SOCIAL COGNITIVE OPTIMIZATION FOR NONLINEAR PROGRAMMING PROBLEMS

XIAO-FENG XIE [1] *, WEN-JUN ZHANG[1], ZHI-LIAN YANG[1]

[1]Institute of Microelectronics, Tsinghua University, Beijing 100084, P. R. China

*Email: xiexiaofeng@tsinghua.org.cn

Abstract: Social cognitive optimization (SCO) for solving nonlinear programming problems (NLP) is presented based on human intelligence with the social cognitive theory (SCT). The experiments by comparing SCO with genetic algorithms on some benchmark functions show that it can get high-quality solutions efficiently, even by only one learning agent.

Keywords: Evolutionary computation, social cognitive theory, vicarious learning, nonlinear programming

1. Introduction

The general nonlinear programming problems (NLP) can be defined as finding $\vec{x} \in S \subseteq \mathbb{R}^{D}$ such that

$$\begin{cases} f(\vec{x}) = \min\{ f(\vec{y}); \vec{y} \in S\}, \\ g_j(\vec{x}) \le 0, \quad for\ j \in [1, m] \end{cases} \tag{1}$$

Where $\vec{x} = (x_1,..., x_d,..., x_D)$ ($d \in [1,\ D]$), $x_d \in [l_d, u_d]$, l_d and u_d are lower and upper boundary values. f and g_j are functions on S; S is a D-dimensional search space defined as a Cartesian product of domains of variables x_d's. The set of points, which satisfying all the constraint functions g_j, is denoted as feasible space (S_F).

Natural evolution, qua self-organized system [2], allows adaptation to the prevailing environment, which makes the systems extraordinarily flexible and robust against perturbation of the outer conditions.

The natural systems have progressively evolved from biological to social systems. The idea of applying the principle of natural evolution to artificial systems has seen impressive growth in the past few years, known as evolutionary algorithms (EAs) [3-9], as shown in Fig. 1. Based on the emergence of species by genetic evolution in biological systems [3], some EAs, such as evolutionary programming (EP) [4], evolution strategies (ES) [5], genetic algorithms (GA) [6], etc, are proposed. And based on the emergence of low-level swarm intelligence [7] in social insect systems, some EAs, such as particle swarm optimization (PSO) [8], ant colony optimization (ACO) [9], etc, are proposed.

Moreover, it seems that human has higher adaptability than insect swarm. The advanced social intelligence of the human learning mechanism than the primate [10], has well studied by the social cognitive theory (SCT) [11-13], which argues that human learning is done by *observing* the behavior of others and the outcomes of other's behaviors with symbolic capability.

In this paper, a new algorithm, called social cognitive optimization (SCO), for solving NLP problem is proposed based on SCT. In the next section, some key constructs of SCT are reviewed. Then in section 3, the algorithm SCO is described. In section 4, eight benchmark functions and the experimental settings for SCO and GA in literature [1] are listed. In section 5, the experimental results are reported and discussed. In the last section, we conclude the paper.

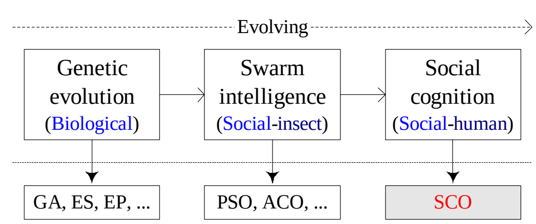

FIG. 1. The evolving of natural systems and the corresponding evolutionary computational techniques

2. Overview of social cognitive theory

The SCT defines human behavior as a reciprocal interaction of personal factors, behavior, and the environment [12]. It contends that behavior is largely regulated antecedently through cognitive processes.

Therefore, response consequences of a behavior are used to form expectations of behavioral outcomes. The ability gives humans the capability to predict the outcomes of their behavior before it is performed.

The SCT suggests that the mind is an active force that constructs one's reality, selectively encodes information, performs behavior on the basis of expectations, and imposes structure on its own actions [14]. Through feedback and reciprocity, a person's own reality is formed by the interaction of the environment and one's cognitions, which change over time as a function of maturation and experience by vicarious learning through observing the symbolic models.

2.1 Symbolizing Capability

S*ymbols* serve as the mechanism for thought although most external influences effect behavior through cognitive processes [12]. Through the formation of symbols, such as images or languages, humans are able to give meaning and contiguity to their experiences. The symbolic capability enables humans to store knowledge in their memory to guide future behaviors even as no live models available. It is through this process that humans are able to model observed behavior, which is distinguish from swarm intelligence.

Symbols provide the mechanism that allows for cognitive problem solving and engaging in foresighted action, i.e., one can think through the consequences of a behavior without actually performing it[11]. Researches indicate that much of human thought is linguistically based, and that there is a correlation between cognitive development and knowledge acquisition [12].

2.2 Vicarious Capability

Vicarious processes refer to the human ability to learn new, never before performed, behaviors through the observation of others by *observational learning* [11]. This information can then be coded (into knowledge) and used as a guide for future action. Vicarious learning is important in that it enables humans to form patterns of behavior quickly, avoiding time-consuming trial and error, as well as avoiding costly and even fatal mistakes. In addition, vicarious capabilities allow one to explore activities for the attainment of new knowledge that would normally be out of reach due to constraints on time, resources, and mobility. For example, Internet has vastly expanded the range of models and behaviors one is exposed to every day, allowing people to transcend the boundaries of their own environment [13].

3. Social cognitive optimization (SCO)

Definition 1: *Knowledge point* is located in the knowledge space (i.e. the search space *S*), which is described by the location $\vec{x}$ and its level (i.e. fitness).

Definition 2: *Library* is a table with a specified size that includes a set of *knowledge points*.

Definition 3: *Learning agent* is a behavioral individual, which possesses of a *knowledge point* in the *library*.

Definition 4: For point $\vec{x}_1$ and $\vec{x}_2$, the *neighborhood searching* for $\vec{x}_2$ based on the reference of $\vec{x}_1$ is selecting a new point $\vec{x}'$, which for the *d*th dimension

$$\vec{x}_d' = \vec{x}_{1,d} + 2*RAND()*(\vec{x}_{2,d} - \vec{x}_{1,d}) \quad (1)$$

Where *RAND*() is a random value in (0,1). $\vec{x}_1$, $\vec{x}_2$ are defined as *reference* and *central point*, respectively.

The principle of SCO is shown in Fig. 2. The optimization process is acted by a set of learning agents. The library is provided for the symbolizing capability. For the Vicarious capability, the tournament selection is used for the model selection, and then a learning agent will perform the observational learning through the neighborhood searching for the better knowledge by observing the selected model.

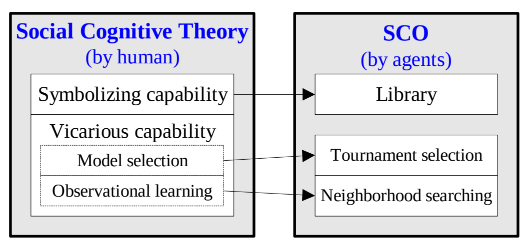

FIG. 2. Principle of social cognitive optimization

Fig. 3 shows the flowchart of SCO. Suppose the number of knowledge points in library is N_{pop}, the number of learning agents is N_c, then it is realized as:

a) Initialization: (1). Create all the N_{pop} knowledge points (include the location $\vec{x}$ and its level) in library at random; (2). Allocate each learning agent to occupy an knowledge point in library at random;

b) Vicarious learning: for each learning agent: (1). *Model selection*: a knowledge point (not at the location of itself) is selected based on tournament selection from two knowledge points in library; (2). *Observational learning*: Compare the level of selected knowledge point with that of possessed by itself, then choose the better knowledge point as central point, and the worse knowledge point as reference point, and the learning agent will move the a new knowledge point (is stored by library) base on the neighborhood searching;

c) Library refreshment: Remove N_c knowledge points with the worst knowledge level in library;

d) Repeat the b)–d) step until a stop condition (e.g., a predefined generation number T) is reached.

The parameters of SCO include: N_{pop}, N_c, and T. The total number of function evaluation is $T_e = N_{pop} + N_c * T$.

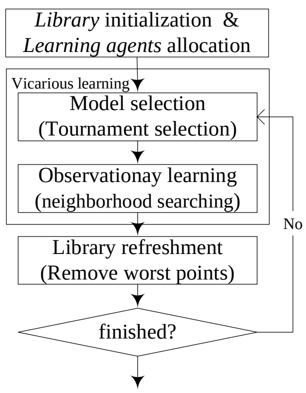

FIG. 3. Flowchart of social cognitive optimization

4. Experimental settings

The test cases from [1] are summarized in table 1. For each test case we list number of variables (D), type of the function *f*, and the relative size of the feasible space given by the ratio S_F/S, the number of constraints (linear inequalities LI, nonlinear inequalities NI), and the number *a* of active constraints at the optimum.

TABLE 1. Summary of test cases [1]

Func.	D	Type of *f*	S_F/S	LI	NI	*a*
G_1	13	quadratic	0.0111%	9	0	6
G_2	20	Nonlinear	unknown	0	2	1
G_4	5	Quadratic	52.1230%	0	6	2
G_6	2	Cubic	0.0066%	0	2	2
G_7	10	Quadratic	0.0003%	3	5	6
G_8	2	Nonlinear	0.8560%	0	2	0
G_9	7	Polynomial	0.5121%	0	4	2
G_{10}	8	Linear	0.0010%	3	3	6

This paper has not discussed the G_3, G_5, G_{11} in [1], which has almost 0% feasible space due to the equations constraints. It needs to replace the equations constraint $g(\vec{x})=0$ by an inequality constraint $|g(\vec{x})| \le \varepsilon$ for some small $\varepsilon > 0$, which the ultimate testing results will not be comparable since ε is not clearly defined in [1].

4.1 Algorithm Settings for GA in [1]

Case **GA#0**: It is the **Experiment #2** tested by Koziel et al [1]. Its main parameters includes: *Pop_size*=70, generation *gap*=100%. The total evaluation times $T_e = Pop\_size \cdot gap \cdot T$. For each run, T=20000 (For G_2, T=30000). Then T_e=1400000 (For G_2, is 2100000). 20 runs were executed for each test case.

4.2 Algorithm setting for SCO

For SCO, 50 runs were executed for each test case:

Case **SCO#1**: N_{pop}=70, N_c=1, T=28000 (For G_8, T=2800). Then T_e=28070 (For G_8, is 2870);

Case **SCO#2**: N_{pop}=70, N_c=14, T=2000 (For G_8, T=200). Then T_e=28070 (For G_8, is 2870);

Case **SCO#3**: N_{pop}=98, N_c=14, T=2000 (For G_8, T=200). Then T_e=28098 (For G_8, is 2898);

Case **SCO#4**: N_{pop}=350, N_c=70, T=2000 (For G_8, T=200). Then T_e=140350 (For G_8, is 14350).

The constraint-handling method for SCO is following the criteria [15]: a) any feasible solution is preferred to any infeasible solution; b) among two feasible solutions, the one having better objective function value is preferred; c) among two infeasible solutions, the one having smaller constraint violation is preferred.

5. Results and discussion

Table 2 gives the summary of F_{best}, which is the mean best fitness value in current generation that found during the evolutionary process, for different algorithm settings on test cases. F_{opt} is the optimum value of each function. Figure 4 to 11 shows the relative fitness value $F_{norm}=|F_{best}-F_{opt}|$, which are performed by SCO#1, #2, and #3, versus N_c*t for different benchmark functions, respectively. Where t is current generation number, and N_c*t is current evaluation times, except for the N_{pop} evaluation times for the initialization of library.

It can be found that SCO#1, which has only one learning agent, shows highly learning capability by providing a library that is large enough. It provides great flexibility than swarm intelligence that is inspired by the collective behavior of social insect colonies.

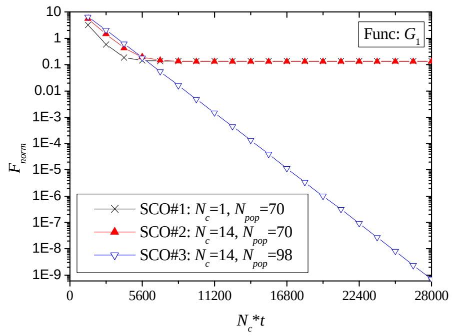

FIG. 4. F_{norm} versus N_c*t for G_1 by different SCO settings

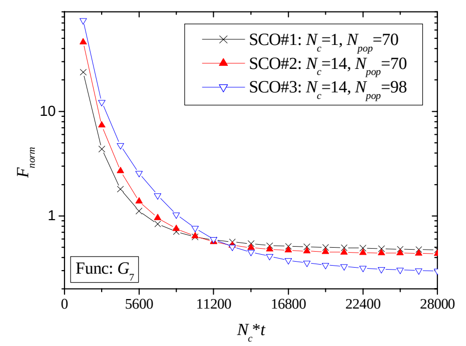

FIG. 8. F_{norm} versus N_c*t for G_7 by different SCO settings

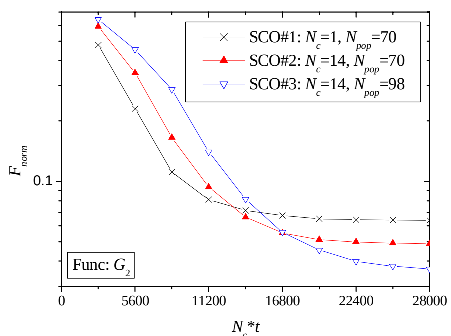

FIG. 5. F_{norm} versus N_c*t for G_2 by different SCO settings

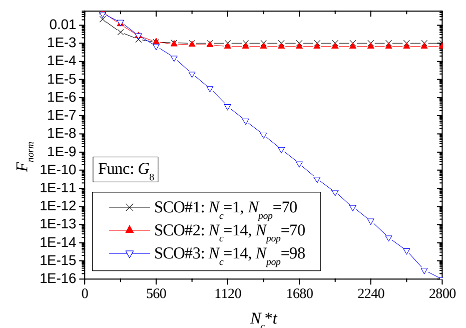

FIG. 9. F_{norm} versus N_c*t for G_8 by different SCO settings

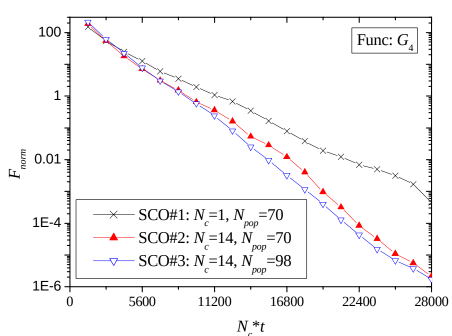

FIG. 6. F_{norm} versus N_c*t for G_4 by different SCO settings

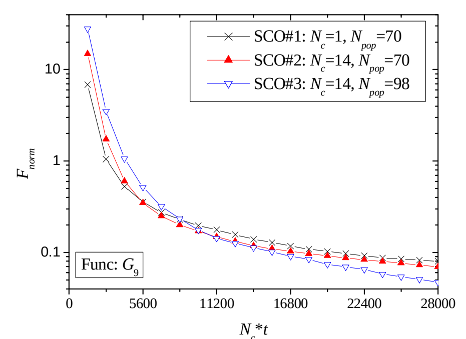

FIG. 10. F_{norm} versus N_c*t for G_9 by different SCO settings

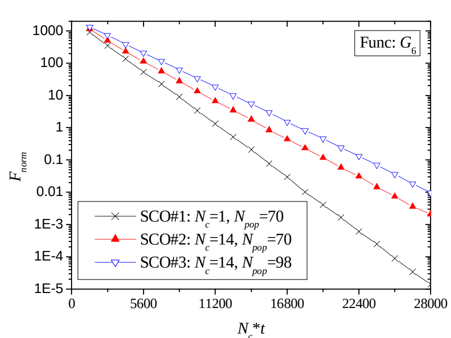

FIG. 7. F_{norm} versus N_c*t for G_6 by different SCO settings

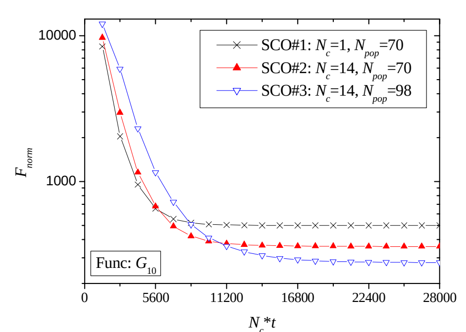

FIG. 11. F_{norm} versus N_c*t for G_{10} by different SCO settings

For SCO#1 and SCO#2, they have same N_{pop} and T_e. The SCO#2, which has more agents, i.e. large N_c, shows lower convergence velocity at early stage of evolution, however, it shows higher sustainable evolutionary capability at the late stage of evolution.

For SCO#2 and SCO#3, the only difference is their N_{pop}. It makes tiny effect on the total evaluation times T_e, however, it have similar effect on the evolutionary performance as that of the N_c. The algorithm with large library size shows lower convergence velocity at early stage of evolution and higher sustainable evolutionary capability at the late stage of evolution. It seems that there has a threshold value for N_{pop} on some functions, such as G_1 and G_8. If the N_{pop} is larger than the threshold value, SCO shows exponential convergence capability, or else it may be stagnated at a certain stage.

Whatever happens, for SCO #1, #2 and #3, they all show higher performance than case GA#0 in almost all cases (except for G_2). In fact, when N_c is enlarged to 70, as in SCO#4, it shows higher performance than GA#0 even for G_2. For the four SCO cases, they all have much less evaluation times that the case GA#0.

6. Conclusion

Human has shown higher adaptability than insect swarm. By applying human social intelligence according to the SCT to artificial system, a new stochastic algorithm, called SCO, is proposed.

The SCO is an extremely simple algorithm, with few parameters and without the mutation operation as in the genetic-based EAs. The experiments by comparing SCO with GA on the benchmark functions show that it can get high-quality solutions in much less evaluation times. Furthermore, as in human society, even one learning agent makes high performance with suitable library size, which shows flexibility than in swarm intelligence.

References

[1] Koziel S, Michalewicz Z. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evolutionary Computation, 1999, 7: 19-44

[2] Kauffman S A. The Origins of Order: Self-organization and selection in evolution, Oxford University Press, NY, 1993

[3] Bäck T, Hammel U, Schwefel H-P. Evolutionary computation: comments on the history and current state. IEEE Trans. Evolutionary Computation, 1997, 1(1): 3-17

[4] Fogel L J, Owens A J, Walsh M J. Artificial intelligence through simulated evolution, John Wiley, Chichester, UK, 1966

[5] Rechenberg I. Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution. Frommann-Holzboog Verlag, Stuttgart, 1973

[6] Holland J H. Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor, 1975

[7] Bonabeau E, Dorigo M, Theraulaz G. Swarm intelligence: from natural to artificial systems. Oxford University Press, UK, 1999

[8] Kennedy J, Eberhart R. Particle swarm optimization. Proc. IEEE Int. Conf. on Neural Networks, 1995: 1942-1948

[9] Dorigo M, ManiezzoV, Colorni A. The ant system: optimization by a colony of cooperating agents. IEEE Trans. on Systems, Man, and Cybernetics - Part B, 1996, 26(1): 1-13

[10] Worden R P. Primate social intelligence. Cognitive Science, 1996, 20: 579-616

[11] Bandura A. Social Learning Theory. Prentice Hall, NJ, 1977

[12] Bandura A. Social foundations of thought and action: a social cognitive theory. Prentice-Hall, NJ, 1986

[13] Bandura A. Social cognitive theory: an agentic perspective. Annual Review of Psychology, 2001, 52: 1-26

[14] Jones J W. Personality and epistemology: cognitive social learning theory as a philosophy of science. Zygon, 1989, 24 (1): 23-38

[15] Deb K. An efficient constraint handling method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering, 2000, 186: 311-338

TABLE 2. Summary of mean results for different algorithm settings on test cases (50 runs for each case)

Func.	*Type*	F_{opt}	GA #0[1]	SCO #1	SCO #2	SCO #3	SCO #4
G_1	Min	-15	-14.7082	-14.8222	-14.8891	-15.0000	-15.0000
G_2	Max	0.80362	0.79671	0.74006	0.75475	0.77908	0.79860
G_4	Min	-30665.539	-30655.3	-30665.539	-30665.539	-30665.539	-30665.539
G_6	Min	-6961.814	-6342.6	-6961.814	-6961.812	-6961.804	-6961.814
G_7	Min	24.306	24.826	24.781	24.742	24.604	24.410
G_8	Max	0.095825	0.089157	0.094990	0.095158	0.095825	0.095825
G_9	Min	680.63	681.16	680.710	680.699	680.677	680.643
G_{10}	Min	7049.33	8163.6	7596.9	7407.7	7327.5	7175.3