Drawing Inspiration From Human Design Teams for Better Search and Optimization: The Heterogeneous Simulated Annealing Teams Algorithm

Christopher McComb
Mem. ASME
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: ccm@cmu.edu

Jonathan Cagan
Fellow ASME
Department of Mechanical Engineering,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: cagan@cmu.edu

Kenneth Kotovsky
Department of Psychology,
Carnegie Mellon University,
Pittsburgh, PA 15213
e-mail: kotovsky@cmu.edu

Insights uncovered by research in design cognition are often utilized to develop methods used by human designers; in this work, such insights are used to inform and improve computational methodologies. This paper introduces the heterogeneous simulated annealing team (HSAT) algorithm, a multiagent simulated annealing (MSA) algorithm. HSAT is based on a validated computational model of human-based engineering design and retains characteristics of the model that structure interaction between team members and allow for heterogeneous search strategies to be employed within a team. The performance of this new algorithm is compared to several other simulated annealing (SA) based algorithms on three carefully selected benchmarking functions. The HSAT algorithm provides terminal solutions that are better on average than other algorithms explored in this work.
[DOI: 10.1115/1.4032810]

1 Introduction

Focused research has uncovered mechanisms of design cognition and revealed insights that can be used to change the methods used by human designers [1]. A number of studies have also sought to inform better computational optimization and design tools from the results of design studies. These include the use of machine learning algorithms to learn stylistic aspects of design [2], design tools that are based on empirical studies of human analogy use [3], and research aimed at enhancing the synergy between humans and computational agents [4].

This work specifically draws upon the cognitively inspired simulated annealing teams (CISAT) framework, an agent-based model of human design teams [5]. CISAT employs a multiagent SA framework that is overlaid with eight characteristics, each describing a facet of human or team behavior that has been described in the design or problem-solving literature. McComb et al. validated the CISAT framework by using it to replicate and analyze the results of a design task solved by human teams [5].

Although computational optimization algorithms are seen as more effective than human-based optimization because of the ability to handle large numbers of parameters through rapid calculation, this work is different. SA, a stochastic optimization algorithm, lies at the root of CISAT because of its ability to mimic aspects of human search [6]. Because of this relationship, the question arose: *Can aspects of human problem-solving be used to inform a new variation of SA via CISAT?* By selectively retaining several CISAT characteristics, this work seeks to create an SA-based numerical optimization algorithm that incorporates beneficial aspects of engineering design teams.

Interaction between members of a team enables them to divergently explore a design space and later convergently focuses their efforts on a diminishing set of alternatives [7]. In order to accomplish the divergent stage, members of a team must be capable of independently tailoring their approach as they search for solutions—a behavior accounted for by the *locally sensitive search* characteristic in CISAT [5]. This specifically reflects the ability of expert designers to use a mixture of depth- and breadth-first solution strategies [8]. To accomplish the convergent stage, the members of the team must have some mechanism for interacting and sharing solutions—a behavior accounted for by the *quality-informed solution sharing* characteristic of CISAT [5]. This reflects the fact that members of a design team factor design quality into decisions, but are also able to pursue designs that may currently display lower quality [9].

This work introduces the HSAT algorithm. This algorithm is based on the CISAT framework and specifically retains the two CISAT characteristics introduced above: locally sensitive search and quality-informed solution sharing. The HSAT algorithm is compared to a variety of SA-based algorithms on three benchmarking functions and consistently provides terminal solutions that are better on average than the other SA-based algorithms explored.

2 Background

The conventional SA algorithm is based on the physical annealing process in which materials are heated and cooled in a controlled manner to minimize residual stresses [10]. A conceptual flowchart of the conventional SA algorithm is provided in Fig. 1.

The annealing schedule dictates the simulated temperature, which in turn dictates the probability of accepting a worse solution. Schedules that adapt to the solution space offer better performance than classical nonadaptive schedules. Two annealing schedules are used in this work: the classical Cauchy schedule and the Triki adaptive schedule [11]. The Triki adaptive annealing schedule is incorporated into the HSAT algorithm, while the Cauchy annealing schedule is used exclusively for comparison. The temperature is not updated after every iteration, but instead updated intermittently every n iterations (i.e., dwell time). For the Cauchy schedule, the temperature is updated as

$$T_{i+1} = \frac{T_0}{1 + \delta_C \cdot i} \tag{1}$$

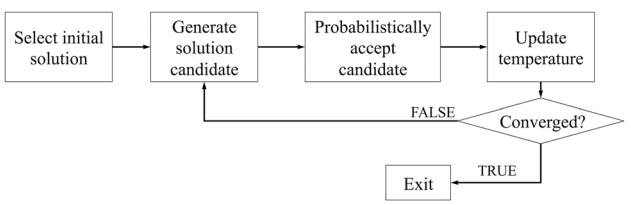

Fig. 1 Generalized flowchart for conventional SA

Contributed by the Design Automation Committee of ASME for publication in the Journal of Mechanical Design. Manuscript received June 3, 2015; final manuscript received February 6, 2016; published online March 2, 2016. Assoc. Editor: Kazuhiro Saitou.

where T_0 is the initial temperature, i is the index of the current iteration, and δ_c is a parameter that allows the schedule to be extended or compressed. The Triki annealing schedule updates temperature as

$$T_{i+1} = T_i\left(1 - \frac{T_i \cdot \delta_T}{\sigma^2_{f(x)}}\right) \tag{2}$$

where δ_T is a parameter that controls how quickly adaptation occurs, and $\sigma^2_{f(x)}$ is the variance of objective function value of candidate solutions entertained since the last temperature update.

MSA algorithms employ software agents to operate on multiple solutions. Existing MSA algorithms utilize principles of differential evolution and particle swarm optimization to accomplish interaction between agents [12,13]. However, the agents in existing algorithms do not possess any sort of individual strategy or preference for exploring solutions, a property which is an integral part of HSAT.

3 The HSAT Algorithm

HSAT is an MSA algorithm (depicted in Fig. 2) that retains two characteristics from the CISAT modeling framework. The interaction between agents in HSAT is structured according to the *quality-informed solution sharing* characteristic from CISAT. The HSAT agents are also provided with individually controlled adaptive temperature schedules, thus implementing the *locally sensitive search* characteristic from CISAT.

When agents are instantiated, each selects a candidate solution at random from the design space. The HSAT algorithm then begins iterations to optimize the objective function. At the beginning of every iteration, the objective function value of every agent's current solution is shared with the other agents in the team through the vector $\mathbf{F}$

$$\mathbf{F} = [f(\mathbf{x}_i^1),\ f(\mathbf{x}_i^2), \ldots, f(\mathbf{x}_i^N)] \tag{3}$$

where $f(\mathbf{x})$ is the objective function. Note that subscripts indicate iteration number, while superscripts indicate different agents. A new vector $\mathbf{W}$ is then defined as the relative function value of each current solution compared against the worst current solution

$$\mathbf{W} = -\mathbf{F} + \max(\mathbf{F}) \tag{4}$$

Note that this formulation of the weighting vector only applies to minimization problems (such as those used in this work). Conversion to a maximization problem can be accomplished by changing the sign of the objective function and changing "max" to "min."

Each agent handles the remaining operations in the iteration independently by first selecting a starting solution using the equation

$$j = \text{mult}\left(\frac{\mathbf{W}}{\sum_i W_i}\right) \tag{5}$$

where the function "mult" returns a draw from the multinomial distribution defined by the vector of probabilities proportional to the weighting vector $\mathbf{W}$. This quality-informed solution sharing procedure probabilistically encourages agents to pursue the best solutions. The solution selected through this process, $\mathbf{x}_i^j$, then replaces the agent's current solution. A new candidate solution for agent k, $\mathbf{x}^k_{\text{new}}$, is created by drawing at random from the Cauchy distribution and adding the resulting vector to the current solution $\mathbf{x}_i^j$. This is accomplished by computing

$$\mathbf{x}^k_{\text{new}} = \mathbf{x}_i^j + T_i^k \tan\left(\text{uniform}(-\pi/2, \quad \pi/2, \quad D)\right) \tag{6}$$

where T_i^k is the current temperature of agent k. The function "uniform" draws a point at random from the continuous D -dimensional space with an upper bound of $\pi/2$ and a lower bound of $-\pi/2$ in each direction. The Cauchy distribution has thicker tails than the Gaussian distribution and thus encourages more extensive search.

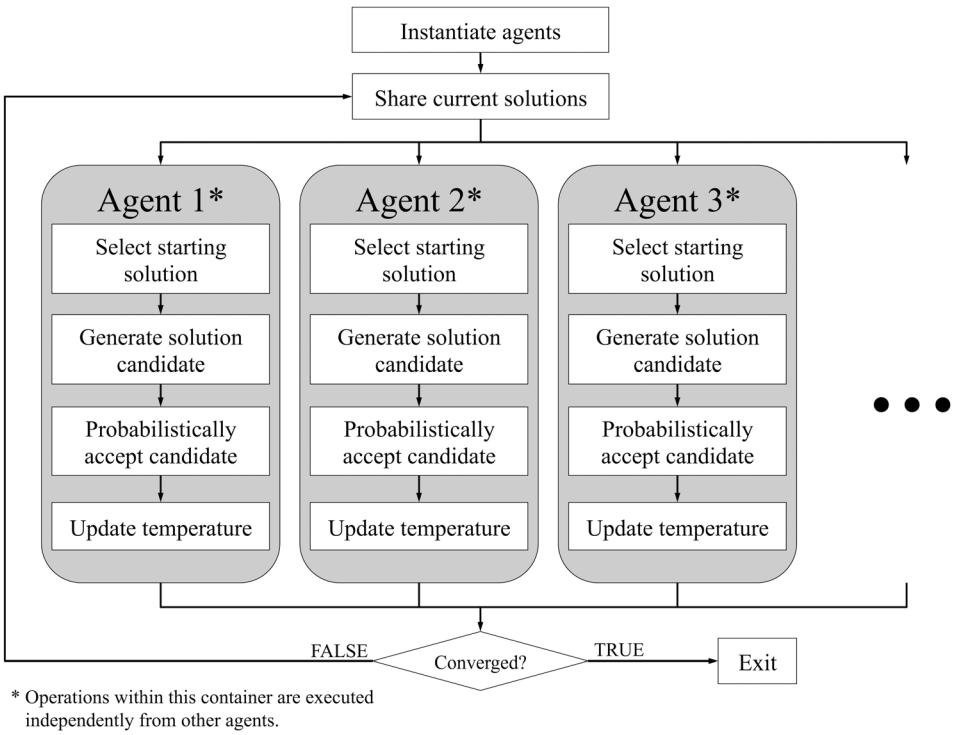

Fig. 2 Flowchart for the HSAT algorithm

Table 1 Summary of SA-based algorithms

	Classical annealing schedule (Cauchy)	Adaptive annealing schedule (Triki)
Single-agent	Nonadaptive SA: single-agent with Cauchy annealing schedule	Adaptive SA: single-agent with Triki annealing schedule
Multi-agent	Nonadaptive MSA: multiple interacting agents with Cauchy annealing schedule	HSAT: multiple interacting agents with Triki annealing schedule for *each* agent

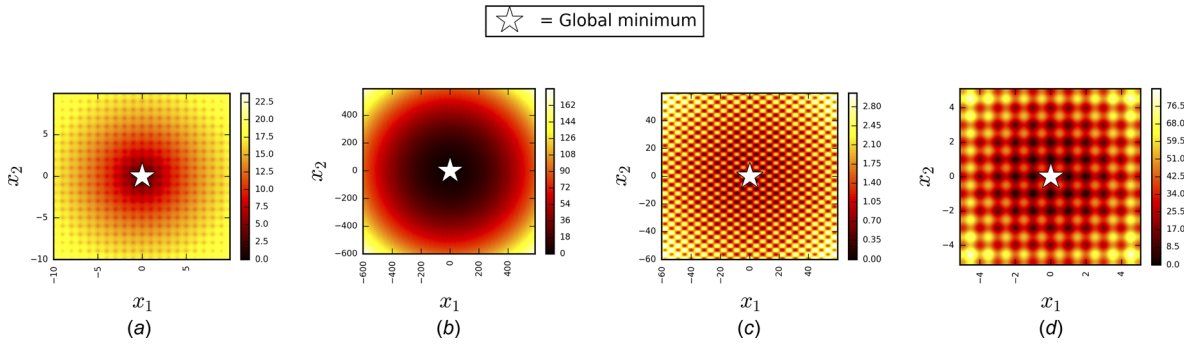

Fig. 3 Two-dimensional representations of benchmarking functions: (*a*) Ackley function, (*b*) Griewank, global, (*c*) Griewank, local, and (*d*) Rastrigin function

If the new solution candidate, $\mathbf{x}_{\text{new}}^k$, is better than the agent's previous solution, $\mathbf{x}_i^k$, the solution candidate is accepted. If it is not better, the agent still accepts the solution with acceptance probability computed as

$$p = \exp\left(\frac{f(\mathbf{x}_{\text{new}}) - f(\mathbf{x}_i)}{T_i}\right) \tag{7}$$

If the new solution is not accepted, the previous solution is carried into the next iteration ($\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i$). Finally, the temperature is updated using the Triki annealing schedule (Eq. (2)). The temperature is updated independently by each agent, thus allowing agents to individually engage in locally sensitive search.

4 Comparison Methodology

The HSAT algorithm employs two features inspired by characteristics observed in human design teams. In order to fully understand the impact of these features, HSAT is compared to three other SA-based algorithms (see Table 1). For equivalent comparison, every algorithm is permitted the same number of objective function evaluations during each run.

Algorithm performance is assessed with respect to three continuous functions. These functions are the Ackley function (Eq. (8)), the Griewank function (Eq. (9)), and the Rastrigin function (Eq. (10)). The variable D indicates the number of dimensions in the search space

$$f(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{D} x_i}{D}}\right) - \exp\left(\frac{\sum_{i=1}^{D}\cos(2\pi x_i)}{D}\right) + 20 + \exp(1)$$
$$-10 \le x_i \le 10 \qquad \forall i \tag{8}$$

$$f(\mathbf{x}) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$
$$-600 \le x_i \le 600 \qquad \forall i \tag{9}$$

$$f(\mathbf{x}) = \sum_{i=1}^{D}(x_i^2 - 10\cos(2\pi x_i) + 10)$$
$$-5.12 \le x_i \le 5.12 \qquad \forall i \tag{10}$$

For the numerical experiments conducted as part of this work, every equation is implemented with 30 dimensions. For purposes of illustration, representations of these functions in two dimensions are provided in Fig. 3. The global minimum is shown with a white star.

Each of these functions presents distinct challenges. The global minimum of the Ackley function resides in a central well, while much of the function is fairly flat. Therefore, minimizing this function requires a broad search to find the well and then a local search to find the global minimum. In other words, an effective search requires breadth, followed by depth. The Griewank function is globally convex, but in the neighborhood of the global minimum there are many minima with very similar values. An effective minimization of this function would require depth (to follow the global behavior) followed by breadth (to search local minima). The Rastrigin function is composed of a number of deep wells, all of which contain local minima that are similar in value to the global minimum. Therefore, this function requires a combination of breadth (to search multiple wells) and depth (to efficiently minimize within each well).

A total of 100,000 function evaluations are allowed when algorithms solve the Ackley and Griewank functions. However, due to its contour, algorithms are permitted 250,000 objective function evaluations when solving the Rastrigin function.

A preprocessing step is used to determine the best parameters for each SA-based algorithm. This preprocessing step performs a pattern search to maximize the mean objective function quality of terminal solutions for a given objective function with respect to the relevant algorithm parameters. The parameters resulting from this process are shown in Table 2.

5 Performance Benchmarking

Using the parameters from Table 2, each benchmarking function is solved 100 times with each of the algorithms. Cumulative

Table 2 Parameters used for SA-based algorithms

Algorithm	Function	Number of agents, N	Initial temperature, T_0	Cauchy parameter, δ_C	Triki parameter, δ_T	Dwell time, n
Nonadaptive SA	Ackley	1	5.77×10^{-1}	7.04×10^{-2}	—	3
	Griewank	1	7.14×10^{-1}	4.08×10^{-3}	—	10
	Rastrigin	1	3.92×10^{-1}	9.87×10^{-4}	—	11
Adaptive SA	Ackley	1	4.17×10^{-1}	—	1.72×10^{-1}	126
	Griewank	1	9.17×10^{-3}	—	4.12×10^{-7}	43
	Rastrigin	1	3.52×10^{-3}	—	6.50×10^{-1}	33
Nonadaptive MSA	Ackley	5	9.84×10^{-1}	4.34×10^{-1}	—	8
	Griewank	8	1.52×10^{0}	2.12×10^{-2}	—	12
	Rastrigin	7	3.30×10^{-1}	8.87×10^{-3}	—	23
HSAT	Ackley	7	1.65×10^{-2}	—	2.91×10^{-1}	71
	Griewank	23	6.33×10^{-2}	—	7.47×10^{-1}	75
	Rastrigin	15	6.52×10^{-3}	—	1.97×10^{1}	31

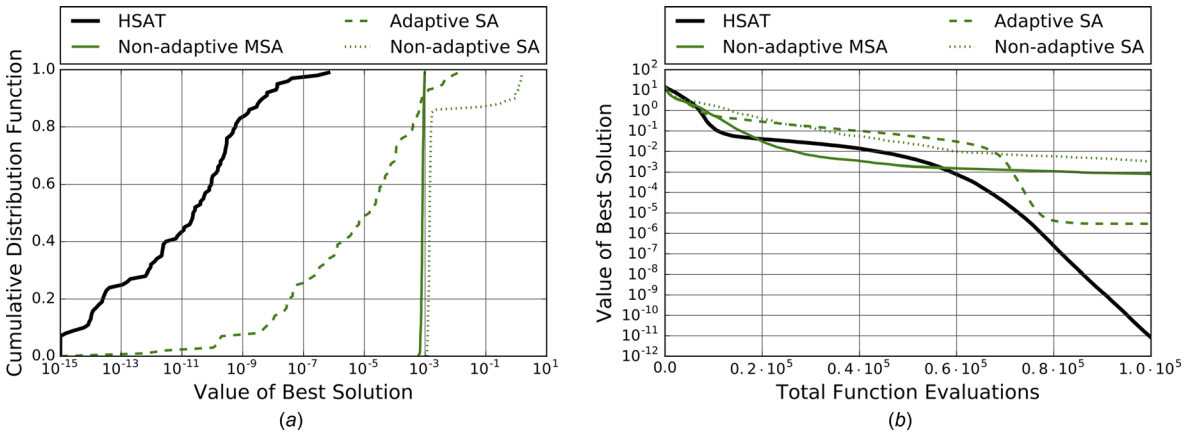

Fig. 4 Comparison of optimization results for Ackley function (error bars omitted for clarity): (*a*) cumulative distribution of terminal solutions and (*b*) geometric mean of best solution found over time

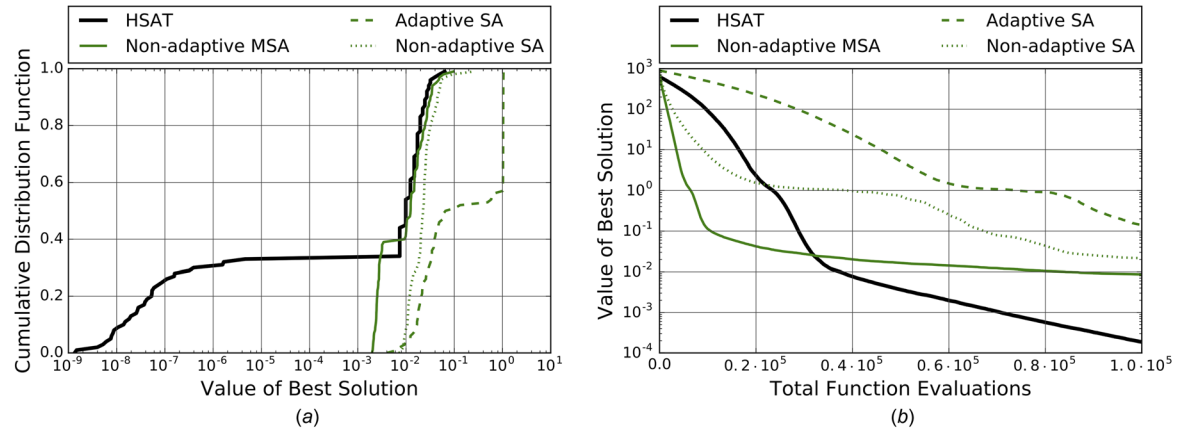

Fig. 5 Comparison of optimization results for Griewank function (error bars omitted for clarity): (*a*) cumulative distribution of terminal solutions and (*b*) geometric mean of best solution found over time

distribution function for the objective function values of the terminal solutions is shown in Figs. 4(*a*), 5(*a*), and 6(*a*).

The best solution achieved by the algorithm over time is shown as a function of total objective function evaluations in Figs. 4(*b*), 5(*b*), and 6(*b*). The geometric mean is used in these plots (rather than the arithmetic mean) because it better communicates central tendency when the data span several orders of magnitude. Error bars are omitted from Figs. 4(*b*), 5(*b*), and 6(*b*) in the interest of visual clarity, but the spread of the terminal solutions can be inferred from the horizontal range of the cumulative distribution functions. Parameters are tuned for near-optimal performance for the given iteration limit, so the value of the best solution continues to improve slowly throughout the allotted runtime.

The HSAT algorithm provides the best mean terminal solutions for every benchmarking function. For the Ackley and Rastrigin functions, the HSAT algorithm returns the best final result by

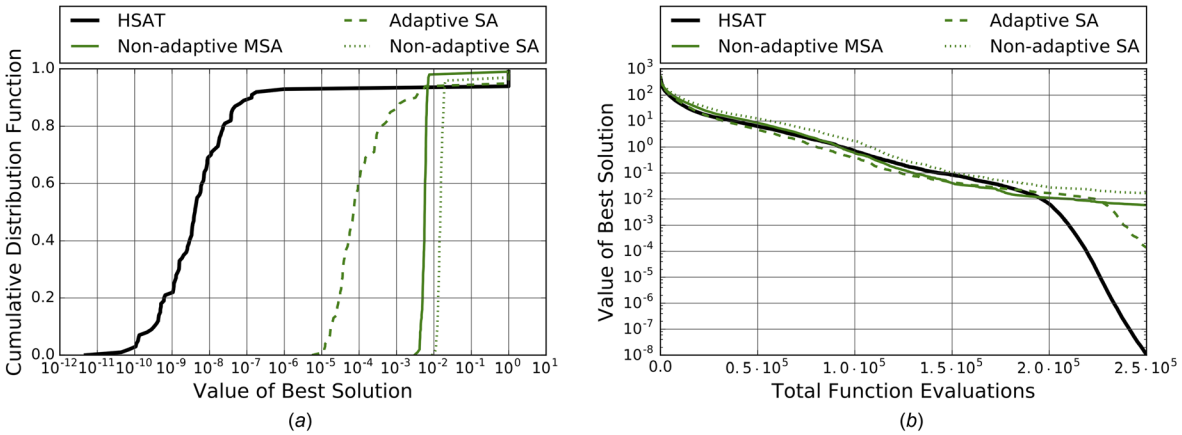

Fig. 6 Comparison of optimization results for Rastrigin function (error bars omitted for clarity): (*a*) cumulative distribution of terminal solutions and (*b*) geometric mean of best solution found over time

several orders of magnitude. HSAT outperforms other algorithms by a smaller margin on the Griewank function.

6 Discussion

Comparing the results of the algorithms across benchmarking functions provides insight into the performance of HSAT. The Ackley function has a large number of local minima that are similar in objective function value, while the global minimum is contained within a central well. The algorithms that employ an adaptive annealing schedule perform best because they are able to transition from a broad search for the central well to a quick descent toward the bottom of the well. In contrast, the Griewank function has convex global behavior, but in the vicinity of the global minimum there are a large number of local minima with similar objective function values. Therefore, the use of multiple interacting agents becomes crucial to success. The Rastrigin function combines the challenges of both the Ackley and Griewank functions, requiring any algorithm to search a number of local minima before beginning a deep dive to the global minimum. For this reason, only the combination of interacting agents and adaptive annealing schedules leads to high performance. These results demonstrate that the unique combination of features found in the HSAT algorithm boosts performance. Further work should use a wider array of benchmarking functions and vary the dimensionality of those functions. It may also be promising to extend HSAT by structuring the interaction between agents to reflect lessons learned from multiteam organizations.

As implemented, this algorithm could be utilized by engineers and designers to optimize highly multimodal parametric design problems; one such application could be layout problems which are known to have fractal-like qualities [14]. With minor changes to how solutions are operated on HSAT could be modified to solve discrete optimization problems as well.

Finally, there are many alternatives to SA-based algorithms for global optimization. These include parallel genetic algorithms which utilize parallelism similar to some MSAs [15], basin-hopping algorithms [16], particle swarm optimization [17], efficient global optimization [18], branch-and-bound methods [19], and pattern search methods [20]. Future work can compare these algorithms to HSAT in order to better delineate the relative advantages or disadvantages derived from the characteristics implemented in this work.

7 Conclusions

This paper introduces the HSAT algorithm, an MSA algorithm based upon a computational model of human design teams. The results of the numerical investigation indicate that HSAT is capable of delivering high performance in highly multimodal environments, and that this performance may also be robust across a variety of function topographies.

Acknowledgment

This material is based upon the work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE125252 and the U.S. Air Force Office of Scientific Research through Grant No. FA9550-12-1-0374. An early version of this work was published in the proceedings of the 2015 International Conference on Engineering Design [21].

References

[1] Dinar, M., Shah, J. J., Cagan, J., Leifer, L., Linsey, J., Smith, S. M., and Hernandez, N. V., 2014, "Empirical Studies of Designer Thinking: Past, Present, and Future," ASME J. Mech. Des., **137**(2), p. 021101.
[2] Tseng, I., Cagan, J., and Kotovsky, K., 2012, "Concurrent Optimization of Computationally Learned Stylistic Form and Functional Goals," ASME J. Mech. Des., **134**(11), p. 111006.
[3] Goel, A. K., Vattam, S., Wiltgen, B., and Helms, M., 2012, "Cognitive, Collaborative, Conceptual and Creative—Four Characteristics of the Next Generation of Knowledge-Based CAD Systems: A Study in Biologically Inspired Design," Comput. Aided Des., **44**(10), pp. 879–900.
[4] Egan, P., Cagan, J., Schunn, C., and LeDuc, P. R., 2015, "Synergistic Human-Agent Methods for Deriving Effective Search Strategies: The Case of Nanoscale Design," Res. Eng. Des., **26**(2), pp. 145–169.
[5] McComb, C., Cagan, J., and Kotovsky, K., 2015, "Lifting the Veil: Drawing Insights About Design Teams From a Cognitively-Inspired Computational Model," Des. Stud., **40**, pp. 119–142.
[6] Cagan, J., and Kotovsky, K., 1997, "Simulated Annealing and the Generation of the Objective Function: A Model of Learning During Problem Solving," Comput. Intell., **13**(4), pp. 534–581.
[7] Fu, K., Cagan, J., and Kotovsky, K., 2010, "Design Team Convergence: The Influence of Example Solution Quality," ASME J. Mech. Des., **132**(11), p. 111005.
[8] Ball, L. J., and Ormerod, T. C., 1995, "Structured and Opportunistic Processing in Design: A Critical Discussion," Int. J. Hum. Comput. Stud., **43**(1), pp. 131–151.
[9] McComb, C., Cagan, J., and Kotovsky, K., 2015, "Rolling With the Punches: An Examination of Team Performance in a Design Task Subject to Drastic Changes," Des. Stud., **36**(1), pp. 99–121.
[10] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., 1983, "Optimization by Simulated Annealing," Science, **220**(4598), pp. 671–680.
[11] Triki, E., Collette, Y., and Siarry, P., 2005, "A Theoretical Study on the Behavior of Simulated Annealing Leading to a New Cooling Schedule," Eur. J. Oper. Res., **166**(1), pp. 77–92.
[12] Zhong, Y., Ning, J., and Zhang, H., 2012, "Multi-Agent Simulated Annealing Algorithm Based on Particle Swarm Optimisation Algorithm," Int. J. Comput. Appl. Technol., **43**(4), pp. 335–342.
[13] Zhong, Y., Wang, L., Wang, C., and Zhang, H., 2012, "Multi-Agent Simulated Annealing Algorithm Based on Differential Evolution Algorithm," Int. J. Bio-Inspired Comput., **4**(4), pp. 217–228.

[14] Cagan, J., Shimada, K., and Yin, S., 2002, "A Survey of Computational Approaches to Three-Dimensional Layout Problems," Comput. Aided Des., **34**(8), pp. 597–611.
[15] Alba, E., and Troya, J. M., 1999, "A Survey of Parallel Distributed Genetic Algorithms," Complexity, **4**(4), pp. 31–52.
[16] Wales, D., and Doye, J. P. K., 1997, "Global Optimization by Basin-Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms," J. Phys. Chem. A, **101**(28), pp. 5111–5116.
[17] Eberhart, R. C., and Shi, Y., 2001, "Particle Swarm Optimization: Developments, Applications and Resources," 2001 Congress on Evolutionary Computation Seoul, Korea, May 27–20, IEEE, New York, pp. 81–86.
[18] Jones, D. R., Schonlau, M., and William, J., 1998, "Efficient Global Optimization of Expensive Black-Box Functions," J. Global Optim., **13**(4), pp. 455–492.
[19] Kearfott, R. B., and Kreinovich, V., eds., 1996, *Applications of Interval Computations*, Springer, Boston, MA.
[20] Yin, S., and Cagan, J., 2000, "An Extended Pattern Search Algorithm for 3D Component Layout," ASME J. Mech. Des., **122**(1), pp. 102–108.
[21] McComb, C., Cagan, J., and Kotovsky, K., 2015, "Heterogeneous Simulated Annealing Teams: An Optimizing Search Algorithm Inspired by Engineering Design Teams," International Conference on Engineering Design, Milan, Italy, July 27–30, pp. 225–234.