Best Guided Backtracking Search Algorithm for Numerical Optimization Problems

Wenting Zhao

E-mail: inweriok@qq.com

MLA Lab, Shandong University

机器学习与数据挖掘实验室

Machine Learning and Data Mining Lab.

Oct. 6. 2016

Outline

- Introduction to Evolutionary Algorithms
- Backtracking Search Optimization Algorithm (BSA)
- Best Guided BSA
- Experiments
- Conclusions
- References

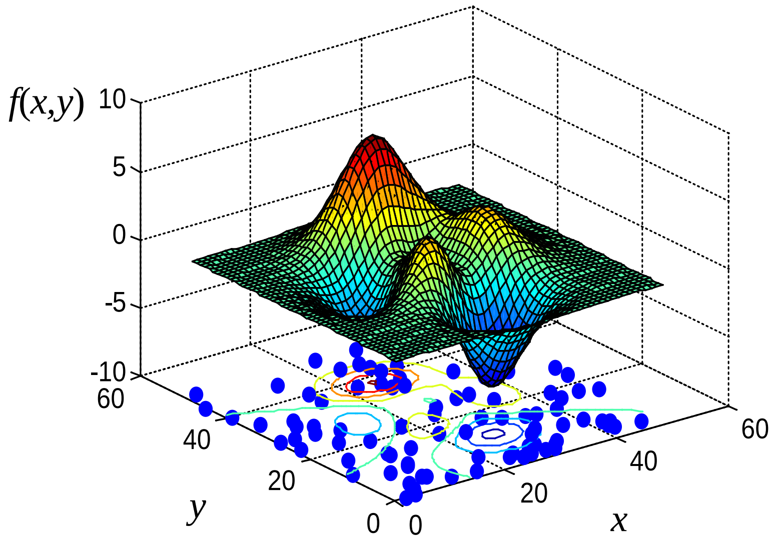
What is optimization problem?

General form:

$$
\begin{array}{lll}
\min\limits_{\mathbf{x}} & f(\mathbf{x}) & \mathbf{x} \in \Re^n \\
\text{s. t.} & g_i(\mathbf{x}) \le 0 & i = 1 \ldots m \\
 & \mathrm{h}_j(\mathbf{x}) = 0 & j = m+1 \ldots q \\
 & l_k \le x_k \le u_k & k = 1 \ldots n
\end{array}
$$

For example:

$$
\begin{array}{ll}
\min\limits_{\mathbf{x}} & f(\mathbf{x}) = x_1 + x_2 \\
\text{s. t.} & -x_1 + x_2 + 50 \le 0 \\
 & 0 \le x_1 \le 100 \\
 & 0 \le x_2 \le 100
\end{array}
$$



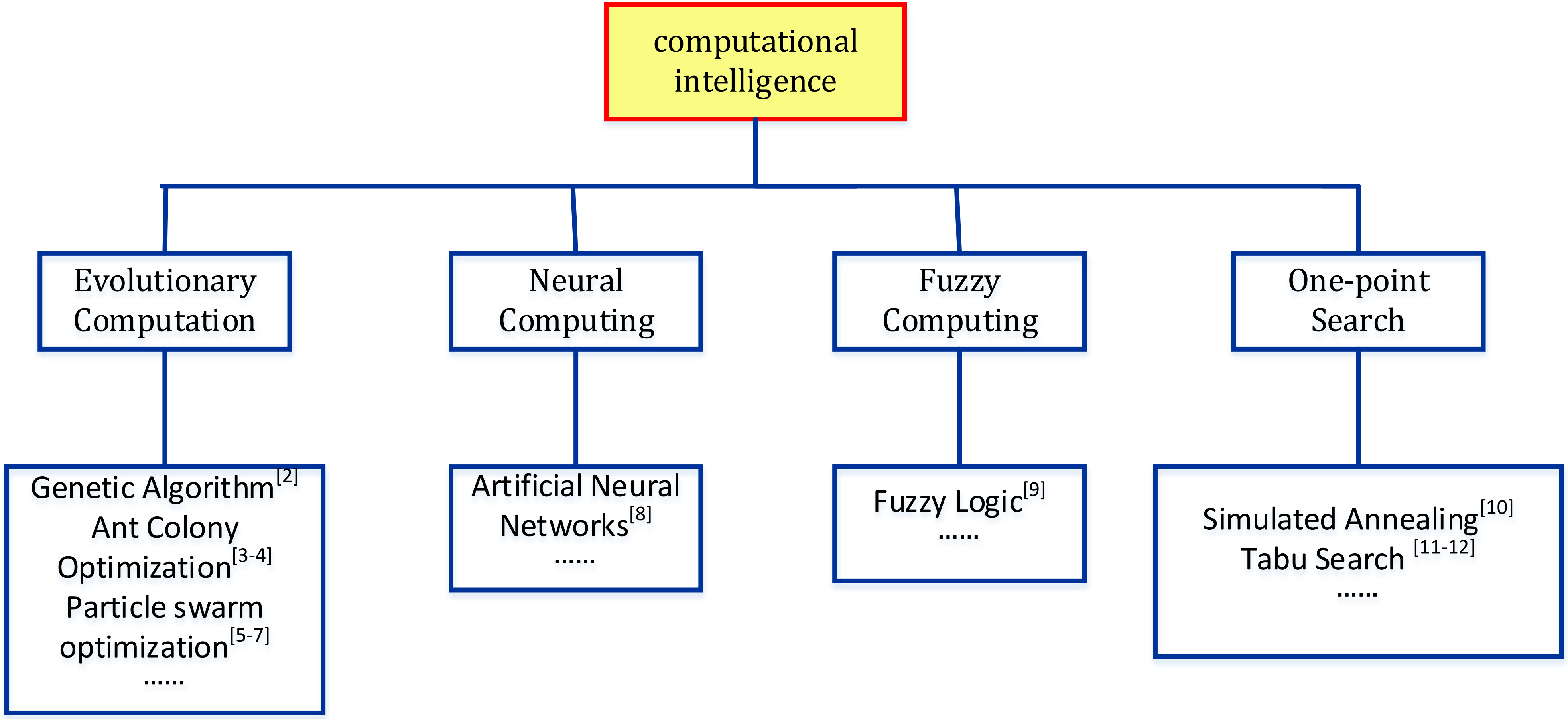
What is computational intelligence?

- **Computational Intelligence**[1] is the study of adaptive mechanisms to enable or facilitate *intelligent behavior* in *complex and changing environments*.

- **Computational Intelligence** is the
  computational part of the artificial intelligence.

- **Computational Intelligence** investigates nature-inspired computational methodologies and approaches to address complex real-world problems where traditional approaches, i.e., first principles modeling or explicit statistical modeling, are ineffective or infeasible.

General categories of CI



Evolutionary computation

- **Evolutionary Computation (EC)**: which is a kind of optimization methodology inspired by the mechanisms of biological evolution and behaviors of living organisms

- **Evolutionary algorithms (EAs)** are a meta-heuristic stochastic search technique based on Darwinian "survival-of-the-fittest" principle, leveraging computationally useful aspects of natural evolution processes.

Evolutionary algorithm

- Genetic Algorithm (*GA*), Differential Evolution (*DE*), Evolutionary Programming (*EP*), Genetic Programming (*GP*), et al.

- Swam Intelligence (*SI*) based Algorithm: Particle swarm optimization (*PSO*), Ant Colony Optimization (*ACO*), Cuckoo Search (*CS*), Harmony Search (*HS*), et al.

- A good SI algorithm survey can be found in [CoRR, abs/1307.4186, 2013]

Evolutionary algorithm

- EAs: can be divided into two groups

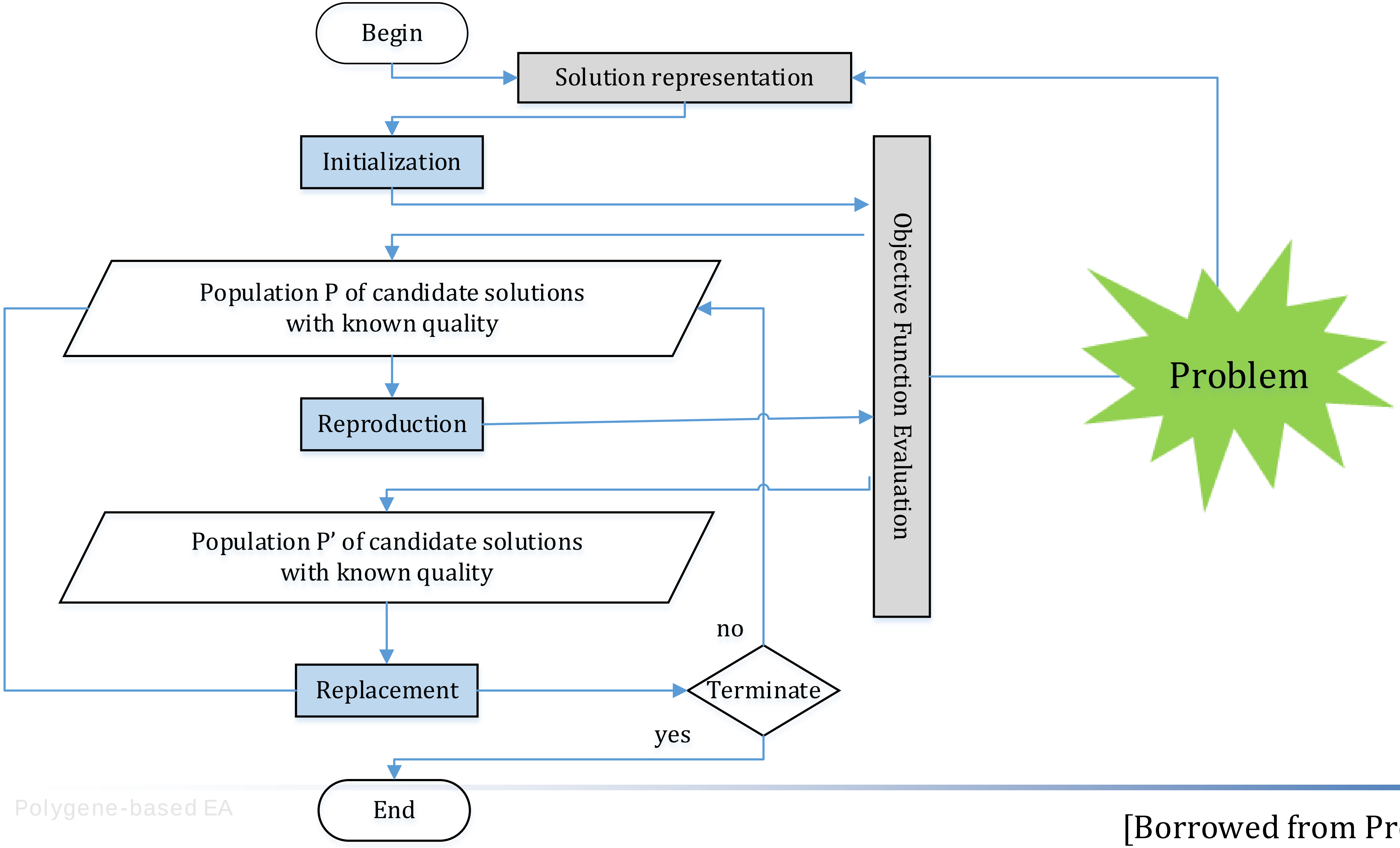
Constructed Algorithms: ACO, HS, GA, et al.

- how to construct a solution during evolutionary process

Evolved Algorithms: PSO, CS, DE, **BSA**, et al.

- how to have a solution evolved during evolutionary process

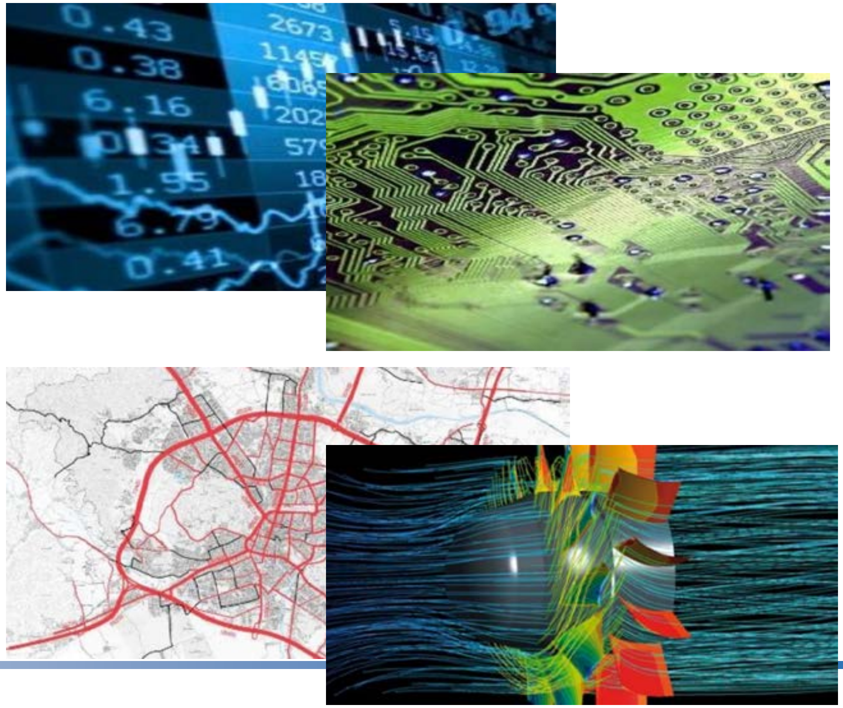
General paradigm of EAs



[Borrowed from Prof. Kai Qin’s slides]

Evolutionary algorithms

- Major characteristics of EAs:
  - Population(s) of candidate solutions
  - Fitness-guided population variations during search
  - Variation with inheritance
  - Replacement (survival-of-the-fittest)



Backtracking Search Optimization Algorithm (BSA)

- A new evolutionary algorithm proposed by Pinar Civicioglu in 2013[13].
- Has a simple structure that is effective, fast and capable of solving multimodal problems.
- Can be divided into five processes:
  - Initialization
  - Selection-I
  - Mutation
  - Crossover
  - Selection-II

Initialization: initializes the population P and historical population $oldP$.

$$P_{i,j} \sim U(low_j, up_j)$$
$$oldP_{i,j} \sim U(low_j, up_j)$$

where i =1,2,… ,N and j=1,2,…,D.

N: population size
D: dimension of the problem
low: lower bound of the element
up: upper bound of the element

$$N\left\{\overbrace{\begin{matrix} p_{1,1} & p_{1,2} & p_{1,3} & p_{1,4} & p_{1,5} \\ p_{2,1} & p_{2,2} & p_{2,3} & p_{2,4} & p_{2,5} \\ p_{3,1} & p_{3,2} & p_{3,3} & p_{3,4} & p_{3,5} \\ p_{4,1} & p_{4,2} & p_{4,3} & p_{4,4} & p_{4,5} \end{matrix}}^{D}\right. = \begin{matrix} 4 & 50 & 2 & 0.5 & 6 \\ 3 & 33 & 1 & 0.2 & 8 \\ 6 & 49 & 3 & 0.7 & 3 \\ 2 & 37 & 6 & 0.4 & 1 \end{matrix}$$

Selection-I: updates the historical population *oldP* to be used for calculating the search direction. This process consists of 2 steps.

Step 1— Redefine *oldP* at the beginning of each iteration.

$$oldP=\begin{cases} P & ,a<b \\ oldP & ,otherwise \end{cases} \quad (a,b\sim U(0,1))$$

Step 2— Randomly change the order of the individuals in *oldP*

$$oldP := permuting(oldP)$$

5	62	6	0.4	9
9	55	8	0.8	2
2	35	9	0.7	5
7	72	5	0.3	6

oldP before permuting

2	35	9	0.7	5
5	62	6	0.4	9
9	55	8	0.8	2
7	72	5	0.3	6

oldP after permuting

Mutation: generates the initial form of the trial population *Mutant*

$$\boxed{Mutant} = \boxed{P} + \boxed{scale\_factor} \cdot \boxed{(oldP-P)}$$

the initial form of the trial population

population P

the amplitude of the search-direction matrix

the search-direction matrix

$$\begin{bmatrix} 0.7 & 25.8 & 13.2 & 0.8 & 4.3 \\ 6.3 & 79.8 & 9.1 & 0.2 & 9.6 \\ 10.8 & 58.7 & 11.0 & 0.8 & 1.3 \\ 10.1 & 93.4 & 4.4 & 0.2 & 9.1 \end{bmatrix} = \begin{bmatrix} 4 & 50 & 2 & 0.5 & 6 \\ 3 & 33 & 1 & 0.2 & 8 \\ 6 & 49 & 3 & 0.7 & 3 \\ 2 & 37 & 6 & 0.4 & 1 \end{bmatrix} + 3 \cdot randn \cdot \left(\begin{bmatrix} 2 & 35 & 9 & 0.7 & 5 \\ 5 & 62 & 6 & 0.4 & 9 \\ 9 & 55 & 8 & 0.8 & 2 \\ 7 & 72 & 5 & 0.3 & 6 \end{bmatrix} - \begin{bmatrix} 4 & 50 & 2 & 0.5 & 6 \\ 3 & 33 & 1 & 0.2 & 8 \\ 6 & 49 & 3 & 0.7 & 3 \\ 2 & 37 & 6 & 0.4 & 1 \end{bmatrix} \right)$$

Crossover: generates the final form of the trial population. This process consists of 2 steps.

Step 1— Calculate a binary integer-valued matrix *map*:

$$
map = \begin{matrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{matrix}
$$

Step 2— Update Mutant:

$$
Mutant_{i,j} = \begin{cases} P_{i,j} & , \quad map_{i,j}=1 \\ Mutant_{i,j} & , \quad otherwise \end{cases}
$$

Mutant

$$
\begin{matrix} 0.7 & 25.8 & 13.2 & 0.8 & 4.3 \\ 6.3 & 79.8 & 9.1 & 0.2 & 9.6 \\ 10.8 & 58.7 & 11.0 & 0.8 & 1.3 \\ 10.1 & 93.4 & 4.4 & 0.2 & 9.1 \end{matrix}
$$

P

$$
\begin{matrix} 4 & 50 & 2 & 0.5 & 6 \\ 3 & 33 & 1 & 0.2 & 8 \\ 6 & 49 & 3 & 0.7 & 3 \\ 2 & 37 & 6 & 0.4 & 1 \end{matrix}
$$

$$
\begin{matrix} 0.7 & 50 & 2 & 0.8 & 4.3 \\ 6.3 & 79.8 & 9.1 & 0.2 & 8 \\ 6 & 58.7 & 11.0 & 0.8 & 3 \\ 2 & 93.4 & 4.4 & 0.2 & 9.1 \end{matrix}
$$

Mutant

Selection-II: select and update the population to be used in the next generation based on greedy selection.

$$P_i^{next} = \begin{cases} Mutant_i\,, & f(Mutant_i) \le f(P_i) \\ P_i\,, & otherwise \end{cases}$$

f is the objective function

$Mutant_i$	$f(Mutant_i)$
(0.7, 50, 2, 0.8, 4.3)	120
(6.3, 79.8, 9.1, 0.2, 8)	270
(6, 58.7, 11.0, 0.8, 3)	331
(2, 93.4, 4.4, 0.2, 9.1)	259

VS

P_i	$f(P_i)$
(4, 50, 2, 0.5, 6)	146
(3, 33, 1, 0.2, 8)	298
(6, 49, 3, 0.7, 3)	210
(2, 37, 6, 0.4, 1)	345

0.7	50	2	0.8	4.3
6.3	79.8	9.1	0.2	8
6	49	3	0.7	3
2	93.4	4.4	0.2	9.1

Pros and Cons of BSA

- Pros:
  - ✓ Simple— has a single parameter and not sensitive to the parameter value
  - ✓ Previous experiences— uses an external archive *oldP* to store experiences gained from previous generation
  - ✓ Exploration ability(global search ability)— the permuting makes the individuals be chosen randomly in the mutation operator
- Cons:
  - ➢ Exploitation ability(local search ability)— by utilizing experiences, BSA may be led to poor exploitation ability on later iteration stage. [15]

Key factors influencing the efficacy of EAs

- Algorithm aspects:
  - Search strategy and the associated parameter settings
  - Population size(s) and structure(s)
  - Computation budget
- Problem aspects:
  - Objective function formulation
  - Solution representation
- Various population structures have been studied
- Key efforts have been put on search strategies

Best Guided BSA

- Best guided operator is proposed to hybridize with BSA: Best guided operator enhances exploitation ability because the mutant individuals are strongly attracted around the current best individual.

new parameter

- BG operator is performed during the later stage of evolution and the original mutation operator is called during the early stage.

  - Best guided operator

$$Mutant_i = P_i + F \times (P_{best} - P_i)$$

4. Experiments

Experiment 1:

- BGBSA is verified on CEC-2013 benchmark test suite including unimodal functions F1–F5, basic multimodal functions F6–F20 and composition functions F21 – F28 [14] compared with BSA.
- Evaluation: error values.
- Parameter setting:

Population size	Dimension	Stage control	FES	DimRate
30	30	0.75	300000	1

The Effect of BGBSA at 30-dim CEC-2013

	BSA	BGBSA		
	AvgEr ±StdEr	AvgEr ±StdEr		P value
Fun1	1.01e-30±3.49e-30	1.01e-30±3.49e-30	=	1.000000
Fun2	1.37e+06±5.35e+05	**4.26e+05±2.13e+05**	+	0.000020
Fun3	**4.54e+06±4.60e+06**	6.44e+06±9.98e+06	=	0.861162
Fun4	1.27e+04±3.58e+03	**4.95e+03±1.93e+03**	+	0.000018
Fun5	0.00e+00±0.00e+00	5.05e-31±2.52e-30	=	1.000000
Fun6	2.74e+01±2.47e+01	2.74e+01±2.64e+01	=	0.492633
Fun7	6.82e+01±1.35e+01	**5.94e+01±1.36e+01**	=	0.082653
Fun8	2.09e+01±6.72e-02	2.09e+01±3.98e-02	=	0.618641
Fun9	2.73e+01±2.75e+00	**2.58e+01±2.86e+00**	=	0.078001
Fun10	1.90e-01±1.42e-01	**1.49e-01±1.59e-01**	=	0.287862
Fun11	7.96e-02±2.75e-01	**3.98e-02±1.99e-01**	=	1.000000
Fun12	8.71e+01±2.14e+01	**8.36e+01±1.74e+01**	=	0.492633
Fun13	1.49e+02±2.53e+01	**1.42e+02±2.19e+01**	=	0.287862
Fun14	3.56e+00±1.73e+00	**1.52e+00±1.28e+00**	+	0.000157

* The **bold in red** indicates the best performance

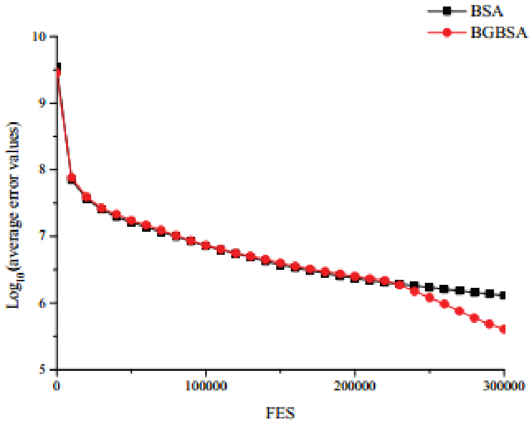
The Effect of BGBSA at 30-dim CEC-2013

Continue…

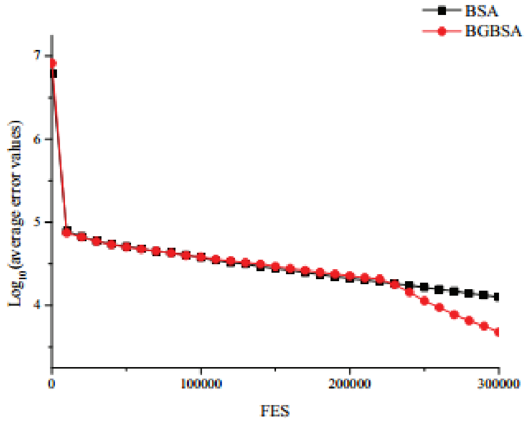
	BSA	BGBSA		
	AvgEr ±StdEr	AvgEr ±StdEr		P value
Fun14	3.56e+00±1.73e+00	**1.52e+00±1.28e+00**	+	0.000157
Fun15	3.81e+03±4.16e+02	**3.48e+03±4.60e+02**	+	0.002259
Fun16	1.26e+00±1.66e-01	**1.10e+00±3.01e-01**	+	0.021418
Fun17	3.09e+01±1.75e-01	**3.06e+01±1.06e-01**	+	0.000029
Fun18	1.16e+02±1.99e+01	**9.78e+01±1.90e+01**	+	0.002947
Fun19	**1.07e+00±2.11e-01**	1.13e+00±2.38e-01	=	0.312970
Fun20	1.14e+01±4.91e-01	**1.10e+01±6.39e-01**	+	0.017253
Fun21	**2.67e+02±8.00e+01**	2.90e+02±4.91e+01	=	0.142970
Fun22	4.33e+01±1.72e+01	**2.59e+01±1.12e+01**	+	0.000602
Fun23	4.36e+03±5.00e+02	**4.09e+03±3.81e+02**	+	0.042207
Fun24	**2.33e+02±1.03e+01**	2.35e+02±1.16e+01	=	0.396679
Fun25	2.89e+02±8.80e+00	**2.81e+02±1.44e+01**	+	0.028314
Fun26	2.00e+02±1.32e-02	2.00e+02±7.07e-03	+	0.000029
Fun27	8.89e+02±1.45e+02	**8.85e+02±1.10e+02**	=	0.798248
Fun28	3.00e+02±1.95e-13	3.00e+02±1.62e-13	=	0.637352
+/=/-				12/16/0

* The **bold in red** indicates the best performance

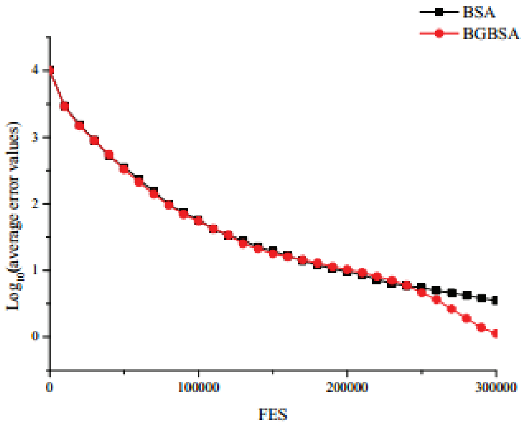
The Effect of BGBSA at 30-dim CEC-2013



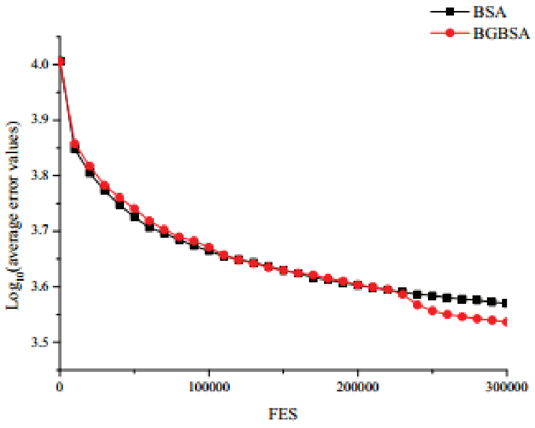
(a) F_2



(b) F_4



(c) F_{14}



(d) F_{15}

the convergence curves of BSA and BGBSA for selected benchmark functions.

Experiment 2:

- Compare with other variants of BSA.
- HBD[15], IBSA[16], COOBSA[17] and unmodified backtracking search algorithm (BSA).
- Evaluation method: Friedman test

D	BGBSA	HBD	IBSA	COOBSA
10	**1.93**	**1.93**	2.14	4
30	**1.84**	1.89	2.27	4
50	**1.75**	1.93	2.32	4

* The **bold in red** indicates the best performance

Experiment 3:

- Compared with other algorithms proposed during CEC-2013.
- NBIPOP-aCMA[17], fk-PSO [18], SPSO2011[19], SPSOABC[20], and PVADE[21] .
- Evaluation: Friedman test ranking

Methods	NBIPOP-aCMA	BGBSA	SPSOABC	fk-PSO	PVADE	SPSO2011
Ranking	**1.8**	3.11	3.3	3.57	3.93	5.29

* The **bold in red** indicates the best performance

Experiment 4:

- The effect of the stage control parameter α.
- Evaluation: Friedman test ranking

α	0.55	0.65	0.75	0.85	0.95
Ranking	3.09	2.71	**2.61**	3.05	3.54

* The **bold in red** indicates the best performance

5. Conclusions

- Best guided operator is designed.
- The proposed algorithm combined the historical experience and the experience from the best individual obtained so far to enhance the convergence speed on the later stage of iteration.
- Experiments demonstrate the competitive performance of the proposed method.

6. References

[1] Engelbrecht A P. Computational intelligence: an introduction. John Wiley & Sons, 2007.

[2] Goldberg D E. Genetic algorithms in search optimization and machine learning. Reading Menlo Park: Addison-wesley, 1989.

[3] Dorigo M, Birattari M, Stützle T. Ant colony optimization. Computational Intelligence Magazine, IEEE, 2006, 1(4): 28-39.

[4] Dorigo M, Gambardella L M. Ant colony system: a cooperative learning approach to the traveling salesman problem. Evolutionary Computation, IEEE Transactions on, 1997, 1(1): 53-66.

[5] Kennedy J. Particle swarm optimization. Encyclopedia of machine learning. Springer US, 2011: 760-766.

[6] Eberhart R C, Kennedy J. A new optimizer using particle swarm theory. Proceedings of the sixth international symposium on micro machine and human science. 1995, 1: 39-43.

[7] Zhang Y, Wang S, Ji G. A comprehensive survey on particle swarm optimization algorithm and its applications. Mathematical Problems in Engineering, 2015, 2015: 1.

[8] Haykin S, Network N. A comprehensive foundation. Neural Networks, 2004, 2(2004).

[9] Kusko B. Fuzzy thinking: The new science of fuzzy logic. 1993.

[10] S P, Morgan B J T. Optimization using simulated annealing. The Statistician, 1995: 241-257.

[11] Glover F. Tabu search-part I. ORSA Journal on computing, 1989, 1(3): 190-206.

[12] Glover F. Tabu search—part II. ORSA Journal on computing, 1990, 2(1): 4-32.

[13] Civicioglu, P. (2013). "Backtracking search optimization algorithm for numerical optimization problems." Applied Mathematics and Computation 219(15): 8121-8144.

[14] Liang J, Qu B, Suganthan P, Hernández-Díaz A G. Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report, 2013

[15] Wang, L., Zhong, Y., Yin, Y., et al.: A hybrid backtracking search optimization algorithm with differential evolution. Mathematical Problems in Engineering (2015)

[16] Zhao, W., Wang, L., Yin, Y.: An improved backtracking search algorithm for constrained optimization problems. In: 7th International Conference on Knowledge Science, Engineering and Management, pp. 222–233. Springer (2014)

[17] Xu, Q., Guo, L., Wang, N., Li, X.: Opposition-based backtracking search algorithm for numerical optimization problems. In: 5th International Conference on Intelligence Science and Big Data Engineering, pp. 223–234. Springer, Switzerland(2015)

[18] Loshchilov, I.: Cma-es with restarts for solving cec 2013 benchmark problems. In:

2013 IEEE Congress on Evolutionary Computation, pp. 369–376. IEEE Press (2013)

[19] Nepomuceno, F.V., Engelbrecht, A.P.: A self–adaptive heterogeneous pso for realparameter optimization. In: 2013 IEEE Congress on Evolutionary Computation, pp.

361–368. IEEE Press (2013)

[20] Zambrano-Bigiarini, M., Clerc, M., Rojas, R.: Standard particle swarm optimisation 2011 at cec–2013: A baseline for future pso improvements. In: 2013 IEEE Congress on Evolutionary Computation, pp. 2337–2344. IEEE Press (2013)

[21] El–Abd, M.: Testing a particle swarm optimization and artificial bee colony hybrid algorithm on the cec13 benchmarks. In: 2013 IEEE Congress on Evolutionary Computation, pp 2215–2220. IEEE Press (2013)

[22] Coelho, L.D.S., Ayala, V.H., Freire, R.Z.: Population's variance-based adaptive differential evolution for real parameter optimization. In: 2013 IEEE Congress on Evolutionary Computation, pp. 1672–1677. IEEE Press (2013)

Thanks for your attention!