A Fireworks-inspired Estimation of Distribution Algorithm

Li Hao
School of Electrical Engineering, Southwest Jiaotong University, Chengdu, China
E-mail: 756048395@qq.com

Abstract. Estimation of distribution algorithm (EDA) is a popular evolutionary algorithm which is obtained widely attention. On the basis of EDA, combining with the idea of fireworks algorithm, we proposed a fireworks-inspired multi-model EDA (FMEDA). In FMEDA, multiple probabilistic models are used and generate different number of solutions by introducing the idea of firework explosion. Next, each model evolves by compared with its own best solution directly. The method of setting the limit of probability is taken to prevent from premature convergence. Applying this algorithm to the function optimization problem and the knapsack problem, the experimental result shows that the proposed algorithm has better performance than CGA and PBIL.

1. Introduction
EDA was proposed in 1996, and developed rapidly after 2000[12]. As a popular evolutionary algorithm (EA), EDAs don’t use crossover and mutation operators. Instead, they describe the distribution of the solutions by the probabilistic model. Further, the candidate solutions can be gotten by sampling the model. Experiments show that the EDAs often has better performance than other EAs in solving complex problems, so it has been widely used in various fields, including function optimization, pattern recognition, system identification, intelligent traffic system and so on[3-8].

Fireworks algorithm (FWA) is an intelligent optimization method which is inspired by fireworks exploration at night. In 2010, it was firstly introduced by Tan and Zhu[9,10]. In 2015, Kumar investigated the effects of major parameters of FWA on unimodel and multi-model benchmark functions[11]. T Zhang have applied the FWA to solve the mean VaR/CVaR model[12]. T.S.Babu use fireworks algorithm to identified the unknown parameters of the double diode solar photo voltaic model, experiment results illustrate the algorithm has superior performance[13].

Because of its simple structure and high efficiency, EDA has attracted wide attention, but on the other hand, it also suffers from search stagnation in the optimization process. In this paper, we combine it with FWA and propose a fireworks-inspired multi-model EDA (FMEDA). In FMEDA, multiple probabilistic models are used to describe the solution space, and the promised solutions are gotten through sampling the models. The idea of firework explosion is introduced into the sampling operation, which enable different models to generate different number of the solutions. Each model update by direct comparison with its own best solution. And by setting the bound of probability, premature convergence may be avoided effectively. To verify the performance of FMEDA, we apply it to solve several typical optimization problems and compare with common genetic algorithm (CGA) and PBIL[14], the experimental results illustrate that the algorithm has better performance.

The rest of the paper is organized as follows. The EDA framework is presented in section 2. FMEDA is introduced in section 3. And the simulation experiment is illustrated in section 4. Finally, conclusion follows in section 5.
2. EDA

Since EDA was put forward, it has made great progress in both theoretical research and actual application. At present, various forms of EDAs have been presented, such as PBIL, UMDA\textsuperscript{15}, MIMIC\textsuperscript{16}, BOA\textsuperscript{17}. In all of them, the statistical method is used to analyze the better individuals in the population, and then a probability model describing the solution space is generated. Further, the next generation of solutions is gotten by sampling the model. Through the alternation of modeling and sampling operation, the evolution process of EDA can be implemented. Next, as a typical example of EDA, PBIL will be presented.

In PBIL, the probability model describing the solution space is a vector $p(x)=(p(x_1), p(x_2), \cdots, p(x_n))$, where $p(x_i)$ indicates the probability of taking $x_i$ as 1 and $1-p(x_i)$ is the probability of getting 0. In each generation, $M$ promised solutions are randomly generated according to $p(x)$, then their fitness are calculated and the $N$ better solutions are chosen to update $p(x)$, where $N < M$. The update process is as follows:

$$p_{t+1}(x) = (1-\alpha) p_t(x) + \alpha \frac{1}{N} \sum_{k=1}^{N} x_i^k$$

where $p_t(x)$ is the probability vector in the $t$th generation, $x_1^k, x_2^k, \cdots, x_N^k$ is the $N$ chosen individuals, $\alpha$ is the learning rate.

The procedure of PBIL can be described as follows:

1) $M$ individuals are generated randomly as the initial population $D_t$, $t=0$.
2) The fitness of $M$ individuals are calculated. If the termination conditions are met, the algorithm will be terminated, otherwise continue.
3) Select $N$ better solutions as the superior group $D_t^s$.
4) A probability model is constructed according to $D_t^s$.
5) Based on the model, generating the new generation of population, return to 2).

3. FMEDA

In FMEDA, multiple probability models are employed, which can be represented as $P = \{p_1, p_2, \cdots, p_m\}$, where $p_i=(p_i(x_1), p_i(x_2), \cdots, p_i(x_n))$, $m$ and $n$ are the number of the models and the variables, respectively. In initialization, let $p_i(x_i) = 0.5$, $i=1, 2, \cdots, m$, $j=1, 2, \cdots, n$, that is, all solutions will be gotten with the same probability.

Next, by sampling the models, the initial population can be obtained. Aimed at the determination of the number of solutions generated from each model, inspired by the phenomenon of fireworks exploration at night, we propose a novel method. When a firework explode, many sparks will generate in the adjacent area. Generally, some superior fireworks will produce more sparks to illuminate the night sky, while the inferior fireworks only produce fewer sparks. Inspiring from this phenomenon, we regard the models as fireworks and the process of getting solutions as that of generating sparks by fireworks explosion. The higher the quality of solutions generated by a model, the more the number of solutions generated by the model in the next generation, which can enhance the local search ability.

Suppose that $m$ models are sorted according to their individual best solution, which are $q_1, q_2, \cdots, q_m$. The number of solutions generated by $q_k$ is determined as follows

$$N_k = round \left( \frac{M \cdot k}{\sum_{i=1}^{m} i} \right)$$

where $M$ is the total number of solutions. By sampling all the models, we can obtain the population.

Then, aimed at the new population, all individuals are evaluated. The individual best solution of models and the global best solution are saved.

In each generation, the probability model is updated based on the formula below.
where $p'_t(x)$ is the probability in the $t$th generation. $\alpha \in (0, 1)$ is the adjusting factor. According to (3), the model learns from its best solution, and $\alpha$ can adjust learning speed. The bigger $\alpha$, the faster the learning speed. Meanwhile, the increase is also in proportion to the difference between the probability and the best solution. When the difference between $x_b$ and $p'_t(x)$ is large, the probability of getting the best solution is small, and the model will evolve towards the best solution at a high speed. And when the difference is small, the probability of getting the best solution is large, and the speed of model evolution is relatively slow, which can keep the randomness of the model.

To prevent from premature convergence of the algorithm, we set upper and lower bounds for all probabilities in the model. It was defined

$$p(x) = \begin{cases} 
\varepsilon & p(x) \leq \varepsilon \\
\frac{p(x) - \varepsilon}{1 - \varepsilon} & \varepsilon \leq p(x) \leq 1 - \varepsilon \\
1 - \varepsilon & p(x) \geq 1 - \varepsilon 
\end{cases}$$

(4)

where $0 < \varepsilon < 1$. Can be seen, this will make $p(x)$ always in $[\varepsilon, 1-\varepsilon]$, which can prevent the algorithm search stagnation because of a probability of 1 or 0.

The procedure of FMEDA is described as follows:

Begin

Initialize $P(t)$ at $t = 0$, in which, all $p_{i}^{0}(x_{i})$ in $P(0)$ are initialized with 0.5.

Generate $M$ individuals $X_1^t, X_2^t, \ldots, X_M^t$ based on uniform distribution.

Evaluate all solutions, save the best for each model and the global best solution.

While (not termination condition) do

$\quad t = t + 1$

$\quad$ Update $P(t)$ according to (3) and (4)

$\quad$ Make $M$ individuals $X_1^t, X_2^t, \ldots, X_M^t$ by sampling $P(t)$.

$\quad$ Evaluate all solutions, save the best for each model and the global best solution

End

End

4. Simulation experiments

In the experiment, several optimization problems are chosen to verify the performance of FMEDA. At the same time, CGA and PBIL are chosen for comparison.

4.1 Function optimization problem

(1) Function1

$$F_1 = 10\cos(2\pi x) + 10\cos(2\pi y) - x^2 - y^2 - 10$$

(5)

where $x, y \in [-5.12, 5.12]$. As shown in Figure 1, this function has multiple local maxima, in which the global maximum point is $(0, 0)$, the corresponding global maximum value is 10.

(2) Function2

$$F_2 = 0.5 - \frac{\sin^2 \sqrt{x^2 + y^2 - 0.5}}{(1 + 0.001(x^2 + y^2))^2}$$

(6)

where $x, y \in [-100, 100]$. As shown in Figure 2, this function has infinite local maxima, where the global maximum point is $(0, 0)$, and the corresponding global maximum is 1.
Figure 1 Function 1

Two functions are runned 30 times by three algorithms respectively. In FMEDA, \( m = 3, M = 20, \varepsilon = 0.01, r = 0.03 \). In PBIL, \( \alpha = 0.01 \). And in CGA, the crossover and mutation probability are 0.7 and 0.01, respectively. The population size in CGA and PBIL is 20, and the maximum number of iteration in the three algorithms are 300. The experimental results are shown in Table 1 and Figure 3-4.

| Function | Algorithm | Best result | Average result | Worst result | Standard deviation |
|----------|-----------|-------------|----------------|--------------|--------------------|
| F1       | CGA       | 10.0000     | 8.8720         | 5.0252       | 1.3757             |
|          | PBIL      | 10.0000     | 9.9256         | 9.0050       | 0.2541             |
|          | FMEDA     | 10.0000     | 9.9999         | 9.9989       | 0.0002             |
| F2       | CGA       | 1.0000      | 0.9420         | 0.5853       | 0.0819             |
|          | PBIL      | 1.0000      | 0.9885         | 0.9628       | 0.0092             |
|          | FMEDA     | 1.0000      | 0.9959         | 0.9903       | 0.0046             |

Can be seen from the result, three algorithms can all find the best result for two functions, which shows their effectiveness. But by comparison, the average and worst result of FMEDA is greater than CGA and PBIL, and its standard deviation is smaller than theirs, which shows that FMEDA has better stability. On the whole, FMEDA has better performance than the other two algorithms.

4.2 Knapsack problem

The knapsack problem is a typical combinatorial optimization problem which can be described as: given \( n \) items and a knapsack, select a subset of the items so as to maximize the profit \( f(X) = \sum_{i=1}^{n} p_i x_i \) and subject to \( \sum_{i=1}^{n} w_i x_i \leq C \), where \( x_i \in \{0, 1\} \), \( 1 \leq i \leq n \), \( C \) is the capacity of the knapsack, \( w_i \) and \( p_i \) is the weigh and the profit of the \( i \)-th item, respectively. \( x_i = 1 \) if the \( i \)-th item is selected, otherwise \( x_i = 0 \).
In the experiments, three knapsack problems with 200, 400, 800 items are considered. And the parameter is set to $w_i = \text{uniformly random \cite{1,10}}$, $p_i = w_i + 5$, $C = (1/2)\sum_{i=1}^{n} w_i$. All problems are runned 100 times by three algorithms respectively. In all of them, the maximum number of iteration is 500. In FMEDA, $m = 5$, $M = 30$, $\varepsilon = 0.01$, $r = 0.03$. In PBIL, $\alpha = 0.01$. And in CGA, the crossover and mutation probability are 0.7 and 0.01, respectively. The population size in CGA and PBIL is 30. For every binary string in the three algorithms, repair method described in\cite{18} is taken. The experimental results are shown in Table 2 and Figure 5.

| Items | Algorithm | Best result | Average result | Worst result | Standard deviation |
|-------|-----------|-------------|----------------|--------------|-------------------|
| 200   | CGA       | 1233.5      | 1232.0         | 1228.1       | 2.3               |
|       | PBIL      | 1233.5      | 1233.5         | 1233.5       | 0                 |
|       | FMEDA     | 1233.5      | 1233.5         | 1233.5       | 0                 |
| 400   | CGA       | 2474.0      | 2468.6         | 2463.8       | 3.3               |
|       | PBIL      | 2479.0      | 2475.5         | 2474.0       | 2.3               |
|       | FMEDA     | 2479.0      | 2478.2         | 2474.0       | 1.8               |
| 800   | PBIL      | 4914.3      | 4909.7         | 4909.3       | 4.5               |
|       | FMEDA     | 4914.3      | 4914.2         | 4909.3       | 0.5               |

![Figure 5 Optimization curves](a) Item 200 (b) Item 400 (c) Item 800

It can be seen from the table 2, in all three cases, FMEDA and PBIL are superior to CGA in both optimization accuracy and stability. Further comparing FMEDA with PBIL, when the number of items is 200, the best solutions can be found by both algorithms and in all tests. And in the cases of 400 and 800 items, the best solutions can be found by the two algorithms, but the average results show that FMEDA is more stable than PBIL, indicating that FMEDA is better than PBIL. Next, figure 5 shows that CGA has faster convergence speed in the former period, but in the later, the premature convergence is easy to occur. PBIL is better than CGA on the whole, but its convergence speed is relatively slow. By contrast, FMEDA is superior to the other two algorithms and can achieve better optimization results.

5. Conclusion

EDA is an intelligent evolutionary algorithm based on the probability model, and has excellent performance. In this paper, learning from the idea of the FWA, a fireworks-inspired multi-models EDA is carried out. Used it to solve several typical optimization problem, and compared with the CGA and PBIL, the experiment results illustrate that FMEDA has better optimization ability, and its performance is better than the other two algorithms.
References

[1] Zhou S.D, Sun Z.Q. (2007) A survey on Estimation of distribution algorithms. ACTA AUTOMATICA SINICA. 33: 113-124.

[2] Hauschild M, Pelikan M. (2011) An introduction and survey of estimation of distribution algorithms. Swarm and evolutionary computation. 1: 111-128.

[3] Fang H, Zhou A.M, Zhang H. (2018) Information fusion in offspring generation: A case study in DE and EDA. Swarm and Evolutionary computation. 014.

[4] Lin Z.Y, Xie G.B, Wang L.J. (2017) Mutual information estimation of distribution algorithm with hybrid sampling mechanism. Control and Decision. 32: 829-835.

[5] Cesar R M, Bengoetxea F, Bloch I. (2005) Inexact graph matching for model based recognition: Evaluation and comparison of optimization algorithms. Pattern Recognition, 38: 2099-2113.

[6] Han P, Zhang T, Dong Z, Zhang Y. (2017) System identification using a hybrid estimation of distribution algorithm with minimum entropy and its application. Proceedings of the CSEE. 37:6363-6372.

[7] Gao Y, Liu W. X. (2017) Functional link artificial neural network with cloud estimation of distribution algorithm for traffic flow forecast. 2017 10th International Symposium on Computational Intelligence and Design(ISCID). 194-197.

[8] Shi W, Chen W. N, Lin Y. (2017) An Adaptive Estimation of Distribution Algorithm for Multi-Policy Insurance Investment Planning. IEEE Transaction on Evolutionary Computation. 1-14.

[9] Tan Y, Zhu Y. (2010) Fireworks algorithm for optimization. International conference in swarm intelligence. Berlin: Springer. 355-364.

[10] Tan Y, Yu C, Zheng S. (2013) Introduction to Fireworks algorithm. International journal of swarm intelligence research. 4: 39-71.

[11] Kumar V, Chhabra J.K, Kumar D. (2015) Optimal Choice of Parameters for Fireworks Algorithm. Procedia Computer Science. 70: 334-340.

[12] Zhang T, Lu Z. (2017) Fireworks algorithm for mean-VaR/CVaR models. Physica A. 483: 1-8.

[13] Babu T.S, Ram J.P, Sangeetha K. (2016) Parameter extraction of two diode solar PV model using fireworks algorithm. Solar energy. 140: 265-276.

[14] Baluja S. (1994) Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning. Technical Report CMU-CS-94-163, Pittsburgh, PA: Carnegie Mellon University.

[15] Mühlenbein H, Paass G. (1996) From recombination of genes to the estimation of distributions I. Binary parameters. Parallel Problem Solving from Nature - PPSN IV, Berlin. 178-187.

[16] De Bonet, J S, Isbell C L. (1997) MIMIC: Finding optima by estimating probability densities. Advances in Neural Information Processing Systems, Cambridge: MIT Press, 9:424-430.

[17] Pelikan M, Goldberg D E, Cantú-Paz E. (2000) Bayesian Optimization Algorithm, Population Sizing, and Time to Convergence. IlliGAL Report No. 2000001, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, Illinois.

[18] Han K. H, Kim J. H. (2002) Quantum-inspired Evolutionary Algorithm for a Class of Combinatorial Optimization. IEEE Trans Evolutionary Computation, 6: 580–593.