AIAA Journal
Vol. 44, No. 12, December 2006

Technical Notes

TECHNICAL NOTES are short manuscripts describing new developments or important results of a preliminary nature. These Notes should not exceed 2500 words (where a figure or table counts as 200 words). Following informal review by the Editors, they may be published within a few months of the date of receipt. Style requirements are the same as for regular contributions (see inside back cover).

Parallel Simulated Annealing Using Simplex Method

Ya-Zhong Luo* and Guo-Jin Tang†
National University of Defense Technology, 410073 Changsha, People's Republic of China

DOI: 10.2514/1.16778

Introduction

SIMULATED annealing (SA) algorithm is an iterative probabilistic algorithm which combines local search with Monte Carlo techniques [1]. Numerous researchers have demonstrated that SA is very effective in many optimization problems. However, the long execution time of SA has been the major drawback in practice. To improve the performance of SA, one widely used method is to parallelize SA by dividing a Markov main chain into subchains on multiple processors, and several representatively parallel SA using this method have been proposed [2–6]. Another widely employed method is to hybridize SA with other local deterministic algorithms [7,8]. The simplex method (SM) is a well-known deterministic search method proposed by Nelder and Mead [9]. Press et al. [10] proposed an new simplex simulated annealing algorithm by combing SA with SM. In Press's new algorithm, the SM is applied for the generation of a new design vector instead of random generation and this algorithm has been applied in a number of studies [11–13]. The search procedure of SM is employed on a simplex consisting of $n+1$ vertices for an n-dimensional optimization problem, so that SM can be regarded as one population-based algorithm with natural parallelism. Therefore, it would be highly desirable to parallelize SA using SM. However, few works have been made on this issue. In this note, an improved edition of hybrid algorithm combining SA with SM, a parallel simulated annealing using simplex method (PSASM), is proposed and applied to structural optimization.

Outline of PSASM

The essence of Press's algorithm is to introduce the Metropolis criterion into SM. Other than Press's algorithm, Kvasnicka and Pospichal [14], and Wang and Zheng [15] proposed different hybrid algorithms combining SA with SM. Kvasnicka and Pospichal's [14] simplex simulated annealing algorithm is similar to Press's algorithm, and their parallel version uses a decomposition of the whole population into disjoint subpopulations, each of which is equal to one simplex. In Wang and Zheng's [15] algorithm, the simulated annealing search is applied on the simplex when a local solution is obtained by the SM. We proposed herein another new hybrid algorithm: the PSASM. The outline of the proposed PSASM algorithm is described as follows:

Step 1) A simplex (including $n+1$ design points), and other parameters such as the initial temperature T_0 and the length of Metropolis sampling L_p are initialized, $k=1$.

Step 2) The simplex is evaluated, and the highest, second-highest, and lowest points are determined.

Step 3) A new simplex is produced by reflection, reflection and expansion, etc.

Step 4) Simulated annealing search is applied on ith point of the simplex (to be referred to as x^i) at probability $P(i,k)$ [described as Eq. (1)], $i=1,2,\ldots,n_0$.

Step 4.1) make the initial solution for Metropolis sampling at the current temperature T_k:$y_l^i = x^i$, $l=1$;

Step 4.2) produce the tentative solution y_{l+1}^i using the neighbor search function;

Step 4.3) replace y_l^i with y_{l+1}^i using Metropolis rule, $l+=1$;

Step 4.4) if $l > L_p$, $x^i = y_l^i$, go to step 5; otherwise go to step 4.1;

Step 5: The PSASM is terminated if the stopping condition is satisfied; otherwise, $k=k+1$, T_k is renewed, go to step 2.

The simplex can be regarded as a population including $(n+1)$ individuals. The excellent individuals of the simplex can be spread out using one of the following operations: reflection, expansion, and contraction, so it is unnecessary to apply simulated annealing search on all points of the simplex. A preliminary probability strategy is used to decide how and when simulated annealing search is employed on a point, which is presented as follows.

Choose $n_0 (n_0 \le n+1)$ points from the simplex at random, the ith $(i=1,2,\ldots,n_0)$ point is chosen to employ simulated annealing search using the following probability function:

$$P(i,k) = \exp(-k/K_{\max}) \tag{1}$$

where k is the annealing iteration index, and $K_{\max}$ is the maximum number of annealing iteration specified by the user in advance.

More details of the PSASM including neighborhood search function, initial temperature, and cooling scheme are provided in [16]. To solve an n-dimensional problem, it is necessary to generate $n+1$ points that form the vertices of the simplex. The vertex of the simplex, which is an initial n-dimensional point, is generated at random in the design space. The other points are generated according to Nelder and Mead's method [9]. A simple stopping condition is employed for the PSASM. $K_{\max}$ is chosen as the stopping condition, if $k \ge K_{\max}$, the PSASM is ended.

It has been shown that the performance of the SA is sensitive to several tuning parameters including T_0, $K_{\max}$, the cooling ratio α, and the length of Metropolis sampling L_s [14,17,18]. Thus, it is important to select proper tuning parameters; Kim et al. [18] used the experimental design method to design the tuning parameters, which can be further optimized by SM. Compared with the SA, the PSASM is less sensitive to tuning parameters. It was observed that several criteria should be followed in choosing the proper tuning parameters: 1) as the initial temperature is defined adaptively [16], $\alpha \in [0.90, 0.97]$ is always effective; 2) when the PSASM is executed in

Presented as Paper 4584 at the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York, 30 August–1 September 2004; received 23 March 2005; revision received 10 June 2006; accepted for publication 25 June 2006.

*Ph.D. Candidate, College of Aerospace and Material Engineering, Student Member AIAA; E-mail: yzluo@sohu.com.

†Professor, College of Aerospace and Material Engineering.

Table 1 Results for test function 1, best solution, 50 runs

Function value range	SE [23]	GESA [23]	RRSA [22]	GEO [21]	SA	PSASM
[1.0000, 1.0001]	66%	100%	100%	100%	100%	100%
[1.0024, 1.0026]	34%	——	——	——	——	——
Average NFE	800,000	800,000	92,254	164,070	57,452	5077

Table 2 Results of five trials using PSASM

Solution	Run 1	Run 2	Run 3	Run 4	Run 5
x_1	3.50001966	3.50004924	3.50000547	3.50019634	3.50000171
x_2	0.70000331	0.70000984	0.70000001	0.70000007	0.70000033
x_3	17	17	17	17	17
x_4	7.30001163	7.30000465	7.30001127	7.30000452	7.30000644
x_5	7.71532052	7.71616417	7.71534120	7.71539161	7.71532769
x_6	3.35021714	3.35046639	3.35021476	3.35021676	3.35021590
x_7	5.28665446	5.28665483	5.28667379	5.28671488	5.28665460
$f(x)$	2994.378443	2994.501795	2994.370088	2994.472965	2994.357835

one computer, n_0 should be a small number, 2–4 is a proper choice; 3) a simple trial-and-error method is used to select $K_{\max}$ and L_p. Relative large numbers of $K_{\max}$ and L_p are first tested, and the convergence process of the PSASM is evaluated. If the PSASM is found to converge earlier, $K_{\max}$ and L_p should be reduced; otherwise, they should be increased. The process can be completed in several experiments. Using these criteria, it becomes easy to find proper tuning parameters quickly. We have demonstrated this issue by the applications of the PSASM to a set of mathematical problems and a structural design problem [16], interplanetary low-thrust trajectory design [19], and rendezvous phasing strategy design problem [20]. However, it needs to be pointed out that this experience-based method does not guarantee the best tuning parameters to be located, and it only provides the proper values.

Functional Optimization

In this section, the PSASM is evaluated by two functional problems. One is a two-dimensional function and the other is Golinski's speed reducer problem. Herein, performance is measured through the best solution obtained by the test algorithms and the number of function evaluations (NFE) needed to find the global solution or the best solution.

Test Function 1

A two-dimensional unconstrained function is selected as the first example:

$$\min f(x) = \frac{1}{2}(x_1^2 + x_2^2) - \cos(20\pi x_1)\cos(20\pi x_2) + 2 \qquad -10 < x_i < 10 \tag{2}$$

This function has only two design variables but is nonseparable and highly multimodal with 40,000 local minima in the range $[-10, 10]$. The global minimum is located at $x^* = (0, 0)^T$, where the optimal objective function value $f(x^*) = 1$. This function is tested with a precision $|f(x) - f(x^*)| \leq 0.0001$. This function has been solved by different stochastic optimization algorithms, including generalized extremal optimization (GEO) [21], region reduction SA (RRSA) [22], simulated evolution (SE) [23], and guided evolutionary SA (GESA) [23]. The SA and PSASM are used to optimize this function, respectively, and 50 runs are executed for each algorithm. The parameters of the SA are $K_{\max} = 300$, $L_s = 300$, and $\alpha = 0.95$, and the parameters of the PSAM are $K_{\max} = 300$, $n_0 = 2$, $L_p = 20$, and $\alpha = 0.95$. The results are listed in Table 1, and the results of SE, GESA, RRSA, and GEO reported by [21–23] are also listed for comparison.

As the PSASM is terminated with a solution precision $|f(x) - f(x^*)| \leq 0.0001$, the PSASM can locate a near-optimal solution with the NFE much less than the number of local minima. As

Table 3 Results of speed reducer design[a]

Solution	Rao [24]	Li and Papalambros [25]	Kuang et al. [26]	Azarm and Li [27]	MDO test suite	Ray [28]	Present
x_1	3.5000000	3.5000000	3.6000000	3.5000000	3.5000000	3.50000002	3.50000171
x_2	0.7000000	0.7000000	0.7000000	0.7000000	0.7000000	0.7000000	0.70000033
x_3	17	17	17	17	17	17	17
x_4	7.3000000	7.3000000	7.3000000	7.3000000	7.3000000	7.30000009	7.30000644
x_5	7.3000000	7.7100000	7.8000000	7.7100000	7.3000000	7.80000000	7.71532769
x_6	3.3500000	3.3500000	3.4000000	3.3500000	3.3502145	3.35021468	3.35021590
x_7	5.2900000	5.2900000	5.0000000	5.2900000	5.2865176	5.28668325	5.28665460
g_1	0.9260847	0.9260847	0.9003601	0.9260847	0.9260847	0.9260847	0.9260834
g_2	0.8020015	0.8020015	0.7797237	0.8020015	0.8020015	0.8020015	0.8020003
g_3	0.5009561	0.5009561	0.4721318	0.5009561	0.5008279	0.5008279	0.5008281
g_4	0.0805668	0.0949185	0.1231443	0.0949185	0.0807793	0.0985283	0.0953563
g_5	**1.0001923**	**1.0001923**	0.9567119	**1.0001923**	**1.0000001**	1.0000000	0.9999989
g_6	0.9980266	0.9981029	**1.1820609**	0.9981029	**1.0000002**	1.0000000	0.9999999
g_7	0.2975000	0.2975000	0.2975000	0.2975000	0.2975000	0.2975000	0.2975001
g_8	1.0000000	1.0000000	0.9722222	1.0000000	1.0000000	1.0000000	0.99999998
g_9	0.4166667	0.4166667	0.4285714	0.4166667	0.4166667	0.4166667	0.4166667
g_{10}	0.94863014	0.94863014	0.95890411	0.94863014	0.94867419	0.94867413	0.94867366
g_{11}	**1.05739726**	**1.00116723**	0.94871795	**1.00116732**	**1.05687249**	0.98914762	0.99999901
Objective	2987.29850	2996.30978	2876.11723	2996.30978	2985.15187	2996.23216	2994.35783

[a]Boldfaced type indicates the violated constraints.

Table 4 Results for 10-bar truss problem

Algorithm	Objective function value				NFE			
	Best	Worst	Average	Standard deviation	Best	Worst	Average	Standard deviation
Hajela and Yoo [29]	1635	1725	1675	29.16	14,803	34,971	23,992	7527.8
Moh and Chiang [22]	1594.7	1606.5	1598.6	3.9918	22,982	28,528	24,899	1981.6
SA	1603.5	1666.1	1623.4	18.4273	65,268	75,096	70,283	3365.1
PSASM	1594.0	1602.5	1597.0	2.9369	43,295	46,951	45,968	1065.1

Table 5 Detailed optimization process of one typical test for 10-bar truss problem

x	Exact solution	NFE $= 10,000$		NFE $= 20,000$		NFE $= 30,000$		NFE $= 40,000$		NFE $= 46,552$, final	
		solution	error, %	solution	error, %	solution	error, %	solution	error, %	solution	error, %
x_1	7.94	7.7297	−2.65	7.9346	−0.07	7.9394	−0.01	7.9329	−0.09	7.9351	−0.06
x_2	0.10	0.4060	306.00	0.1785	78.50	0.1121	12.10	0.1129	12.90	0.1092	9.20
x_3	8.06	8.5296	5.83	8.1117	0.64	8.1028	0.53	8.0871	0.34	8.0688	0.11
x_4	3.94	3.7894	−3.82	3.9195	−0.52	3.9225	−0.44	3.9336	−0.16	3.9330	−0.18
x_5	0.10	0.1683	68.30	0.1404	40.40	0.1006	0.60	0.1031	3.10	0.1020	2.00
x_6	0.10	0.4084	308.40	0.1374	37.40	0.1214	21.40	0.1029	2.90	0.1003	0.30
x_7	5.74	6.2399	8.71	5.8244	1.47	5.7763	0.63	5.7530	0.23	5.7527	0.22
x_8	5.57	5.2659	−5.46	5.5495	−0.37	5.5504	−0.35	5.5627	−0.13	5.5625	−0.13
x_9	5.57	5.2596	5.57	5.5515	−0.33	5.5518	−0.33	5.5846	0.26	5.5636	−0.11
x_{10}	0.10	0.6498	549.80	0.1168	16.80	0.1379	37.90	0.1087	8.70	0.1116	11.60
f	1593.2	1643.7	3.17	1602.8	0.60	1597.1	0.24	1595.7	0.16	1594.0	0.05

shown in Table 1, the PSASM clearly demonstrates better performance than SE, GESA, RRSA, GEO and the SA. The PSASM locates the near-optimal solution at probability 100%, and its computational cost is the lowest, only $1/130 \sim 1/10$ of the other algorithms.

Test Function 2 (Golinski's Speed Reducer Problem)

Golinski's speed reducer problem is one of the most well-studied problems of the NASA Langley Multidisciplinary Design Optimization (MDO) Test Suite. Many researchers, for example, Rao [24], Li and Papalambros [25], Kuang et al. [26], Azarm and Li [27], and Ray [28], have reported solutions of this problem. However, all of the reported solutions other than Ray's are not feasible, including the one that appears in the MDO Test Suite itself. This paper presents the best-known feasible solution obtained by the PSASM. The mathematical model of Golinski's speed reducer problem can be found in [16,28]. Because the Golinski's speed reducer problem is constrained, a simple penalty function [16] is used to handle constraints. The parameters of the PSASM are same as that used in test function 1.

Results of five successive runs are presented in Table 2. The best solution obtained by the PSASM is compared with other reported results in Table 3. From Table 3, it is clear that the solutions reported by Rao [24], Li and Papalambros [25], Kuang et al. [26], Azarm and Li [27], and NASA are infeasible. Only Ray [28], obtained a feasible solution using a swarm algorithm. The solution obtained by the PSASM is better than Ray's. Besides, Ray's solution was located after 70,000 function evaluations, while PSASM located its solution after approximately 7,000 function evaluations, only one-tenth of the former, clearly demonstrating the efficacy of PSASM.

Structural Optimization

In this example, the PSASM is applied to a 10-bar truss design problem. This problem has been used as one of the classical structural optimization design problems to test different stochastic optimization algorithms [22,29]. The mathematical model of this problem is provided in [16].

This problem has been solved with 10 different seeds for the random number generator. The parameters of the SA are $K_{\max} = 500$, $L_s = 200$, and $\alpha = 0.95$, and the parameters of the PSASM are $K_{\max} = 500$, $L_p = 80$, $n_0 = 2$, and $\alpha = 0.95$. The results are summarized in Table 4. The results presented by Hajela and Yoo [29], and Moh and Chiang [22] are also listed for comparison. Hajela and Yoo [29] solved this problem using a genetic algorithm, and Moh and Chiang [22] used an RRSA.

As can be seen in Table 4, the NFE in PSASM is generally higher than that in GA and RRSA, but the optimization results found by the PSASM are slightly better and more consistent. Besides, our penalty function is the simplest. Again, the PSASM excels over the SA both in solution quality and computational cost in solving this structural optimization problem. One detailed optimization process of the PSASM is presented in Table 5. From Table 5, we can find that even for such a simple structural optimization problem, it is very difficult to get an exact global solution. Even if the relative error of function value is small (0.24%), it still could be a relative optimum (the relative error of one variable is 37.90%). The PSASM consumes 20,000 function evaluations only to improve the function value by 0.44% (from 0.6% to 0.16%). Moh and Chiang [22] pointed out that some variables such as x_{10} were so small that it hardly affected the objective function, which might be the reason why an exact solution was so difficult to locate. In Moh and Chiang's study, they used an augmented Lagrange method to deal with size constraints and obtained solutions better than the results presented in this note. However, their method is problem dependent.

It is shown that the PSASM performs efficiently in the optimization of the mathematical functions, but not so well in the 10-bar truss problem, mainly because of the less-effective constraint handling method. Future work will find more effective constraint dealt methods for the PSASM.

Conclusions

An improved edition of hybrid simulated annealing and simplex method algorithm, a parallel simulated annealing using simplex method (PSASM), is proposed in this note. The PSASM was evaluated in two functional problems and one 10-bar truss structural design problem. and was compared with other popular stochastic optimization algorithms. It is shown that PSASM performed competitively when compared with other popular stochastic optimization algorithms. However, it must be remembered that a best optimization method does not exist [30], and it is not expected that the PSASM will work better than other methods for all problems. On the other hand, other implementations for the PSASM should be further examined to improve its performance, for example, the use of other parallelization schemes and penalty functions.

References

[1] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by Simulated Annealing," *Science*, Vol. 220, No. 4598, 1983, pp. 671–680.

[2] Lee, S. Y., and Lee, K. G., "Synchronous and Asynchronous Parallel Simulated Annealing with Multiple Markov Chains," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 7, No. 10, 1996, pp. 993–1007.

[3] Gallego, R. A., Alves, A. B., Monticelli, A. A., and Romero, R., "Parallel Simulated Annealing Applied to Long Term Transmission Network Expansion Planning," *IEEE Transactions on Power Electronics*, Vol. 12, No. 1, 1997, pp. 181–189.

[4] Witte, E. E., Chamberlain, R. D., and Franklin, M. A., "Parallel Simulated Annealing Using Speculative Computation," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 2, No. 4, 1991, pp. 483–494.

[5] Nabhan, T. M., and Zomaya, A. Y., "A Parallel Simulated Annealing Algorithm with Low Communication Overhead," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 6, No. 12, 1995, pp. 1226–1233.

[6] Yamanmoto, A., and Hashimoto, H., "Application of Temperature Parallel Simulated Annealing to Loading Pattern Optimization of Pressurized Water Reactors," *Nuclear Science and Engineering*, Vol. 136, No. 2, 2000, pp. 247–257.

[7] Luo, Y. Z., Tang, G. J., and Zhou, L. N., "Simulated Annealing for Solving Near-Optimal Low-Thrust Trajectory Transfer," *Engineering Optimization*, Vol. 37, No. 2, 2005, pp. 201–216.

[8] Yiu, K. F. C., Liu, Y., and Teo, K. L., "A Hybrid Descent Method for Global Optimization," *Journal of Global Optimization*, Vol. 28, No. 2, 2004, pp. 229–238.

[9] Nelder, J. A., and Mead, R., "A Simplex Method for Function Minimization," *Computer Journal (UK)*, Vol. 7, No. 4, 1965, pp. 308–313.

[10] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Art of Scientific Computing: Numerical Recipes in C*, 2nd ed., Cambridge Univ. Press, New York, 1993, pp. 408–412, 444–455.

[11] Cardoso, M. F., Salcedo, R. L., and De Azevedo, S. F., "Simplex-Simulated Annealing Approach to Continuous Non-Linear Optimization," *Computers and Chemical Engineering*, Vol. 20, No. 9, 1996, pp. 1065–1080.

[12] Moltoda, T., and Stengel, R. F., "Robust Control System Design Using Simulated Annealing," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 2, 2002, pp. 267–274.

[13] Behzadi, B., Ghotbi, C., and Galindo, A., "Application of the Simplex Simulated Annealing Technique to Nonlinear Parameter Optimization for the SAFT-VR Equation of State," *Chemical Engineering Science*, Vol. 60, No. 23, 2005, pp. 6607–6621.

[14] Kvasnicka, V., and Pospichal, J., "A Hybrid of Simplex Method and Simulated Annealing," *Chemometrics and Intelligent Laboratory Systems*, Vol. 39, No. 2, 1997, pp. 161–173.

[15] Wang, L., and Zheng, D. Z., "Effective Hybrid Optimization Strategy for Complex Functions with High-Dimension," *Journal of Tsinghua University*, Vol. 41, No. 9, 2001, pp. 118–121 (in Chinese).

[16] Luo, Y. Z., and Tang, G. J., "Parallel Simulated Annealing Using Simplex Method," *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA Paper 2004-4584, 2004.

[17] Park, M. W., and Kim, Y. D., "Systematic Procedure for Setting Parameters in Simulated Annealing Algorithms," *Computers and Operations Research*, Vol. 25, No. 3, 1998, pp. 207–217.

[18] Kim, Y. D., Lim, H. G., and Park, M. W., "Search Heuristics for a Flowshop Scheduling Problem in a Printed Circuit Board Assembly Process ," *European Journal of Operational Research*, Vol. 91, No. 1, 1996, pp. 857–868.

[19] Luo, Y. Z., and Tang, G. J., "Near-Optimal Low-Thrust Orbit Transfer Generated by Parallel Simulated Annealing," *55th International Astronautical Congress*, International Astronautical Congress Paper 04-A.6.10, Canada, 2004.

[20] Luo, Y. Z., Tang, G. J., and Li, H. Y., "Optimization of Rendezvous Phasing Maneuvers Using Multiple-Revolution Lambert Solution," *Journal of Guidance, Control, and Dynamics* (submitted for publication).

[21] Sousa, F., Ramos, F., Paglione, P., and Girardi, R. M., "New Stochastic Algorithm for Design Optimization," *AIAA Journal*, Vol. 41, No. 9, 2003, pp. 1808–1817.

[22] Moh, J. S., and Chiang, D. Y., "Improved Simulated Annealing Search for Structural Optimization," *AIAA Journal*, Vol. 38, No. 10, 2000, pp. 1965–1973.

[23] Yip, P. P. C., and Pao, Y. H., "Combinatorial Optimization with Use of Guided Evolutionary Simulated Annealing," *IEEE Transactions on Neural Networks*, Vol. 6, No. 2, 1995, pp. 290–295.

[24] Rao, S. S., *Engineering Optimization*, 3rd ed., Wiley, New York, 1996.

[25] Li, H. L., and Papalambros, P. A., "Production System for Use of Global Optimization Knowledge," *Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 107, No. 2, 1985, pp. 277–284.

[26] Kuang, J. K., Rao, S. S., and Chen, L., "Taguchi-Aided Search Method for Design Optimization of Engineering Systems," *Engineering Optimization*, Vol. 30, No. 1, 1998, pp. 1–23.

[27] Azarm, S., and Li, W. C., "Multi-Level Design Optimization Using Global Monotonicity Analysis," *Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 111, No. 2, 1989, pp. 259–263.

[28] Ray, T., "Golinski's Speed Reducer Problem Revisited," *AIAA Journal*, Vol. 41, No. 3, 2003, pp. 556–558.

[29] Hajela, P., and Yoo, J., "Constraint Handling in Genetic Search Using Expression Strategies," *AIAA Journal*, Vol. 41, No. 12, 2003, pp. 2414–2420.

[30] Wolpert, D. H., and Macready, W. G., "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, 1997, pp. 67–82.

S. Saigal
Associate Editor