Generalized Crowding for Genetic Algorithms

Severino F. Galan
Dept. of Artificial Intelligence
National Distance University of Spain
C/ Juan del Rosal 16
28040 Madrid, Spain
seve@dia.uned.es

Ole J. Mengshoel
Carnegie Mellon University
NASA-Ames Research Center
Mail Stop 263-3, P.O. Box 1
Moffett Field, CA 94035
ole.mengshoel@sv.cmu.edu

ABSTRACT

Crowding is a technique used in genetic algorithms to preserve diversity in the population and to prevent premature convergence to local optima. It consists of pairing each offspring with a similar individual in the current population (*pairing phase*) and deciding which of the two will remain in the population (*replacement phase*). The present work focuses on the replacement phase of crowding, which usually has been carried out by one of the following three approaches: Deterministic, Probabilistic, and Simulated Annealing. These approaches present some limitations regarding the way replacement is conducted. On the one hand, the first two apply the same selective pressure regardless of the problem being solved or the stage of the genetic algorithm. On the other hand, the third does not apply a uniform selective pressure over all the individuals in the population, which makes the control of selective pressure over the generations somewhat difficult. This work presents a *Generalized Crowding* approach that allows selective pressure to be controlled in a simple way in the replacement phase of crowding, thus overcoming limitations of the other approaches. Furthermore, the understanding of existing approaches is greatly improved, since both Deterministic and Probabilistic Crowding turn out to be special cases of Generalized Crowding. In addition, the temperature parameter used in Simulated Annealing is replaced by a parameter called *scaling factor* that controls the selective pressure applied. Theoretical analysis using Markov chains and empirical evaluation using Bayesian networks demonstrate the potential of this novel Generalized Crowding approach.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Markov processes; Probabilistic algorithms (including Monte Carlo); I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO'10, July 7–11, 2010, Portland, Oregon, USA.
Copyright 2010 ACM 978-1-4503-0072-8/10/07 ...$10.00.

General Terms

Algorithms

Keywords

Genetic algorithms, niching, deterministic crowding, probabilistic crowding, Markov chain analysis, Bayesian networks, experiments

1. INTRODUCTION

Genetic algorithms (GAs) [6, 4] use stochastic search methods based on natural evolution in order to solve optimization problems in fields like design, learning, or planning, among others. Two processes form the basis of GAs: variation (recombination and mutation) and selection. While the former facilitates diversity and novelty, the latter favors quality. The goal of a GA is to find either an optimal (or near-optimal) solution or to identify a set of high-fitness solutions. Premature convergence to local optima is one of the most frequent difficulties that arise when applying GAs to complex problems. It is associated with the loss of diversity in the population; however, too much population diversity can lead to a dramatic deterioration of GA efficiency. Therefore, an important issue in GA research and practice is the trade-off between exploitation of the best individuals and exploration of alternative regions of the search space.

Crowding was introduced by De Jong [2] as a technique for preserving population diversity and preventing premature convergence. Crowding is applied in the survival selection step of GAs in order to decide which individuals among those in the current population and their offspring will pass to the next generation. Crowding consists of two main phases: pairing and replacement. In the pairing phase, offspring individuals are paired with individuals in the current population according to a similarity metric. In the replacement phase, a decision is made for each pair of individuals as to which of them will remain in the population. A review of crowding approaches for GAs can be found in [13].

Depending on how the replacement phase is carried out, there are three main types of crowding: Deterministic [8, 9], Probabilistic [12, 11], and based on Simulated Annealing [10]. Deterministic Crowding selects the fittest individual in each pair in the replacement phase. Probabilistic Crowding selects the surviving individual for each pair based on a probabilistic formula that takes fitness into account. Finally, crowding inspired by Simulated Annealing uses well-known rules like Metropolis or Boltzmann, which include a temperature parameter, in the replacement phase.

The design of the replacement rule has great impact on the performance of crowding. Deterministic Crowding develops an exploitative replacement strategy. This may be a disadvantage, since exploitative methods in GAs frequently lead to premature convergence. Probabilistic Crowding promotes the exploration of alternative (less fit) solutions. The degree of exploration is defined through a probabilistic formula that is not changed throughout the GA execution. However, a trade-off between exploration and exploitation can only be achieved if selective pressure can be adapted to the problem being solved and to the GA stage. Crowding based on Simulated Annealing replacement rules allows the degree of exploration to be changed through the temperature parameter. In spite of that, given a temperature and a population, a uniform selective pressure is not applied over all the individuals in the population as shown in Section 2.2. This makes the control of selective pressure over the generations more difficult than it perhaps needs to be. Giving a solution to the three limitations enumerated above has been one of the motivations for the research described in this paper.

This work focuses on the replacement phase of crowding and presents a *Generalized Crowding* approach that allows selective pressure to be controlled in a simple way. Generalized Crowding is inspired by Probabilistic Crowding, but differs from it in that the degree of exploration can be changed by means of a parameter named the *scaling factor*. Both Deterministic and Probabilistic Crowding turn out to be special cases of Generalized Crowding, obtained by assigning the scaling factor values 0 and 1 respectively. A specific scaling factor value provides the population with a way of applying a uniform selective pressure over all its individuals. Like temperature, the scaling factor can be lowered throughout the generations by following a determined schedule.

The rest of this paper is organized as follows. First, we present preliminaries including previous research on crowding in GAs. The following section introduces our Generalized Crowding approach, which is founded on the use of a parametric scaling factor. Next, we analyze our novel approach by means of (discrete time, discrete state space) Markov chains. We then present experimental results for the use of Generalized Crowding to estimate the most probable explanation in Bayesian networks (BNs), while the last section concludes and outlines areas for future research.

2. THE CROWDING APPROACH

We first introduce a few definitions: P_C is crossover probability; P_M is mutation probability; M is population size; and S is family size [13]. Without loss of generality, we assume maximization of a fitness function $f : \{0,1\}^n \longrightarrow \Re$ in this section.

The original crowding scheme developed by De Jong in the seventies [2] consists of randomly selecting for each offspring γ individuals from the current population. The offspring will replace the most similar selected individual. Parameter γ is known as *crowding factor*, and usually $\gamma = 2$ is used.

De Jong's scheme was changed slightly in the nineties by Mahfoud [8, 9]. Since an offspring is likely to be similar to its parents, the following scheme can be used to efficiently preserve diversity in the population:

1. The individuals in the current population are randomly paired. (Since parent selection is not usually applied under crowding, every individual in the population becomes a parent.)
2. With probability P_C, the parents in each pair (p_1, p_2) are recombined. The two resulting children (c_1, c_2) are mutated with probability P_M.
3. Each child competes with one of its two parents to be included in the population of the next generation. Let $d(i_1, i_2)$ denote the distance between two individuals, i_1 and i_2:

   If $d(p_1, c_1) + d(p_2, c_2) < d(p_1, c_2) + d(p_2, c_1)$
   $p_1 \leftarrow$ winner of competition between p_1 and c_1
   $p_2 \leftarrow$ winner of competition between p_2 and c_2
   else
   $p_1 \leftarrow$ winner of competition between p_1 and c_2
   $p_2 \leftarrow$ winner of competition between p_2 and c_1

Under this scheme, each offspring tends to compete for survival with its most similar parent. Other variants exist that pick up more than two parents and children before the similarity metric is applied [13, Section 4.2]. This idea has been the basis of a number of widely-used modern crowding approaches [8, 9, 12, 11, 10]. One key difference between these approaches is the rule used to decide the winner for each competition, as we discuss next.

2.1 Competition and Replacement

In crowding, the way a competition takes place between parent p and child c is defined through a so-called *replacement rule*. We now discuss four of the most widely used replacement rules: *deterministic replacement*, *probabilistic replacement*, *Boltzmann replacement*, and *Metropolis replacement*. The two latter are based on the application of Simulated Annealing ideas [7] to crowding.

In *Deterministic Crowding* [8, 9], the winner of a competition between parent p and child c is the one with higher fitness. Let P_c denote the probability that child c replaces parent p in the population. This probability can be expressed in the following way for deterministic replacement:

$$P_c = \begin{cases} 1 & \text{if } f(c) > f(p) \\ 0.5 & \text{if } f(c) = f(p) \\ 0 & \text{if } f(c) < f(p) \end{cases}.$$

Unlike Deterministic Crowding, *Probabilistic Crowding* [12, 11] uses a non-deterministic rule to establish the winner of a competition between parent p and child c. The probability that c replaces p in the population is the following:

$$P_c = \frac{f(c)}{f(c) + f(p)}. \tag{1}$$

Boltzmann Crowding [10] is based on the well-known Simulated Annealing method, implemented with the Boltzmann acceptance rule [1, 5]. Under this replacement method, P_c adopts the expression:

$$P_c = \frac{1}{1 + \exp\left(\frac{f(p) - f(c)}{T}\right)}, \tag{2}$$

where T is the temperature at the current generation. Temperature is usually reduced over the generations by means of a cooling schedule. Two typical cooling schedules consist of repeating throughout generations one of the following operations: multiplying the current temperature by a constant

lower than unity, or subtracting a constant amount from the current temperature. Some authors have also used fixed temperature.

Like Boltzmann Crowding, *Metropolis Crowding* [10] applies Simulated Annealing in the replacement phase. However, the Metropolis acceptance rule [14] is used instead of the Boltzmann rule. The probability that child c replaces parent p in the population is calculated as follows:

$$P_c = \begin{cases} 1 & \text{if } f(c) \geq f(p) \\ \exp\left(-\frac{f(p)-f(c)}{T}\right) & \text{if } f(c) < f(p) \end{cases}, \quad (3)$$

where T is the temperature at the current generation. The same cooling schedule as in Boltzmann Crowding can be applied; alternatively, a fixed temperature can be used.

2.2 Discussion

The probability P_c that child c replaces parent p in the population is calculated in different ways depending on the specific replacement rule being used. While in the rules based on Simulated Annealing, Boltzmann replacement and Metropolis replacement, P_c is a function of $f(p) - f(c)$ as shown in Equations 2 and 3, in probabilistic replacement P_c is a function of $f(p)/f(c)$ as can be seen by dividing numerator and denominator in Equation 1 by $f(c)$.

Given parent p with fitness $f(p)$ and child c with fitness $f(c)$, quite different selective pressures are obtained in general from $P_c = g_1(f(p) - f(c))$ and $P_c = g_2(f(p)/f(c))$, where g_1 and g_2 are real functions. For example, let S_1 and S_2 be the following sets of parent-child pairs: $S_1 \equiv \{(f(p_1) = 2,\ f(c_1) = 1),\ (f(p_2) = 12,\ f(c_2) = 11),\ (f(p_3) = 102,\ f(c_3) = 101)\}$ and $S_2 \equiv \{(f(p_1) = 2,\ f(c_1) = 1),\ (f(p_2) = 20,\ f(c_2) = 10),\ (f(p_3) = 200, f(c_3) = 100)\}$.

While applying $P_c = g_1(f(p) - f(c))$ would produce constant P_c values for the elements in S_1 ($f(p) - f(c) = 1$ for each pair in S_1), applying $P_c = g_2(f(p)/f(c))$ would produce non-constant P_c values for S_1. The opposite would take place for S_2. This work focuses on using a function of $f(p)/f(c)$ in the replacement phase, due to the fact that selective pressure is more frequently applied in GAs by following this option rather than the option using a function of $f(p) - f(c)$. For example, *fitness proportional selection* [6] is a widely used selection method in GAs that is mathematically equivalent to Probabilistic Crowding: Given two individuals p and c, the latter is selected with probability $\frac{f(c)}{f(c)+f(p)}$ under fitness proportional selection, which is equivalent to the formula applied in probabilistic replacement.

It is interesting to analyze the values that Simulated Annealing would generate for P_c in the case of the pairs in S_2:

$$P_{c_1} = \exp\left(-\tfrac{2-1}{1}\right) = 0.36788$$

$$P_{c_2} = \exp\left(-\tfrac{20-10}{1}\right) = 4.54 \times 10^{-5}$$

$$P_{c_3} = \exp\left(-\tfrac{200-100}{1}\right) = 3.7201 \times 10^{-44}$$

where $T = 1$ and Metropolis replacement has been applied. As shown in Table 1, while under probabilistic replacement P_{c_i} would be equal to 1/3 for $i \in \{1, 2, 3\}$, under Metropolis replacement a non-uniform set of P_c values is obtained. For instance, $P_{c_1} = 0.36788$ is similar to applying probabilistic replacement, whereas $P_{c_3} = 3.7201 \times 10^{-44}$ is almost the same as applying deterministic replacement.

Set	$f(p_i)$	$f(c_i)$	$f(p_i) - f(c_i)$	$\frac{f(p_i)}{f(c_i)}$	$P_{c,\mathrm{PR}}$	$P_{c,\mathrm{MR}}$
S_1	2	1	1	2	0.333	0.368
	12	11		1.09	0.478	
	102	101		1.01	0.498	
S_2	2	1	1	2	0.333	0.368
	20	10	10			4.54×10^{-5}
	200	100	100			3.72×10^{-44}

Table 1: P_c values for the pairs in S_1 and S_2 under probabilistic replacement ($P_{c,\mathbf{PR}}$) and Metropolis replacement ($P_{c,\mathbf{MR}}$). $P_{c,\mathbf{PR}}$ is invariant for S_2, whereas $P_{c,\mathbf{MR}}$ is invariant for S_1.

In order to create a replacement rule with the advantages of both probabilistic replacement (where P_c is a function of $f(p)/f(c)$ for all the individuals in the current population) and Simulated Annealing replacement (where P_c can be controlled over generations through a parameter, *temperature* in this case), the next section introduces a new crowding method, which we name *Generalized Crowding*.

3. GENERALIZED CROWDING

Building on the schemes discussed in the previous section, Generalized Crowding consists of a pairing phase and a replacement phase. The novelty is that the replacement phase relies on the use of a scaling factor ϕ, which allows us to carry out a broad range of replacement rules just by adjusting ϕ, compared to having to deal with potentially very different replacement algorithms.

In Generalized Crowding, the winner of a competition between parent p and child c is established using:

$$P_c = \begin{cases} \frac{f(c)}{f(c)+\phi\times f(p)} & \text{if } f(c) > f(p) \\ 0.5 & \text{if } f(c) = f(p) \\ \frac{\phi\times f(c)}{\phi\times f(c)+f(p)} & \text{if } f(c) < f(p) \end{cases}, \quad (4)$$

where P_c is the probability that child c replaces parent p in the population, f is the fitness function to be maximized, and $\phi \in \Re^+ \cup \{0\}$ denotes a parameter named *scaling factor*.

The key idea in Generalized Crowding is that fitness scaling of the least fit individual, among p and c, is done before Probabilistic Crowding is applied. As shown in (4), if $f(p) < f(c)$ then $f(p)$ is transformed into $\phi \times f(p)$; otherwise, if $f(c) < f(p)$ then $f(c)$ is transformed into $\phi \times f(c)$.

Specifically, when $\phi = 0$ in Equation 4, Generalized Crowding becomes equivalent to Deterministic Crowding. When $0 < \phi < 1$, it is possible that the least fit among p and c wins in the replacement phase. When $\phi = 1$, Generalized Crowding turns into Probabilistic Crowding. Finally, when $\phi > 1$ the probability that the least fit of p and c wins is greater than in Probabilistic Crowding, which may be beneficial for deceptive or highly multimodal problems.

To avoid having to deal with three separate cases in (4), one can use the logistic function $\ell(x) = 1/(1+e^{-x})$ to obtain:

$$P_c = \frac{\phi^{\ell(f(p)-f(c))} \times f(c)}{\phi^{\ell(f(p)-f(c))} \times f(c) + \phi^{\ell(f(c)-f(p))} \times f(p)}. \quad (5)$$

This scaling of (4) or (5) is different from the scaling in Simulated Annealing for two main reasons:

1. In Generalized Crowding either $f(p)$ or $f(c)$ is changed, while in Simulated Annealing the magnitude changed is $f(p)$ - $f(c)$. In other words, following the discussion in Section 2.2, Generalized Crowding applies selective pressure as a function of $f(p)/f(c)$.

2. The scaling operation is kept simple in Generalized Crowding, that is, just a multiplication with no divisions or exponentials. This allows Generalized Crowding to include Deterministic Crowding and Probabilistic Crowding as special cases.

The scaling factor ϕ allows a wide range of selective pressures to be applied: The greater ϕ is, the more probable it is that locally non-optimal parts of the search space are explored. Similar to Simulated Annealing, ϕ can be subjected to a lowering schedule that favors exploration at the first GA generations and progressively increments the degree of exploitation of the best solutions.

4. MARKOV CHAIN ANALYSIS

In this section, we use discrete time, discrete state space Markov chains to analyze our Generalized Crowding approach. We first present the general case for N niches, and then we solve the case for two niches by using the general equations for N niches.

4.1 Multiple Niches

We now perform an analysis in which each niche that a parent can be in has a corresponding Markov chain state. There are additional Markov chain states, one for each combination of all possible niche locations for pairs of a child and a parent. Formally, we introduce the following definitions: p is a parent in the current generation; p' is a parent in the next generation; c is a child in the current generation; N is the number of niches; and[1]

$$\begin{aligned}\Pr([p=i,\, c=j] \mid [p=i]) &= A_{i,j},\ i,\, j \in \{0,\ldots,N-1\}\\ \Pr([p'=j] \mid [p=i,\, c=j]) &= B_{i,j},\ i,\, j \in \{0,\ldots,N-1\},\end{aligned}$$

where $A_{i,j}$ denotes the probability that, given parent p at niche i, it is paired with child c at niche j, and $B_{i,j}$ denotes the probability that, given parent p at niche i paired with child c at niche j, the resulting parent p' for the next generation is at niche j after replacement.

We need to calculate the following, prior to solving the balance equations for $i, j \in \{0,\ldots,N-1\}$:

$$\Pr([p'=j] \mid [p=i]) = \sum_{k=0}^{N-1} \Pr([p'=j] \mid [p=i, c=k]) \times \Pr([p=i, c=k] \mid [p=i]).$$

We have here two cases: (i) If $i \neq j$ then $\Pr([p'=j] \mid [p=i]) = A_{i,j}B_{i,j}$; (ii) if $i=j$ then $\Pr([p'=j] \mid [p=i]) = 1 - \sum_{k=0}^{N-1} A_{i,j}B_{i,j}$.

[1]Note that while $\Pr(A, B \mid B) = \Pr(A \mid B)$ for events A and B, we have $\Pr([p=i,\, c=j] \mid [p=i]) \neq \Pr([c=j] \mid [p=i])$. In other words, $[p=i,\, c=j]$ is a distinct event and cannot be simplified to $[c=j]$ even if we condition on $[p=i]$.

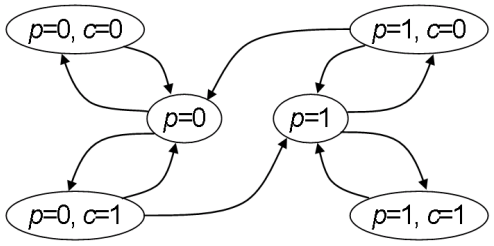

Figure 1: Augmented Markov chain (AMC).

We now obtain the following balance equations:

$$\sum_{k=0,k\neq i}^{N-1} \Pr([p'=k] \mid [p=i]) \Pr([p=i]) = \sum_{k=0,k\neq i}^{N-1} \Pr([p'=i] \mid [p=k]) \Pr([p=k]),$$

and in addition we have $\sum_{k=0}^{N-1} \Pr([p=k]) = 1$. By solving this system of N equations from the formulas just obtained in (i) and (ii) for $\Pr([p'=j] \mid [p=i])$, we can obtain $\Pr([p=k])$ for $k \in \{0,\ldots,N-1\}$. However, the resulting expressions are quite complex for the general case. So, in this work, we analyze the case of two niches in more detail.

4.2 Two Niches

To analyze the case of two niches, we introduce two Markov chains. A more detailed augmented Markov chain, which we discuss first, is used as an aid to define the transition probabilities in a two-state generational Markov chain, where each state corresponds to a niche.

4.2.1 Augmented Markov Chain

The graph for the augmented Markov chain (AMC) is shown in Figure 1. The transition matrix is shown in Table 2; it explicitly shows the transition probabilities A, A', B, B' and others. States $[p=0]$ and $[p=1]$ represent the contents of a population location at a certain time; it contains an individual (parent) that is either in niche 0 (so $p=0$) or niche 1 (so $p=1$). Consider a location that contains an individual in niche 1, so $p=1$. At the competition step (after crossover, mutation, and pairing), there are two options for p: either the competing child is in niche 0 (so $c=0$) or in niche 1 (so $c=1$). Consequently, we have two states, namely $[p=1,\, c=0]$ and $[p=1,\, c=1]$, and there are edges from $[p=1]$ to each of these. After the competition we are left with an individual in niche 0 or 1, which becomes a parent in the next generation, consequently there are two edges from $[p=1,\, c=0]$—into $[p=0]$ and $[p=1]$ respectively—and two similar edges from $[p=1,\, c=1]$. A similar logic applies to states $[p=0,\, c=0]$ and $[p=0, c=1]$.

First, we consider how transition probabilities depend on crossover, mutation, and pairing in the crowding algorithm (A and A'). We introduce the following definitions: $V(0)$ is volume of niche 0; $V(1)$ is volume of niche 1; $A(0)$ is surface area of niche 0; and $A(1)$ is surface area of niche 1.

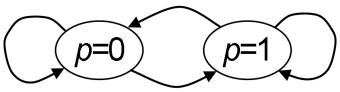

Figure 2: Generational Markov chain (GMC).

For A and A' we have:

$$A = \frac{c_A \times P_C \times P_M \times A(1)}{S \times V(1)}$$
$$A' = \frac{c_{A'} \times P_C \times P_M \times A(0)}{S \times V(0)},$$

where c_A and $c_{A'}$ are constants. For simplicity, we assume that $c_A = c_{A'}$ in the rest of this paper.

Second, we consider probabilities that depend on replacement (B and B'). We introduce these definitions: $f(0)$ is the fitness for niche 0 and $f(1)$ is the fitness for niche 1. Using (5) we now have:

$$B = \frac{f(0)}{f(0) + \phi^{\ell(f(0)-f(1))-\ell(f(1)-f(0))} \times f(1)}$$
$$B' = \frac{f(1)}{f(1) + \phi^{\ell(f(1)-f(0))-\ell(f(0)-f(1))} \times f(0)}$$

4.2.2 Generational Markov Chain

The graph for the generational Markov chain (GMC) is shown in Figure 2. Its two states correspond, respectively, to the two states with the same labels in the AMC. The remaining four AMC states are abstracted into the four GMC transition probabilities, as we will see shortly. In preparation for stating the GMC transition matrix, we introduce the following definitions for the AMC states: $0 \equiv [p = 0]$; $1 \equiv [p = 1]$; $00 \equiv [p = 0, c = 0]$; $01 \equiv [p = 0, c = 1]$; $10 \equiv [p = 1, c = 0]$; and $11 \equiv [p = 1, c = 1]$.

The four GMC transition probabilities can now easily be derived from the AMC transition probabilities:

$$\Pr(0 \mid 0) = \Pr(0 \mid 00)\Pr(00 \mid 0) + \Pr(0 \mid 01)\Pr(01 \mid 0) = 1 - B'A',$$

and in a similar way we obtain $\Pr(0 \mid 1) = BA$, $\Pr(1 \mid 0) = B'A'$, and $\Pr(1 \mid 1) = 1 - BA$. Assuming balance, it is now easy to obtain $\Pr(0)\Pr(1 \mid 0) = \Pr(1)\Pr(0 \mid 1)$ and $\Pr(0) + \Pr(1) = 1$, which gives the following stationary distributions:

$$\Pr(0) = 1/\left(1 + \left(A'B'/AB\right)\right)$$
$$\Pr(1) = 1/\left(1 + \left(AB/A'B'\right)\right).$$

By introducing the definitions for A, A', B, and B', we obtain the following expressions for the stationary distributions:

$$\Pr(0) = \frac{1}{1 + \frac{A(0)V(1)f(1)}{A(1)V(0)f(0)}\phi^{\ell(f(0)-f(1))-\ell(f(1)-f(0))}}$$
$$\Pr(1) = \frac{1}{1 + \frac{A(1)V(0)f(0)}{A(0)V(1)f(1)}\phi^{\ell(f(1)-f(0))-\ell(f(0)-f(1))}}$$

The following conclusions can be established from these two formulas:

1. When $\phi = 0$, the niche with highest fitness gets all the population. This is Deterministic Crowding.

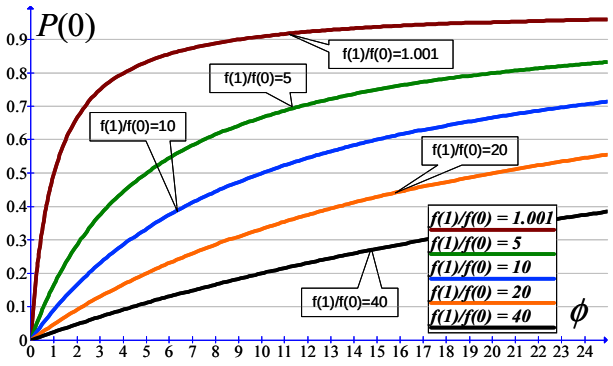

Figure 3: The stationary probability $\Pr(0)$ (along y-axis) as a function of ϕ (along x-axis) for different fitness function ratios $f(1)/f(0)$.

2. When $\phi = 1$, the probability for niche i is $f(i)/(f(i) + f(j))$, with $i, j \in \{0, 1\}$. This is Probabilistic Crowding.

3. When $\phi = f(j)/f(i)$, where $f(i) < f(j)$, the population distributes equally between both niches.

4. When $\phi > f(j)/f(i)$, where $f(i) < f(j)$, the niche with less fitness gets more population than the other niche. As ϕ increases, even more population will go to the less fit niche with respect to the other niche in the stationary distribution.

Case 4 can be beneficial in the case of multimodal problems, where, for example, Case 1 may lead to premature convergence. For example, in the experiments of Section 5 with the Bayesian network denoted Alarm0.2, $\phi = 10$ was the value giving best results when $M = 20$.

4.2.3 Population

Given that we have the above results for one population location, how can they be extended to a population size M? We use M Markov chains instead of just one Markov chain. In the case of one Markov chain and two niches $\{0, 1\}$, a stationary distribution $\Pr(i)$ for $i \in \{0, 1\}$ is essentially a Bernoulli trial for niche i. So for M population locations (Markov chains), we have a Binomial. Consequently, the mean is $M\Pr(i)$, and the variance is $M\Pr(i)(1 - \Pr(i))$. The mean is the expected number of individuals we expect to see in the niche. Compared to previous population sizing results [13], we now have more general expressions involving ϕ for the stationary distributions.

4.2.4 An Example

Figure 3, where we consider two niches 0 and 1, shows ϕ along the x-axis and the stationary probability for niche 0, $\Pr(0)$, along the y-axis. We assume that $f(0) < f(1)$ and have drawn graphs for different $f(1)/f(0)$ values. In this case of $f(0) < f(1)$ we have:

$$\Pr(0) = \frac{1}{1 + \frac{f(1)}{\phi f(0)}}$$

which we have obtained by (i) putting $V(0) = V(1)$ and $A(0) = A(1)$ and (ii) not using the logistic function, due

	$\mathbf{p=0}$	$\mathbf{p=1}$	$\mathbf{p=0, c=0}$	$\mathbf{p=0, c=1}$	$\mathbf{p=1, c=0}$	$\mathbf{p=1, c=1}$
$\mathbf{p=0}$	0	0	$1-A'$	A'	0	0
$\mathbf{p=1}$	0	0	0	0	A	$1-A$
$\mathbf{p=0, c=0}$	1	0	0	0	0	0
$\mathbf{p=0, c=1}$	$1-B'$	B'	0	0	0	0
$\mathbf{p=1, c=0}$	B	$1-B$	0	0	0	0
$\mathbf{p=1, c=1}$	0	1	0	0	0	0

Table 2: Transition matrix for the AMC shown in Figure 1.

to the fact that Pr(0) cannot be expressed as a function of $f(1)/f(0)$ if the logistic function is used. The following observations can be made from these graphs:

- The greater ϕ is, the larger a proportion of the population goes to the lesser fit niche 0.
- For low ϕ values ($0 \leq \phi \leq 1$), the more fit niche 1 gets a higher percentage of the population.
- If ϕ is kept constant, the higher $f(1)/f(0)$ is, the more population goes to the more fit niche 1.
- Although $f(0) < f(1)$, if $\phi = f(1)/f(0)$, then $\Pr(0) = \Pr(1) = 0.5$ and the population is equally distributed between niches 0 and 1. For example, the ϕ-values where the curves reach $\Pr(0) = 0.5$ vary as follows: curve $f(1)/f(0) = 5$ crosses $\Pr(0) = 0.5$ at $\phi = 5$ and curve $f(1)/f(0) = 40$ crosses $\Pr(0) = 0.5$ at $\phi = 40$.
- For high enough ϕ values, even if $f(0) < f(1)$, niche 0 receives most of the population.

Consider the question: Why do values of $\phi > 1$ work so well in some cases? We hypothesize that in highly multimodal problems the degree of deceptiveness is high in general and consequently, locally, an individual with higher fitness may be farther away from the global optimum than another less fit individual is.

5. EXPERIMENTS

In these experiments, we are investigating how Generalized Crowding's performance depends on the scaling factor ϕ. For experimental purposes, we consider the estimation of most probable explanations (MPEs) in BNs, a computationally hard problem of interest in many applications including image recognition, diagnosis, and error correction decoding.

5.1 Variants of Alarm Bayesian Network

In these experiments, we used variants of the well-known Alarm BN to investigate the performance of Generalized Crowding. The Alarm BN has 37 nodes and 752 CPT values (none are zero).[2] We randomly introduced zeros into the CPTs of the BN in order to vary the search space characteristics. Specifically, zeros were introduced with probability $\rho \in \{0.0, 0.1, 0.2, \ldots\}$ [3] as reflected in the notation Alarmρ. An Alarmρ BN is not the same as the Alarm BN. They have the same graph structure, node cardinalities, and number of CPT values, but CPT values are generated in a randomized way for Alarmρ [3]. For Alarmρ, the number of zero CPT values is approximately $\rho\times$ 100%. In experiments with these Alarmρ BNs we varied ϕ for Generalized Crowding, and investigated $\phi = 0$ (Deterministic Crowding), $\phi = 0.5$, $\phi = 1$ (Probabilistic Crowding), $\phi = 10$, and $\phi = 100$. In these latter two cases, more often than not the worse fit among a parent and child wins, and surprisingly we found that $\phi = 10$ out-performed $\phi = 0$ and $\phi = 1$ in some experiments. In addition, we varied the population size M.

Parameterized uniformed crossover has been used with $P_C = 1$. For a BN with N nodes, we used $P_M = 1/N$, and mutation amounted to changing a node's state uniformly at random. Experimental results are averaged over 350 runs.

The results of the experiments are summarized in Figure 4. We derive the following conclusions from the experiments. For Alarm0.0, the higher the scaling factor ϕ is, the worse performance is for all population sizes[3] $M \in \{20, 40, 80, 160\}$. For Alarm0.1, and with $M \in \{40, 80, 160\}$, the same as for Alarm0.0 applies. However, there is an important change for $M = 20$ (see Figure 4(c)): the higher ϕ is, the better performance is at generation 2,000 (with the exception of $\phi = 100$). What we said for Alarm0.1 applies again for Alarm0.2, but with more intensity and the twist that $M = 40$ is now more similar to Alarm0.1's $M = 20$ (see Figure 4(c)) rather than its $M = 160$ (see Figure 4(d)).

These experimental results suggest that, as the number of zeros in the CPTs grows and consequently the complexity of the search space increases, it is useful to increase ϕ in order to increase the degree of GA exploration. This tendency does not apply to Alarm0.25 and Alarm0.3. This was initially a little surprising to us, but we now hypothesize that plateaus with fitness $f = 0$ begin to gain in importance in the search process. Since those plateaus are explored in the same manner by Generalized Crowding, regardless of the ϕ value (the child has 0.5 probability of being chosen), the performance of the GA for different ϕ values is more and more similar. There are ways to explore these plateaus more efficiently, an issue beyond the scope of this paper.

Key points in these experiments are the following:

- The fact that Generalized Crowding allows selective pressure to be varied through the ϕ parameter is important, as the optimal value of ϕ depends on whether Alarm0.0, Alarm0.1, or Alarm0.2 is processed and which value for M is used.
- Deterministic Crowding ($\phi = 0$) and Probabilistic Crowding ($\phi = 1$) are generally strong performers. Surprisingly, in some cases Generalized Crowding with $\phi > 1$ produces better results. For example, this is shown by Alarm0.2 as the population size is reduced—see Figure 4(e) and to a lesser extent Figure 4(c).

[2]See http://compbio.cs.huji.ac.il/Repository/Datasets/alarm/alarm.htm for details on Alarm.

[3]We show in detail only a subset of the experimental results, due to space restrictions, but summarize a broader range of experiments in the text.

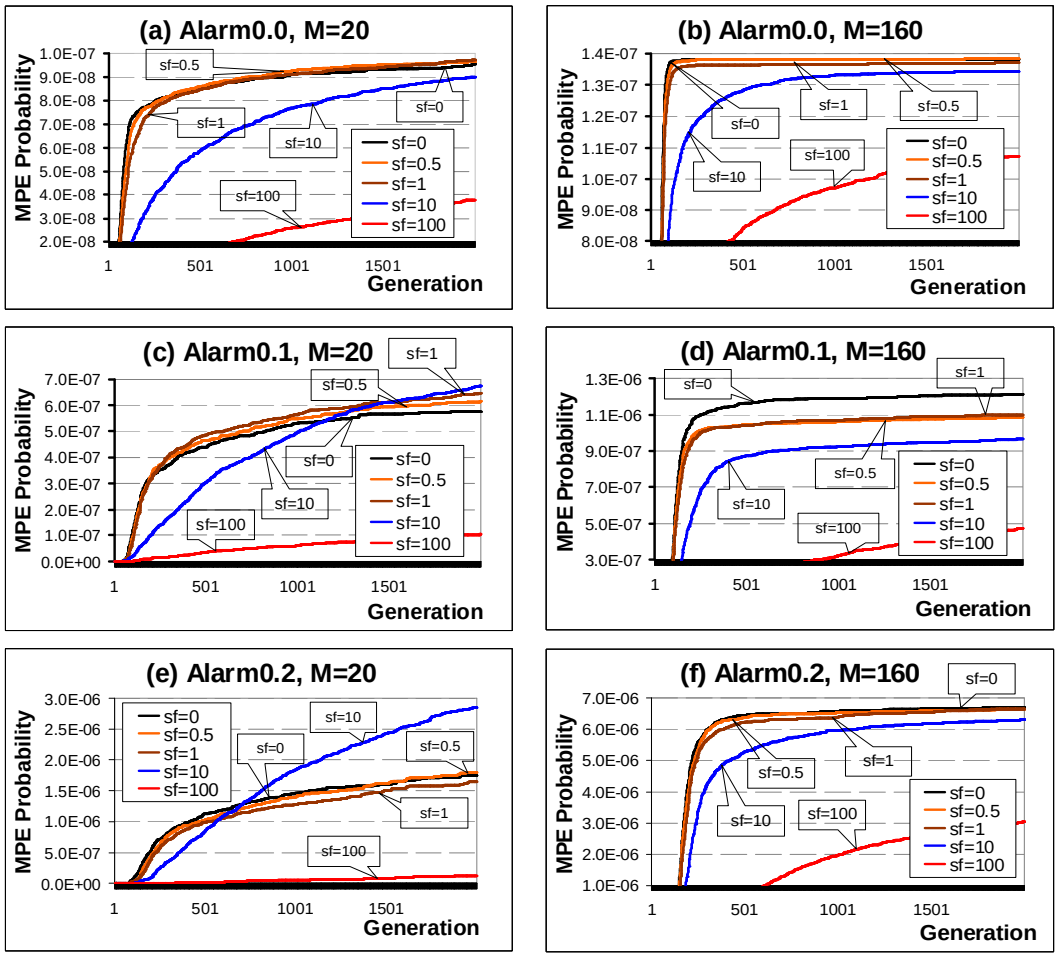

Figure 4: The effect of varying the scaling factor (sf or ϕ) when estimating the most probable explanation (MPE) for different variants of the Alarm BN (Alarm0.0, Alarm0.1, and Alarm0.2) for two different population sizes $M = 20$ and $M = 160$.

While we kept ϕ constant during search here, it could be dynamically adjusted. There are different ways to do this, subject to future research, but one could (i) perform linear decrements throughout generations or (ii) decrement proportionally to population entropy.

5.2 Various Bayesian Networks

In this section, we discuss the results of experiments using a few different BNs, namely Barley, HeparII, and Water. The Barley BN has 48 nodes with 130180 CPT values (none of them equal to 0).[4] The HeparII BN contains 70 nodes with 2139 CPT values (none of them equal to 0).[5] The Water BN has 32 nodes with 13484 CPT values (3972 of them equal to 0).[6] Experimental results are averaged over 350 runs, except for Barley where 50 runs were used.

Experiments with these three BNs are summarized in Figure 5, from which we make a few observations. (i) For the HeparII and Barley BNs, it is clear that a lower ϕ leads to better performance. These two BNs have no zeros in their CPTs. (ii) For the Water BN (29.45% of zeros in its CPTs), $\phi = 1$ (Probabilistic Crowding) is superior to the rest. Both (i) and (ii) are in accordance with the results obtained for Alarmρ: When there are no zeros in the CPTs, Deterministic Crowding is the best option; however, when zeros are present in the CPTs, $\phi > 0$ leads to the best performance. The optimal ϕ-value depends on the BN at hand.

[4]See http://compbio.cs.huji.ac.il/Repository/Datasets/barley/barley.htm for details on Barley.
[5]See http://www.pitt.edu/~druzdzel/abstracts/springer00-.html for details on HeparII.
[6]See http://compbio.cs.huji.ac.il/Repository/Datasets/water/water.htm for details on Water.

6. CONCLUSION AND FUTURE WORK

In this work, a general method for crowding in GAs has been defined, analyzed, and evaluated. The method is based on making the replacement phase in crowding more flexible, specifically by introducing a scaling factor ϕ, which enables the selective pressure to be easily controlled. This flexibility allows the new method, Generalized Crowding, to be successfully applied to a wide range of problems. Furthermore, Generalized Crowding has as special cases other widely-used crowding methods such as Deterministic Crowding ($\phi = 0$) and Probabilistic Crowding ($\phi = 1$).

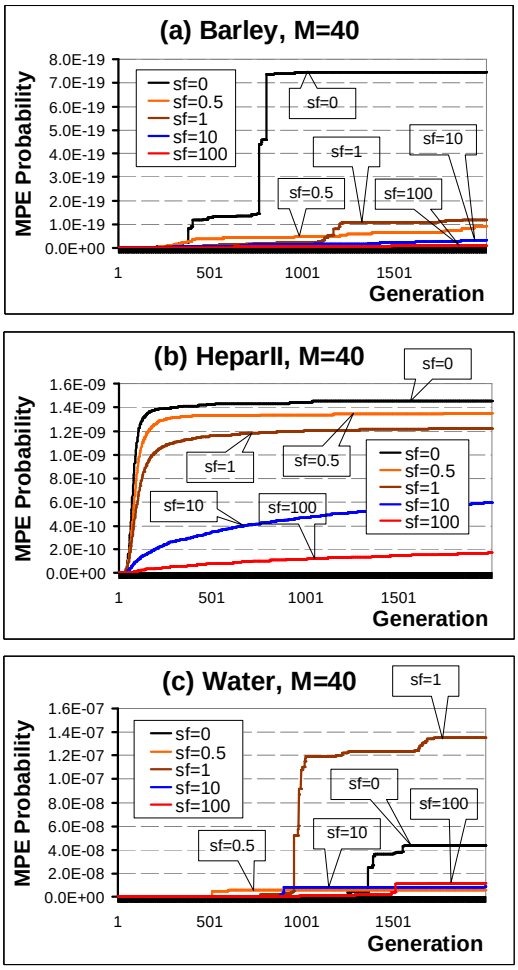

Figure 5: MPE estimation results for varying scaling factor (sf or ϕ) for three different Bayesian networks, population size $M = 40$.

The proper control of the scaling factor ϕ, depending on the problem being processed by Generalized Crowding, constitutes an important future research direction. This paper has analytically and empirically shown the importance of ϕ, including how its optimal value depends on the difficulty of the specific problem and population size. Since problem difficulty is usually not known in advance, adaptive or self-adaptive techniques could be used to control the value of ϕ in an on-line fashion. A second direction for future work is to build on the experiments reported here, where we have focused on MPE estimation. It would be interesting to consider the estimation of multiple highly probable explanations, in particular in situations where such explanations are spread throughout the search space.

7. ACKNOWLEDGEMENTS

This material is based, in part, upon work by Ole J. Mengshoel supported by NSF grants CCF-0937044 and ECCS-0931978. The anonymous reviewers are acknowledged for their comments, which helped improve the paper.

8. REFERENCES

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169, 1985.

[2] K. A. de Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1975.

[3] S. F. Galán and O. J. Mengshoel. Constraint handling using tournament selection: Abductive inference in partly deterministic Bayesian networks. *Evolutionary Computation*, 17(1):55–88, 2009.

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

[5] D. E. Goldberg. A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing. *Complex Systems*, 4:445–460, 1990.

[6] J. H. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor, MI, 1975. Second Edition, The MIT Press, Boston, MA, 1992.

[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[8] S. W. Mahfoud. Crowding and preselection revisited. In R. Männer and B. Manderick, editors, *Proceedings of the 2nd International Conference on Parallel Problem Solving from Nature (PPSN II)*, pages 27–36, Brussels, Belgium, 1992. Elsevier, Amsterdam, The Netherlands.

[9] S. W. Mahfoud. *Niching Methods for Genetic Algorithms*. PhD thesis, Department of General Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 1995.

[10] S. W. Mahfoud and D. E. Goldberg. Parallel recombinative simulated annealing: A genetic algorithm. *Parallel Computing*, 21:1–28, 1995.

[11] O. J. Mengshoel. *Efficient Bayesian Network Inference: Genetic Algorithms, Stochastic Local Search, and Abstraction*. PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, 1999.

[12] O. J. Mengshoel and D. E. Goldberg. Probabilistic crowding: Deterministic crowding with probabilistic replacement. In W. Banzhaf, J. M. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. J. Jakiela, and R. E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-1999)*, pages 409–416, Orlando, FL, 1999. Morgan Kaufmann, San Francisco, CA.

[13] O. J. Mengshoel and D. E. Goldberg. The crowding approach to niching in genetic algorithms. *Evolutionary Computation*, 16(3):315–354, 2008.

[14] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.