Fast Evolutionary Programming

Xin Yao and Yong Liu
Computational Intelligence Group, School of Computer Science
University College, The University of New South Wales
Australian Defence Force Academy, Canberra, ACT, Australia 2600
Email: xin@csadfa.cs.adfa.oz.au, WWW: http://www.cs.adfa.oz.au/~xin

Abstract

Evolutionary programming (EP) has been applied to many numerical and combinatorial optimisation problems successfully in recent years. One disadvantage of EP is its slow convergence to a good near optimum for some function optimisation problems. In this paper, we propose a fast EP (FEP) which uses a Cauchy instead of Gaussian mutation operator as the primary search operator. The relationship between FEP and classical EP (CEP) is similar to that between the fast simulated annealing and the classical version. Extensive empirical studies have been carried out to evaluate the performance of FEP for different function optimisation problems. Fifty runs have been conducted for each of the 23 test functions in our studies. Our experimental results show that FEP performs much better than CEP for multi-modal functions with many local minima while being comparable to CEP in performance for unimodal and multi-modal functions with only a few local minima. We emphasise in the paper that no single algorithm can be the best for *all* problems. What we need is to identify the relationship between an algorithm and a class of problems which are most amenable to the algorithm.

1 Introduction

Although evolutionary programming (EP) was first proposed as an evolutionary approach to artificial intelligence [1], it has been recently applied to many numerical and combinatorial optimisation problems successfully [2, 3, 4]. Optimisation by EP can be summarised into two major steps:

1. Mutate all the solutions in the current population, and
2. Select the next generation from the mutated and the current solutions.

These two steps can be regarded as a population-based variant of the classical generate-and-test algorithm. The advantage of reformulating EP as a special variant of the generate-and-test algorithm is that we can study EP along with other popular search algorithms, and thus encourage cross-fertilisation among different research areas.

One disadvantage of EP in solving some of the high-dimensional optimisation problems is its slow convergence to a good near optimum, e.g., f_8 to f_{13} studied in this paper. The generate-and-test formulation of EP indicates that we can either improve EP's mutation operator or its selection scheme in order to improve EP's performance. This paper concentrates on the mutation operator. A new mutation operator based on Cauchy random numbers is proposed and tested on a suite of 23 functions. The new version outperforms the classical EP, which uses Gaussian mutations, on multimodal functions with many local minima while being comparable to the classical EP for unimodal and multimodal functions with only a few local minima. We describe the new EP as fast EP (FEP) while the EP proposed by Fogel [2] as the classical EP (CEP) in this paper. The motivation of introducing Cauchy mutation into EP comes from fast simulated annealing [5, 6, 7].

We have carried out many empirical studies of both FEP and CEP in order to evaluate the relative strength and weakness of FEP and CEP for different problems. Such results show that Cauchy mutation is an efficient mutation operator for at least a large class of multimodal function optimisation problems. FEP's performance can be expected to improve further since all the parameters used in the FEP were set the same as those used in CEP in order to isolate the effect of the Cauchy mutation operator. Suitable parameter settings for CEP may not be suitable for FEP.

The rest of this paper is organised as follows. Section 2 describes the global minimisation problem considered in this paper and the CEP used to solve them. The CEP algorithm given follows suggestions from Fogel [3, 8] and Bäck *et al.* [9]. Section 3 describes the FEP and its implementation. Section 4 gives the 23 functions used in our studies. Section 5 presents our experimental results and discussions. Finally, Section 6 concludes with some remarks and future research directions.

2 Function Optimisation By CEP

A global minimisation problem can be formalised as a pair (S, f), where $S \subseteq R^n$ is a bounded set on R^n and $f : S \mapsto R$ is an n-dimensional real-valued function. The problem is to find a point $x_{min} \in S$ such that $f(x_{min})$ is a global minimum on S. More specially, it is required to find an $x_{min} \in S$ such that

$$\forall x \in S : f(x_{min}) \leq f(x)$$

Here f does not need to be continuous but it must be bounded. We only consider unconstrained function optimisation in this paper.

Fogel [3] and Bäck *et al.* [9] have indicated that CEP with adaptive mutation usually performs better than CEP without adaptive mutation. Hence we will consider the CEP with adaptive mutation in this paper. According to the description by Bäck *et al.* [9], the CEP is implemented as follows in our studies:

1. Generate the initial population of μ individuals, and set $k = 1$. Each individual is taken as a pair of real-valued vectors, (x_i, η_i), $\forall i \in \{1, \cdots, \mu\}$.

2. Evaluate the fitness score for each individual (x_i, η_i), $\forall i \in \{1, \cdots, \mu\}$, of the population based on the objective function, $f(x_i)$.

3. Each parent (x_i, η_i), $i = 1, \cdots, \mu$, creates a single offspring (x_i', η_i') by: for $j = 1, \cdots, n$,

$$x_i'(j) = x_i(j) + \eta_i(j)N(0,1), \quad (1)$$
$$\eta_i'(j) = \eta_i(j)\exp(\tau' N(0,1) + \tau N_j(0,1)) \quad (2)$$

   where $x_i(j)$, $x_i'(j)$, $\eta_i(j)$ and $\eta_i'(j)$ denote the j-th component of the vectors x_i, x_i', η_i and η_i', respectively. $N(0,1)$ denotes a normally distributed one-dimensional random number with mean zero and standard deviation one. $N_j(0,1)$ indicates that the random number is generated anew for each value of j. The factors τ and τ' have commonly set to $\left(\sqrt{2\sqrt{n}}\right)^{-1}$ and $\left(\sqrt{2n}\right)^{-1}$ [9, 8].

4. Calculate the fitness of each offspring (x_i', η_i'), $\forall i \in \{1, \cdots, \mu\}$.

5. Conduct pairwise comparison over the union of parents (x_i, η_i) and offspring (x_i', η_i'), $\forall i \in \{1, \cdots, \mu\}$. For each individual, q opponents are chosen randomly from all the parents and offspring with an equal probability. For each comparison, if the individual's fitness is no greater than the opponent's, it receives a "win."

6. Select the μ individuals out of (x_i, η_i) and (x_i', η_i'), $\forall i \in \{1, \cdots, \mu\}$, that have the most wins to be parents of the next generation.

7. Stop if the stopping criterion is satisfied; otherwise, $k = k + 1$ and go to Step 3.

3 FEP

In order to investigate the impact of the Cauchy mutation operator on EP, we hold everything in FEP constant as that in CEP except for the mutation operator. The one-dimensional Cauchy density function centred at the origin is defined by:

$$f_t(x) = \frac{1}{\pi}\frac{t}{t^2 + x^2}, \quad -\infty < x < \infty,$$

where $t > 0$ is a scale parameter [10](pp.51). The corresponding distribution function is

$$F_t(x) = \frac{1}{2} + \frac{1}{\pi} arctan\left(\frac{x}{t}\right).$$

The shape of $f_t(x)$ resembles that of the Gaussian density function but approaches the axis so slowly that an expectation does not exist. As a result, the variance of the Cauchy distribution is infinite. Some studies [11] have indicated the benefit of increasing the variance in Monte-Carlo algorithms, which can be regarded as a type of generate-and-test algorithms. Fast simulated annealing is an example where an increased variance improves the efficiency of the global search. The reason for such improvement is often explained as an increased probability of escaping from a local optimum. However, it is unknown whether there exists an optimal distribution for a function optimisation problem.

The FEP studied in this paper is exactly the same as the CEP described in Section 2 except for Eq.(1) which is replaced by the following:

$$x_i'(j) = x_i(j) + \eta_i(j)\delta_j \quad (3)$$

where δ_j is an Cauchy random number variable with the scale parameter $t = 1$, and is generated anew for each value of j. It is worth indicating that we leave Eq.(2) unchanged in FEP in order to keep our modification of CEP to a minimum. η in FEP plays the role of the scale parameter t not the variance in the Cauchy distribution. It is not unreasonable to expect improvement of FEP's performance if a different mutation scheme for η is used in FEP since Eq.(2) is designed to work with the Gaussian not Cauchy mutation operator.

4 Test Functions

We use 23 well-known functions [2, 12, 13, 9] in our experimental studies. This number is larger than the number used in most other papers. However, we believe it is necessary. Our purpose is not to show FEP is better or worse than CEP, but to find out when FEP is better (or worse) than CEP for what kind of problems. Wolpert and Macready [14] have proved that no single search algorithm is best on average for all problems. If the number

of test problems is small, it would be very difficult to make a generalised conclusion. It also has the potential risk that the algorithm is biased (optimised) towards the small number of test problems while such bias might not be useful for other problems of interest.

The 23 test functions used in our experiments are listed in Table 1. Functions f_1 to f_{13} are high-dimensional problems. Functions f_1 to f_5 are unimodal functions. Function f_6 is the step function which has one minimum and is discontinuous. Function f_7 is a noisy quartic function, where $random[0, 1)$ is a uniformly distributed random variable in $[0, 1)$. Functions f_8 to f_{13} are multimodal functions where the number of local minima increases exponentially with the problem dimension [12, 13]. Functions f_{14} to f_{23} are low-dimensional functions which have only a few local minima [12]. For unimodal functions, we are most interested in the convergence rate of FEP and CEP. We are less interested in the final results of the optimisation as there are other methods which are specifically designed to optimise unimodal functions. For multimodal functions, we are most concerned with the issue whether an algorithm can find a better solution in a shorter time.

5 Experimental Studies

5.1 Experimental Setup

For all experiments; the same self-adaptive method (i.e., Eq.(2)), the same population size $\mu = 100$, the same tournament size $q = 10$ for selection, the same initial standard deviations 3.0, and the same initial population, were used.

5.2 Unimodal Functions

The first experiment was aimed to compare the convergence rate of CEP and FEP for Functions f_1 to f_7. The progress of the mean best solutions and the mean of average values of population found by CEP and FEP over 50 runs for each function is shown in Figures 1–2. The average results are summarised in Table 2. It is clear that FEP performs better than CEP in terms of convergence rate. Note that the solutions found by CEP are better than those found by FEP for Functions f_1 and f_2 in the final stage. This seems to suggest that FEP has better global search capability while CEP is better at local fine-tuning.

The largest difference in performance between CEP and FEP occurs with Function f_6. In this case, CEP has difficulty dealing with plateaus because it mainly searches in a relatively local neighbourhood in spite of adaptive mutation. All the points within a local neighbourhood will have the same fitness value except for a few boundaries between plateaus. FEP can jump to a point further away from the current one with a relatively high probability. This increases the probability of jumping to the next lower plateau. The rapid convergence of FEP shown in Figure 2 seems to confirm our explanations.

5.3 Multimodal Functions

5.3.1 Multimodal Functions With Many Local Minima

Multimodal functions having many local minima are often regarded as being most difficult to optimise. Techniques like fitness sharing which can handle functions with a few local minima are no longer effective for functions with many local minima. We have carried out experiments on some multimodal functions, i.e., f_8–f_{13} where the number of local minima increases exponentially as the dimension of the function increases. The dimensions of f_8–f_{13} are all set to 30.

In order to evaluate FEP fully, we also include some other well-known multimodal functions in our test set, i.e., f_{14}–f_{23} where the number of local minima for each function and the dimension of the function are small. Figures 3–4 illustrate the performances of CEP and FEP on f_8–f_{15}. Similar figures for f_{16}–f_{23} were omitted due to the page limitation of the paper. Table 3 summarises our results in terms of the mean best solutions in the last generation of the runs and their standard deviations over 50 runs for each function.

For 30-dimensional functions f_8–f_{13}, it is found that FEP performs much better than CEP. It has a faster convergence rate and discovers better solutions. CEP seems to stagnate after certain generations. For most functions, it stagnates quite early as well. This may be caused by the thinner tails of the Gaussian distribution used by CEP. In theory, the adaptive mutation scheme used by CEP should increase the variance of the Gaussian distribution and help CEP to escape from local minima. However, this does not seem to happen in practice. We are currently investigating why the theory does not match the reality.

By observing Figures 3–4, we can discover that FEP seems to converge roughly at a linear rate or even faster with respect to the number of generations. However, further theoretical analysis is necessary before a convincing conclusion can be drawn.

5.3.2 Multimodal Functions With Only a Few Local Minima

Interestingly, quite different results have been observed from functions f_{14} to f_{23}. FEP's performance is either slightly better than CEP's, e.g., for f_{14}, or almost the same as CEP's, e.g., for functions f_{16}, f_{17}, f_{19}, and f_{20}. FEP performs worse than CEP for functions f_{15}, f_{18}, and f_{21} to f_{23}. In fact, CEP's and FEP's performances are very close for f_{15} and f_{18}.

Table 1: The 23 test functions used in our experimental studies, where n is the dimension of the function, f_{min} is the minimum value of the function, and $S \subseteq R^n$.

Test function	n	S	f_{min}
$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]^n$	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]^n$	0
$f_3(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	$[-100, 100]^n$	0
$f_4(x) = \max_i \{ \lvert x_i \rvert, 1 \leq i \leq n \}$	30	$[-100, 100]^n$	0
$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]^n$	0
$f_6(x) = \sum_{i=1}^{n} \lfloor x_i + 0.5 \rfloor$	30	$[-100, 100]^n$	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]^n$	0
$f_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	$[-500, 500]^n$	-12569.5
$f_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10)]$	30	$[-5.12, 5.12]^n$	0
$f_{10}(x) = -20 \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos 2\pi x_i\right) + 20 + e$	30	$[-32, 32]^n$	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]^n$	0
$f_{12}(x) = \frac{\pi}{n}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{1}{4}(x_i + 1)$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0, & -a \leq x_i \leq a, \\ k(-x_i - a)^m, & x_i < -a. \end{cases}$	30	$[-50, 50]^n$	0
$f_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_i - 1)^2[1 + sin^2(3\pi x_{i+1})] + (x_n - 1)[1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	$[-50, 50]^n$	0
$f_{14}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right]^{-1}$	2	$[-65.536, 65.536]^n$	1
$f_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	$[-5, 5]^n$	0.0003075
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]^n$	-1.0316285
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5, 10] \times [0, 15]$	0.398
$f_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	$[-2, 2]^n$	3
$f_{19} = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{4} a_{ij}(x_j - p_{ij})^2\right]$	4	$[0, 1]^n$	-3.86
$f_{20} = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right]$	6	$[0, 1]^n$	-3.32
$f_{21} = -\sum_{i=1}^{5} [(x - a_i)^T(x - a_i) + c_i]^{-1}$	4	$[0, 10]^n$	$-1/c_1$
$f_{22} = -\sum_{i=1}^{7} [(x - a_i)^T(x - a_i) + c_i]^{-1}$	4	$[0, 10]^n$	$-1/c_1$
$f_{23} = -\sum_{i=1}^{10} [(x - a_i)^T(x - a_i) + c_i]^{-1}$ where $c_1 = 0.1$	4	$[0, 10]^n$	$-1/c_1$

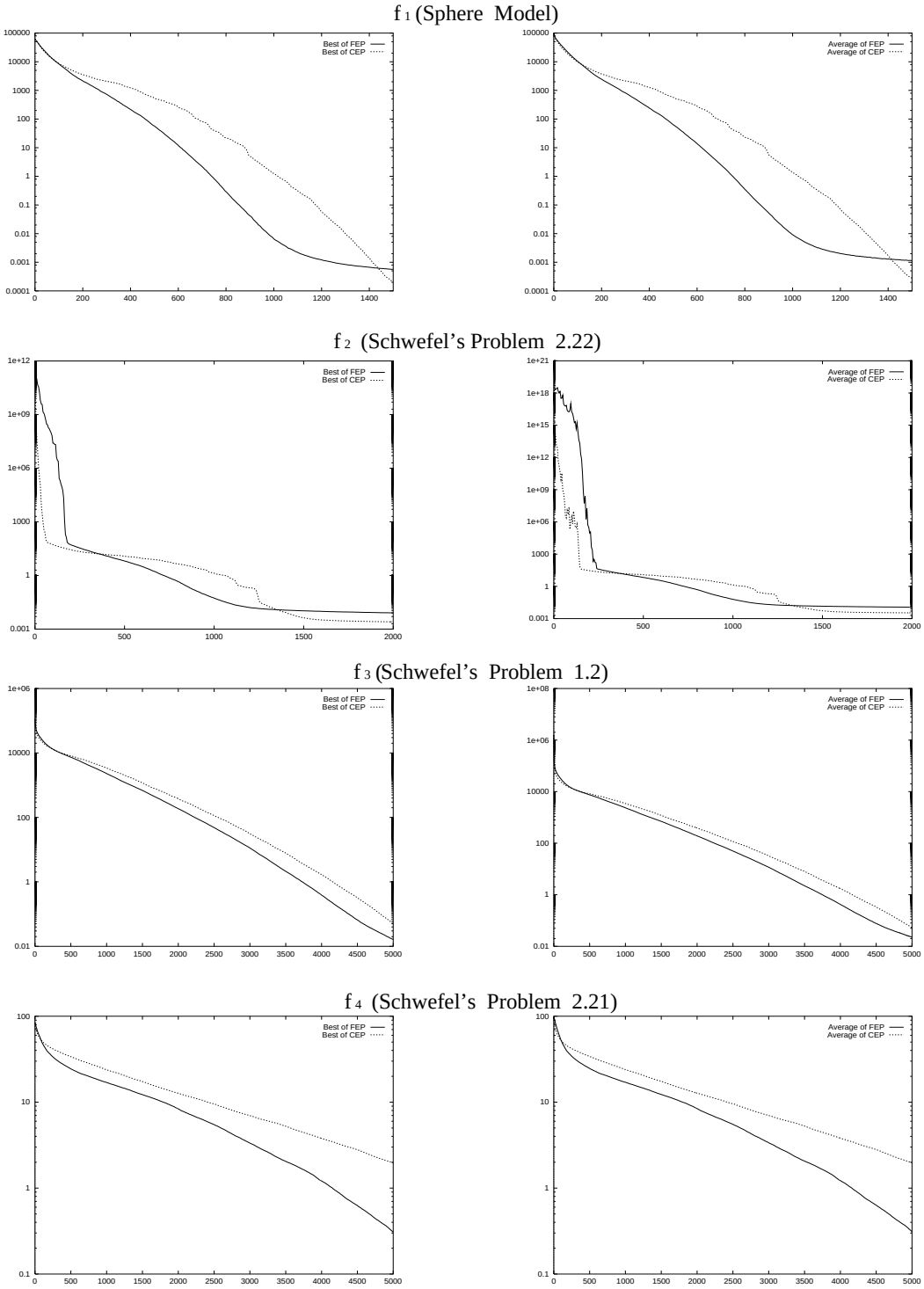
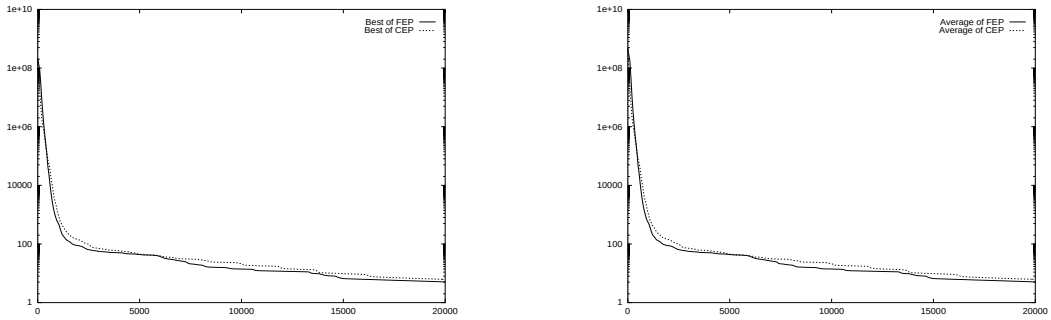
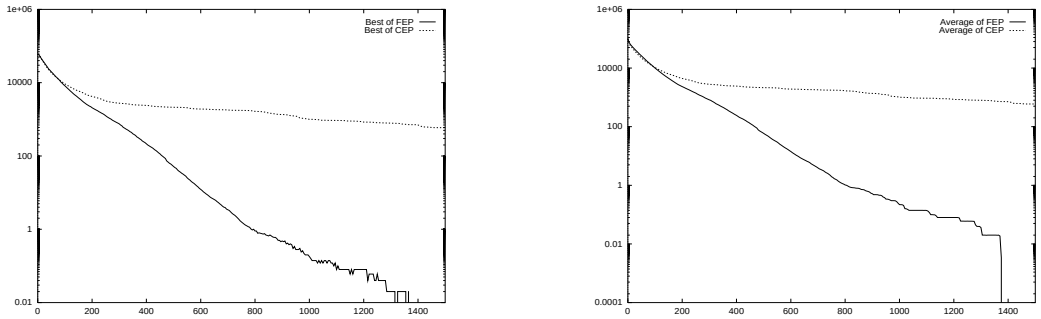


Figure 1: Comparison between CEP and FEP on f_1–f_4. The vertical axis is the function value and the horizontal axis is the number of generations.

f 5 (Generalized Rosenbrock's Function)



f 6 (Step Function)



f 7 (Quartic Function with Noise)

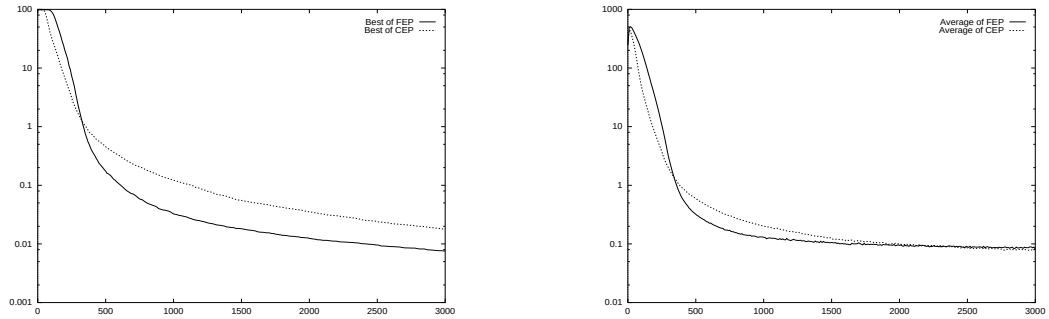


Figure 2: Comparison between CEP and FEP on f_5–f_7.The vertical axis is the function value and the horizontal axis is the number of generations.

Table 2: Comparison between CEP and FEP on f_1–f_7. All results have been averaged over 50 runs, where "Mean Best" indicates the mean best function values found in the last generation, and "Std Dev" stands for the standard deviation.

Function	Number of	FEP		CEP		FEP−CEP
	Generations	Mean Best	Std Dev	Mean Best	Std Dev	t-test
f_1	1500	5.7×10^{-4}	1.3×10^{-4}	2.2×10^{-4}	5.9×10^{-4}	$4.06^{\dagger}$
f_2	2000	8.1×10^{-3}	7.7×10^{-4}	2.6×10^{-3}	1.7×10^{-4}	$49.83^{\dagger}$
f_3	5000	1.6×10^{-2}	1.4×10^{-2}	5.0×10^{-2}	6.6×10^{-2}	$-3.79^{\dagger}$
f_4	5000	0.3	0.5	2.0	1.2	$-8.25^{\dagger}$
f_5	20000	5.06	5.87	6.17	13.61	-0.52
f_6	1500	0	0	577.76	1125.76	$-3.67^{\dagger}$
f_7	3000	7.6×10^{-3}	2.6×10^{-3}	1.8×10^{-2}	6.4×10^{-3}	$-10.72^{\dagger}$

†The value of t with 49 degrees of freedom is significant at $\alpha = 0.025$ by a two-tailed test.

Table 3: Comparison between CEP and FEP on f_8–f_{23}. The results are averaged over 50 runs, where "Mean Best" indicates the mean best function values found in the last generation, and "Std Dev" stands for the standard deviation.

Function	Number of	FEP		CEP		FEP−CEP
	Generations	Mean Best	Std Dev	Mean Best	Std Dev	t-test
f_8	9000	-12554.5	52.6	-7917.1	634.5	$-51.39^{\dagger}$
f_9	5000	4.6×10^{-2}	1.2×10^{-2}	89.0	23.1	$-27.25^{\dagger}$
f_{10}	1500	1.8×10^{-2}	2.1×10^{-3}	9.2	2.8	$-23.33^{\dagger}$
f_{11}	2000	1.6×10^{-2}	2.2×10^{-2}	8.6×10^{-2}	0.12	$-4.28^{\dagger}$
f_{12}	1500	9.2×10^{-6}	3.6×10^{-6}	1.76	2.4	$-5.29^{\dagger}$
f_{13}	1500	1.6×10^{-4}	7.3×10^{-5}	1.4	3.7	$-2.76^{\dagger}$
f_{14}	100	1.22	0.56	1.66	1.19	$-2.21^{\dagger}$
f_{15}	4000	5.0×10^{-4}	3.2×10^{-4}	4.7×10^{-4}	3.0×10^{-4}	0.49
f_{16}	100	-1.03	4.9×10^{-7}	-1.03	4.9×10^{-7}	0.0
f_{17}	100	0.398	1.5×10^{-7}	0.398	1.5×10^{-7}	0.0
f_{18}	100	3.02	0.11	3.0	0	1.0
f_{19}	100	-3.86	1.4×10^{-5}	-3.86	1.4×10^{-2}	-1.0
f_{20}	200	-3.27	5.9×10^{-2}	-3.28	5.8×10^{-2}	0.45
f_{21}	100	-5.52	1.59	-6.86	2.67	$3.56^{\dagger}$
f_{22}	100	-5.52	2.12	-8.27	2.95	$5.44^{\dagger}$
f_{23}	100	-6.57	3.14	-9.10	2.92	$4.24^{\dagger}$

†The value of t with 49 degrees of freedom is significant at $\alpha = 0.05$ by a two-tailed test.

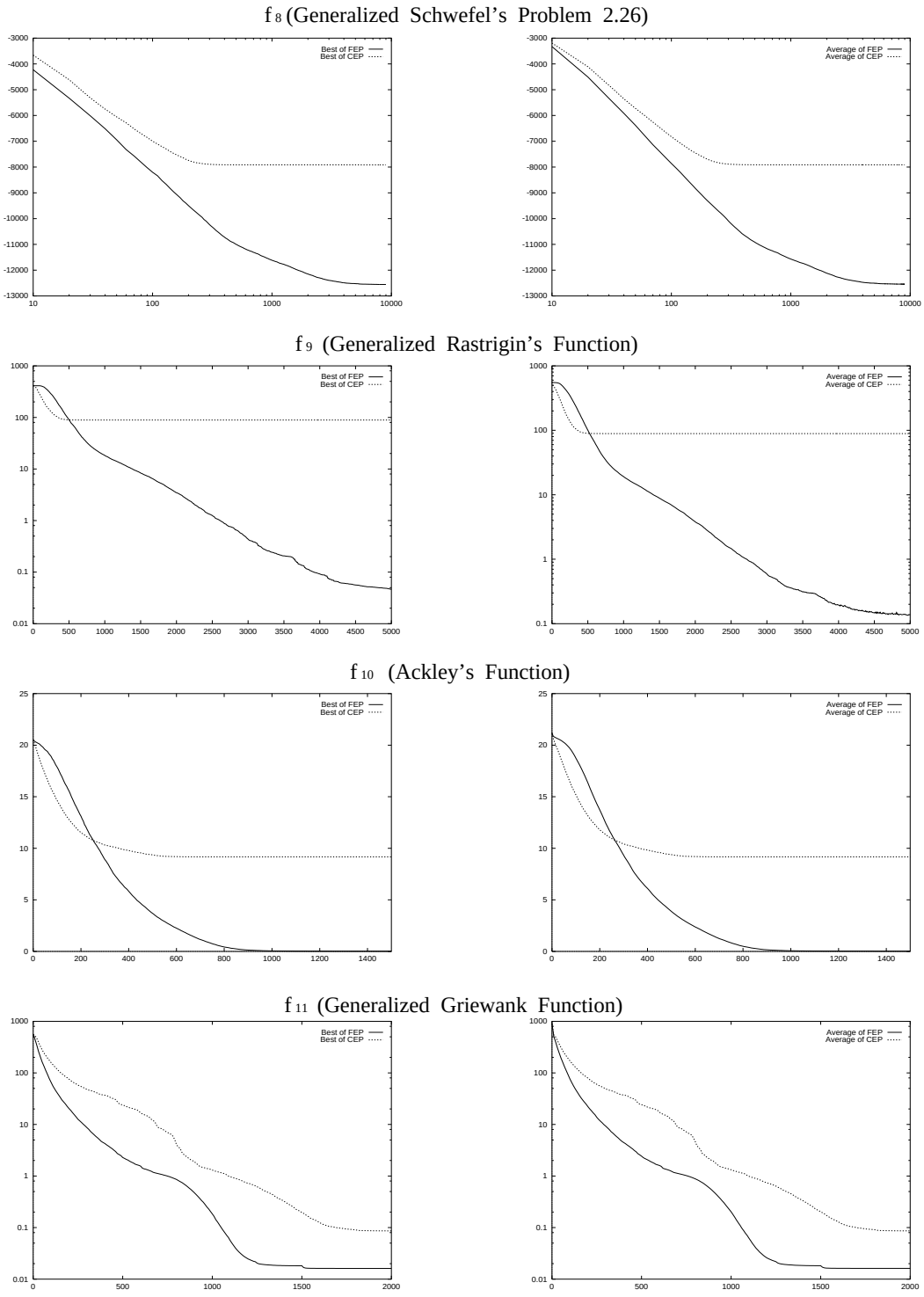


Figure 3: Comparison between CEP and FEP on f_8–f_{11}. The vertical axis is the function value and the horizontal axis is the number of generations.

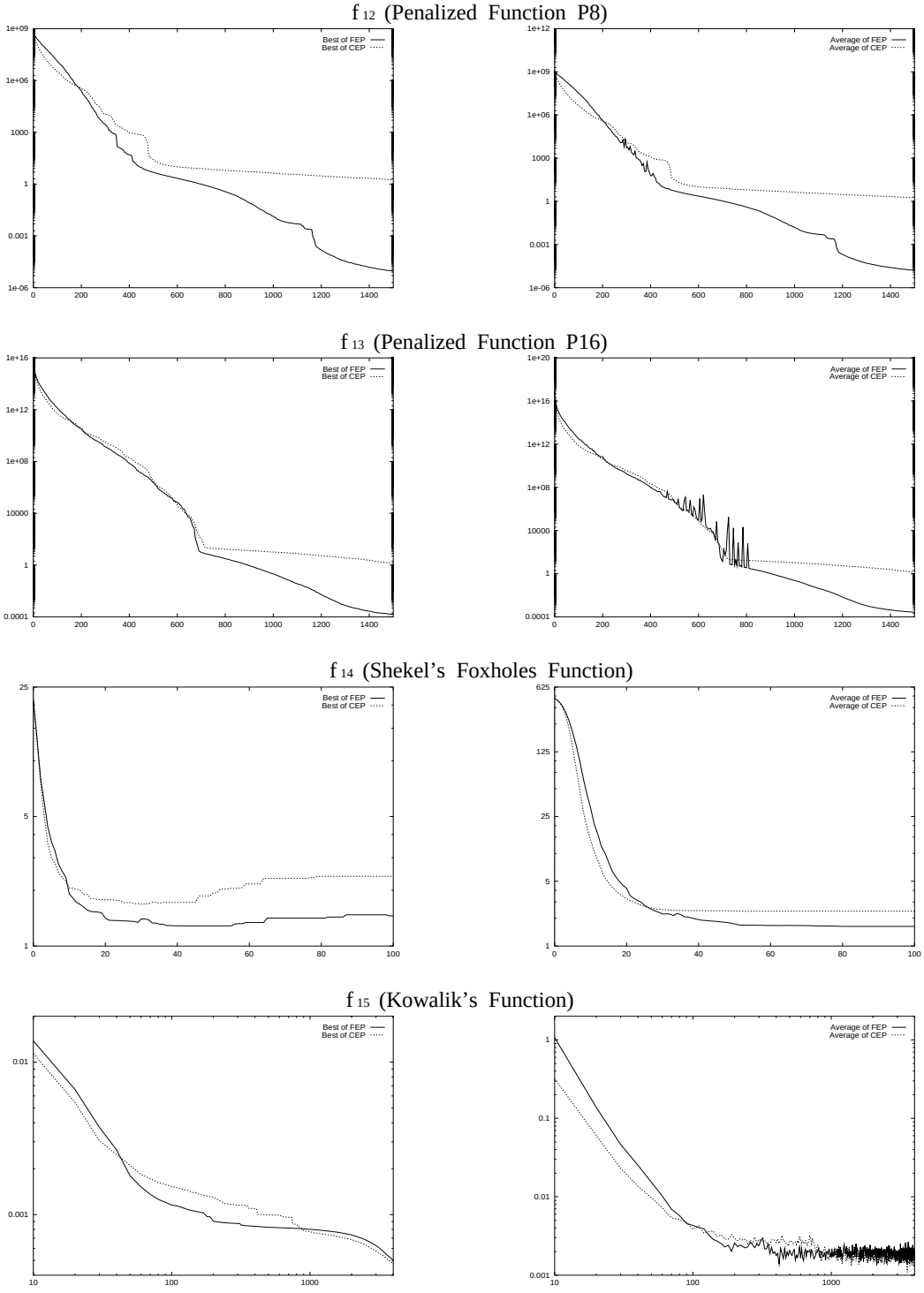


Figure 4: Comparison between CEP and FEP on f_{12}–f_{15}. The vertical axis is the function value and the horizontal axis is the number of generations.

It is possible that f_{21} to f_{23}, which are Shekel functions, pose particular difficulties to FEP. Shekel functions might prefer fine-tuned local search to a combined local and global search. This is something we would like to find out in our future research. The difference between FEP and CEP may also be caused by unsuitable parameter settings for FEP.

There is another potential source which might cause the difference between CEP's and FEP's performance — the dimensionality of the function. For function f_8 to f_{13}, their dimensions are all 30, while the dimensions of f_{14} to f_{23} are all no more than 6. Is it possible that FEP is better for high-dimensional functions while CEP is better for low-dimensional ones? We have carried out another set of experiments on the low-dimensional version of f_8 to f_{13} to find out the answer. The experimental results confirmed that FEP converged much faster to a better solution than CEP. The difference between CEP'sand FEP's performance is not caused by the dimensionality of the function.

6 Conclusions

An FEP algorithm using the Cauchy mutation operator is proposed in this paper. The Cauchy mutation operator provides FEP with a faster convergence rate due to its better global search capability. We have tested a basic form of FEP on 23 well-known functions. Our experimental results show that FEP converges to a better near optimal solution much faster than CEP for multimodal functions with many local minima. FEP performs worse than CEP on the three Shekel functions. It is comparable to CEP on other functions. The worse-than-expected performance of FEP on the three Shekel functions is caused either by the simple-minded implementation of FEP or by the characteristics of the functions.

There are a number of interesting research topics we would like to pursue in the future, some of which may not be specific to EP. First, we are interested in the characteristics of the Shekel functions and plan to analyse whether they are the source for FEP's poor performance. Second, we will design and evaluate a potentially more appropriate FEP using a different adaptive mutation scheme and different set of parameters. Third, we intend to carry out theoretical analysis of FEP's convergence rate for some well-known functions and compare the result with CEP's. This is an important topic, but may be quite difficult. The difficulty can be seen from the lack of vigorous analysis of fast simulated annealing's convergence rate. Fourth, we will look at the impact of adaptive mutation on EP's capability of escaping from local minima. This research is inspired by CEP's poor performance on multimodal functions with many local minima. Last, we will investigate the scalability of FEP by conducting experiments with dimension larger than 30.

Acknowledgement — This work is partially supported by the Australian Research Council through its small grant scheme. The authors are grateful to Dr David Fogel for his constructive comments on the paper.

References

[1] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence Through Simulated Evolution*, John Wiley & Sons, New York, NY, 1966.

[2] D. B. Fogel, *System Identification Through Simulated Evolution: A Machine Learning Approach to Modeling*, Ginn Press, Needham Heights, 1991.

[3] D. B. Fogel, *Evolving Artificial Intelligence*, PhD thesis, University of California, San Diego, 1992.

[4] D. B. Fogel, "Applying evolutionary programming to selected traveling salesman problems," *Cybernetics and Systems*, 24:27–36, 1993.

[5] H. H. Szu and R. L. Hartley, "Nonconvex optimization by fast simulated annealing," *Proceedings of IEEE*, 75:1538–1540, 1987.

[6] X. Yao, "Simulated annealing with extended neighbourhood," *Int. J. of Computer Math.*, 40:169–189, 1991.

[7] X. Yao, "A new simulated annealing algorithm," *Int. J. of Computer Math.*, 56:161–168, 1995.

[8] D. Fogel, "An introduction to simulated evolutionary optimisation," *IEEE Trans. on Neural Networks*, 5(1):3–14, 1994.

[9] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary Computation*, 1(1):1–23, 1993.

[10] W. Feller, *An Introduction to Probability Theory and Its Applications*, volume 2, John Wiley & Sons, Inc., 2nd edition, 1971.

[11] G. S. Fishman and V. G. Kulkarni, "Improving Monte Carlo efficiency by increasing variance," *Management Science*, 38(10):1432–1444, 1992.

[12] A. Törn and A. Žilinskas, *Global Optimisation*, Springer-Verlag, Berlin, 1989. Lecture Notes in Computer Science, Vol. 350.

[13] H.-P. Schwefel, *Evolution and Optimum Seeking*, John Wiley & Sons, New York, 1995.

[14] D. H. Wolpert and W. G. Macready, "No free lunch theorems for search," TR SFI-TR-95-02-010, Santa Fe Institute, NM 87501, July 1995.