Genetic and Evolutionary Computation Conference (GECCO), LNCS 3102, Seattle, WA, 2004: 261-262 (Extended Version)

Solving Engineering Design Problems by Social Cognitive Optimization

Xiao-Feng Xie, Wen-Jun Zhang

Institute of Microelectronics, Tsinghua University, 100084 Beijing, China
xiexf@ieee.org, zwj@tsinghua.edu.cn

Abstract. Social cognitive optimization (SCO) is a simple behavioral model based on human social cognition. By formalizing the fundamental social cognitive agent, the single-agent and multiagent models of SCO are studied. After realizing the goodness evaluation, the experiments results of SCO are compared with existing results on five engineering design problems, which show that SCO can get high-quality solutions efficiently, even by the single-agent model.

1 Introduction

The engineering optimization problem [8, 19] can be defined as finding $\vec{x} \in S$:

$$\begin{cases} \text{Miniminze}: f(\vec{x}) \\ \text{Subject to: } g_j(\vec{x}) \le 0 \quad (1 \le j \le m, j \in \mathbb{Z}) \end{cases} \quad \mathbf{(1)}$$

where $\vec{x} = (x_1, ..., x_d, ..., x_D)$ ($1 \le d \le D$, $d \in \mathbb{Z}$), $x_d \in [l_d, u_d]$ can be continuous or integer variable, l_d and u_d are lower and upper boundary values, respectively. $f(\vec{x})$ and $g_j(\vec{x})$ are *objective function* and *constraints*, respectively. S is a D-dimensional *search space*, which includes the set of possible $\vec{x}$.

With the gain of adaptability to the problem at hand, in combination with global optimization characteristics and robust performance, the idea of applying the principle of natural evolution to artificial adaptive systems has seen impressive growth [2, 12, 14].

Swarm systems, such as bird flock, are products of natural evolution. The complex collective behavior can emerge from a society of autonomous cognitive entities, called as agents [6, 10, 20], following simple rules operated on a *symbolic space*. Symbols provide the mechanism that allows for cognitive problem solving [1] by enabling agents storing *knowledge* [16] in their memory [18] for guiding future actions. Each agent acquires knowledge in a mix of two complementary ways [7, 13]: a) individual learning [20], which performs reinforced practice to its own experience; b) social learning [3], which allows one to benefit from the successful sharing experiences, with a “head start” [11]. For the behavioral models based on the concepts of swarm, each $\vec{x} \in S$ is a *knowledge point*, and its goodness is evaluated by the *goodness function* $F(\vec{x})$, which a typical example is particle swarm optimization (PSO) [15].

Social interaction plays the key role for the collective behavior [6, 15]. However, for most animals, the knowledge is often adhered to each agent. Hence an agent can hardly benefit from social learning when few other agents are observable.

Language is perhaps the most interesting trait of humans [17]. The powerful symbol processing capability allows for cognition on a grand scale, since individual can acquire information that is no longer limited to direct observation [11].

Social cognitive optimization (SCO) [21] is a simple model based on human social cognition [3, 5, 11]. In this paper, SCO is formalized from the viewpoint of agent-based modeling (ABM) [6]. In section 2, the single-agent and multiagent models of SCO are studied. In section 3, the goodness evaluation methods for engineering design problems are described. In section 4, the experimental results on five problems are compared with existing results [8, 19]. In the last section, we conclude the paper.

2 Social cognitive optimization (SCO)

For human, the extrasomatic arbitrary symbols that manipulated by language are efficient means of knowledge transfer and storage. Human cognition can easily benefit from social learning, since the sharing experiences can be available from the formation of symbols, instead of from direct observing other persons. Furthermore, selective social learning on success experiences enables human to form patterns of behavior quickly by avoiding time-consuming trial-and-error [11]. In addition, such capabilities allow one to explore activities for the attainment of new knowledge that would normally be out of reach due to constraints on time and resources. The individual learning then only plays secondary role due to the ubiquity and efficiency of social learning [3].

SCO is based on human cognition. Firstly, as the fundamental element, the social cognitive (SC) agent is formalized; secondly, it works in the single agent model (SAM); thirdly, the original SCO version is represented by a full sharing multiagent model (FSM); then a partial sharing multiagent model (PSM), which is similar to the real world cases, is realized for reducing the probability of the premature convergence.

2.1 Social Cognitive (SC) Agent

The foundational entity for simulating human cognition is social cognitive (SC) agent. Each SC agent includes a memory (M_D) and a set of action rules (R_A). The cognition is realized by interplaying between learning and memory [18], which the M_D is employed for storing the N_M knowledge point(s) to guide future actions, and learning behaviors are achieved by executing R_A for acquiring memory. As a frugal version, N_M is fixed as 1. Specially, the agent acquires *social sharing information* (called $\underline{I}$) not only from the M_D of other agents, but also from the medium, called *library* ($\underline{L}$), which stores N_L points. The collection of knowledge in its M_D and $\underline{I}$, i.e. all the accessible knowledge for the agent, is defined as knowledge pool ($\underline{K}$), which the size is N_K.

Each agent is worked in iterated learning cycles. Suppose T is the number of maximum learning cycles. At the tth ($1 \le t \le T, t \in \mathbb{Z}$) learning cycle, the M_D stores the most recently knowledge point $\vec{x}^{(t)}$. The knowledge point with best goodness value in its $\underline{K}$ is defined as $\vec{g}^{(t)}$, and then the goal is to find $\vec{g}^{(T)}$ with good enough goodness.

For the convenience of discussion, a tournament-selection(NUM, STATE, SET) function is defined as following: a) constructs the sample set $\underline{X}$: select NUM different points from SET at random; b) returns the point with the STATE goodness in the $\underline{X}$.

The action rules (R_A) for selective social learning on success experiences include:

a) Selects a successful knowledge point: $\vec{\kappa}_B$=tournament-selection (τ_B, *best*, $\underline{I}$);

b) Infers a new knowledge point $\vec{x}^{(t+1)}$ around $\vec{x}^{(t)}$ and $\vec{\kappa}_B$. For the dth dimension:

$$x_d^{(t+1)} = U_{\mathbb{R}}(X_{R,d}, X_{B,d}) \tag{2}$$

where $U_{\mathbb{R}}(a, b)$ is a random value within [a, b], and normally $X_{B,d} = 2 \cdot X_{M,d} - X_{R,d}$. Here if $F(\vec{\kappa}_B) \le F(\vec{x}^{(t)})$, then $\vec{X}_M = \vec{\kappa}_B$, $\vec{X}_R = \vec{x}^{(t)}$, else $\vec{X}_R = \vec{\kappa}_B$, $\vec{X}_M = \vec{x}^{(t)}$.

Besides, it is essential to ensure that $\vec{x}^{(t+1)} \in S$. Hence for equation (2), if $X_{B,d} < l_d$, then $X_{B,d} = l_d$, if $X_{B,d} > u_d$, then $X_{B,d} = u_d$.

c) Stores $\vec{x}^{(t+1)}$ into M_D, and pushes its old $\vec{x}^{(t)}$ into $\underline{L}$.

To keep the constant library size, the redundant points, which each point is selected as $\vec{\kappa}_W$=tournament-selection (τ_W, *worst*, $\underline{L}$) in sequence, are discarded from $\underline{L}$ at the end of each learning cycle.

The default values of τ_B and τ_W are 2 and 4, respectively.

2.2 Single-agent model (SAM)

As shown in figure 1, the SC agent in SAM communicates with a library, which is also exactly the $\underline{I}$. Here the $\underline{K}$ concludes the knowledge points in its M_D and $\underline{L}$.

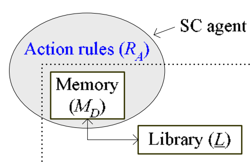

Fig. 1. Single-agent cognitive model

The total evaluation times of SAM are T_E=N_L+1+T.

If $\tau_B = N_L$, then $\vec{\kappa}_B$ is always the best point in $\underline{L}$, it is easy to prove that $\| \vec{\kappa}_B - x^{(t)} \|$ will be damped quickly as t increasing. If N_L> τ_B, then the agent is referring to multiply points. If $\tau_B = \tau_W = 1$, then the trajectory of agent should be in chaos. However, if τ_B>1, then only successful points are referred, and if τ_W>1, then the $\underline{L}$ is under elitist selection, hence the $\vec{g}^{(t)}$ will be improved by reinforcement learning on successful points. Here N_L controls the trade-off between exploitation and exploration. Notes the intermediate $x^{(t)}$ can be temporal badly, which may benefit for long-term performance. Besides, similar to the case $N_L = \tau_B$, the variation of $x^{(t)}$ will be small as the diversity of the successful points in $\underline{L}$ becomes small along with learning cycles.

By considering $\underline{L}$ as an external memory for the agent, the successive dynamics of $x^{(t)}$ in SAM can also be considered as simplified thinking process [3].

2.3 Full sharing multiagent model (FSM)

A simple multiagent model is sharing of a library by a society with N agents, as shown in figure 2. The $\underline{K}$ for all the agents is same, which includes the knowledge points in all the M_D and $\underline{L}$. Foe each agent, its $\underline{I}$ is points in $\underline{K}$ that except for its own M_D.

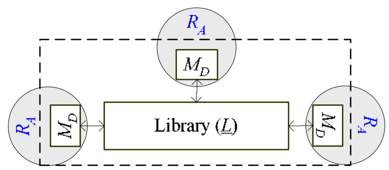

Fig. 2. Full sharing multiagent model

For each learning cycle, each agent in FSM performs their action rules as SAM in turn. At the end of each learning cycle, totally N obsolete points are discarded from $\underline{L}$.

The total evaluation times are $T_E = N_L + (1+T) \cdot N$. The FSM is equivalent to the original SCO version [21]. And if N=1, then it becomes the SAM.

Since all the intermediate knowledge points are updated into the $\underline{L}$, the diversity of knowledge points in $\underline{K}$ may be decreased too fast to induce premature convergence.

2.4 Partial sharing multiagent model (PSM)

For real world cases, the size of social sharing knowledge points is very large, and each agent in a society can only access a part of them. Internet is a typical example.

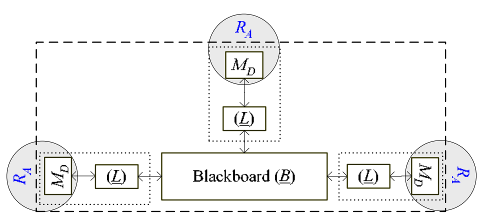

Fig. 3. Partial sharing multiagent model

As shown in figure 3, the *blackboard* ($\underline{B}$) serves as a central data repository, which the size is N_B. The communication among the agents happens through their actions for modifying the $\underline{B}$. Each agent has own library and allows a specified *thinking time* (T_T) for learning deliberately. For simplicity, all the agents have same N_L and same T_T.

When $t = n_t \cdot T_T + 1 \ \ (0 \le n_t \le T / T_T, n_t \in \mathbb{Z})$, each agent updates its $\underline{L}$ by selecting N_L knowledge points from $\underline{B}$ at random. Then each agent performs as an SAM with its M_D and $\underline{L}$ as $n_t \cdot T_T + 1 \le t \le (n_t + 1) \cdot T_T$. After the $t = (n_t + 1) \cdot T_T$ cycle is finished, each agent only updates the point with best goodness value in its $\underline{K}$ into $\underline{B}$.

For PSM, the total evaluation times are $T_E = N_B + T \cdot N$.

3 Goodness Function for Engineering Design Problems

The goodness function $F(\vec{x})$ is employed for evaluating the goodness of each knowledge point $\vec{x}$. For engineering design problems, the most frequently parts encountered are the handling for the constraints and integer variables.

3.1 Constraint Handling

The basic goodness function is defined as $F(\vec{x}) =< F_{OBJ}(\vec{x}), F_{CON}(\vec{x}) >$, where $F_{OBJ}(\vec{x}) = f(\vec{x})$ and $F_{CON}(\vec{x}) = \sum_{j=1}^{m} r_j G_j(\vec{x})$ are the goodness functions for objective function and constraints, respectively, r_j are positive weight factors, which default value is 1, and $G_j(\vec{x}) = \max(0, g_j(\vec{x}))$.

To avoid adjusting penalty coefficient [8], and to follow the criteria by Deb [9], the goodness evaluation is realized by comparing any two points $\vec{x}_A$, $\vec{x}_B$:

$$F(\vec{x}_A) \le F(\vec{x}_B) \text{ when } \begin{cases} F_{CON}(\vec{x}_A) < F_{CON}(\vec{x}_B) \ \text{ OR} \\ F_{OBJ}(\vec{x}_A) \le F_{OBJ}(\vec{x}_B),\ F_{CON}(\vec{x}_A) = F_{CON}(\vec{x}_B) \end{cases} \quad \textbf{(3)}$$

3.2 Integer Variable Handling

Some engineering design problems may be mixed-integer-continuous problem, which some variables are integer variables.

The SCO can handle with continuous variables only. However, extending it for integer variables is rather easy. The discrete space is mapping into a continuous step function, i.e. $F(\vec{x}) = F(\vec{x}\,')$, where for the *d*th dimension, $x_d\,' = INT(x_d)$, and *INT*() is a function for converting a real value to a closest integer value.

4 Experimental Results

Five engineering design problems [8, 19] have been tested in order to study the performance of SCO. They are speed reducer (*SR*), three-bar truss (*TB*), welded beam (*WB*), tension spring (*TS*), and pressure vessel (*PV*) design problems. The last two problems, i.e. *TS* and *PV*, are mixed-integer-continuous problems. Table 1 summaries the global optimum F^* and the existing results [8, 19] by other algorithms, include the mean values (F_B) and the evaluation times (T_E).

Table 1. Global optimum and existing results [8, 19] for engineering design problems

F.	F^*	F_B	T_E
SR	2994.471	2998.027 [19]	110235 [19]
TB	263.8958	263.8989 [19]	36113 [19]
WB	2.38113	2.96070 [19]	64862 [19]
TS	0.012666	0.012923 [19]	25167 [19]
PV	6059.714	6263.793 [8]	50000 [8]

When t=0, all the knowledge points are initialized at random in the S. The testing results of different versions of SCO are mean goodness results in specified evaluation times, which 500 runs were performed for each problem.

Table 2 gives the mean results $F_B^{(T)}$ by the SAM, which T=2E4, the sizes of N_K are 40, 120, 200, and 280, respectively. It means the evaluation times T_E are 20040, 20120, 20200, and 20280, respectively. When N_K is large than 40, the SAM gets better results for the first three continuous design problems, i.e. *SR*, *TB*, and *WB*, but gets worse results for the two problems with integer variables than the existing results in table 1. The highly adaptivity for small N provides great flexibility than PSO [15].

Table 2. Mean results by SAM, where T=20000

F.	N_K=40	N_K=120	N_K=200	N_K=280
SR	2994.471	2994.471	2994.471	2994.471
TB	264.0311	263.8977	263.8972	263.8969
WB	3.80066	2.61903	2.47514	2.44056
TS	0.01432	0.01377	0.01365	0.01365
PV	6702.254	6589.230	6531.708	6468.242

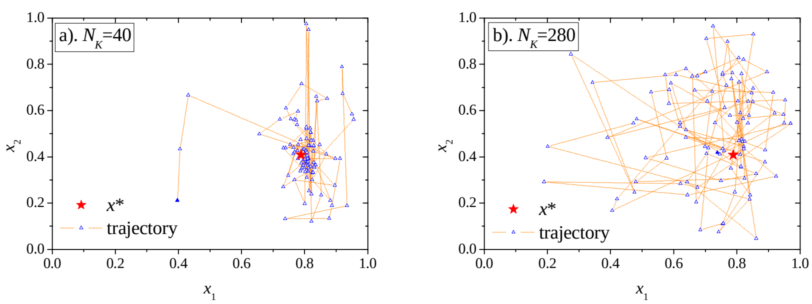

Fig. 4. The trajectory examples of the agent in SAM, where T=100, and: a) N_K=40; b) N_K =280

To demonstrate the characteristics of SC agent in SAM, figure 4 give the trajectory examples of the agent in 100 learning cycles, for the *TB* problem, which the sizes of

N_K are 40 and 280, respectively. The pentacle indicates the location of the global optimum point $\vec{x}^*$. It shows that the small N_K (=40) facilitates the exploitation, i.e. the local search capability while the large N_K (=280) facilitates the exploration, i.e. the global search capability. Besides, the agent is mainly interested in the promising part of S, by referring to the multiply successful knowledge points in the library.

Table 3 gives the mean results $F_B^{(T)}$ by the FSM, which N=40, T=500, and the sizes of N_L/N are 2, 4, and 6, respectively. It means T_E is 20120, 20200, and 20280, respectively. It can be found that the FSM can get better results than SAM in same evaluation times. However, it still gets worse results for the two problems with integer variables than the existing results in table 1. Since for the mapped step landscape, the goodness values for all the points at a same step are same. As the learning cycles goes on, many intermediate points at the same step with $\vec{g}^{(t)}$ will be prosperously in the knowledge pool $\underline{K}$, and induce the FSM to be converged prematurely.

Table 3. Mean results by FSM, where N=40, T=500

F.	N_L=2N	N_L=4N	N_L=6N
SR	2994.471	2994.471	2994.478
TB	263.8970	263.8970	263.8967
WB	2.47458	2.42723	2.42162
TS	0.01346	0.01344	0.01342
PV	6460.137	6457.597	6456.929

Table 4 gives the mean results $F_B^{(T)}$ by the PSM of SCO, which N=40, N_L=39, T=500, T_T=15, and the sizes of N_B/N are 3, 5, and 7, respectively. It means T_E is 20120, 20200, and 20280, respectively. For the *SR* problem, the PSM shows worse results than SAM and FSM due to its slower convergence. However, it can get better results for all the design problems than the existing results in table 1. Since the intermediate knowledge points during the T_T are no longer allowed to guide the actions of agents, the probability of premature convergence is decreased.

Table 4. Mean results by PSM, where N=40, N_L=39, T=500, T_T=15

F.	N_B=3N	N_B=5N	N_B=7N
SR	2994.600	2995.407	2997.403
TB	263.8962	263.8963	263.8966
WB	2.38508	2.39042	2.40589
TS	0.01284	0.01283	0.01287
PV	6259.600	6242.481	6236.937

5 Conclusions

Human beings have shown higher adaptability than other animals, since the extra-somatic symbolic capability by language allows social learning and cognition in rather convenient forms, even for only a single individual.

The SCO is based on human cognition. From the viewpoint of agent-based modeling, its fundamental element, called social cognitive agent, learns knowledge by action rules according to the experience in its own memory and social sharing information, which not only from the memory of other agents, but also from the medium of knowledge, called library. Several models, include SAM, FSM, and PSM, were presented. Then the goodness evaluation methods are realized after handling with the constraints and the discrete variables.

The SCO is a simple algorithm with few control parameters that can be readily adjusting. Firstly, the number of agents (N) can be adjusted flexibly. Moreover, the trade-off between exploitation and exploration can be achieved by adjusting the size of library (N_L) while no significant impacts on the total evaluation times, which is determined by the N and the learning cycles (T).

The experiments results of SCO on five benchmark functions have compared with existing results. It shows that even SAM can get high quality solutions on three problems with suitable library size. Moreover, the cooperating models, include FSM and PSM, are performed better than SAM for most testing cases. For the mixed-integer-continuous problems, the PSM shows better performance than SAM and FSM, which reduces the probability of the premature convergence by preventing some intermediate knowledge points from guiding the actions of agents. At last, the PSM performs better than two existing algorithms for all problems in fewer evaluation times.

6 References

1. Anderson, J.R.: A theory of the origins of human knowledge. Artificial Intelligence, 40 (1989) 313-351
2. Bäck, T., Hammel, U., Schwefel, H.-P.: Evolutionary computation: comments on the history and current state. IEEE Trans. Evolutionary Computation, 1 (1997) 3-17
3. Bam, A., Wills, L., Domeshek, E., Nersessian, N., Kolodner, J.: Understanding the creative mind. Artificial Intelligence, 79 (1995) 111-128
4. Bandura, A.: Social Learning Theory. Prentice Hall, NJ (1977)
5. Bandura, A.: Social cognitive theory: an agentic perspective. Annual Review of Psychology, 52 (2001) 1-26
6. Bonabeau, E.: Agent-based modeling: methods and techniques for simulating human systems. Proc. Natl. Acad. Sci. USA, 99 (2002) 7280–7287
7. Boyd, R., Richerson, P.J.: The Origin and Evolution of Cultures. Oxford Univ. Press (2003)
8. Coello, C.A.C.: Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Computer Methods in Applied Mechanics and Engineering, 191 (2002) 1245-1287
9. Deb, K.: An efficient constraint handling method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering, 186 (2000) 311-338
10. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: optimization by a colony of cooperating agents. IEEE Trans. Systems, Man, and Cybernetics - Part B, 26 (1996) 1-13
11. Flinn, M.V.: Culture and the evolution of social learning. Evolution and Human Behavior, 18 (1997) 23-67
12. Fogel, L.J., Owens, A.J., Walsh, M.J.: Artificial intelligence through simulated evolution. John Wiley, Chichester, UK (1966)

13. Galef, B.G.: Why behaviour patterns that animals learn socially are locally adaptive. Animal Behaviour, 49 (1995) 1325-1334
14. Holland, J.H.: Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor, MI (1975)
15. Kennedy, J., Eberhart, R.C.: Particle swarm optimization, IEEE Int. Conf. on Neural Networks, Perth, Australia (1995) 1942-1948
16. Newell, A.: The knowledge level. Artificial Intelligence, 18 (1982) 87-127
17. Nowak, M.A., Komarova, N.L., Nlyogl, P.: Computational and evolutionary aspects of language. Nature, 417 (2002) 611-617
18. Okano, H., Hirano, T., Balaban, E.: Learning and memory. Proc. Natl. Acad. Sci. USA, 97 (2000) 12403-12404
19. Ray, T., Liew, K.M.: Society and civilization: an optimization algorithm based on the simulation of social behavior. IEEE Trans. Evolutionary Computation, 7 (2003) 386-396
20. Thorndike, E.L.: Animal intelligence: An experimental study of the associative processes in animals. Psychological Review, Monograph Supplement, 2 (1898)
21. Xie, X.F., Zhang, W.J., Yang Z.L.: Social cognitive optimization for nonlinear programming problems, Int. Conf. on Machine Learning and Cybernetics, Beijing, China (2002) 779-783