Distribution UC–405

SAND97–#
Unlimited Release
Printed February 1997

CONF-970740--1

On the Application of Evolutionary Pattern Search Algorithms

William E. Hart†
Applied and Numerical Mathematics Department
Sandia National Laboratories
Albuquerque, NM 87185-0111

Abstract

This paper presents an experimental evaluation of evolutionary pattern search algorithms (EPSAs). Our experimental evaluation of EPSAs indicates that EPSAs can achieve similar performance to EAs on challenging global optimization problems. Additionally, we describe a stopping rule for EPSAs that reliably terminated them near a stationary point of the objective function. The ability for EPSAs to reliably terminate near stationary points offers a practical advantage over other EAs, which are typically stopped by heuristic stopping rules or simple bounds on the number of iterations. Our experiments also illustrate how the rate of the crossover operator can influence the tradeoff between the number of iterations before termination and the quality of the solution found by an EPSA.

*This work was supported by the United States Department of Energy under Contract DE-AC04-94AL85000. Sanda is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the Unite States Department of Energy.

†Dept. 9222; email wehart@cs.sandia.gov.

1 Introduction

Evolutionary pattern search algorithms (EPSAs) are a class of evolutionary algorithms (EAs) that adapt the step size of the mutation operator to guarantee convergence to a stationary point of an objective function, where the gradient is zero, with probability one. Formally, the convergence theory guarantees that for a continuously differentiable function the sequence of best solutions in each generation, $\{x_k^*\}$, has the property that

$$P\left(\liminf_{k\to\infty} \| \bigtriangledown f(x_k^*)\| = 0\right) = 1,$$

where $\bigtriangledown f(x)$ is the gradient of f at x. Hart [6, 5] proves this convergence theory by demonstrating that EPSAs can be cast as stochastic versions of pattern search methods, which have recently been shown to have a weak stationary point convergence theory [13]. This result can be extended to prove convergence on continuous nondifferentiable functions, where the limit points of an EPSA include points where the gradient does not exist or where the gradient is not continuous [6].

While EAs are typically described as methods for global optimization, this convergence analysis does not guarantee that the global optimum is found. However, the constraints imposed on EPSAs to ensure converge to a stationary point provide insight into the requirements needed to converge to a global optimum. For example, the convergence analysis of pattern search methods highlights the fact that the rate of convergence of EPSAs is likely to decrease as the dimension of the problem increases [13]. This analysis also demonstrates that a poor choice of the scale and orientation of the search steps taken by an EA can adversely affect its performance.[1] Proofs that EAs converge to a global optimum typically rely in weaker analyses like the Borel-Cantelli lemma, which fail to provide insights like this about the factors that influence the behavior of these algorithms.

Aside from theoretical considerations, experience with pattern search algorithms suggests that EPSAs can be successfully applied to a wide range of optimization problems [12, 7, 14]. For example, Meza and Martinez [7] have successfully applied a pattern search method, parallel directed search, to the global optimization of the conformational energy of a simple chain molecule. Their comparison of parallel directed search to a genetic algorithm and simulated annealing algorithm indicates that parallel directed search can be equally effective at performing global optimization for this problem.

We present an experimental comparison between EPSAs and EAs that confirms that EPSAs and EAs find equally good solutions on several standard global optimization test functions. Furthermore, we describe a stopping rule for EPSAs that reliably terminates when the best solution found is in the neighborhood of a stationary point. Because of the constrained manner in which EPSAs adapt the mutation operator's step length, this stopping rule provides a greater confidence that the algorithm has converged to a minimum than stopping rules that terminate an EA when the distribution of solutions in a population has converged to a small neighborhood of the search space.

We also consider how the choice of the mutation operator can influence the expected number of iterations between updates to the mutation operator's step length. This analysis suggests that EPSAs using mutation operators based on a multinomial selection of the mutation point will converge more quickly than EPSAs using mutation operators based on

[1] This type of performance degradation is not limited to pattern search algorithms. The empirical results in Salamon [10] coroborate that EAs can exhibit a performance degradation when a simple coordinate rotation is applied to some standard test functions.

a binomial selection of the mutation point(s). This analysis also incorporates the effect of crossover on the expected number of iterations between updates, which are consistent with our experimental data.

2 Evolutionary Pattern Search Algorithms

This section reviews the definition of EPSAs given in Hart [5], which extends the definition of EPSAs in Hart [6]. The definition of EPSAs is motivated by the similarity between between EAs and pattern search methods. Pattern search methods use an exploratory moves algorithm to conduct a series of exploratory moves about the current iterate before identifying a new iterate. These moves can be viewed as a search about the current iterate for a trial point with a lower function value. Exploratory moves are generated by the exploratory moves algorithm using a pattern matrix which is constrained to contain a set of $2n$ columns of the form $[-M\ M]$, where M is a nonsingular matrix in $\mathbf{Z}^{n\times n}$, which define the *basic steps* of the pattern search method.[2] The stochastic pattern search methods described in Hart [5] use an exploratory moves algorithm that is only probablistically guaranteed to terminate. To ensure convergence, the probability of selecting each of the basic steps is bounded below by $\nu > 0$.

The design of EPSAs incorporates restrictions which enable EPSAs to be cast as stochastic pattern search methods. This correspondence is established by viewing the best individual in each generation of an EPSA as the current iterate. The vectors in the pattern matrix correspond to the vectors from the best individual to each of the possible individuals that can be generated from the current population. Figure 1 shows the pseudo code describing EPSAs. Each iteration of the loop indexed by h corresponds to an iteration of an exploratory moves algorithm for a stochastic pattern search algorithm. The restriction on the replacement strategy ensures that the best individual found is kept for further processing.

The crossover operator generates individuals using coordinate-wise swaps, and the mutation operator adds a discrete random variable to a dimension of an individual, which may assume values $\{\pm\sigma_1, \ldots, \pm\sigma_m\}$, $\sigma_i \in \mathbf{Q}$ and $\sigma_i > \sigma_1 = 1$. Let ξ be probability that a mutation with value ± 1 is generated. Let e_i be the standard unit vector in the i-th dimension. Let $\eta \in \{0,1\}^{2n}$ be a counter that indicates whether the $2n$ vectors $\pm\sigma_1 e_i$ have been examined. The contraction factor is $\theta = 1/a$ and the expansion factor is $\lambda_k \in \{1, a\}$, $a \in \mathbf{Z}$.

3 Algorithmic Design

The abstract definition of EPSAs provides a lot of flexibility for the different implementations of these algorithms. This section discusses two algorithmic issues that affect the number of iterations required by an EPSA: (1) the mutation operator and (2) the stopping rule.

3.1 Mutation Operators

The following lemma bounds the expected length of time spent by the exploratory moves algorithm of a stochastic pattern search method. Because of the correspondence between EPSAs and stochastic pattern search, this lemma bounds the expected length of time spent in the inner loop of an EPSA. Recall that the probability of selecting each of the $2n$ trial steps is greater than or equal to $\nu > 0$.

[2] We use the notation $X = [Y\ Z]$ to denote that the matrix X is partitioned into the matrices Y and Z.

```
1  Initialize the population of individuals {x_1^0, ..., x_N^0} such that x_i^0 ∈ Q^n
2  Compute the fitness of each individual and reorder such that x_1^0 ≤ x_i^0, ∀i
3  Let t = 0 and η = {0}^{2n}
4  Repeat k = 0, 1, ...
5     Repeat h = 1, 2, ...
6        Perform selection with a policy which guarantees that the
            individual with the smallest fitness value is selected for processing
            at each generation with a probability of at least π > 0
7        Perform crossover with probability χ ∈ [0, 1⟩
8        Perform mutation with probability μ ∈ ⟨0, 1] of mutating each dimension;
            mutation adds a vector of the form Δ_k ω, ω ∈ {0, ±σ_1, ..., ±σ_m}^n
9        If the mutation x_1^t + Δ_k σ_1 e_i is generated, then η_i = 1
         Else, if the mutation x_1^t − Δ_k σ_1 e_i is generated, then η_{i+n} = 1
10       Compute the fitness of each individual and reorder such that x_1^t ≤ x_i^t, ∀i
11       Perform replacement with an elitist replacement policy which guarantees that
            the individual with the smallest fitness value among the previous population
            and the newly generated individuals is included in the next generation.
12       t = t + 1; x_t^* = arg min{f(x_{t-1}^*), f(x_1^t), ..., f(x_N^t)} and y_t^* = f(x_t^*)
13    Until (y_t^* < y_{t-1}^*) or (|η| = 2n)
14    If (|η| = 2n) then Δ_{k+1} = θ Δ_k
15    Else Δ_{k+1} = λ_k Δ_k
16    η = {0}^{2n}
17 Until some stopping criterion is satisfied
```

Figure 1: **Pseudo Code for EPSAs**

Lemma 1 Let Y_n be the number of iterations spent in the stochastic exploratory moves algorithm. Then

$$E(Y_n) \leq \frac{1}{\nu} \sum_{i=1}^{2n} \frac{1}{k} = \frac{1}{\nu} O(\log n).$$

Proof. Consider each iteration of the exploratory moves algorithm as a sample with replacement from a set of exploratory moves. In the worst case, the exploratory moves algorithm terminates when all of the basic steps have been sampled.

Let $S_{k,2n}$ be the number of samples until k of the $2n$ elements are sampled, and let $X_{k,2n} = S_{k,2n} - S_{k-1,2n}$. $X_{k,2n} - 1$ is the number of elements sampled between the $(k-1)$th new basic step and the kth new basic step. This value is geometrically distributed with parameter $p \geq \nu(2n - k + 1)$. Thus

$$E(X_{k,2n}) = 1 + \frac{1-p}{p} = \frac{1}{p} \leq \frac{1}{\nu(2n-k+1)}.$$

Now $S_{2n,2n}$ is the number of samples required to ensure that all $2n$ trial steps have been sampled. Since $S_{2n,2n} = X_{2n,2n} + \ldots + X_{1,2n}$, we have

$$E(Y_n) = E(S_{2n,2n}) \quad \leq \quad \frac{1}{\nu} \sum_{k=1}^{2n} \frac{1}{2n-k+1}$$

$$= \frac{1}{\nu}\sum_{k=1}^{2n}\frac{1}{k}$$
$$= \frac{1}{\nu}O(\log n).$$

■

In general, it is not possible to determine the value of ν, even if the mutation and crossover rates are provided. The reason is that selection mechanism can greatly influence the probability that the best individual in the population is selected. However, for rank selection the value of ν is easily computed.

Let r_i be the rank of the ith individual in the population, such that the best rank is 0 and the worst rank is $P-1$, where P is the population size. Linear ranking [4] assigns a probability of selection to an individual x_i according to the equation

$$p(x_i) = \frac{C - 2(C-1)r_i/(P-1)}{P},$$

where $1 \leq C \leq 2$. It follows that the probability of selecting the best individual in a population is at least C/P (it could be greater if multiple copies of the best individual exist).

We use rank selection to consider the effects of two different discrete mutation operators. The *binomial* mutation operator is the analogue of the mutation operator commonly used in binary GAs. The probability of mutating a given dimension is μ, and given that a mutation occurs the probability of mutating by a particular value σ_j is $1/(2m)$, where m is the number of different mutation offsets. The *multinomial* mutation operator mutates an individual with probability μ. Given that mutation occurs the dimension mutated is selected with probability $1/n$ and the probability of mutating by σ_j is $1/(2m)$. Multinomial mutation is introduced here because it has a much higher probability of generating singleton mutations that are used by EPSAs to control the step length parameter.

For EPSAs using binomial mutation, the probability of generating each basic step is C/P times the probability that crossover is not applied and mutation generates a basic step, or crossover is applied and the individuals selected to perform crossover are both best individual in the population:

$$\nu \geq \mu(1-\mu)^{n-1}\frac{(1-\chi)CP + \chi C^2}{2mP^2}.$$

Similarly, the probability that multinomial mutation generates a basic step is

$$\nu \geq \mu\frac{(1-\chi)CP + \chi C^2}{2mnP^2}$$

The following corollary applies Lemma 1 to bound the length of the exploratory moves for EPSAs that use these mutation operators.

Corollary 1 Let an EPSA use linear rank selection with parameter C. If binomial mutation is used, then

$$E(Y_n) \leq \frac{2mP^2}{\mu(1-\mu)^{n-1}((1-\chi)CP + \chi C^2)}O(\log n)$$

If multinomial mutation is used, then

$$E(Y_n) \leq \frac{2mP^2}{\mu((1-\chi)CP + \chi C^2)}O(n\log n)$$

Let μ_b be the mutation rate used for binomial mutation, and let μ_m be the mutation rate used for multinomial mutation. The bound on the expected length of the exploratory moves algorithm will be shorter for EPSAs using binomial mutation if

$$\frac{1}{\mu_b(1-\mu_b)^{n-1}} < \frac{n}{\mu_m}.$$

If $\mu_b = \mu_m$, this implies that $\mu_m = \mu_b < 1 - n^{-1/(n-1)}$. Even for moderately large n, the expected length of the exploratory moves algorithm for EPSAs using binomial mutation will be shorter. Intuitively, the reason for this is that the expected number of mutations per individual is $\mu_b n$ for binomial mutation but only μ_m for multinomial mutation. If we equalize the expected number of mutations, then the expected length of the exploratory moves algorithm will *always* be shorter for EPSAs using multinomial mutation.

3.2 Stopping Rules

The proof of convergence for EPSAs relies soley on the properties of the mutation operator. However, the constrained nature of the exploratory moves should make stopping rules for EPSAs applicable in practice. Torczon [11] discusses a number of stopping rules for direct search methods that might be applicable to EPSAs. These stopping rules cannot guarantee that the algorithm is terminated at a stationary point because the gradient is not available and is not explicitly estimated. Instead, these stopping rules measure whether the algorithm has "ground to a halt," either because it has stalled or converged.

The stopping rules discussed by Torczon [11] are applied to direct search methods for which all points in the simplex of possible new points are examined. In generalized pattern search methods, the stopping rule is checked when the exploratory moves algorithm fails to find an improving search direction. At this point, we are only guaranteed to have examined the basic steps. Consequently, the simplex used to evaluate the stopping rule for pattern search methods is simply the simplex of basic steps.

Let $v_1^t, \ldots, v_{2n}^t$ be the $2n$ basic steps in a pattern search algorithm at time t, such that $f(v_1^t) \leq f(v_i^t)$ for all i. The stopping rule recommended by Torczon [11] terminates the algorithm when

$$\frac{1}{\delta} \max_{1 \leq i \leq 2n} \left| v_i^t - v_j^t \right| \leq \epsilon,$$

where $\delta = \max(1, |v_1^t|)$. The division by δ makes this stopping rule terminate based on the relative change of the step lengths. Dennis and Schnabel [2] note that this this type of stopping rule may be satisfactory if the change in the step lengths is greater than one, but will be unsatisfactory if the change in the step lengths is always much smaller than 1. Consequently, we use $\delta = \max(\xi, |v_1^t|)$ for some suitably small value of ξ. Dennis and Schnabel [2] suggest a guideline for choosing the value of the step tolerance ϵ: if p significant digits are desired, then set ϵ to 10^{-p}.

For EPSAs, $\max_{1 \leq i \leq 2n} \left| v_i^t - v_j^t \right| \leq 2\Delta_k$, so it suffices to stop when

$$\frac{2\Delta_k}{\max(\xi, |v_1^t|)} \leq \epsilon,$$

which is essentially a stopping rule based on the value of the step length Δ_k.

4 Experimental Evaluation

The experiments described in this section offer some preliminary evidence about the performance of EPSAs. These experiments address a number of practical questions concerning the application of EPSAs:

- How important is an adaptive step size? If EAs using fixed step lengths perform as well as EPSAs, then there may be little need for the machinery needed to adapt the step length.
- What is the effect of the rate of crossover on the total cost of EPSAs? Our analysis in Section 3.1 enables us to bound the expected cost of the inner loop of EPSAs, which is one factor of the total cost. Other factors such as the rate of convergence are related to the role of crossover in EAs. This question also addresses the ability of the stopping rule to terminate in the presence of crossover steps. Since the stopping rule depends upon the fitness landscape with respect to mutation steps, it is not clear whether or not the stopping rule would be less useful for EPSAs using crossover.
- How does the performance of EPSAs compare with that of canonical EAs? Specifically, what is the effect of using a discretized mutation operator in EAs?
- How well do the stopping rules work? Do they terminate EPSAs near critical points of the search space?

Three standard test functions where used in our experiments:

- Ackley's function [1]:

$$\begin{aligned} g(\bar{x}) &= -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) \\ &\quad -\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e \end{aligned}$$

  for $x_i \in [-30.0, 30.0]$.

- Griewank's function [8]:

$$h(\bar{x}) = \sum_{i=1}^{n} x_i^2/4000 - \prod_{i=1}^{n} x_i/\sqrt{i} + 1$$

  for $x_i \in [-600.0, 600.0]$.

- Rastrigin's function [8]:

$$f(\bar{x}) = 10n + \sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i)\right),$$

  for $x_i \in [-5.12, 5.12]$.

These functions are highly multimodal, and they all have a minimum energy of 0 at $\bar{x} = 0$.

To normalize the effects of mutation, the vectors were scaled by a constant factor to make the search domain for each dimension of these functions $[-100, 100.0]$. Similarly, these problems were tested for $n = 10$ to normalize the effects of the dimension on the performance of the EAs.

The EA used in the experiments is a real-coded GA that uses a binomial mutation operator that adds a normally distributed random deviate to the dimensions of an individual that are mutated. The EPSA used in the experiments initializes individuals by discretizing the range of each dimension into 10^5 values. A multinomial mutation operator is used to add a value of Δ_k or $-\Delta_k$ to a dimension of an individual. The mutation rate of the EPSA was 1.0, and the mutation rate of the EA was 0.1, so the expected number of mutations per solution was the same for both algorithms. The initial step length used in the experiments is $\Delta_0 = 20$. The contraction factor is $\theta = 1/2$ and the step length is not expanded. The value $\xi = 10^{-8}$ is used by the stopping rule, and the threshold is $\epsilon = 10^{-4}$.

In all experiments, the EPSA and EAs use a population size of 100 and results are averaged over 30 different seeds for the random number generator. Linear rank selection is used to select individuals for recombination and mutation. Elitist replacement is used to ensure that the best individual seen is kept in subsequent populations. Efforts were made to reduce the total number of function evaluations by keeping track of individuals that were generated from identical parents and individuals that were simply copied to the next population (e.g. the elitist individual). Furthermore, efforts were made to determine whether there were multiple copies of the best individual in a given population, which was used to accelerate the collection of statistics that were used by EPSA to modify the step length.

4.1 Effects of Adaptive Step Sizes

This experiment compares the performance of the EPSA to the performance of EAs that use a fixed step length for the mutation operator. Fixed step lengths of 1.0, 10.0 and 20.0 were used, and comparisons were made with no crossover and high crossover probabilities. Table 1 summarizes the final performance of the EPSA and EAs. When $\chi = 0.0$, the EAs were terminated after 250000, 100000 and 100000 function evaluations for Ackley, Griewank and Rastrigin's function respectively. When $\chi = 0.8$, the EAs were terminated after 850000, 400000 and 400000 function evaluations respectively. These bounds on the number of function evaluations are an upper bound on the average number of function evaluations used by the EPSA averaged over the 30 trials. These results suggest that there is a best fixed step size for the EAs. However, it is clear that the adaptation of step sizes by the EPSA allows it to find better solutions.

4.2 Effects of Crossover

Our analysis in Section 3.1 enables us to bound the expected cost of each outer iteration of EPSAs, which is one factor of the total cost. Specifically, this analysis predicts that in the worst case, a crossover rate of χ will multiply the number of iterations used by the EPSA by a factor of approximately $1/(1-\chi)$. This analysis does not incorporate the effects that crossover may have on the convergence rate of the EPSA, since it is a worst case analysis.

Table 2 compares the performance of the EPSA when crossover is not used $\chi = 0.0$, and when crossover is used frequently, $\chi = 0.8$. For this experiment, the analysis predicts

		Ackley			Griewank			Rastrigin		
		Best	Mean	Worst	Best	Mean	Worst	Best	Mean	Worst
$\chi = 0.0$	EPSA	0.000	0.000	0.000	0.007	0.130	0.374	2.985	6.069	11.940
	$\Delta = 1$	0.463	8.975	18.362	0.391	0.662	1.009	18.459	50.647	86.089
	$\Delta = 10$	3.615	4.785	5.967	1.398	1.771	2.197	14.435	32.455	48.722
	$\Delta = 20$	5.205	7.235	8.598	2.566	3.707	5.595	32.086	57.468	89.903
$\chi = 0.8$	EPSA	0.000	0.000	0.000	0.010	0.050	0.128	0.000	2.089	4.975
	$\Delta = 1$	0.372	0.707	1.061	0.298	0.606	1.011	3.412	14.447	28.357
	$\Delta = 10$	3.409	4.554	5.136	1.269	1.683	2.137	7.625	20.269	37.804
	$\Delta = 20$	4.971	6.908	8.580	1.962	3.601	5.036	6.578	16.671	31.442

Table 1: Comparison of performance for the EPSA with adaptive step length against EPSAs with fixed step length Δ.

χ	Ackley	Griewank	Rastrigin
0.0	217680	86927	67793
0.8	826418	288478	358540

Table 2: Comparison of the average number of function evaluations for EPSAs when $\chi = 0.0$ and $\chi = 0.8$.

that the EPSA using crossover will require on average five times as many generations (calls to the exploratory moves algorithm) as the EPSA that does not use crossover. For the Rastrigin function, the ratio between the average number of function evaluations used by the EPSAs is remarkably close to this prediction, which suggests that introducing crossover does not accelerate the EPSA's convergence to a solution. Conversely, for the Ackley and Griewank functions, the ratio is less than five. In both cases, the use of crossover improves the average performance of the EPSAs. The use of a stopping rule to terminate the EPSAs clarifies the relationship between the time spent searching and the quality of the solution found. The EPSAs using crossover find better solutions at the expense of additional function evaluations.

4.3 EPSAs vs EAs

This experiment compares the performance of the EPSA with and without crossover to the performance of the EA, with and without crossover. The EAs were terminated after the same number of function evaluations used in Section 4.1. Table 3 summarizes the results of this experiment.

These results indicate that the relative performance of EAs and EPSAs depends upon the objective function that is minimized as well as the standard deviation of the mutation operator used by the EA. None of the three algorithms compared in this experiment is clearly superior to the others. Furthermore, in all cases the performance of the EPSA is reasonably close to the performance of the best of the three algorithms.

Although an analysis of the search dynamics of these algorithms is beyond the scope of this paper, we offer the following explanation for their relative performance on these problems.

Ackley An inspection of the rate of convergence of the EPSA and EAs shows that all of the algorithms quickly find solutions with values below 1.0, but that the EPSAs

		Ackley			Griewank			Rastrigin		
		Best	Mean	Worst	Best	Mean	Worst	Best	Mean	Worst
EPSA	0.0	0.000	0.000	0.000	0.007	0.130	0.374	2.985	6.069	11.940
	0.8	0.000	0.000	0.000	0.010	0.050	0.128	0.000	2.089	4.975
EA-20	0.0	0.030	0.107	0.190	0.147	0.483	1.001	0.036	0.269	1.067
	0.8	0.058	0.132	0.230	0.230	0.439	0.963	0.038	0.182	0.735
EA-1	0.0	0.003	9.222	17.934	0.017	0.086	0.173	17.910	50.445	85.567
	0.8	0.002	0.005	0.011	0.011	0.040	0.089	2.985	13.466	29.850

Table 3: Comparison of performance for EPSAs with EAs. EA-α denotes results for an EA using a normal mutation operator with standard deviation α.

continue to find improving solutions while the performance of the EAs plateaus. This suggests that the EPSA's ability to guarantee improving steps through the constrained reduction of the step length plays an important role for this function.

Griewank The rate of convergence shows that the EAs with standard deviation of 1.0 quickly identified solutions with values below 1.0 (despiter their small step size) and continued to refine them (albeit slowly). The EPSAs were also able to find solutions with low values, but their performance plateaus and they converge to nonoptimal minima near the global optima. Since there are many minima around the global minima that all have similar values, we think that the variable nature of the EA's mutation operator allows it to continue to make progress by jumping to neighboring minima, thereby continuing to make progress even if it's population has converged to a small neighborhood.

Rastrigin With the exception of the EAs with standard deviation 20.0, all of the algorithms plateau at nonoptimal minima. Rastrigin has fewer minima than the Griewank function, that are correspondingly wider. The comparison between the EAs with standard deviation 1.0 and 20.0 indicate that having a large step size is crucial to locating the global optimum, perhaps because it allows the EA to jump from minima to minima. This would also explain why the EAs with standard deviation 20.0 outperform the EPSAs, which get trapped in nonoptimal minima.

4.4 Efficacy of the Stopping Rule

To evaluate the efficacy the stopping rules, I refined the final solutions from the optimization trials generated by the EPSA with no crossover. The final solutions were minimized using conjugate gradient [9], a gradient-based optimization method. The refined solutions were compared to the final solutions generated by the EPSA. In all cases, the final solutions were within the epsilon tolerance (10^{-4}) of the local optima.

5 Discussion

Our experimental evaluation of EPSAs indicates that EPSAs can achieve similar performance to EAs on challenging global optimization problems. No algorithmic configuration of the EPSA and EAs was clearly superior in our experiments, which suggests that EPSAs should be considered for optimization problems on $\mathbf{R}^n$. In fact, the ability for EPSAs to

reliably terminate near stationary points offers a practical advantage over other EAs, which are typically stopped by a bound on the number of iterations that is specified by the user.

The fact that EPSAs can perform as well as EAs perhaps surprising since EPSAs cannot in general perform a global optimization of an objective function. Instead, EPSAs are better described as *nonlocal optimizers* since they perform nonlocal search through the use of the crossover operator and the stochastic competition between disparate solutions. This raises the question of whether EAs are better described as global or nonlocal optimizers. Although global convergence for many EAs can be shown using the Borel-Cantelli lemma, once a population has converged to a subregion the expected time needed to search all neighborhoods outside of that subregion is so long that for all practical purposes the EA no longer performs a global search. Although further study is needed to resolve this question, experience with the CHC algorithm [3] suggests that treating EAs as nonlocal optimizers can be advantageous. The CHC algorithm uses an EA that cannot in general perform a global optimization since it does not include a mutation operator, but Eschelman [3] demonstrates that the randomized restart mechanism used by CHC allows it to apply the EA multiple times to perform global optimization more effectively than traditional GAs.

The experiments we have described can be extended in a number of ways to provide a more complete analysis of EPSAs. First, a greater variety of test functions need to be considered to provide a careful characterization of the relative performance of EPSA and EAs. Second, other stopping rules need to be evaluated for EPSAs. In particular, it would be useful to try and utilize the information generated by the crossover steps to terminate the algorithm, especially when the population appears to have converged. Also, it is important to consider other stopping rules which include the measure of the relative change in the objective function as well as the magnitude of the step length. Next, an experimental comparison of the effects of different types of mutation operators on the EPSA's performance is needed to confirm the analysis is Section 3.1. Additionally, a further analysis of mutation is needed to determine what an appropriate rate of mutation is for EPSAs. The effects of the problem dimension on the performance of EPSAs also need to be considered. Although the analysis in Torczon [13] indicates that the rate of convergence in EPSAs will slow with increasing dimension, it is unclear whether the relative rates of EPSA and EAs will remain the same. Finally, although the convergence analysis in Hart [5] allows the step lengths to expand, our experiments have used EPSAs that only decrease the step length. It remains to determine what is a *good* way of controlling the balance between increasing and decreasing the step length. Preliminary experiments have shown that increases in the step length should be tied to improvements due to points that are generated by mutation alone [5].

References

[1] D. H. Ackley. *A Connectionist Machine for Genetic Hillclimbing.* Kluwer Academic Publishers, Boston, 1987.

[2] J. J. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, 1983.

[3] L. J. Eshelman. The chc adaptive search algorithm: How to have a safe search when engaging in nontraditional genetic recombination. In G. J. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 265–285. Morgan-Kaufmann, 1991.

[4] D. E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In G. J. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 301–315. Morgan-Kauffmann, San Mateo, CA, 1991.

[5] W. E. Hart. A generalized stationary point convergence theory for evolutionary algorithms. In *ICGA97*, 1997. (submitted).

[6] W. E. Hart. A stationary point convergence theory for evolutionary algorithms. In *Foundatations of Genetic Algorithms 4*, 1997. (to appear).

[7] J. Meza and M. L. Martinez. Direct search methods for the molecular conformation problem. *Journal of Computational Chemistry*, 15(6):627–632, 1994.

[8] H. Mühlenbein, M. Schomisch, and J. Born. The parallel genetic algorithm as function optimizer. *Parallel Computing*, 17:619–632, 1991.

[9] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipies in C - The Art of Scientific Computing.* Cambridge University Press, 1990.

[10] R. Salomon. Performance degradation of genetic algorithms under coordinate rotation. In L. J. Fogel, P. Angeline, and T. Bäck, editors, *Proc. of the Fifth Annual Conf. on Evolutionary Programming*, pages 155–161, Cambridge, Massachusetts, 1996. MIT Press.

[11] V. Torczon. Multi-directional search: A direct search algorithm for parallel machines. Technical Report TR90-7, Rice University, May 1989.

[12] V. Torczon, May 1996. Personal Communication.

[13] V. Torczon. On the convergence of pattern search methods. *SIAM J Optimization*, 7(1):1–25, Feb 1997.

[14] M. Wright. Direct search methods: Once scorned, now respected. Technical Report 96-4-01, Computing Science Research Center, AT&T Bell Laboratories, 1996.