A Multi-Objective Optimization Procedure with Successive Approximate Models

Pawan K. S. Nain and Kalyanmoy Deb

Kanpur Genetic Algorithms Laboratory (KanGAL)
Department of Mechanical Engineering
Indian Institute of Technology, Kanpur
Kanpur, PIN 208016, India
{pksnain,deb}@iitk.ac.in,
WWW home page: http://www.iitk.ac.in/kangal

KanGAL Report Number 2005002

Abstract. This paper explores the possibility of using approximate model for fitness landscape in multi-objective optimization. A multi-objective genetic algorithm based optimizer, namely, NSGA-II is integrated with artificial neural network (ANN). This presented technique makes use of successive fitness landscape modeling for reducing the precise function evaluation calls while retaining the basic robust search capability of genetic algorithms (GA). The procedure is tried on some of the standard test problems available in literature on multi-objective optimization. The simulation results show a considerable savings in precise function evaluations and a good diversity in the obtained Pareto-front.

1 Introduction

Often the real world optimization problems require high computational loads. This computational load increases further for multi-objective optimization problems. Most real world optimization problems need robust optimizer with proven search capabilities. Evolutionary optimization methods are often used for such problems. Generally, all evolutionary optimizers require large number of function evaluation in order to reach the optima or near optima solutions. Sometimes, the time required for single precise function evaluation of the problem can take few hours to several days. Computational fluid dynamics problem and finite element analysis based problems are two such examples. Under such conditions, a strong need to reduce the number of precise function evaluations during the course of optimization is felt. This problem can be handled by using an approximate fitness landscape model of the original problem [1, 2]. However, the use of approximation model raises few questions. The first question is that whether it will be able to reach the same optima as that of original problem. The second question is about the diversity in the final set of optimal solutions. The present work addresses both of these concerns.

2 Past Studies

Many groups have reported their work on use of approximation models in evolutionary algorithms. A complete survey on the use of fitness approximation in evolutionary algorithms is reported by Jin and Sendhoff [1]. In this paper, authors have broadly classified the approximation methods, which are used currently, in three categories, namely response surface methodology, kriging models and artificial neural networks. Branke and Schmidt [6] have used two estimation methods, namely, regression and interpolation, to achieve faster convergence to the optima. Sastry et al. [7] have used fitness inheritance to reduce number of function evaluations. Rasheed and Hirsh [8] have used informed operators for speeding up the genetic algorithms. El-Beltagy et al. [9] have suggested the use of metamodels to reduce the computational burden on the evolutionary algorithms. They have used a metamodeling technique, namely the kriging approach. A metamodel based reconstruction algorithm is proposed by Ratle [10] while Emmerich et al. [11] have proposed a metamodel assisted evolution strategy for reducing the computational cost.

3 Generic Optimization Procedure

Here we present a generic procedure to combine a evolutionary optimization technique with the approximation technique. The focus of this study is to use a successive approximation of the optimization problem. Figure 1 depicts this

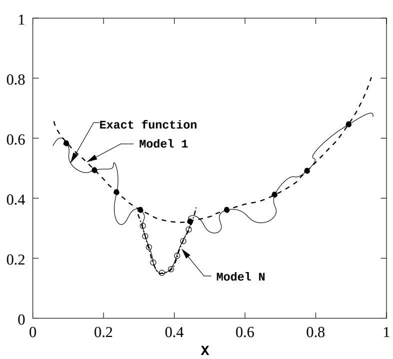

Fig. 1. Approximate modeling

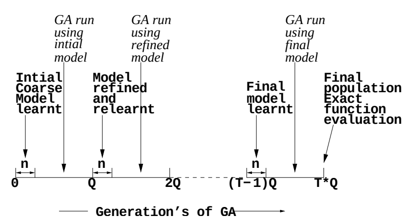

Fig. 2. A line diagram of the generic procedure

procedure. The figure shows a hypothetical one-dimensional objective function for minimization in a solid line. Since this problem may have a number of local minimum solutions, it would be a difficult problem for any optimization technique. Figure 1 also shows a coarsely approximated function in the entire range of the function with a dashed line. The optimization problem can be evaluated exactly at a few finite number of pre-specified solutions in the entire range of the decision variables. Thereafter, an approximating function can be fitted through these function values using regression or some other sophisticated techniques such as ANN. If this approximating function is optimized, it is likely that a evolutionary optimizer will proceed in the right direction. Since the population diversity will be reduced while approximating the first approximated function, the second approximating function need not be defined over the whole range of the decision variables, as shown in Figure 1. Since the approximating function will be defined over a smaller search region, more local details can appear in successive approximations. This process may continue till no further approximation results in an improvement in the function value. Figure 2 outlines a schematic of a plausible plan for the generic procedure. The combined procedure begins with a set of randomly created N solutions, where N is the population size. Since an adequate size of solutions are required to arrive at an approximated problem, we execute a evolutionary optimizer with exact function evaluations for n generations, thereby collecting a total of $N' = nN$ solutions for approximation. At the end of n generations, the approximation technique is invoked with N' solutions and the first approximated problem is created. The evolutionary optimizer is then run for the next $(Q - n)$ generations with this approximated problem. Thereafter, the evolutionary optimizer is performed with the exact function for the next n generations and a new approximated problem is created. This procedure is continued till the termination criterion is met. Thus, this procedure requires a fraction n/Q of total evaluations in evaluating the problem exactly.

4 Principle Results: Some Test Problems

In this section, some results of the multi-objective optimization based on successive approximations are presented. The multi-objective optimizer is non-dominating sorting genetic algorithm (NSGA-II) [4]. Artificial neural networks based on standard error back-propagation algorithm with unipolar sigmoidal activation function with logistics 0.5, momentum factor 0.1 and learning rate 0.3 is used for generating approximate model of the problem [5]. For the ANN, input neurons represent problem variables and output neurons represent different objective functions and constraint functions. The combined procedure will be referred as NSGA-II-ANN simulation. The ANN is trained using two different models, namely, batch training and incremental training. If a simulation is performed with batch training in which after every Q generation the approximation model is refined and the training database is collected over n generations, It is called as the B-Q-n model. If the training method is incremental, then it is called as the I-Q-n model. Here, two test problems, namely TNK, ZDT4 [3] are selected for testing the procedure.

4.1 TNK:

It is a two real-valued variable constrained test problem. The Pareto-optimal front is disconnected and have three parts as visible in Figs. 3 and 4. The number

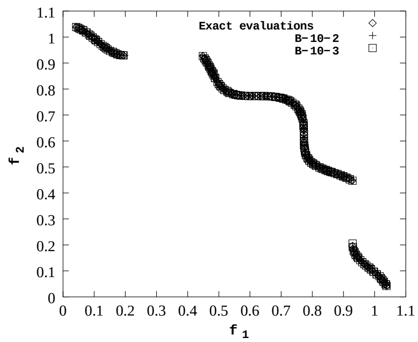

Fig. 3. Batch model simulation results for the TNK test problem

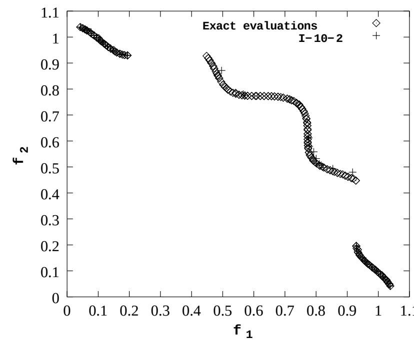

Fig. 4. Incremental model simulation results for the TNK test problem

of precise function evaluations taken by simulations of various models in order to reach the Pareto front are given in Table 1. Fig. 3 shows that *B*-10-2 and *B*-10-3 NSGA-II-ANN simulations reach the Pareto front with a saving of about 50% in precise function evaluations. Figs. 4 shows that *I*-10-2 NSGA-II-ANN simulations reach to Pareto-optimal front without any saving in precise function evaluations. The spread of the above three simulations and that of NSGA-II alone

Table 1. Spread metric results of NSGA-II-ANN simulations on the TNK test problem

Model Name	Part-I	Part-II	Part-III	Precise Function Evaluations	No. of hidden neurons
NSGA-II	0.644	0.872	0.704	5,00,400	NA
B-10-2	0.597	0.810	0.613	2,50,183	9
B-10-3	0.588	0.833	0.724	2,50,256	11
I-10-2	0.540	1.047	0.507	4,99,933	9

is compared in the Table 1. The spread metric calculation method is as suggested by Deb et al. [3]. In this metric calculation, the Euclidean distance of two extreme ends of simulation from corresponding ends of Pareto-optimal front as well as the uniformity of distribution for intermediate solutions is considered. A smaller value of the spread metric means a better spread. Batch model simulations show better spread results as well as savings in precise function evaluation. Based on the spread metric value, *B*-10-2 simulation is the best and is closely followed by the *B*-10-3 simulation. However, incremental mode simulations have difficulty in capturing the middle portion of the Pareto-optimal front which is also evident from the high spread metric value obtained for the *I*-10-2 simulation.

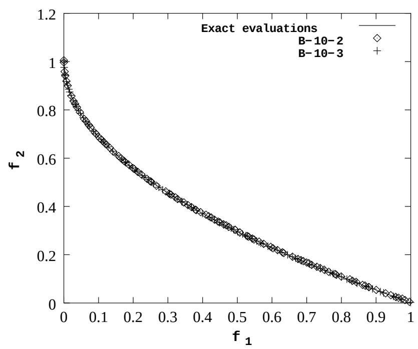

Fig. 5. Batch model simulation results for the ZDT4 test problem

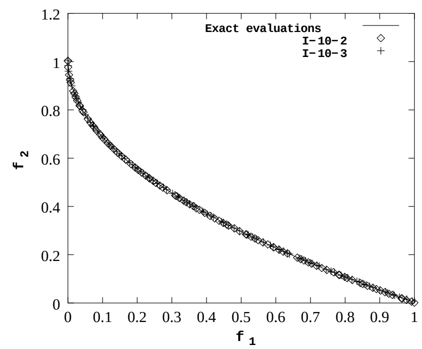

Fig. 6. Incremental model simulation results for the ZDT4 test problem

Table 2. Spread metric results of NSGA-II-ANN simulations on the ZDT4 test problem

Model Name	Spread metric value	Precise Function Evaluations	No. of hidden neurons
NSGA-II	0.386	30,200	NA
B-10-2	0.422	22,675	17
B-10-3	0.332	22,730	15
I-10-2	0.344	22,675	13
I-10-3	0.388	21,781	17

4.2 ZDT4:

It is a ten real-valued variable problem. It has a convex global Pareto-optimal front. This is a difficult unconstrained test problem as it has 100 distinct Pareto-optimal fronts, out of which only one is global. Here, all NSGA-II-ANN simulations have reached the global Pareto-optimal front. But due to presence of local sub-optimal Pareto fronts, the savings in precise function evaluations are slightly lower. Figs. 5 and 6 show the NSGA-II-ANN simulations in batch and incremental mode, respectively. Both figures show that simulations are capturing the global Pareto-optimal front. The number of precise function evaluations taken by simulations of various models in order to reach the Pareto front are given in Table 2. The saving in both batch models namely, *B*-10-2 and *B*-10-3 models are approximately 25%. The spread metric value are given in Table 2. Here, the best spread metric value is achieved by NSGA-II-ANN simulation in batch mode of training, namely, for *B*-10-3 model. The two incremental models, *I*-10-2 and *I*-10-3 also show better spread metric results. Only one simulation, i.e. *B*-10-2 gives slightly poor spread when compared to NSGA-II simulation running only with precise function evaluations.

5 Conclusions

The real world application of optimization finds many computationally expensive problems. This computational load is still higher for multi-objective optimization. Hence there is a need for a generic multi-objective optimization procedure which can work reliably with approximate models. In this paper, NSGA-II is used in conjunction of ANN. The NSGA-II-ANN procedure performance is tested on two test problems. The aspects tested here are real variable unconstrained problems with discontinuous non-convex and continuous convex Pareto fronts. NSGA-II-ANN simulations show savings in precise function evaluations of about 25% to 50% with good convergence and spread. The batch model B-10-3 shows consistently better performance in all problems and is recommended. The question of using different approximation technique other then ANN remains an open question which needs further detailed explorations.

References

1. Jin, Y., and Sendhoff, B.: Fitness approximation in evolutionary computation - A survey. In *Proceedings, Genetic and Evolutionary Computation Conference, 2002*. Morgan Kaufmann, 2002, pp. 1105-1112.
2. Nain, P.K.S., and Deb, K.: Computationally effective search and optimization procedure using coarse to fine approximation. In *Proceedings, Congress on Evolutionary Computation, 2003*. IEEE Computer Society Press, 2003, pp. 2081-2088.
3. Deb, K.: *Multi-Objective Optimization Using Evolutionary Algorithms*. First Edition, Chichester, Uk: Wiley, 2001.
4. Deb, K. , Agarwal, S. , Pratap, A. , and Meyarivan, T.: A fast elitist non-dominating sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Proceedings, 6th International Conference on Parallel Problem Solving from Nature - PPSN VI*. Lecture Notes in Computer Science 1917 Springer, 2000, pp. 849-868.
5. Haykin, S.: *Neural networks a comprehensive foundation*. second edition, Singapore: Addison Wesley, 2001. pp. 208.
6. Branke, J., and Schmidt, C.: Faster convergence by means of fitness estimation. In *Soft Computing Journal*. (in press).
7. Sastry, K., Goldberg, D. E., and Pelikan, M.: Don't evaluate, inherit. In *Proceedings, Genetic and Evolutionary computation Conference, 2001*. Morgan Kaufmann, 2001, pp. 551-558.
8. Rasheed, K., and Hirsh, H.: Informed operators: speeding up genetic-algorithm-based design optimization using reduced models. In *Proceedings, Genetic and Evolutionary Computation Conference, 2000*. Morgan Kaufmann, 2000, pp. 628-635.
9. El-Beltagy, M. A., Nair, P. B., and Keane, A. J.: Metamodelling techniques for evolutionary optimization of computationally expensive problems: promises and limitations. In *Proceedings of the Genetic and Evolutionary Computation Conference, 1999*. Morgan Kaufman, 1999, pp. 196-203.
10. Ratle, A.: Accelerating the convergence of evolutionary algorithms by fitness landscape approximation. In *Proceedings, Parallel Problem Solving from Nature, 1998*. volume V, 1998, pp. 87–96.
11. Emmerich, M., Giotis, A., Ozdenir, M., Back, T., and Giannakoglou, K.: Metamodel-assisted evolution strategies. In *Proceedings, Parallel Problem Solving from Nature, 2002*. Springer, 2002, pp. 371-380.