MULTI-RESOLUTION OPTIMISATION: APPLICATION OF META-HEURISTIC IN FUNCTION REMODELLING

Cheng Wai Kheng, Siang Yew Chong, Andrzej Bargiela
School of Computer Science,
The University of Nottingham, Malaysia Campus,
Jalan Broga, Semenyih, 43500, Selangor, Malaysia
Email: {cheng-wai.kheng, siang-yew.chong,}@nottingham.edu.my, andrzej.bargiela@nottingham.ac.uk

KEYWORDS

Optimisation, Function Remodelling, Meta-Heuristic, Approximation, Estimation, Smoothing.

ABSTRACT

Multi-resolution approach to optimisation involving remodelling of objective function has great received interests lately. There are two different goals of remodelling for the objective function. First, the reduction of computational complexity of the objective functions in order to accelerate the searching process. Second, the removal of local optima in the objective functions to avoid premature convergence. Most of the approach shares the problem: the setting of the smoothness pressure value over the objective function. We proposed a meta-heuristic approach to determine the suitable smoothness pressure value throughout the searching process. The results show that meta-heuristic has significant improvement compared to state-of-the-art Classic Evolutionary Programming (CEP) and Fast Evolutionary Programming (FEP).

INTRODUCTION

Continuous Optimisation is a non-linear programming problem that involves finding the optimal solution in a given solution space. Often, the form of optimisation is defined as an objective function, which determines the fitness of given solution (Goldberg 1989; Lange 2004, Muhammad et al 1999, 2001). A solution is defined as a vector or a set of real values. If the problem is bounded, the values are restricted in the lower bound and upper bound of the problem. Such as follow:

$$\begin{aligned} &\min \mathrm{f}(x) \; \textit{such that} \; \forall x' \neq x, \mathrm{f}(x') > \mathrm{f}(x) \\ &\textit{where } x, x' \in X \textit{ and } X \subset \mathbf{R}^N \\ &\mathrm{low}_i < x_i < \mathrm{high}_i \end{aligned} \tag{1}$$

The application of Evolutionary Algorithm (EA) in optimisation received attention due to their stochastic characteristic and capability of escaping local optima in the solution space (Goldberg 1989). Two issues that always surrounded by EA are the convergence speed and the quality of converged solution. Several algorithms that are based on EA have been proposed. These includes Memetic Algorithm (Krasnogor 2002), Cooperative Co-Evolutionary Algorithm (Popovici and Jong 2006). Other researches include improving the three major components of EA, which are the problem representation, selection and variation methods. Recently, variation method with higher probabilities of escaping local optima have been studied, such as Cauchy Mutation (Xin et al. 1999), Levy Mutation (Chang-Yong and Xin 2004), both producing promising results over benchmark testing functions.

On the other hand, improvement on problem representation is the remodelling of objective function. There are two different types of remodelling the objective function. First, the reduction of computational complexity of the objective functions in order to accelerate the searching process. Several regression techniques have been proposed, such as polynomial regression (Branke and Schmidt 2005), Gaussian/Kriging regression (Ratle 1998, 1999; Buche et al. 2005; Nakayama et al. 2001), these techniques have been implemented in Memetic Algorithm as local surrogates model for local searches (Zhou et al. 2007).

Second, the removal of local optima in the objective functions to avoid premature convergence. Techniques implemented in 2nd type includes: Polynomial (Quadratic) Approximation (Liang et al. 2000), Gaussian Kernel Smoothing (Yang and Flockton 1995), Truncated Fourier Series (Wang 2004). These techniques work under the condition that the remodelled function provides correct smoothness and certain degree of similarity compared to the objective function. As concluded by Liang et al, the Quadratic Approximation model performs worst than without remodelling if the problem does not carry quadratic characteristic (Liang et al. 2000). Gaussian Kernel Smoothing shares the same problem which the kernel size affects the smoothness of the remodelled function. Dekun Yang et al proposed a coarse-to-fine mechanism, where the kernel size reduces according to the searching progress. Bernardetta Addis et al discussed the same problem in (Addis et al. 2005) where Gaussian Smoothing Kernel is used together with numerical optimisation. In this paper, we further the investigation of the effect of Gaussian Smoothing

doi: 10.7148/2009-0834-0840

Kernel in optimisation, and proposed a metaheuristic approach in choosing the kernel size which reacts according to the current population.

BACKGROUND STUDY

Noise reduction is always an interesting issue in signal/image processing. The main purpose of noise reduction is to filter the unwanted signals that are caused by external factors. Local optima in optimisation share this characteristic, where the solution's fitness does not reflect the actual distance from optimal solution. To smooth continuous signals, a convolution transformation has to be done. As depicted in (Yang and Flockton 1995), the convolution transformation, or the approximation of function *f* is the integration of multiplication of smoothing kernel and the function. Let F_β be the approximated function with β as the smoothness pressure.

$$F_\beta(x) = \int_{y \in \mathbf{R}^N} h_\beta(y) f(x-y) dy \quad (2)$$
$$\text{where } h(y) \text{ satisfied } \lim_{\beta \to 0} h_\beta(y) = \delta(y)$$

Gaussian function satisfied condition in (2) and is given as below:

$$h_\beta(x) = \frac{1}{(2\pi)^{N/2} \beta^N} \exp(-x^T x / 2\beta^2) \quad (3)$$

The β and N are the kernel size and number of dimension respectively. That is, the higher the β, the smoother the approximated function. As Evolutionary Algorithm treats the optimisation as a black box searching process, the convolution of the function itself is prohibited. However, statistical learning suggested that if the sample size is large enough, then the convolution yields the similar result as the expected value (mean) of the fitness value of neighbour solutions. These solutions are sampled randomly according to the probability of Gaussian distribution, as depicted as y in (4).

$$F_\beta(x) = E_y[f(x-y)] \quad (4)$$

And the expected value of these neighbour solutions is its sample mean:

$$F_\beta(x) = \frac{1}{M} \sum_{i=1}^{M} f(x - y_i) \quad (5)$$

with y_i is randomly selected according to Gaussian distribution with β as the standard deviation. (2) to (5) are summarized from Dekun Yang et al (Yang and Flockton 1995). As discussed above, the β is the smoothness pressure of the approximated function. The β has to be carefully selected because if it is too small, the approximated function does not remove the local optima, whereas if it is too high, it removes the global optimum as well (Yang and Flockton 1995). In (Yang and Flockton 1995), a coarse-to-fine mechanism is used, where β is arbitrary set to 3.0 initially and reduced 30% every eighty generation. First, the coarse-to-fine mechanism does not reflect the underlying optimisation problem. Second, the approximated function F_β does not guarantee to have the same global optimum as original fitness function F, this is shown in figure 1, where β=3.0 and 2.0 shifted the global optimum of the function. This is referred as "curse of uncertainty", a term was coined in (Ong et al. 2006). It indicates that the improper approximation causes the local search returns non-local-optimum solution in F, but the solution appears to be local optimum in F_β. In fact, the smoothed function only provides the "area of interest" where there no local optima. Consider figure 2 and 3. Figure 2 suggested that the smooth function removed local optima while the global optimum remains. However, the finer interval in figure 3 shows that the global optimum in objective function remains in single point while the approximated function does not carry any optimum. Thus, the approximated function should only provides the "area of interest" while the actual searching process continues with solutions found in the approximated function.

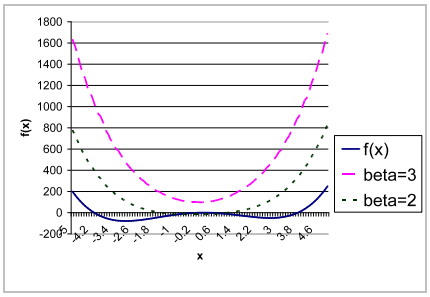

Fig 1. Global Optimum shifted

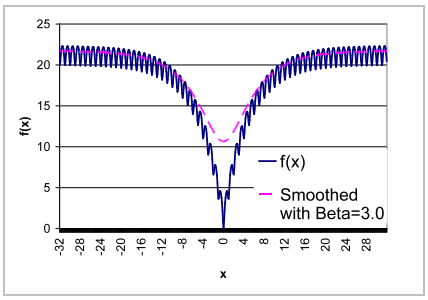

Fig 2. Effect of Gaussian Smoothing on Ackley

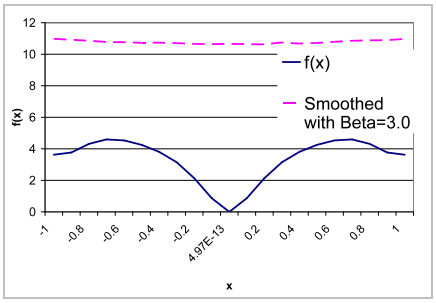

Fig 3. Ackley with finer interval between -1 to 1.

doi: 10.7148/2009-0834-0840

METHODOLOGY

From the hyperheuristic/metaheuristic point of view, the approximated functions with different smoothness are different heuristics for solving the problem. Choosing the good approximation may result significant improvement on the algorithm, which includes faster convergence speed and better converged solution. A heuristic is informal action used in solving problem, often is cheap in computation complexity and yet a natural way of solving the problem(Cowling et al. 2002). A hyperheuristic is a heuristic that chooses low level heuristic independent from problem, such as CPU execute cycles, improvement of a consecutive heuristic, etc(Kendall and Soubeiga 2002; Cowling et al. 2002). Metaheuristic is a hyperheuristic but the choosing function is often problem dependent (Cowling et al. 2002), such as pre-clustering of class of problem and choose corresponding heuristic according to the problem class (Ong and Keane 2004).

Approximated function F_β should be balanced between the smoothness and the similarity. Smoothness is referred to the number of local optima in the searching space. It can be represented by the inverse of roughness. However, there is no other way to quantify local optima in the function except by finding all of them. Fortunately, the fluctuation of the function also reflects the number of local optima. The fluctuation of the function is associated with the standard deviation of itself, σ(F).

$$\text{roughness of F} \propto \text{fluctuation of F} \propto \sigma(F) \tag{6}$$

$$\mu = \int x\,\mathrm{F}(x)dx \tag{7}$$

$$\sigma(F) = \sqrt{\int (x-\mu)^2\,\mathrm{F}(x)dx} \tag{8}$$

Similar to (2), the integration is prohibited, and can be replaced with unbiased estimator with uniform sampling:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{9}$$

$$\sigma(F) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\bar{x})^2} \tag{10}$$

Similarity is referred here to the difference between the global optimum in F_β and F. Also, this cannot be assessed as the searching process does not know the actual global optimum in the searching space. However, the area difference between F_β and F may reflect the similarity as well. Thus, the comparison of the area of the underlying function is defined as λ(F), which is normalised from root mean square error "*rmse*" from (Ong et al. 2006):

$$\text{area difference} \propto \lambda(F_\beta)$$

$$\lambda(F_\beta) = \frac{\| \int F_\beta(x) - F(x)dx \|}{\int F(x)dx} \tag{11}$$

$$for\ \beta \in \mathrm{R}^+$$

Followed by an unbiased estimator with uniform sampling:

$$\lambda(F_\beta) = \frac{\| \sum_{i=1}^{N} \mathrm{F}_\beta(x) - \mathrm{F}(x) \|}{\sum_{i=1}^{N} \mathrm{F}(x)} \tag{12}$$

In order for (9-10), (12) to estimate the roughness and area difference between the approximated and objective function accurately, sample size N must be very large. Fortunately, the sample size can be reduced if the quantifying process only takes place for the current searching area, which are the solutions in the current population. Let p be the current population, and I_{best} and I_{worst} be the best and worst solution in population p. The current searching space is defined as γ, with abbreviation γ (i) indicate value of i^{th} element:

$$\begin{aligned} &\text{for min problem} \\ &\gamma(\mathrm{i}) = \{\mathrm{x} \in \mathrm{R} \mid \mathrm{I}_{best}(\mathrm{i}) < \mathrm{x} < \mathrm{I}_{worst}(\mathrm{i})\} \\ &\text{for max problem} \\ &\gamma(\mathrm{i}) = \{\mathrm{x} \in \mathrm{R} \mid \mathrm{I}_{best}(\mathrm{i}) > \mathrm{x} > \mathrm{I}_{worst}(\mathrm{i})\} \end{aligned} \tag{13}$$

The uniform sampling processes for (8) and (12) are performed within γ. The empirical model for the metaheuristic in choosing F_β out of a predefined set $\beta=\{\beta_1, \beta_2, \beta_3, ...\beta_n\}$ is the β value that produce the minimum output of M(β):

$$M(\beta) = \alpha\sigma(F_\beta) + (1-\alpha)\lambda(F_\beta) \tag{14}$$

Where α is the weight associate with the roughness and area difference for each approximated function F_β. In this paper we select α value as 0.3. The authors are aware that M(β) is a multi-objective optimisation problem. We avoid using multi-objective optimisation in this event due to the demonstration here is not about to choose the exact β but to estimate the good balance between smoothness and similarity.

The proposed algorithm is outlined in pseudo code in Fig 4. The procedure starts by randomly generates solutions into the population. To determine which approximation function F_β to use, the algorithm has to choose the suitable β, which M(β) evaluates all beta values and returns the suitable β. There are two phase in the searching process. The first phase search in F_β and when it convergence condition is met, the process continues on the second phase, which searches in F. The convergence condition we experimented is the number of generation without improvement, and it is set to two hundred generations.

doi: 10.7148/2009-0834-0840

```
Define Population as pop
Populate pop
Initialize Beta = {3.0, 2.0, 1.0, 0.5, 0.3, 0.1}
Initialize mutation rate=3.0
Initialize generation=1
Choose B from Beta based on M(β)
Evaluate pop based on Fβ
While(termination condition)
DO
   set improvement=0
   While(i<population size)
   DO
      Select individual i from pop
      offspring = GaussianMutation(i, mutation rate)
      IF(offspring fitness better than i)
         improvement+1
      ADD offspring into pop
   ENDDO
      IF converged on Fβ
         Evaluate pop based on F
      ELSE
         Evaluate pop based on Fβ
      IF (generation%20 =0)
      DO
         IF not converged on Fβ
            Set B based on M(β)
            Evaluate pop based on Fβ
         IF improvement/population size > 0.2
            Set mutation rate = mutation rate*1.2
         ELSE
            Set mutation rate = mutation rate*0.8
      ENDDO
      Truncate pop
ENDDO
Return best solution of pop
```

Fig 4. Pseudocode of EA+S

EXPERIMENTAL SETTING

For the purpose of comparison, we implemented the metaheuristic and smoothing in Evolutionary Algorithm with 20% adaptive mutation rule, called EA+S. this algorithm is compared with Classic Evolutionary Programming (CEP), Fast Evolutionary Programming (FEP), and Evolutionary Algorithm with 20% adaptive mutation rule (EA). For simplicity, selection method implemented in these algorithms is truncation. EA+S revise the β value every twenty generations. For EA and EA+S, the adaptive mutation rate is also revised every twenty generations. There are totally ten benchmark functions tested with these experiments settings. Table 1 outlines all benchmark functions. F_1 to F_9 are taken from (Xin et al. 1999) while F_{10} is originally from (Yang and Flockton 1995).

To minimize the bias toward different type of problems between EA and EP, instead of using uniform selection for parents, each parent in EA produces exactly one offspring, as suggested in CEP and FEP (Xin et al. 1999). The only difference between EA and EP in this case is the adaptive mutation mechanism. For EA+S, the searching process is divided into two phases. In the first phase, solutions in population are evaluated with approximated function F_β. After the population has converged on F_β, the algorithm then continues the searching with the original function F.

Table 1:The Following Ten Benchmark Functions Have Been Used in Evaluating Effect of Smoothing. N Indicate The Number Of Dimension Of The Problem, S and F_{min} Indicates Function Range and Function Minimum respectively.

Test Function	n	S	f_{min}
$f_1(x) = \sum_{i-1}^{n} x_i^2$	30	$[-100,100]^n$	0
$f_2(x) = \max_i \{\| x_i \|, 1 \le i \le n\}$	30	$[-100,100]^n$	0
$f_3(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30,30]^n$	0
$f_4(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\| x_i \|})$	30	$[-500, 500]^n$	-12569.5
$f_5(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]^n$	-1.0316285
$f_6(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i) + 20 + e$	30	$[-32,32]^n$	0
$f_7(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$[-600,600]^n$	0
$f_8(x) = \sum_{i=1}^{n-1} (\lfloor x_i + 0.5 \rfloor)^2$	30	$[-100,100]^n$	0
$f_9(x) = a(x_2 - bx_1^2 + cx_1 - d)^2 + e(1-f)\cos(x_1) + e$ $a = 1, b = \frac{5}{4\pi^2}, c = \frac{5}{\pi}, d = 6, e = 10, f = \frac{1}{8\pi}$	2	$[0,15]^2$	0.397887
$f_{10}(x) = n + \sum_{i=1}^{n} x_i^2 - \cos(2\pi x_i)$	30	$[-5.12, 5.12]^n$	0

This is to avoid the situation where the approximated function shifted the global optimum as shown in Fig 1 and 3. For (5), the number of samples used here is the same as suggested in (Yang and Flockton 1995), which is 10. The sample size for (10) and (12) are 500 for both. As the sample size increases, (5), (10) and (12) becomes more accurate. Further investigation of the minimum sample size is required in the future.

doi: 10.7148/2009-0834-0840

The population size for all algorithms is set to 100, while the initial mutation rate is set to 3.0. The Gaussian random is time seeded. To avoid having different random sequence between algorithms, random sequence for (5), (10), and (12) are separated from mutation. Twenty individual runs for each algorithm is tested with all benchmark problem in table 1.

RESULT AND DISCUSSION

Recall that the purpose of smoothing in optimisation is to "pull" the population into the "area of interest" where the local optima do not exist. The ideal population that converged on approximated function should always within the "area of interest". However, the results suggested that it may not be this case. For EA+S, chances of "pulling" into the area are relatively high for F_1, F_3, F_5, F_6, F_8 and F_{10}. Consider figure 5, 6, and 7, majority of the individual runs successfully reach the area of interest, and converged on the global optimum in the second phase. EA+S suggested a huge improvement over these functions compared to EA. The reason we choose to compare our algorithm with CEP and FEP because both of these algorithms are the state-of-the-art in Evolutionary Computation. As we compare the result in table 2, EA+S always have the better quality of solution than EA, as well as FEP. The only exceptional cases are F_2 and F_4. Function F_2 is neither polynomial nor trigonometry, which is very hard for Gaussian to estimate the pattern of the function, whereas F_4 has very sharp local optima, which requires higher β value. In fact, as shown in figure 8, function F_4 requires at lest β=80 for the approximation to be smooth. Since the maximum setting for β is 3.0 in our experiment, we expect that EA+S perform worst compare to FEP and CEP. This raises another issue where the β range has to be predetermined before the searching process actually starts. Pre-processing such as performing random walk in β space is required.

As for the comparison of the convergence speed, figure 9 and 10 shows the significant difference between the convergence speeds for EA+S and FEP for function F_8. That is, all individual runs converged to global optimum within 500 generations for EA+S whereas only one individual run converged within 700 generations for FEP. However, this may due to several factors other than choosing the correct smoothness to approximate the function. Recall that the adaptive mutation rate for EA and EP are different. EA uses single mutation rate for all dimensions of the problem, whereas in EP, each dimension of the problem has their own adaptive mutation rate.

CONCLUSION AND FUTURE WORK

Based on the empirical results of the benchmark functions in table 2 and 3, we concluded that the metaheuristic does assist in the choosing the correct smoothness, providing a compartible underlying EA. However, its compartibility with other parameters in EA is still an open question. Further studies are required in order to understand the behaviour of Gaussian approximation in EA. We will analyse different adaptive mutation mechanism to maintain the population diversity without decreasing the probability of getting into the "area of interest". Throughout statistical learning, we may discover more meaningful information regarding the underlying problem and propose different meta-heuristic that is capable of handling various problems in the future.

Table 2: Comparison Between Es and Es+S. The Mean Best and Standard Deviation Values Have Been Averaged Over 20 Runs. Termination Condition is 3500 Generations. *Stdev is Abbreviation of Standard Deviation.

	EA		CEP		FEP		EA+S	
	Mean Best	STDEV	Mean Best	STDEV	Mean Best	STDEV	Mean Best	STDEV
F_1	**104.05**	14.33	1.57	3.73	0.31	1.14	*2.30E-19*	5.3E-19
F_2	3.91	0.26	12.59	5.77	7.84	4.67	6.59	6.91
F_3	**8.89E+06**	1228655	**18913.85**	16422.29	**13347.76**	20508.87	*2201.243*	4703.648
F_4	-7269.24	664.2538	-7503.03	593.4426	**-10587.7**	412.3559	-7374.01	605.8585
F_5	-1.03147	0.000171	-1.03163	4.56E-16	-1.03163	4.56E-16	-1.03163	4.56E-16
F_6	**7.936829**	0.273333	**8.236878**	2.199107	**2.271381**	1.688895	*4.65E-11*	7.3864E-11
F_7	**0.327072**	0.060595	**0.23566**	0.153455	**0.168185**	0.083916	*0.01958*	0.010644
F_8	**105.7**	10.87779	**922.3**	917.0538	**32.35**	40.7008	*0*	0
F_9	0.397976	8.45E-05	0.397887	1.14E-16	0.397887	1.14E-16	0.397887	1.14E-16
F_{10}	**122.505**	8.64936	**27.78619**	8.814827	**3.384301**	1.814504	*0.665995*	0.625033

*Mean Best with **BOLD** indicates that there are 95% confindence that they are statistically significant different compare to EA+S in T-Test. Mean Best with *Italic* in EA+S column indicates it is the best among others.

doi: 10.7148/2009-0834-0840

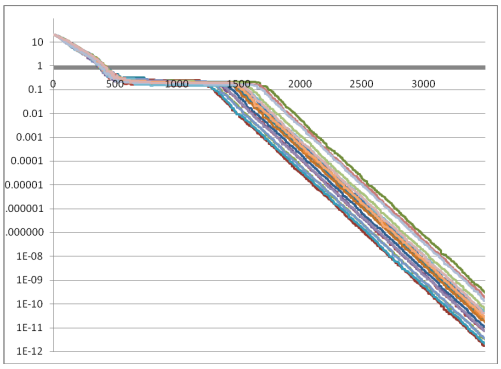

Fig 5. Natural Log Of Best Fitness of 20 Individual Runs of EA+S In 3500 Generations for Function F_1.

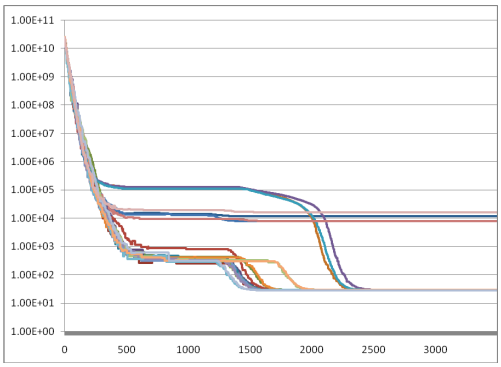

Fig 6. Natural Log Of Best Fitness of 20 Individual Runs of EA+S In 3500 Generations for Function F_3.

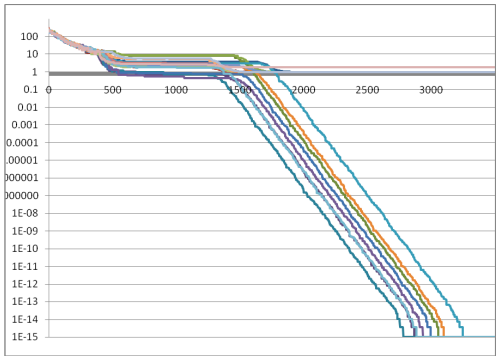

Fig 7. Natural Log of Best Fitness of 20 Individual Runs of EA+S In 3500 Generations for Function F_{10}.

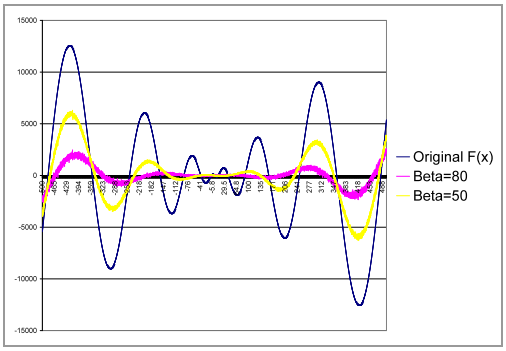

Fig 8. Single Dimension of Function F_4 With y-axis as The Fitness Value Against x-axis, and Beta=50, 80.

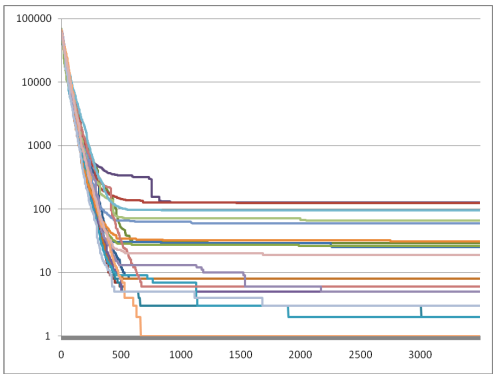

Fig 9. Natural Log of Best Fitness of 20 Individual Runs of FEP In 3500 Generations for Function F_8.

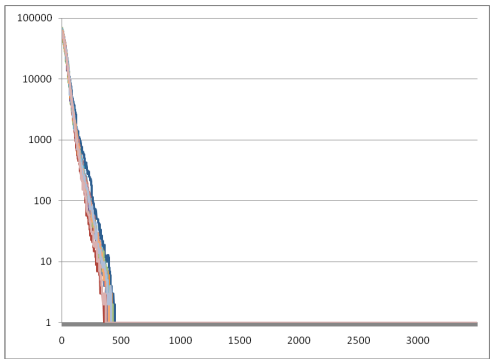

Fig 10. Natural Log of Best Fitness of 20 Individual Runs of EA+S In 3500 Generations for Function F_8.

REFERENCES

Addis, B., M. Locatelli, and F. Schoen. 2005. Local optima smoothing for global optimization. *journal Optimization Methods and Software* 20 (4 & 5):417-437.

Branke, J., and C. Schmidt. 2005. Faster convergence by means of fitness estimation. *Soft Computing - A Fusion of Foundations, Methodologies and Applications* 9 (1):13-20.

Buche, D., N. N. Schraudolph, and P. Koumoutsakos. 2005. Accelerating evolutionary algorithms with Gaussian process fitness function models. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 35 (2):183-194.

Chang-Yong, L., and Y. Xin. 2004. Evolutionary programming using mutations based on the Levy probability distribution. *Evolutionary Computation, IEEE Transactions on* 8 (1):1-13.

Cowling, P., G. Kendall, and E. Soubeiga. 2002. Hyperheuristics: A Tool for Rapid Prototyping in Scheduling and Optimisation. In *Applications of Evolutionary Computing*, 269-287.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization & Machine Learning*: Addison-Wesley.

doi: 10.7148/2009-0834-0840

Kendall, G., and E. Soubeiga. 2002. Choice function and Random Hyperheuristics. In *Proceedings of the fourth Asia-Pacific Conference on Simulated Evolution And Learning*. SEAL: Springer, 667--671.

Krasnogor, N. 2002. Studies on Theory and Design Space of Memetic Algorithm, Faculty of Computing, Engineering and Mathematical Sciences, University of the West of England, Bristol.

Lange, K. 2004. *Optimization*. Edited by G. Casella, S. Fienberg and I. Olkin: Springer-Verlag.

Liang, K.-h., X. Yao, and C. Newton. 2000. Evolutionary search of approximated n-dimensional landscapes. *International Journal of Knowledge-based Intelligent Engineering Systems* 4:172--183.

Muhammad A, A Bargiela, G King, Fine-grained parallel genetic algorithm: A global convergence criterion, International journal of computer mathematics 73 (2), 139-155, 1999, doi: 10.1080/00207169908804885

Muhammad A, A Bargiela, G King, Fuzzy and evolutionary modelling of nonlinear control systems, Mathematical and computer modelling 33, 4, 2001, 533-551, doi: 10.1016/S0895-7177(00)00259-4

Nakayama, H., M. Arakawa, and R. Sasaki. 2001. A Computational Intelligence Approach to Optimization with Unknown Objective Functions. In *Artificial Neural Networks — ICANN 2001*, 73-80.

Ong, Y.-S., Z. Zhou, and D. Lim. 2006. Curse and Blessing of Uncertainty in Evolutionary Algorithm Using ApproximationIn *IEEE Congress on Evolutionary Computation*.

Ong, Y. S., and A. J. Keane. 2004. Meta-Lamarckian learning in memetic algorithms. *Evolutionary Computation, IEEE Transactions on* 8 (2):99-110.

Popovici, E., and K. D. Jong. 2006. The effects of interaction frequency on the optimization performance of cooperative coevolution. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. Seattle, Washington, USA: ACM.

Ratle, A. 1998. Accelerating the Convergence of Evolutionary Algorithms by Fitness Landscape Approximation. In *Parallel Problem Solving from Nature — PPSN V*: Springer Berlin / Heidelberg, 87.

———. 1999. Optimal sampling strategies for learning a fitness model. *This paper appears in: Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress* 3:2085

Wang, Y. 2004. Improving Evolutionary Algorithms by a New Smoothing Technique. In *Intelligent Data Engineering and Automated Learning – IDEAL 2004*, 746-751.

Xin, Y., L. Yong, and L. Guangming. 1999. Evolutionary programming made faster. *Evolutionary Computation, IEEE Transactions on* 3 (2):82-102.

Zhou, Z., Y. Ong, P. Nair, A. Keane, and K. Lum. 2007. Combining Global and Local Surrogate Models to Accelerate Evolutionary Optimization. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 37 (1):66-76.

AUTHORS BIOGRAPHIES

CHENG WAI KHENG is a Postgraduate Student of School of Computer Science, University of Nottingham, Malaysia Campus. He is a member of Intelligent Decision Support Systems Research Group, University of Nottingham, Malaysia. His current research interests include the application of Hyper/Meta Heuristic, Evolutionary Computation and Optimisation. He received his B.Sc in Computer Science from University of Nottingham, Malaysia in 2007.

SIANG YEW CHONG is an Assistant Professor with the School of Computer Science, University of Nottingham, Malaysia Campus, a member of the Automated Scheduling, Optimization and Planning (ASAP) Research Group, University of Nottingham, UK, and an Honorary Research Fellow with the School of Computer Science, University of Birmingham, UK. He was a Research Associate with the Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA), University of Birmingham, UK, in 2007. His major research interests include evolutionary computation, neural networks, and evolutionary game theory. Dr. Chong received the Ph.D. degree in Computer Science from the University of Birmingham, UK, in 2007, and was awarded the 2009 IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation Award for his work on theoretical frameworks for performance analysis in co-evolutionary learning.

ANDRZEJ BARGIELA is Professor and Director of Computer Science at the University of Nottingham, Malaysia Campus. He is member of the Automated Scheduling and Planning research group in the School of Computer Science at the University of Nottingham. Since 1978 he has pursued research focused on processing of uncertainty in the context of modelling and simulation of various physical and engineering systems. His current research falls under the general heading of Computational Intelligence.

doi: 10.7148/2009-0834-0840