10.1109/SIS.2008.4668317

2008 IEEE Swarm Intelligence Symposium
St. Louis MO USA, September 21-23, 2008

Roach Infestation Optimization

Timothy C. Havens, *Student Member, IEEE*, Christopher J. Spain, *Student Member, IEEE*, Nathan G. Salmon, *Student Member, IEEE*, and James M. Keller, *Fellow, IEEE*

***Abstract*—There are many function optimization algorithms based on the collective behavior of natural systems — *Particle Swarm Optimization* (PSO) and *Ant Colony Optimization* (ACO) are two of the most popular. This paper presents a new adaptation of the PSO algorithm, entitled *Roach Infestation Optimization* (RIO), that is inspired by recent discoveries in the social behavior of cockroaches. We present the development of the simple behaviors of the individual agents, which emulate some of the discovered cockroach social behaviors. We also describe a "hungry" version of the PSO and RIO, which we aptly call *Hungry PSO* and *Hungry RIO*. Comparisons with standard PSO show that Hungry PSO, RIO, and Hungry RIO are all more effective at finding the global optima of a suite of test functions.**

***Index Terms*—cockroach, optimization, particle swarm.**

I. Introduction

THERE is a strong precedent for biologically-inspired algorithms in computational intelligence — the motto of the IEEE Computational Intelligence Society is, "Mimicking nature for problem solving". Researchers have modeled algorithms on the behavior of natural systems such as flocking birds, shoals of fish, bacterial growth, and colonies of ants [1, 2]. The collective behavior of these natural systems is surmised to be an aggregate result of decentralized and simple behaviors of the individuals [3]. Hence, the development of algorithms that mimic collective behavior is a type of reverse-engineering, in which one uses available information about the natural system to create or discover the simple behaviors of the individuals.

Recent discoveries in the behavior of cockroaches are the inspiration for our proposed algorithm, *Roach Infestation Optimization* (RIO). Studies have shown that cockroaches not only have a distaste for the light, but they also enjoy the company of friends. Jeanson et al. [4] determined, through experiment, that cockroach larvae exhibit a complex collective behavior that ultimately results in the formation of aggregates — in other words, cockroaches like to hang out with friends. Halloy et al. [5] and Ame et al. [6] further studied the social behavior of cockroaches by guiding groups of cockroaches with cockroach-like robots. Interestingly, this experiment showed that the individual decisions of cockroaches modulate the collective behavior of the entire group, which supports the hypothesis that collective behavior is aggregated from simple decentralized behavior. This experiment also accentuated the hypothesis that cockroaches prefer to be in groups, as well as in the dark. Garnier et al. [7] were able to mimic the behavior of cockroaches with a group of cockroach-like robots, each programmed with a simple set of behaviors. We discuss these studies in more detail in Section II.

Our algorithm, RIO, is inspired by the collective and individual behaviors of cockroaches. Section III describes our adaptation of cockroach behaviors to *Particle Swarm Optimization* (PSO) [as described in 8–10]. In essence, we are augmenting the optima-searching behavior of the PSO with cockroach behaviors. Section IV presents numerical examples of the algorithms. We compare the ability of PSO, RIO, and the hungry versions of these algorithms in finding the global optima of several test functions. There were several cases in which each algorithm converged to local optima. However, Hungry PSO, RIO, and Hungry RIO were able to find the global optima in more tests than the PSO. We summarize this paper in Section V.

II. Cockroach Behaviors

Recent research has shown that cockroaches are more social than we would like to believe. After all, we would rather think that one of the worst asthma triggers [11] is just an uncivilized insect. To the contrary, cockroaches have shown complex social behavior and reasoning in multiple studies [4, 5, 12]. It is these behaviors that we use as a model for our function optimization algorithms.

Jeanson et al. [4] conducted an experiment where the behavior of individual cockroach larvae was tracked in a enclosed circular arena. They discovered that roaches upon entering the arena would first exhibit a wall-following behavior along the periphery. Some individuals would then choose to explore the central area of the experiment arena. The cockroaches would also form aggregates or groups. Table I summarizes the numerical results of this study [4]. An encounter is defined as two cockroaches coming within 6mm of one another and a collision is defined as when a moving cockroach encounters a stopped cockroach that is a member of an aggregate. Notice that the probability of leaving an aggregate (or moving) is very low when the size of the group is greater than one. This is indicative of the cockroaches' desire to stay in aggregates or groups. The behaviors outlined in Table I will be important in the design of the individual behaviors of the RIO agents.

Manuscript received June 15, 2008; revised July 19, 2008.

T.C. Havens is with the Department of Electrical and Computer Engineering, University of Missouri. (e-mail: havenst@gmail.com).

C.J. Spain is with the Department of Electrical and Computer Engineering, University of Missouri. (e-mail: cjs2pc@mizzou.edu).

N.G. Salmon is with the Department of Electrical and Computer Engineering, University of Missouri. (e-mail: ngsp68@mizzou.edu).

J.M. Keller is with the Department of Electrical and Computer Engineering, University of Missouri. (e-mail: kellerj@missouri.edu).

TABLE I
NUMERICAL RESULTS OF COCKROACH AGGREGATION BEHAVIOR STUDY [4]

Characteristic	Measured value	Description
$1/\tau_{\text{stop,p}}$	0.08/s	Probability per unit time of stopping in periphery
$1/\tau_{\text{exit}}$	0.13/s	Probability per unit time of exiting periphery
$F_{\text{stop,c}}$	0.21	Fraction of paths in the center of arena that ended before reaching periphery
$1/\tau_{\text{stop,c}}$	0.03/s	Probability per unit time of stopping in center
$1/\tau_{\text{stop,N}}$	**0.49**/s $(N=1)$	Probability per unit time of stopping when encountering N friends
	0.63/s $(N=2)$	
	0.65/s $(N=3)$	
$1/\tau_{\text{collision,N}}$	0.27/s $(N=1)$	Probability per unit time of moving after collision for group of N size
	0.052/s $(N=2)$	
	0.021/s $(N=3)$	

Halloy and Ame et. al [5, 6] further revealed intricate social behavior in roaches with a novel experiment. These studies showed that cockroaches can be persuaded to aggregate under the lighter of two dark discs by cockroach-like robots. In essence, they determined that cockroaches prefer to optimize the number of friends and the darkness of the shelter simultaneously. Halloy and Ame also developed a simple behavior model for the cockroaches that was able to accurately predict how they aggregate under shelters. This model relied on two rates - R the rate of entering a shelter, and Q the rate of quitting a shelter. These rates were defined by the capacity of the shelter and the number of individuals (and robots) in the shelter.

Garnier et al. [7] showed that the collective behavior of cockroaches could be simulated by groups of robots, each programmed with a simple set of behaviors based on the findings in [4]. For this reason, we think that cockroaches are an excellent model for swarm intelligence algorithms.

Algorithm 1: Particle Swarm Optimization (PSO) [9, 10]

Input: Fitness function $F(\vec{x}) \in \mathbb{R}^D$
Parameters:
$N_p = 20$, Number of particles
$t_{max} = 1000$, Maximum iterations
$C_0 = 0.7$, $C_{max} = 1.43$, Swarm parameters
Initialize population, $\vec{x}_i$ and $\vec{v}_i$, randomly
for $t = 1$ *to* t_{max} **do**
  for $i = 1$ *to* N_p **do**
    if $F(\vec{x}_i) < F(\vec{p}_i)$ **then** $\vec{p}_i = \vec{x}_i$
    $\vec{p}_g = \arg\min_{\vec{p}_{\text{neighbors}}} F(\vec{p}_{\text{neighbors}})$
    $\vec{v}_i = C_0\vec{v}_i + C_{max}\vec{R}_1(\vec{p}_i - \vec{x}_i) + C_{max}\vec{R}_2(\vec{p}_g - \vec{x}_i)$ (2)
    $\vec{x}_i = \vec{x}_i + \vec{v}_i$

III. ROACH INFESTATION OPTIMIZATION

Based on the studies outlined in the previous section, we defined three simple behaviors of cockroach agents:

1) Cockroaches search for the darkest location in the search space. The level of darkness at a location $\vec{r} \in \mathbb{R}^D$ is directly proportional to the value of the fitness function at that location $F(\vec{r})$.
2) Cockroaches enjoy the company of friends and socialize with nearby cockroaches with a probability equal to the **bolded** values shown in Table I.
3) Cockroaches periodically become hungry and leave the comfort of darkness or friendship to search for food.

These simple behaviors allowed us to begin designing the algorithm. We now examine each behavior individually, but within the context of the whole set of behaviors. Behavior 1), which we denote as *Find_Darkness*, is the main goal of every optimization algorithm — find the minimum (or maximum) of a fitness function $F(\vec{r})$ upon a given search space. Within the cockroach paradigm, we define the level of darkness to be the fitness function value. Hence, if one is searching for a minimum, $-\infty$ is perfectly dark, while ∞ is perfectly light. The RIO is a cockroach-inspired PSO; hence, we model the *Find_Darkness* behavior with a portion of the velocity update equation in the PSO. The PSO algorithm is outlined in Algorithm 1. Equation (2) is the velocity update equation. The part of this equation that models the *Find_Darkness* behavior is

$$\vec{v}_i = C_0\vec{v}_i + C_{max}\vec{R}_1 .* (\vec{p}_i - \vec{x}_i), \tag{1}$$

where $\vec{v}_i$ is the velocity of the ith agent, $\vec{x}_i$ is the current location, $\vec{p}_i$ is the best location found by the ith agent, $\{C_0, C_{max}\}$ are parameters, and $\vec{R}_1$ is a vector of uniform random numbers. The $.*$ in Eq.(1) is element-by-element vector multiplication, as in Matlab [13]. This equation emulates *Find_Darkness* because $(\vec{p}_i - \vec{x}_i)$ is a velocity change in the direction of the darkest known location for that agent.

Behavior 2), which we call *Find_Friends*, is an important element in the RIO algorithm. We assume that all cockroach agents begin as individuals and are governed by only the *Find_Darkness* behavior. If a cockroach agent comes within a detection radius of another cockroach agent, then there is a probability of $1/\tau_{\text{stop,N}}$ (see Table I) that these roaches will socialize (or group). This socializing is emulated in the algorithm by a sharing of information, where this information is the darkest known location. In essence, when two cockroach agents meet, there is a chance that they will communicate their knowledge of the search space to each other. They share their knowledge by setting the darkest local location $\vec{l}$

according to

$$\vec{l}_i = \vec{l}_j = \arg\min_k \{F(\vec{p}_k)\},\ k = \{i, j\}, \tag{3}$$

where $\{i, j\}$ are the indices of the two socializing cockroaches and $\vec{p}_k$ is the darkest known location for the individual cockroach agent (personal best). Equation 1 can now be extended to include the *Find_Friends* behavior,

$$\vec{v}_i = C_0\vec{v}_i + C_{max}\vec{R}_1.*(\vec{p}_i - \vec{x}_i) + C_{max}\vec{R}_2.*(\vec{l}_i - \vec{x}_i). \tag{4}$$

It is immediately obvious that this is very much like the standard PSO velocity update, Eq.(2). However, the global best solution is replaced by a group best solution $\vec{l}_i$. In other words, we are defining the neighborhood of the agents to emulate the cockroach behavior described in Section II. Algorithm 2 outlines the RIO and *Hungry RIO* (HRIO) algorithms.

The last behavior we defined is *Find_Food* — when a cockroach agent becomes hungry it searches for food. We emulate this behavior algorithmically by defining a hunger counter for each agent $hunger_i$. When this counter reaches a chosen threshold the cockroach agent is immediately transported to a random food location $\vec{b}$. These food locations are initialized randomly within a chosen hypercube in the search space. When a piece of food is eaten by a hungry cockroach agent, that piece of food is randomly placed at another location. Hence, there is always food present in the search space. The numbered lines in Algorithm 2 show the *Find_Food* portion of the RIO algorithm. Line 1 checks to see if the ith cockroach agent has a hunger that is greater than the chosen threshold t_{hunger}. Lines 2 and 3 are the RIO update equations as described in the preceding paragraphs and are implemented if the cockroach agent is not hungry. If the cockroach agent is sufficiently hungry then Line 4 transports it to a random food location $\vec{b}$. And then this piece of food is randomly relocated in Line 5. Finally, Line 6 checks to see if one is running the "Hungry" version of the algorithm and, if so, increments the hunger counters. The *Find_Food* behavior periodically perturbs the population, ideally minimizing the chance of converging to a local optima. Algorithm 3 outlines the *Hungry PSO* (HPSO).

Matlab code of the algorithms presented in this paper can be downloaded at http://co-cluster.com.

Algorithm 2: Roach Infestation Optimization (RIO)

Input: Fitness function $F(\vec{x}) \in \mathbb{R}^D$
Parameters:
$N_a = 20$, Number of cockroach agents
$t_{max} = 1000$, Maximum iterations
$C_0 = 0.7$, $C_{max} = 1.43$, Roach parameters
$A_1 = 0.49$, $A_2 = 0.63$, $A_3 = 0.65$, Group parameters
$t_{hunger} = 100$, Hunger interval
Initialization:
set hungers $hunger_i = \text{rand}\{0, t_{hunger} - 1\}$
set population, $\vec{x}_i$ and $\vec{v}_i$, randomly
set food locations $\vec{b}$ randomly
for $t = 1$ *to* t_{max} **do**
  $\mathbf{M} = [M_{jk}] = [||\vec{x}_j - \vec{x}_k||_2]$
  $d_g = \text{median}\{M_{jk} \in \mathbf{M} : 1 \le j < k \le N_a\}$
  for $i = 1$ *to* N_a **do**
    if $F(\vec{x}_i) < F(\vec{p}_i)$ **then** $\vec{p}_i = \vec{x}_i$
    Compute the neighbors of roach i as,
    $\{j\} = \{k : 1 \le k \le N_a, k \ne i, M_{ik} < d_g\}$
    $N_i = \text{number of neighbors } |\{j\}|$
    for $q = 1$ *to* N_i **do**
      if rand[0,1]$< A_{\min\{N_i,3\}}$ **then**
      $\vec{l}_i = \vec{l}_{j_q} = \arg\min_k \{F(\vec{p}_k)\},\ k = \{i, j_q\}$
1   **if** $hunger_i < t_{hunger}$ **then**
2     $\vec{v}_i =$
      $C_0\vec{v}_i + C_{max}\vec{R}_1(\vec{p}_i - \vec{x}_i) + C_{max}\vec{R}_2(\vec{l}_i - \vec{x}_i)$
3     $\vec{x}_i = \vec{x}_i + \vec{v}_i$
    else
4     $\vec{x}_i = \text{Random food location } \vec{b}$
5   Relocate eaten food randomly
      $hunger_i = 0$
6   **if** *Hungry* **then**
      Increment $hunger_i$ counters

IV. Numerical Examples

This section presents a number of numerical examples that show the relative ability of the PSO, HPSO, RIO, and HRIO in finding global minima of multi-dimensional test functions. We also show a number of illustrative results that display the difference in behavior of the cockroach inspired algorithms and the PSO.

Each algorithm was run on each of the test functions presented in the Appendix. The bounds shown in the Appendix for each test function were the bounds on the initial positions of the agents. However, after the algorithm was started the agents were not bound to a test space — they were allowed to explore unlimitedly. This represents the most general type of optimization problem as many optimization problems have an unknown search space; thus, we did not limit the agents to a prescribed search space. To maintain parity between each set of tests, we used the same initialization position and velocity for each of the algorithms. The algorithms were also presented the same sequence of random number vectors for $\vec{R}_1$ and $\vec{R}_2$ (as shown in Algorithms 1-3). This procedure was repeated 11 times for each test function, each run with a different set of initialized positions and velocities, and random number vector sequence. This test procedure ensured that each algorithm could be compared fairly in both overall results and in individual test cases. Algorithm 4 outlines our test procedure in algorithmically.

Table II outlines the results of our testing of PSO, HPSO, RIO, and HRIO, on nine different test functions. These test functions are outlined in detail in the Appendix. Seven of the functions were tested in both 2 and 10 dimensions, while two of the functions were strictly 2-dimensional. The bold

Algorithm 3: Hungry Particle Swarm Optimization (HPSO)

Input: Fitness function $F(\vec{x}) \in \mathbb{R}^D$
Parameters:
$N_p = 20$, Number of particles
$t_{max} = 1000$, Maximum iterations
$C_0 = 0.7$, $C_{max} = 1.43$, Swarm parameters
$t_{hunger} = 100$, hunger interval
Initialization:
set hungers, $hunger_i = \text{rand}\{0, t_{hunger} - 1\}$, randomly
set population, $\vec{x}_i$ and $\vec{v}_i$, randomly
set food locations $\vec{b}$, randomly
for $t = 1$ *to* t_{max} **do**
  for $i = 1$ *to* N_p **do**
    if $F(\vec{x}_i) < F(\vec{p}_i)$ **then** $\vec{p}_i = \vec{x}_i$

$$\vec{p}_g = \arg\min_{\vec{p}_{\text{neighbors}}} F(\vec{p}_{\text{neighbors}}) \tag{5}$$

    if $hunger_i < t_{hunger}$ **then**
      $\vec{v}_i =$
      $C_0\vec{v}_i + C_{max}\vec{R}_1(\vec{p}_i - \vec{x}_i) + C_{max}\vec{R}_2(\vec{p}_g - \vec{x}_i)$
      $\vec{x}_i = \vec{x}_i + \vec{v}_i$
    else
      $\vec{x}_i =$ Random food location $\vec{b}$
      Relocate eaten food randomly
      $hunger_i = 0$
  Increment $hunger_i$ counters, $\forall i$

Algorithm 4: Testing Procedure

for $i = 1$ *to* 11 **do**
1 Draw random initialization position $\vec{x}_i$ and velocity $\vec{v}_i$
2 Draw random vector sequences $\{\vec{R}_1\}_{1000}$ and $\{\vec{R}_2\}_{1000}$
  for *each test function* **do**
    Run PSO, HPSO, RIO, and HRIO using initialization and random vectors drawn in lines 1 and 2
    Record results

values in Table II indicate the algorithm(s) that performed best for each test function. The last row in the table shows the number of tests in which the HPSO, RIO, and HRIO performed as good or better than the PSO. As Table II shows, the RIO performed as good or better than the PSO in 9 out of 14 tests, while the HPSO and HRIO performed as good or better than the PSO in 7 out of 14 tests. In essence, this shows that the cockroach-inspired algorithms are as good or better than the PSO. This is an interesting result, as the goal of our algorithm development was not to create a "better" PSO, it was inspired by the documented social behavior of cockroaches.

Now we discuss the results more specifically. The HPSO, RIO, and HRIO performed better than the PSO in almost every 10-dimensional test. We expect the PSO to be the best algorithm for the Sphere function as this function is monotonically decreasing to the global optima, with no local optima. The Rastrigin, Ackley, and Griewank are all highly-modal functions and the proposed algorithms, on average, are more effective at finding the global optima for these functions. The Rosenbrock function is special in that there are no local optima for $D < 4$ dimensions. And we see that the PSO is more effective in the 2-dimensional case, while the RIO and HRIO and the most effective for the 10-dimensional Rosenbrock (there is one local minima in $D \geq 4$ dimesional Rosenbrock functions [14]). Interestingly, the PSO appears to be more effective than the RIO and HRIO for the 10-dimensional Ackley function. However, we examined the results more closely and came to a different conclusion.

Table III details the results for each of the 11 runs for the 10-dimensional Ackley function. The bold values in the table indicate that the algorithm converged to a local minima. The last row contains the tally of the number of runs in which the algorithm converged to the global optima (within the monotonically decreasing area around the global minima). The mean result, shown in Table II, appears to suggest that the PSO is more effective than the RIO and HRIO for this function. However, it is clear, from the results shown in Table III, that the HPSO, RIO, and HRIO are more effective in finding the global optima of the 10-dimensional Ackley function. Additionally, the HPSO is the most effective algorithm for this case, converging to the global optima in all 11 test runs.

Table IV details the results for each of the 11 runs on the 2-dimensional Griewank function. Again, the bold values in these tables indicate that the algorithm converged to a local minima. The last line tallies the number of runs in which the algorithm converged to the global optima. It is clear that the proposed RIO algorithms (RIO and HRIO) are able to more effectively find the global optima. We note that the Griewank function is highly-modal. Examining the results for the 10-dimensional Griewank function, shown in Table II, shows that all the algorithms performed comparably, with the HPSO having a slightly better result.

Lastly, the results of the Michaewicz function tests were especially interesting. Incredibly, the RIO and HRIO found an average global optima value of -1.9 in the 2-dimensional test, while the published global optima value is only -1.8013. However, upon further examination, the published global optima is within the search space $-\pi \leq x_i \leq \pi$ [15]. The RIO and HRIO explored beyond this region (recall that food is only located within the predefined search space) and found a lower optima value. The PSO and HPSO did not exhibit this behavior. While this result is an artifact of how we coded our algorithms — we did not constrain the agents to a predefined region, except upon initialization — it does exemplify the exploration properties of the RIO and HRIO.

TABLE II
MEAN OPTIMA VALUE FOUND BY ALGORITHMS FOR SOME TEST FUNCTIONS — OVER 11 RUNS, BOLD INDICATES BEST ALGORITHM FOR EACH TEST FUNCTION

Test Function	Dimensions	PSO	HPSO	RIO	HRIO	Actual Value
Sphere	2	**0**	$4.5 \cdot 10^{-10}$	**0**	$6.5 \cdot 10^{-23}$	0
Sphere	10	$\mathbf{4.5 \cdot 10^{-33}}$	$4.4 \cdot 10^{-7}$	$2.7 \cdot 10^{-31}$	$4.8 \cdot 10^{-14}$	0
Rastrigin	2	**0**	$1.2 \cdot 10^{-10}$	**0**	**0**	0
Rastrigin	10	12.4	**4.5**	8.3	11.6	0
Rosenbrock	2	$\mathbf{7.1 \cdot 10^{-28}}$	$4.6 \cdot 10^{-9}$	$9.2 \cdot 10^{-18}$	$4.8 \cdot 10^{-12}$	0
Rosenbrock	10	11.0	14.7	7.3	**3.8**	0
Ackley	2	$\mathbf{8.9 \cdot 10^{-16}}$	$1.6 \cdot 10^{-6}$	$\mathbf{8.9 \cdot 10^{-16}}$	$1.4 \cdot 10^{-11}$	0
Ackley	10	0.61	$\mathbf{8.2 \cdot 10^{-4}}$	1.6	2.6	0
Griewank	2	$2.0 \cdot 10^{-3}$	$1.3 \cdot 10^{-3}$	$\mathbf{1.3 \cdot 10^{-12}}$	$1.4 \cdot 10^{-5}$	0
Griewank	10	$1.0 \cdot 10^{-1}$	$\mathbf{8.2 \cdot 10^{-2}}$	$1.5 \cdot 10^{-1}$	$1.4 \cdot 10^{-1}$	0
Michalewicz	2	-1.8	-1.8	**-1.9**	**-1.9**	-1.8013
Michalewicz	10	-7.7	**-8.1**	-5.8	-6.1	-9.66015
Easom	2	**-1**	**-1**	**-1**	**-1**	-1
Hump	2	$\mathbf{4.7 \cdot 10^{-8}}$	$\mathbf{4.7 \cdot 10^{-8}}$	$\mathbf{4.7 \cdot 10^{-8}}$	$\mathbf{4.7 \cdot 10^{-8}}$	0
No. as good or better than PSO			7	9	7	

TABLE III
OPTIMA VALUES FOUND BY ALGORITHMS FOR 10-DIMENSIONAL ACKLEY FUNCTION — BOLD INDICATES ALGORITHM CONVERGED TO LOCAL OPTIMA, LAST ROW INDICATES NUMBER OF TESTS IN WHICH GLOBAL OPTIMA WAS FOUND

PSO	HPSO	RIO	HRIO
1.2	$1.3 \cdot 10^{-3}$	**16.4**	$1.9 \cdot 10^{-7}$
$4.4 \cdot 10^{-15}$	$0.5 \cdot 10^{-3}$	$8.0 \cdot 10^{-15}$	$1.7 \cdot 10^{-7}$
1.6	$0.6 \cdot 10^{-3}$	$4.4 \cdot 10^{-15}$	$2.4 \cdot 10^{-7}$
$4.4 \cdot 10^{-15}$	$1.2 \cdot 10^{-3}$	**1.2**	$3.9 \cdot 10^{-7}$
$8.0 \cdot 10^{-15}$	$0.4 \cdot 10^{-3}$	$4.4 \cdot 10^{-15}$	**20.1**
1.2	$1.3 \cdot 10^{-3}$	$1.5 \cdot 10^{-14}$	$3.5 \cdot 10^{-7}$
$8.0 \cdot 10^{-15}$	$0.3 \cdot 10^{-3}$	$4.4 \cdot 10^{-15}$	$4.0 \cdot 10^{-7}$
1.2	$0.5 \cdot 10^{-3}$	$4.4 \cdot 10^{-15}$	$2.6 \cdot 10^{-7}$
$8.0 \cdot 10^{-15}$	$0.9 \cdot 10^{-3}$	$4.4 \cdot 10^{-15}$	**4.7**
1.6	$0.4 \cdot 10^{-3}$	$8.0 \cdot 10^{-15}$	**3.3**
$4.4 \cdot 10^{-15}$	$1.2 \cdot 10^{-3}$	$4.4 \cdot 10^{-15}$	$4.1 \cdot 10^{-7}$
6	11	9	8

TABLE IV
OPTIMA VALUES FOUND BY ALGORITHMS FOR 2-DIMENSIONAL GRIEWANK FUNCTION — BOLD INDICATES ALGORITHM CONVERGED TO LOCAL OPTIMA, LAST ROW INDICATES NUMBER OF TESTS IN WHICH GLOBAL OPTIMA WAS FOUND

PSO	HPSO	RIO	HRIO
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	$1.2 \cdot 10^{-13}$
0	0	$3.5 \cdot 10^{-15}$	$1.6 \cdot 10^{-4}$
$\mathbf{7.4 \cdot 10^{-3}}$	$\mathbf{7.4 \cdot 10^{-3}}$	0	$1.2 \cdot 10^{-15}$
$\mathbf{7.4 \cdot 10^{-3}}$	$\mathbf{7.4 \cdot 10^{-3}}$	0	0
$\mathbf{7.4 \cdot 10^{-3}}$	0	0	0
0	0	$1.5 \cdot 10^{-12}$	0
8	9	11	11

Considering the results and analysis described in this section, we conclude that the HPSO, RIO, and HRIO are more effective for optimizing highly-modal functions. This result is intuitively pleasing as, in essence, the adaptations we propose encourage the optimization algorithms to explore the search space more vigorously than the standard PSO. Additionally, the HPSO, RIO, and HRIO perform comparably to the PSO for the 2-dimensional test functions.

V. CONCLUSIONS AND FUTURE WORK

The results presented in Tables II-IV clearly show that the addition of simple cockroach-inspired behaviors to the PSO have a positive effect on its ability to find global optima. This is quite an interesting result as we were not trying to develop a "better" PSO. Our aim was to examine the effect of adapting the PSO with the social behavior of cockroaches. Thus we concluded that the examples presented in this paper show that the social behavior of cockroaches is an effective model for algorithm development.

In the future we will further analyze the RIO and Hungry algorithms with regard to other non-standard PSO formulations, such as PSO with mass extinction [16], dissipative PSO [17], and PSO with near neighbor interactions [18]. These formulations of the PSO all have similar aspects of our RIO algorithm. PSO with mass extiction and dissipative PSO both have a randomizing feature similar to our *Find_Food* behavior that discourages the swarm from converging to local optima. The PSO with near neighbor interactions uses a similar neighborhood concept as our *Find_Friends* behavior. These algorithms are only a sample of the multitude of modified PSO formulations that exist; hence, our future work aims to investigate how RIO fits in with the wide spectrum of PSO research.

Additionally, we will further leverage the collective behavior of cockroaches to create algorithms that find global optima of multi-dimensional highly-modal functions. In this paper we adapted the PSO with a loose model of the social collective behavior of cockroaches, only using a small fraction of the available information on roaches (Table I). We are furthering this work by developing an optimization algorithm that is more intricately tied to the behaviors of the cockroach. This algorithm will most likely be far different from the PSO-based RIO that was presented in this paper. We

also hope to apply the cockroach model to other real-world problems such as robot goal-seeking and navigation.

APPENDIX

The D-dimensional test functions used in this paper are:

- Sphere (De Jong)

$$F(\vec{r}) = \sum_{i=1}^{D} (r_i^2 - 1), \tag{6}$$

$-50 \leq r_i \leq 50,\ i = 1, \ldots, D;$

- Rastrigin

$$F(\vec{r}) = 10D + \sum_{i=1}^{D} \left[r_i^3 - 10\cos(2\pi r_i) \right], \tag{7}$$

$-50 \leq r_i \leq 50,\ i = 1, \ldots, D;$

- Rosenbrock

$$F(\vec{r}) = \sum_{i=1}^{D-1} \left[(1 - r_i)^2 + 100(r_{i+1} - r_i^2)^2 \right], \tag{8}$$

$-50 \leq r_i \leq 50,\ i = 1, \ldots, D;$

- Ackley [19]

$$F(\vec{r}) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}(r_i^2)} \right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}[\cos(2\pi r_i)] \right), \tag{9}$$

$-50 \leq r_i \leq 50,\ i = 1, \ldots, D;$

- Griewangk

$$F(\vec{r}) = \sum_{i=1}^{D} \frac{r_i^2}{4000} - \prod_{i=1}^{D} \cos(r_i/\sqrt{i}) + 1, \tag{10}$$

$-600 \leq r_i \leq 600,\ i = 1, \ldots, D;$

- Michalewicz [15]

$$F(\vec{r}) = -\sum_{i=1}^{D} \sin(r_i)(\sin(ir_i^2/\pi))^{20}, \tag{11}$$

$-\pi \leq r_i \leq \pi,\ i = 1, \ldots, D.$

The strictly 2-dimensional test functions used are:

- Easom [20]

$$F(\vec{r}) = -\cos(r_1)\cos(r_2)e^{-(r_1-\pi)^2}e^{-(r_2-\pi)^2}, \tag{12}$$

$-100 \leq r_i \leq 100,\ i = 1, 2;$

- Hump [21]

$$F(\vec{r}) = A + 4r_1^2 - 2.1r_1^4 + r_1^6/3 + r_1r_2 - 4r_2^2 + 4r_2^4, \tag{13}$$

where $A = 1.0316285$, and $-50 \leq r_i \leq 50,\ i = 1, 2.$

Table V displays the location and value of the global minima for each of these test functions.

TABLE V
LOCATION AND VALUE OF GLOBAL MINIMA OF TEST FUNCTIONS

Function	Location	Value
Sphere	$\vec{r} = \{1, \ldots, 1\}$	0
Rastrigin	$\vec{r} = \{0, \ldots, 0\}$	0
Rosenbrock	$\vec{r} = \{1, \ldots, 1\}$	0
Ackley	$\vec{r} = \{0, \ldots, 0\}$	0
Griewangk	$\vec{r} = \{0, \ldots, 0\}$	0
Michalewicz	varies with D	varies with D
Easom	$\vec{r} = \{\pi, \pi\}$	-1
Hump	$\vec{r} = \{0.0898, -0.7126\}$	0
	$\vec{r} = \{-0.0898, 0.7126\}$	0

ACKNOWLEDGMENT

Spain and Salmon are supported in part by the University of Missouri College of Engineering Honors Undergraduate Research Program.

REFERENCES

[1] C. Zimmer, "From ants to people, an instinct to swarm," *The New York Times*, November 2007.

[2] P. Miller, "The genius of swarms," *National Geographic*, July 2007.

[3] G. Beni and J. Wang, "Swarm intelligence in cellular robotic systems," in *Proceedings of NATO Advanced Workshop on Robots and Biological Systems*, Tuscany, Italy, June 1989.

[4] R. Jeanson, C. Rivault, J. Deneubourg, S. Blancos, R. Fournier, C. Jost, and G. Theraulaz, "Self-organized aggregation in cockroaches," *Animal Behaviour*, vol. 69, pp. 169–180, 2005.

[5] J. Halloy et al., "Social integration of robots into groups of cockroaches to control self-organizined choices," *Science*, vol. 318, November 2007.

[6] J. Ame, J. Halloy, C. Rivault, C. Detrain, and J. Deneubourg, "Collegial decision making based on social amplification leads to optimal group formation," *Proc. Natl. Acad. Sci.*, vol. 103, no. 15, pp. 5835–5840, April 2006.

[7] S. Garnier et al., "Collective decision-making by a group of cockroach-like robots," in *Proceedings of 2005 Swarm Intelligence Symposium*, June 2005, pp. 233–240.

[8] J. Kennedy and R. Eberhardt, "Particle swarm optimization," in *Proceedings of the IEEE Int. Conf. on Neural Networks*, Piscataway, NJ, 1995, pp. 1942–1948.

[9] M. Clerc, *Particle Swarm Optimization*. Newport Beach, CA: ISTE USA, 2006.

[10] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," *IEEE Int. Conf. on Evolut. Comp.*, pp. 69–73, May 1998.

[11] R. Gruchalla et al., "The inner city asthma study: Relationships between sensitivity, exposure and morbidity," *J. of Allergy and Clinical Immunology*, vol. 115, pp. 584–591, March 2005.

[12] H. Watanabe and M. Mizunami, "Pavolv's cockroach: Classical conditioning of salivation in an insect," *PLoS ONE*, vol. 2, no. 6, p. e529, June 2007.

[13] *Using MATLAB*, The Mathworks, Natick, MA, November 2000.

[14] Y. Shang and Y. Qiu, "A note on the extended Rosenbrock function," *Evolutionary Computation*, vol. 14,

no. 1, pp. 119–126, April 2006.

[15] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Berlin, Heidelberg, New York: Springer-Verlag, 1992.

[16] X. Xie, W. Zhang, and Z. Yang, "Hybrid particle swarm optimizer with mass extinction," in *Proc. ICCCAS*, Chengdu, China, June 2002, pp. 1170–1173.

[17] ——, "A dissipative particle swarm optimization," in *Proc. Cong. on Evolut. Comp.*, Honolulu, HI, May 2002, pp. 1456–1461.

[18] K. Veeramachaneni, T. Peram, C. Mohan, and L. Osadciw, "Optimization using particle swarm with near neighbor interactions," in *Proc. Genetic and Evolut. Comp. Conf.*, Chicago, IL, July 2003, pp. 110–121.

[19] D. Ackley, *A Connectionist Machine for Genetic Hill-climbing*. Boston: Kluwer Academic Publishers, 1987.

[20] E. Easom, "A survey of global optimization techniques," Master's thesis, Univ. Louisville, Louisville, KY, 1990.

[21] L. Dixon and G. Szego, *Towards Global Optimization II*. New York: North Holland, 1978, ch. The optimization problem: An introduction.