalvenes (1996) termed this a *trigger phenomenon*. We see this phenomenon as a naturally occurring one observed in evolutionary systems-one termed *punctuated equilibria*. Here a seemingly stable system suddenly evolves new genetic material.

Another reason for using a GA framework is that a body of formal, mathematical theory has been developed to describe the expected behavior of the simple genetic algorithm. If GSS can be modeled as a GA, this theory could provide numerous insights into the group decision-making process. Variables and different environmental pressures thought to influence the process could be related to GA parameters and then factors such as the expected behavior of the system could be determined or optimized. There also exists a large body of heuristic knowledge available for the genetic algorithm, which could be used to determine group size and other group characteristics relating to GA parameters, such as crossover and mutation rates. Finally, by establishing an analogy for the behavior of such groups, we can preserve the knowledge shared by the group and the interaction knowledge among group members.

Several variables are thought to have an influence on group processes and outcomes. Among these variables are task, communication mode, group characteristics, incentive structures, and environmental variables. There are several different task types mentioned in the GSS literature. Negotiation, idea ranking, and idea generation are several examples. Examples of communication mode include Face-to-Face (FTF) communication, where all participants are in the same physical location and have visible contact with the other group members and Computer-Mediated-Communication (CMC) where the participants are geographically dispersed. Group characteristics include group size, the presence (or absence) of a leader and group makeup or composition. Incentive structures include localized incentive schemes, organizational (global) incentive schemes, leader incentive schemes (Barkhi, 1995), hybrid incentive schemes and the lack of external incentive, which can be classified further as either identification (the intangible benefit of group membership) or internalization (behavior based on the belief in group norms) (Shamir, 1990; Guzzo and Dickson, 1996). The environment is the surrounding within which the group decision-making process takes place. Elements of the environment include (but are not limited to) the actual GSS tool used in the decision-making process, the information quality available to group members, and the surrounding corporate culture.

Several of the above variables could be further partitioned, for example, group characteristics could be separated into group size, diversity, and cohesiveness to name a few (Guzzo and Dickson, 1996). However, for the sake of simplicity, this model will aggregate as many of these related factors as possible. There is also the admission that many of these components are not easily measured, or even described, for example, corporate culture and leadership.

The group decision-making process is an adaptive, iterative process that eventually results in a final group solution, which can be measured by several metrics. Overall, these metrics, which might be used simultaneously, include solution quality (which may or may not be objective), time taken to arrive at the final solution and user satisfaction with both the final solution and the process used to arrive at the solution.

It is certainly acknowledged that GSS takes on many shapes, sizes, and variations in both field and laboratory settings. For purposes of this model, we will assume that GSS use entails a simplistic scenario. The group using the GSS is provided a one-time problem to address. The group proposes possible solutions to the problem through the GSS until a solution or set of solutions adopted.

4. The Genetic Algorithm Evolutionary Model

The simple genetic algorithm moves from population to population using three operators: selection, crossover, and mutation. There are many implementations of these. Below we summarize the most common implementations.

Selection

Although there are many others, one of three versions of the selection operation is commonly employed (Goldberg, 1989). They are roulette-wheel selection, tournament selection, and rank selection. Roulette-wheel selection assigns a probability of selection proportional to a string's fitness (multiplied by the number of instances of the string in the population) relative to the sum of the fitness values of all the strings in the current population. Tournament selection operates by first using another selection process to pick k strings (typically $k = 2$) from the current population (with replacement). Then the "fittest" individual of the k strings is inserted into the new population. This continues until the next generation is complete. Rank selection, considered a non-parametric procedure, sorts the strings in the population according to fitness value. Copies of individual strings are inserted into the next generation according to a function of the original ranking. Essentially, the higher ranked the proposed solution, the more likely it will influence subsequent generations.

Crossover

Two strings are mated with probability χ (the crossover rate). Typically, mating is accomplished using single-point, multi-point, or uniform crossover. With single-point crossover, a point along the string is selected with uniform probability. The bits following this point are swapped between the two strings. Multi-point crossover is similar except that multiple points are selected and genetic material swapped from selected intervals.

Uniform crossover works by moving bit-wise down the pair of strings, exchanging bits with probability χ. The appeal of uniform crossover is the ability to exchange a variable number of information segments between the string pairs, which is a more dynamic approach than either single-point or multi-point crossover.

Mutation

Uniform mutation works by moving bit-wise down the string and altering the particular bit according to a "flip" of a weighted coin. The "weighting" is called the *mutation rate*, μ.

Fitness Functions

Some group tasks or specific meetings provide for an explicit operationalization of a particular incentive scheme or reward that can be translated into a fitness function for the genetic algorithm. Usually any group that possesses an explicit economic incentive structure system for its members can relatively easily encode the incentive into a fitness function. However, many groups do not operate under an explicit incentive system. Other functions could possibly be used, such as the objectives of the group. Various voting procedures could be used as fitness functions, as they provide a means to measure the fitness of proposed solutions. Therefore, several methods exist for evaluating proposed solutions exchanged among the group and these methods can be mapped to the fitness functions of GAs.

Model Implementation

We suggest a two-phase process for using the GA model to guide GSS usage. The first phase consists of gathering data from various actual GSS experiments and using these data to determine the best-fit GA parameters. Fine-tuning of model details would occur during this first phase. Many GSS experiments have already been performed that could provide the data necessary for parameter estimation. The second phase would consist of using the best-fit parameters to determine various controllable parameters such as group size, time until acceptable solution, etc. These could be determined by using the mathematical models of GA behavior, by simulation using GA software, or by using rules of thumb developed by the GA community.

In the next section we summarize the mathematical models for the simple GA and show how these can be used to determine a maximum-likelihood estimate for the observed sequence of GSS proposed solutions (we call this a *trajectory of solutions*).

5. Markov Chain Model for GAs and Parameter Estimation

Vose (1990) and Vose and Liepins (1991) provided the first exact mathematical model of a simple GA (for the proportional-selection, one-point crossover, and uniform-mutation case). This initial model has been extended to include many variants (see Vose 1999). Let Ω be a collection of binary strings of length ℓ and let $r = |\Omega| = 2^\ell$ be the number of possible strings. These strings can be equivalently considered as the integer equivalents $0, 1, \ldots, r-1$. Let $M_{g,k}$ be the probability that the string of all zeros is the child of the mating process between parent strings g and k (where g and k are the integer values corresponding to the strings). Under one-point crossover and uniform mutation, Vose (1990) and Vose and Liepins (1991) showed that

$$M_{g,k} = \frac{(1-\mu)^\ell}{2} \times \left\{ \eta^{|g|}\left(1-\chi+\frac{\chi}{\ell-1}\sum_{h=1}^{\ell-1}\eta^{-\Delta_{g,k,h}}\right) + \eta^{|k|}\left(1-\chi+\frac{\chi}{\ell-1}\sum_{h=1}^{\ell-1}\eta^{+\Delta_{g,k,h}}\right)\right\} \tag{1}$$

where

$$\eta = \frac{\mu}{1-\mu}, \tag{2}$$

division by zero at $\mu = 1$ is removed by continuity and where

$$\Delta_{g,k,h} = |(2^h-1)\otimes g| - |(2^h-1)\otimes k| \tag{3}$$

and $\otimes$ is the bitwise "AND" operator. Let P be a population of elements from Ω where $n = |P|$ is the population size and N is the number of possible populations. N is computed by the formula,

$$N = \begin{pmatrix} n+r-1 \\ r-1 \end{pmatrix} \tag{4}$$

A population is a multiset, meaning that it may contain multiple copies of the same string. Consider the Markov chain where the possible populations of size n are the states. Express a state by the vector of length r, ϕ_i, having as its kth component the number of copies of string k in the population. Let e be a vector of 1's and e' its transpose. Each ϕ_i is defined by

$$e'\phi_i = n \tag{5}$$

$$(\phi_i)_j \in \{0, 1, \ldots, n\} \qquad j = 0, 1, \ldots, r-1. \tag{6}$$

The transition probabilities from state (population) i to j are computed by

$$P_{i,j} = n! \prod_{g=0}^{r-1} \frac{q_{i,g}^{(\phi_j)_g}}{(\phi_j)_g!} \tag{7}$$

where

$$q_{i,g} = \mathcal{M}(\mathcal{F}(\phi_i))_g. \tag{8}$$

Vose (1999) uses $\mathcal{F}$ to capture the selection process and $\mathcal{M}$ the mixing operators (mutation and crossover). In particular

$$\mathcal{M}(x)_i = (\sigma_i x) M \sigma_i x \tag{9}$$

where the permutation of x, $\sigma_k x$, is defined by

$$\sigma_k x = \begin{pmatrix} x_{k\oplus 0} \\ \vdots \\ x_{k\oplus(r-1)} \end{pmatrix}. \tag{10}$$

A general form of the mixing matrix, M, was given by Vose and Wright (1995) as

$$M_{x,y} = \sum_{j,k} \mu_j \frac{\chi_k + \chi_{\bar{k}}}{2} \delta(x \otimes k \oplus \bar{k} \otimes y = j). \tag{11}$$

Here μ_j and χ_k are called *mutation and crossover masks* and $\delta(x)$ is 1 when x is true and 0 otherwise. The various mutation and crossover schemes can be captured using appropriate choices for these masks. For example, letting

$$\chi_i = \begin{cases} \chi c_i & \text{if } i > 0 \\ 1 - \chi + \chi c_0 & \text{if } i = 0 \end{cases} \tag{12}$$

with $c_i = 2^{-\ell}$ gives uniform crossover. For one-point crossover

$$c_i = \begin{cases} \dfrac{1}{(\ell - 1)} & \text{if } \exists\, k \in (0, \ell) \text{ and } i = 2^k - 1 \\ 0 & \text{otherwise} \end{cases}. \tag{13}$$

For uniform mutation we have

$$\mu_i = (\mu)^{e'i}(1-\mu)^{\ell - e'i} \tag{14}$$

where $e'i$ is the number of non-zero bits of i.

The selection process is captured through $\mathcal{F}$. For roulette-wheel selection

$$\mathcal{F}(\phi) = \frac{F\phi}{e'F\phi} \tag{15}$$

where F is a diagonal matrix and $F_{i,i} = f(i)$ is the fitness of string i. Rank selection is given by

$$\mathcal{F}(\phi)_i = \int_{\sum(\phi)_j \delta(f(j) < f(i))}^{\sum(\phi)_j \delta(f(j) \le f(i))} \rho(y)\, dy \tag{16}$$

where ρ is any continuous increasing probability density over $[0, 1]$ (see Vose 1999). Finally, tournament selection is given as follows. Here k (usually $k = 2$) strings are uniformly drawn from a population. These k strings compete for selection based on their ranking under a ranking selection scheme $\overline{\mathcal{F}}$. Vose (1999) showed that

$$\mathcal{F}(\phi)_i = k! \sum_{v \in X_r^k} \overline{\mathcal{F}}\left(\frac{v}{k}\right)_i \prod_{j<r} \frac{(\phi)_j^{v_j}}{v_j!} \tag{17}$$

where

$$X_r^k = \{x \ge 0, x \text{ integral}, e'x = k\}. \tag{18}$$

The previous models were developed specifically for binary strings. Bhattacharyya and Koehler (1994) extended the Vose model for strings with digits selected from 2^v cardinality alphabets. Later, Koehler et al. (1998) generalized the Vose-Liepins model for strings composed of digits having alphabets of arbitrary cardinality z, where z is an integer greater than 1. This is accomplished by replacing the Boolean EXCLUSIVE-OR and AND operators with multiplication and addition over rings of integers. Their analysis also requires Fourier transforms instead of Walsh transforms, typically used in GA theory.

These models can be easily extended to varying-sized populations as follows. Let $P_{i,j}(I, J)$ be the probability of going from state i (where populations are of size I) at time t (the current generation) to state j (where populations are of size J) at time $t+1$. Then we have

$$P_{i,j}(I, J) = I! \prod_{g=0}^{r-1} \frac{q_{i,g}^{(\phi_j)_g}}{(\phi_j)_g!} \tag{19}$$

where

$$e'\phi_i = I \tag{20}$$

and

$$e'\phi_j = J. \tag{21}$$

Parameter Estimation

Given an observed trajectory of a GA process, we wish to estimate the underlying parameters used by the GA. That is, we wish to estimate rates χ and μ. The likelihood of an observed trajectory is proportional to the product of the transition probabilities along the path. Hence, the likelihood of a given chain going from j_1 to j_1 to $j_2 \ldots$ is

$$P_{j_1,j_2}(J_1, J_2)P_{j_2,j_3}(J_2, J_3)\cdots P_{j_{T-1},j_T}(J_{T-1}, J_T), \tag{22}$$

where $J_1, J_2, \ldots, J_T$ are the population sizes at times $t = 1, 2, \ldots, T$. We use a simple maximum-likelihood procedure in deriving estimates. Maximum-likelihood estimators have several desirable properties, including invariance, sufficiency (if the parameter itself is sufficient), and efficiency (Mood, 1950). To find the maximum-likelihood estimate for each parameter of interest, namely, crossover χ, and mutation μ, we maximize the likelihood function given above. Therefore, we must solve

$$\max_{\chi,\mu} \prod_{i=1}^{T-1} P_{j_i,j_{i+1}}(J_i, J_{i+1}) \tag{23}$$

where T is the number of observed populations and J_i is population i's size.

In order to find the maximum-likelihood estimate for our Markov chain, we could set the partial derivatives with respect to the mutation and crossover operators to zero and solve for μ and χ. The partial derivatives, with respect to χ, are relatively easy to derive but those for μ are highly non-linear. Furthermore, it is unlikely that first-order conditions would be sufficient. Besides, it appears that the equations would be nearly impossible to solve. Therefore, an approximately exhaustive search over a grid should be performed to determine the (near) optimal values of the crossover and mutation rates. An iteration through the values of μ from 0.0 to 0.5 (where 0.5 represents a random search in the binary case) and the values of χ from 0.0 to 1.0, inclusive, is appropriate.

6. Modeling a GSS as a GA: Model Details

Population Sizing

A population consists of a number of proposed solutions. In GSS settings, proposed solutions are offered by group members. As these solutions are not typically offered in a round-robin fashion, but rather as soon as the proposed solutions are generated by group members, a dynamic population size scheme was designed for use in the model. Several possibilities exist for population sizing schemes. When learning GA parameters from GSS experimental data, several schemes have been proposed and several have undergone preliminary testing. We propose two particular schemes for modeling the populations: Peer-Influenced and Data-Driven.

The Peer-Influenced scheme is quite simple. Each group member is assigned an identifier by the system. When anonymity is desired, this identifier could be used at the system level and not revealed to GSS users. As each proposal is submitted, that proposal is encoded and placed into the current population. As the next proposal enters the system, the identifier is checked to see who submitted the proposal. If the identifier is the same as the identifier of the previous solution proposal, the proposal is placed into the population and the process continues. This indicates that the same group member submitted the proposal, perhaps to reemphasize the point or clarify the previous statement. When an identifier enters the system that does match the previous identifier, the proposed solution is encoded and placed into the population and the GA operators then act upon the population. The rationale for this is that group members most likely require the proposed solutions from multiple group members before the proposed solutions are evaluated and combined. Consecutive proposed solutions from the same group member are more likely seen as repeating or clarifying a previous statement and are not as "powerful" in driving the group towards its final, accepted solution. Once the operations are finished, the next proposed solution seeds the next population or generation and the process continues. This scheme is presented in algorithm form in Figure 2 below.

The advantages of this scheme are that the scheme (in algorithm form below) is relatively simple to implement and conceptualize. Preliminary tests (Rees and Koehler, 1999) show that it captures the maximum-likelihood estimates of the transition probabilities better than previously proposed schemes. As each proposal is treated equally, this scheme seems suitable for GSS usage where the participants are anonymous, considered an important feature of GSS. However, some GSS implementations allow the identification of group members by either rank or title and situations potentially exist where this identification might be desirous or necessary. The drawback to this population scheme is when group member identifications are known during GSS use, certain participant proposals carry different "weight" than others

```
WHILE NOT end-of-session DO:
    READ (input string) INTO Proposed_Solution;
    READ (System_Identifier) INTO New_ID;
    IF New_ID == Prev_ID
    THEN append Proposed_Solution to Current_Population
    ELSE start New_Population;
    Prev_ID = New_ID;
END WHILE.
```

Figure 2 Description of Peer-Influenced Population Sizing Scheme

and the GA model should capture this phenomenon. Therefore, another population-sizing scheme, namely, Data-Driven, has been proposed (but not yet tested) that would hopefully address this concern.

The Data-Driven population-sizing scheme would use data from actual GSS experiments to estimate the population sizes from the data itself. By computing the parameter-estimates and maximum likelihood estimates for the transition probabilities for different population sizing schemes, the "best-fit" population size scheme can be determined. Initially, this computation would most likely be by a "brute-force" approach. However as more is learned about the model, heuristics and intelligence could be applied to the search for the most representative population sizes. The advantage of this scheme is that it best reflects what occurs in actual GSS use. Different population configurations are likely to occur from different implementations of GSS, which would be interesting from a research perspective. The primary disadvantage is that it appears computationally expensive in terms of time to run the data through the different configurations of population sizes.

Both schemes result in variable population sizes, meaning that the number of strings in each population can vary from generation to generation. Equation 19 showed how the Markov-chain model is extended to accommodate variable-sized populations. The GA model takes each proposed solution as it is submitted to the group and places it into the current population or begins a new population, depending on the scheme employed. The process ends when the group ends its GSS session. Therefore, the total number of actual solutions proposed by the group is the same as the total number of strings processed by the GA model.

String Encoding

Some problems have solutions that are more easily represented by strings than others. Binary strings are the most common implementation form but higher-cardinality strings are natural in some situations. The analytical model has forms for all cardinalities. This allows for much flexibility in encoding the problem. However, Rees and Koehler (1998) showed that the higher the cardinality, the more difficult the computation of quantities, such as expected time to solution, could be. Due to the inherent complexity and dynamic characteristics of this type of system, most analyses will be limited anyway. String lengths can either be fixed or variable-length. However, computational analysis will be much more unwieldy in the case of variable-length strings. Often it is not complicated to create fixed-length strings from variable-length strings.

7. Illustration

Barkhi (1995) performed experiments examining the effects of communication channel, leadership and incentive structure on group decision-making within a GSS context. The groups were provided a resource-allocation task to solve that required negotiation and conflict resolution. The group task was a production-planning problem. Each group was given a predetermined set of customer orders and were to determine the most optimal set of customer orders to fill based on revenue and cost data for each order and capacity constraints. Group members submitted proposed solutions to the task (for example, fill order numbers 1, 5, and 8) using templates provided with the GSS. The data from the experiments, including the solutions proposed (and the order in which they were proposed) and the group decisions (the final solution from each group) were used to validate the model. We needed to test whether or not groups using GSS behaved like a GA with random search. In order to test this, we compared maximum-likelihood estimates of the path probabilities (the probability of the search moving from a particular state to another) estimated from the experimental data with the path probabilities seen in a GA using random search.

We modeled the GA explicitly using uniform crossover, uniform mutation, and rank selection. Tournament selection is also a likely process. Better proposed solutions are successively compared two-by-two with the better solution of the pair going on for future consideration. However, rank selection is useful in cases of small population sizes, which is the most likely scenario in applying the model to GSS use. In any case, GA practitioners have observed little performance difference between rank and tournament selection methods.

Previous experimentation has indicated that single-point crossover is probably not an effective operator for the GSS model (Rees and Koehler, 1999). This is most likely due to a lack of diversity in the search and also the manner in which single-point crossover is operationalized—which is not intuitively suggestive of how people form solutions. Solutions are more likely formed by taken one or more pieces of previous proposed solutions rather than exchanging entire segments between solutions. Previous research (Rees and Koehler, 1999) also supports the use of uniform mutation as an implementation of the mutation operator. The peer-influenced population-sizing heuristic was used due to the relative simplicity of the scheme. Implementation of the data-driven population-sizing heuristic is beyond the scope of this research.

The values of χ and μ consistent with random search are $\chi = 0.0$ and $\mu = 0.5$. Table 1 shows the log-likelihood values for path probabilities found from a random GA path and those based on estimates from the Barkhi data using the estimation procedure detailed in Section 5. We pose the null hypothesis that the estimated GA is indistinguishable from a purely random GA process. The significance of these differences was measured using the Wilcoxon matched-pairs signed-ranks test (Siegel, 1956). The null hypothesis was rejected for the 0.05 level ($T = 1081$, $w_{.95} = 691.1$). The model was fitted using data from 48 groups from Barkhi's study. The average estimated uniform crossover rate was computed at 0.15994, and the average uniform mutation rate was computed at 0.0231. From this example, we concluded groups using GSS do not behave like random GA processes but follow a significantly different GA process (as estimated). This gives support to using GAs to model GSS search processes.

Another finding of interest is the set of estimated crossover and mutation rates. The average estimated mutation rate certainly falls within normal GA mutation rate parameters (between 0.001 and 0.1). However, the average estimated crossover rate is far lower than what is considered normal by practitioners (usually this value is around 0.6). After applying a GA simulation to the groups' problem, the solutions were

Table 1 Log Differences in Path Probabilities Between Actual (Fitted) and Random GA Paths for All Groups Under Peer-Influenced Populations, Rank Selection, Uniform Mutation, and Crossover

Actual Path Probability (Log)	Random Path Probability (Log)	Difference (Log)
−102.923	−332.711	229.788
−33.337	−235.670	202.333
−38.013	−124.766	86.753
−83.074	−318.848	235.774
−7.926	−152.492	144.566
−41.889	−235.670	193.781
−97.825	−402.025	304.199
−85.865	−180.218	94.353
−61.312	−318.848	257.534
−35.382	−97.041	61.659
−142.889	−318.848	175.959
−8.830	−124.766	115.936
−46.830	−304.985	258.155
−38.723	−194.081	155.358
−28.929	−69.315	40.386
−9.040	−97.041	88.000
−5.205	−83.178	77.972
−33.655	−138.629	104.974
−61.213	−180.218	119.005
−32.494	−207.944	175.450
−24.120	−138.629	114.509
−40.949	−235.670	194.721
−60.054	−207.944	147.890
−31.839	−180.218	148.379
−25.442	−124.766	99.324
−45.942	−124.766	78.824
−31.479	−110.904	79.425
0.000	0.000	0.000
−0.308	−83.178	82.869
−43.461	−194.081	150.620
−6.827	−166.355	159.528
−57.639	−180.218	122.579
−34.693	−69.315	34.621
0.000	0.000	0.000
−55.953	−221.807	165.854
−1.909	−41.589	39.679
−91.283	−207.944	116.661
−37.326	−152.492	115.166
−28.638	−207.944	179.306
−7.115	−180.218	173.103
−169.015	−318.848	149.833
−13.013	−235.670	222.657
−46.868	−180.218	133.350
−72.118	−249.533	177.415
−55.611	−166.355	110.744
−6.709	−138.629	131.919
−17.548	−124.766	107.218
−31.893	−97.041	65.147

maximized at lower crossover rates than at more "typical" crossover rates. The apparent explanation is that the production-planning problem was highly constrained, resulting in many infeasible solutions. Therefore, both the GA simulation and the GSS users had to rely mostly on enumeration, rather than high levels of information exchange to solve the problem. This issue is further explored in Rees and Barkhi (2001).

8. Using the GA Mathematical Model

We have presented a new model for GSS activities using a Genetic Algorithm as the foundation. This model has a built-in mathematical framework that could prove useful in analyzing group processes under GSS. Through this framework, we have the ability to examine the exact expected behavior of groups using GSS.

The vector of expected times (where the value in row k corresponds to starting in state k) until string i is observed is found by computing

$$(I - D_i P)^{-1} e - e \tag{24}$$

where D_i is a diagonal matrix having zero rows corresponding to states having string i in the population and a diagonal of one in rows corresponding to states not having string i. Prior probabilities are given by

$$\pi_j = \frac{n!}{r^n \prod_{g=0}^{r-1}[(\phi_j)_g!]} \tag{25}$$

when the initial population is drawn randomly from Ω with replacement and

$$\pi_j = \frac{\delta(\phi_j \le e)}{\binom{r}{n}} \tag{26}$$

without replacement. A straightforward application of these equations, however, is impractical since the size of P is very large. Koehler (1999) explored using matrix iterative methods for this purpose, but more work is needed.

De Jong et al. (1995) used the binary Markov chain model to compute the exact expected performance of small, simple problems

—while varying various GA parameters (such as the crossover rate and mutation rate);

—while changing the scaling on the fitness function; and

—while altering the fitnesses of schema-style building blocks.

Another practical use of this theory is to derive stopping criteria. Aytug and Koehler (1996, 2000) used the binary GA model to derive bounds on GA running times. These (upper) bounds provide sufficient conditions on the number of GA iterations needed to guarantee that one has seen an optimal solution with some stated confidence.

9. Conclusions and Future Directions

This research has shown that in the GSS groups studied, the groups engaged in a directed search process as opposed to engaging in a random search. Estimates of GA search parameters were obtained for experimental groups, indicating that these parameter values can possibly be examined and manipulated in order to achieve improved GSS outcomes. The advantages of the evolutionary method relative to others are many, including access to the existing body of theory of the exact expected behavior of GAs that might be applied to GSS research, and the large set of practitioner heuristics in the GA community that might be applied to GSS research. For example the expected waiting time to a particular solution can be computed as shown in Section 8. This technique could be immediately useful to managers looking for rudimentary quality control on the length of GSS sessions. Also, this method is well suited for developing realistic simulation models for GSS research, discussed in greater detail below.

Perhaps the most interesting capability of this model is the ability to capture the search processes of the experimental GSS groups. The search processes do vary from group to group as demonstrated in Table 1. Now that a preliminary computational method of examining these search processes has been identified, the factors affecting the search can be better studied. Specifically, how do the task, communication mode, group characteristics, incentive structures, and environmental variables affect the nature of this search in light of this model? Experimental data, when available, can be examined within the context of the model to see how the search varies between treatment conditions. Preliminary research (Rees and Koehler, 1999) shows that there is promise in examining the search processes of groups in this manner.

More data from actual GSS experimental use is required to further validate the model. Due to the highly constrained problem used in the Barkhi (1995) example, more data sets, especially data sets incorporating less constrained search spaces, are required. At the least, the GA estimation process should explicitly consider constraints on the search space. For example, in the Barkhi data capacity constraints in the underlying manufacturing problem were not explicitly modeled in the GA context. As a result, the GA search space was larger than it needed to be.

Also, more work needs to be completed with respect to examining the potential role crossover and mutation masks could play in the model. Instead of positing models where the mutation and crossover masks are constrained to forms dependent on mutation and crossover *rates*, the mask values could be estimated directly. This would increase the estimation problem from two parameters to 2ℓ. In other words, the size of the estimation problem would increase from two parameters (crossover and mutation in the current model) to two times the string length. In the example presented in this paper, the strings were twenty digits in length, representing a significant increase in computation time.

Other operators and behaviors have been studied in addition to crossover and mutation (Goldberg, 1989) and should be carefully examined in light of the GA evolutionary model. Examples of potential operators include niching behaviors and dominance operators.

We handled population size changes under the peer-influenced heuristic. Other schemes should be examined including the data-driven heuristic. Another possible scheme is one where the population size continuously increases with new solutions being added to the non-changing old ones. This could be modified to drop the oldest solutions or the least beneficial solutions. The mathematical models and estimation equations would have to be changed accordingly.

Once better GA models have been designed to capture GSS activities and their parameters estimated, further validation can be attained by using the derived GA model to predict a group's performance on a set of new problems and have these compared to actual outcomes. These "holdout" cases will help to distinguish between the apparent "random-looking" behavior of groups trying to solve complex, combinatorial problems under uncertainty from random behavior not systematically (on average) captured by the GA model.

One particularly attractive feature of the GA model is the ability to create simulations based on characteristics of the model. The ability to simulate groups and experimentally vary incentive schemes, group sizes, and composition, and other variables would be invaluable to many researchers. Simulation studies have the potential to shed new light on previously examined variables and allow researchers to carefully examine relationships in great detail with lower cost than repeating costly human subject studies.

Another exciting future application of this model is the creation of an "intelligent" GSS based on this model. We envision a GSS where GAs are built into the system, acting as an additional (albeit virtual) team member. The system itself could provide suggested solutions to the group problem as the GSS participants themselves use the system for problem solving. We do not suggest replacing the group itself with the system. Instead, the GA-based system would add rationality to and remove biases from the decision-making process, and the group members would add considerable domain knowledge and common sense to the process. In addition, the GA could be used to assist in the capture of organizational knowledge incorporated into such groups, greatly assisting in organizational knowledge management activities. The combination of the two forces could prove a formidable foe to many organizational problems.

One issue of interest is whether any conclusions can be drawn linking GA parameters, such as crossover and mutation rates, to GSS variables such as leadership, communication channel, group size, incentive structure, and others. Examining the crossover operator's role as an exploitation operator and the mutation operator's role as an exploration operator would hopefully lead to better insights into group decision-making processes. Rees and Koehler (1999) reported ambiguous and often contradictory results when examining such possible relationships. Most likely, more work needs to be performed in fine-tuning the model and testing the model on a wide variety of data sets performed before such relationships can be fruitfully explored.

References

Aytug, H., G. J. Koehler. 1996. Stopping criteria for finite length genetic algorithms. *ORSA Journal on Computing* **8** 183–191.

Aytug, H., G. J. Koehler. 2000. New stopping criteria for genetic algorithms. *European Journal of Operational Research* **126** 662–674.

Barkhi, R. 1995. An empirical study of the impact of proximity, leader and incentives on negotiation process and outcomes in a group decision support setting. Doctoral dissertation, Department of Decision Sciences and Information Systems, The Ohio State University, Columbus, OH.

Bhattacharyya, S., G. J. Koehler. 1994. An analysis of genetic algorithms of cardinality 2^V. *Complex Systems* **8** 227–256.

Chidambaram, L., R. P. Bostrom, B. E. Wynne. 1991. A longitudinal study of the impact of group decision support systems on group development. *Journal of Management Information Systems* **7** 7–25.

De Jong, K. A., W. M. Spears, D. F. Gordon. 1995. Using Markov chains to analyze GAFOs. *Foundations of Genetic Algorithms 3.* Morgan Kaufmann, San Francisco, CA, 115–137.

DeSanctis, G., R. B. Gallupe. 1987. A foundation for the study of group decision support systems. *Management Science* **33** 589–609.

Gallupe, R. B., G. DeSanctis. 1988. Computer-based support for group problem-finding: an experimental investigation. *MIS Quarterly* **12** 277–296.

Gavish, B., J. Kalvenes. 1996. Economic issues in group decision support systems. H. Pirkul, M. Shaw, eds. *Proceedings of the First INFORMS Conference on Information Systems and Technology*. INFORMS, Washington, DC, 18–27.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison Wesley, Reading, MA.

Gopal, A., R. P. Bostrom, W. W. Chin. 1993. Applying adaptive structuration theory to investigate the process of group support systems use. *Journal of Management Information Systems* **9** 45–69.

Guzzo, R. A., M. W. Dickson. 1996. Teams in organizations: recent research on performance and effectiveness. J. T. Spence, J. M. Darley, D. J. Foss, eds. *Annual Review of Psychology* **47** 307–338.

Hiltz, S. R. 1988. Productivity enhancement from computer-mediated communications: a systems contingency approach. *Communications of the ACM* **31** 1438–1454.

Hirokawa, R., D. Johnson. 1989. Toward a general theory of group decision making development of an integrated model. *Small Group Behavior* **20** 500–523.

Koehler, G. J. 1999. Computing simple GA expected waiting time. *Genetic and Evolutionary Computation Conference*. Morgan Kaufmann, Orlando, FL, 795.

Koehler, G. J., S. Bhattacharyya, M. D. Vose. 1998. General cardinality genetic algorithms. *Evolutionary Computation* **5** 439–459.

Mood, A. M. 1950. *Introduction to the Theory of Statistics*. McGraw-Hill, New York.

Nunamaker, J. F., A. R. Dennis, J. S. Valacich, D. R. Vogel, J. F. George. 1991. Electronic meeting systems to support group work. *Communications of the ACM* **34** 40–61.

Poole, M. S., G. DeSanctis. 1990. Understanding the use of group decision support systems: the theory of adaptive structuration. C. W. Steinfeld, J. Fulk, eds. *Organizations and Communication Technology.* Sage Press, Newbury Park, CA, 173–193.

Rao, V. S., S. L. Jarvenpaa. 1991. Computer support of groups: theory-based models for GDSS research. *Management Science* **37** 1347–1362.

Rees, J., R. Barkhi. 2001. The problem of highly constrained tasks in group decision support systems. *European Journal of Operational Research* **135** 220–229.

Rees, J., G. J. Koehler. 1998. An investigation of GA performance results for different cardinality alphabets. D. Davis, K. De Jong, M. Vose, D. Whitley, eds. *IMA Volumes in Mathematics and its Applications Proceedings from the IMA Workshop on Evolutionary Algorithms*. Springer, New York.

Rees, J., G. J. Koehler. 1999. Brainstorming, negotiating and learning in group decision support systems: an evolutionary approach. *Proceedings from the Thirty-Second Annual Hawaii International Conference on System Sciences*. IEEE Computer Society Press, Los Alamitos, CA.

Shamir, B. 1990. Calculations, values, and identities: the sources of collectivistic work motivation. *Human Relations* **43** 313–332.

Siegel, S. 1956. *Nonparametric Statistics for the Behavioral Sciences.* McGraw-Hill, New York.

Steiner, I. D. 1966. Models for inferring relationships between group size and potential group productivity. *Behavioral Science* **11** 273–283.

Steiner, I. D. 1972. *Group Process and Productivity.* Academic Press, New York.

Valacich, J. S., A. R. Dennis. 1994. A mathematical model of performance of computer-mediated groups during idea generation. *Journal of Management Information Systems* **11** 59–72.

Vose, M. D. 1990. Formalizing genetic algorithms. *Proceedings of the IEEE Workshop on Genetic Algorithms, Neural Networks, and Simulated Annealing Applied to Signal and Image Processing*. Sinauer Associates, Glasgow, Scotland 212–232.

Vose, M. D. 1999. *The Simple Genetic Algorithm: Foundations and Theory*. MIT Press, Cambridge, MA.

Vose, M. D., G. E. Liepins. 1991. Punctuated equilibria in genetic search. *Complex Systems* **5** 31–44.

Vose, M. D., A. H. Wright. 1995. Simple genetic algorithms with linear fitness. *Evolutionary Computation* **2** 347–368.

Watson, R. T., G. DeSanctis, M. S. Poole. 1988. Using a GDSS to facilitate group consensus: some intended and unintended consequences. *MIS Quarterly* **12** 463–477.

Accepted by John W. Chinneck; received September 2000; revised July 2001, November 2001; accepted November 2001.