Digital Enzymes: Agents of Reaction Inside Robotic Controllers for the Foraging Problem

Chad M. Byers, Betty H.C. Cheng, and Philip K. McKinley
BEACON Center for the Study of Evolution in Action
Department of Computer Science and Engineering
Michigan State University
East Lansing, Michigan, USA
{byerscha, chengb, mckinle3}@msu.edu

ABSTRACT

Over billions of years, natural selection has continued to select for a framework based on (1) parallelism and (2) cooperation across various levels of organization within organisms to drive their behaviors and responses. We present a design for a bottom-up, reactive controller where the agent's response emerges from many parallelized, enzymatic interactions (bottom-up) within the biologically-inspired process of signal transduction (reactive). We use enzymes to explore the potential for evolving simulated robot controllers for the central-place foraging problem. The properties of the robot and stimuli present in its environment are encoded in a digital format ("molecule") capable of being manipulated and altered through self-contained computational programs ("enzymes") executing in parallel inside each controller to produce the robot's foraging behavior. Evaluation of this design in unbounded worlds reveals evolved strategies employing one or more of the following complex behaviors: (1) swarming, (2) coordinated movement, (3) communication of concepts using a primitive language based on sound and color, (4) cooperation, and (5) division of labor.

Categories and Subject Descriptors

I.2.8 [**Computing Methodologies**]: Artificial Intelligence—*Problem Solving, Control Methods, and Search*

General Terms

Experimentation

Keywords

Artificial life, digital evolution, digital enzyme, digital signal transduction, robot controller, simulated robotics, evolution, foraging, cooperative behavior, self-organization, division-of-labor.

1. INTRODUCTION

For many, the term "foraging" evokes images of ants racing across the forest floor along hidden trails [19], or honeybees making vast treks across open spaces using only a fellow hive member's "shake" and their internal solar compass for navigation [18]. However, foraging likely established its roots in a more primitive, single-celled era, long before the appearance of multicellular organisms such as ants and bees. At a time predating the biocomplexity surrounding us today, single-celled organisms effectively foraged and survived solely through the cooperative behavior of their internal biomolecular "machinery" [5]. Molecular receptors on the cell's external surface provided inputs that triggered a series of biochemical reactions inside the cell and, in turn, the cell's response. As shown on the left-hand side of Figure 1, a receptor on the cell's extracellular surface binds to and detects the presence of a molecule. This molecule's presence might indicate to the cell a necessary resource in the direction of the receptor. Next, a cascading of events occurs within the cell, where secondary messengers receive the signal molecule, possibly alter its chemical structure, and produce an output that drives the cell's flagella, enabling it to move toward the valuable resource [13]. (The right-hand side of Figure 1 depicts our digital design of the "cell" that will be described in detail as the focus of this paper.)

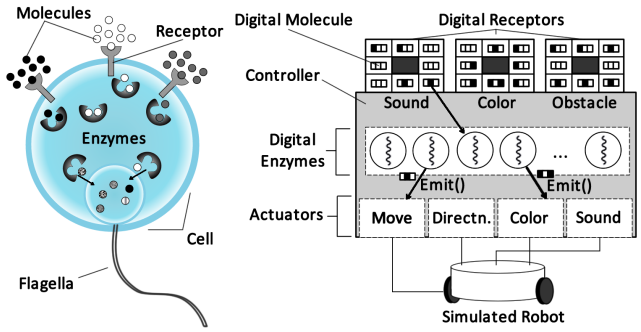

Figure 1: Biological signal transduction in a single-celled bacterium (left). Digital signal transduction within a robot controller based on digital enzymes (right).

Over time, cooperation and colonization allowed single-celled organisms to band together and may even have provided the first steps toward multicellularity [1]. As we fast forward to the present, the complex behaviors exhibited by ants, honeybees, hyenas, and even human beings still emerge from a vast number of interactions occurring across the cells in their bodies. Over billions of years, natural selection has *continued* to select for a framework of concurrently executing entities, whether it be the cells in a multicellular organism such as the honeybee or the enzymes and biomolecules within each of the honeybee's cells. These parallel executing en-

GECCO'11, July 12–16, 2011, Dublin, Ireland.

tities cooperate to provide the honeybee's response to its environment. Ongoing research has demonstrated that life's "program" arises not simply from static information encoded in a genome, but also from complex interactions orchestrated at the cellular level [14]. The expression of genes, production of enzymes, transfer of biomolecules, and regulation across metabolic pathways are the result of "agents of reaction" driving the cell's response.

In this paper, we explore parallel internal reactions of cooperation and the benefits this framework might provide for evolving systems to handle unknown environments such as those found in the *foraging problem*. The foraging problem is commonly defined as a group activity in which agents cooperate to gather a resource in the environment and return that resource to a specified location. Foraging is not required to be a group task, however, many strategies and behaviors in biology rely upon other members of the colony or group to aid in gathering the resource [18][19]. Although familiar biology examples have been discussed, many engineered applications also require solutions to the foraging problem. In deployed robotic systems, successful foraging behaviors benefit search-and-rescue missions where the resource is survivors being led to a location of known safety, reconnaissance missions where the resource is intelligence information being retrieved to a location of safe dissemination and transmission, and relief efforts for chemical spills where the resource is hazardous chemical waste being returned for containment.

This paper presents a design for a reaction-based, bottom-up controller inspired by the biological processes of *signal transduction* and *enzymatic catalysis*, to be used by simulated robots for the foraging problem. In biology, the combination of these two processes provides a cell the ability to react to both the (1) presence and (2) concentration of stimuli occurring in its external environment. We present a first step towards realizing these processes in a digital model, shown in the right-hand side of Figure 1, with the intent not only to gain insight into effective mechanisms for foraging behavior, but also to study why parallelism has continued to persist through natural selection.

The remainder of this paper is organized as follows: Section 2 describes the biological inspiration of this work and provides a working vocabulary of concepts and elements that will be captured in our digital system. We introduce the enzyme-based controller in Section 3. The experimental setup and results are presented in Sections 4 and 5. Section 6 describes related work. Finally, we draw conclusions and briefly describe future work in Section 7.

2. BIOLOGICAL INSPIRATION

The proposed approach is inspired by two cellular processes that occur in nature: (1) *signal transduction* - the process by which external stimulus molecules in the cell's environment lead to an overall response from the cell and (2) *enzymatic catalysis* - the process by which internal cellular entities (enzymes) exhibit a specificity towards molecules surrounding them, bind to those molecules, alter the molecules' chemical structures, and release the altered products to various parts of the cell [1].

2.1 Signal Transduction

Signal transduction refers to the process by which a molecule in the cell's external environment triggers a specific response, often through the use of intracellular, cascading biochemical reactions. As shown in Figure 2, signal transduction comprises three key steps: (1) sensing a stimulus molecule, (2) internal interactions, and (3) the generation of a response. Molecular detection components, referred to as receptors, are embedded on the extracellular surface of the cell exposed to the environment. Stimulus molecules in the cell's external environment can bind successfully to a receptor and trigger subsequent reactions within the cell via internal entities referred to as *secondary messengers*. In response to an environmental stimulus, a chemical "relay" using additional molecules can take place within the cell that may activate or deactivate the expression of certain genes, alter metabolic pathways, stimulate or suppress locomotion, and so on [1].

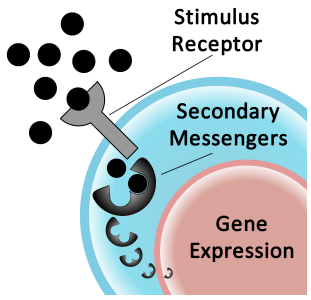

Figure 2: The three components of biological signal transduction: reception of signal, internal interactions among secondary messengers, and a response of the cell, for example, gene expression.

2.2 Enzymes

One of the actors often found in signal transduction pathways is the enzyme [7]. Biological enzymes are a specific class of proteins that catalyze the chemical reactions occuring within a specific cellular system [1]. As illustrated in Figure 3, molecules termed *substrate* act as input and bind to the enzyme forming an enzyme-substrate complex. The substrate molecules are then manipulated and altered by the enzyme to produce output molecules called products with properties generally differing from the substrate. As all enzymes are catalysts, the rate of chemical reaction and expression in the cell is controlled by both (1) the *concentration* of substrate molecules and (2) the *amount of enzymatic participation*.

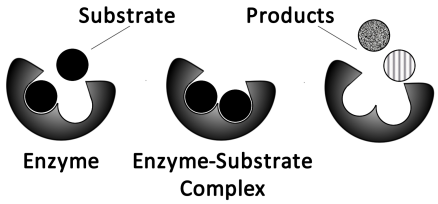

Figure 3: Biological enzyme binding to a molecule, altering its structure, and emitting the products (adapted from [10]).

3. SIMULATED ROBOT CONTROLLER

In this work, we have designed an evolutionary computation system to evolve robotic controllers that operate using *digital-based* signal transduction. In order to do so, we first had to develop the "material" (molecules) upon which the digital signal transduction process operated. Of course in the digital world, this material is not composed of chemical elements (or even a simulation thereof). Rather, digital molecules are simply bitstrings. In this study each molecule comprises three bits to encode the various properties of

the robot's world. The meaning of a particular bit set within a digital molecule arises solely out of the *context in which it is being interpreted*, as shown in Figure 4. This design allows the digital molecule to be a universal material that can be altered and exchanged throughout the different components of the system.

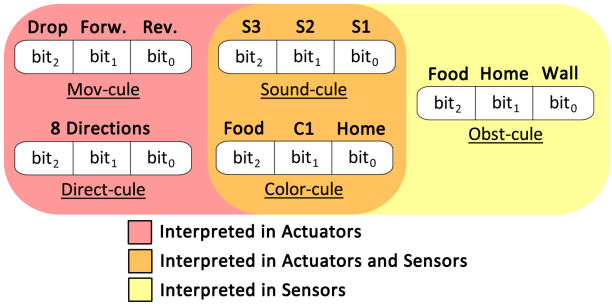

Figure 4: The mapping of bits within a digital molecule used to encode properties of the robot's environment and actuators

Digital-based signal transduction takes place within the controller for each of the robot's moves and comprises the following phases: (1) sensing stimuli in the robot's environment using **digital receptors**, (2) interactions within the robot by executing parallel, single-threaded processes termed **digital enzymes**, and (3) gathering the digital molecule "products," or votes, for each of the robot's actuators and translating those products into the robot's response.

3.1 Sensing The Environment

As previously shown in Figure 2, extracellular receptors on the surface of a biological cell can bind to and detect specific types of molecules in the vicinity. Similarly, a robot contains many different hardware sensors that reside on its external surfaces and are used to detect sound, color, and obstacles in the robot's vicinity. The analogue of encoding sounds, colors, and obstacles into a digital molecule form (i.e. sound-cules, color-cules, and obst-cules) inspired the design of the **digital stimulus receptor**.

The first phase of each robot's execution begins with the robot using its sensors, referred to as digital stimulus receptors throughout the remainder of this paper, to detect stimuli occurring in the robot's current surroundings. Within our two-dimensional world, stimuli can be sensed from eight compass directions. Therefore, each of the robot's digital stimulus receptors is an 8-element digital molecule array, as shown in Figure 5. An element within this array is analogous to a biological *binding site* storing one digital molecule, and bits set within this molecule correspond to stimuli that have been sensed. Stimuli occurring from the North are represented through the set bits in the digital molecule stored at index 0 and the indices increase clockwise to index 7 in the Northwest. Using the mapping of bits shown in Figure 4 and the digital obstacle receptor's array shown in Figure 5, we see that the robot is detecting food to the North and Northwest directions and Home to the South and Southeast directions. Using this information, the robot can determine that it is at a position between the two detected stimuli.

3.2 Parallel Interactions Within the Controller

The second phase of digital signal transduction is for the robot's response to emerge from the interactions of many parallel-executing **digital enzymes**. In this work, we implemented digital enzymes as single-threaded processes based on the Avida virtual hardware [12], although many other designs are possible. Figure 6 is used to guide

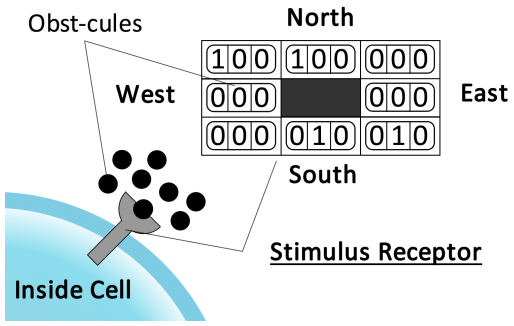

Figure 5: The digital stimulus receptor containing the accumulation of environmental stimulus bits in a digital molecule for each compass direction.

the discussion of each virtual hardware component found within a digital enzyme.

Each digital enzyme executes instructions from a circular sequence of instructions in a Turing-complete programming language. Two heads, or indices, are maintained by the digital enzyme for execution: an instruction pointer (IP head) indicates the current instruction to be executed and a flow head can be manipulated by instructions to jump the digital enzyme's execution to a different point within the program. Three "molecular" registers (AX, BX, and CX) exist within a digital enzyme; each register can store one digital molecule. Lastly, each digital enzyme contains two "molecular" stacks that can be used for additional storage of digital molecules pushed from one of the enzyme's registers.

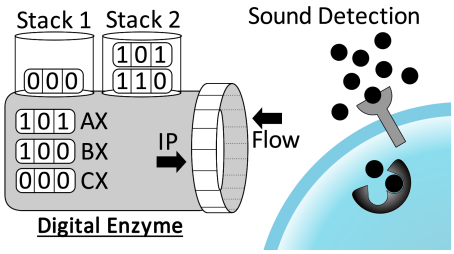

Figure 6: Digital Enzyme: a self-contained computing entity.

A digital enzyme is considered to be an "instantiation" of the circular sequence of instructions that it executes. For this system to undergo evolution, a genetic material must be exposed to the processes of mutation and inheritance. In this work, a controller contains three programs, one for each of its sensing capabilities: (1) obstacle, (2) color, and (3) sound. The instructions in these three programs act as the genetic material and are subject to insertion, deletion, and point mutations as well as being inherited by offspring.

The set of virtual computer instructions supported in this system is similar to the Avida [12] instruction set, including conditionals, stack operations, arithmetic operations, bitwise operations, and control flow operations. In addition, we introduced four new operations to interact with the "cellular" structures in digital signal transduction. A Retrieve operation can load a digital molecule into a register from (1) any of the digital receptor's array cells through indexing, (2) the "previous move" maintained for each of the robot's actuators, and (3) the current "majority vote" for an actuator's next behavior found in an actuator membrane discussed next in Section

3.3. An Emit operation enables the digital enzyme to send a digital molecule in one of its registers to a specified actuator where the bits set within the emitted molecule represent "votes" for the actuator's next behavior. An If_Food conditional enables the enzymes to detect whether the simulated robot is carrying food, and the Load_ID instruction encodes the robot's unique identifier into binary within a digital molecule in one of the registers. The Load_ID is specifically designed to provide an enzyme with a means to assess the robot's identity and potentially alter the robot's behavior.

3.3 Producing the Robot's Response

The last phase of both biological and digital signal transduction is determining the system response. The digital molecules in a robot's environment are manipulated by enzymes to produce molecules to control the robot's actuators. Each simulated robot contains an actuator for: (1) *direction* to face, (2) *movement* to make, (3) *sound* to emit, and (4) *color* to display.

An encapsulating "membrane" for each of the robot's actuators, called an **actuator membrane**, provides a model for aggregating the molecules produced by the digital enzyme reactions. This data structure within the controller represents a "persistent voting box," where the bits set within digital molecules stored in the actuator membrane represent "votes" for the actuator's next behavior. The term "persistent" refers to how the votes within each actuator's membrane are maintained across executions of the robotic controller during an evaluation. Once the user-defined size limit on the actuator's membrane has been reached, newly emitted molecules randomly replace existing ones within the membrane. There is no internal structure or ordering within an actuator membrane, intended to capture the Brownian motion often found within the cytoplasm of biological cells [16]. Figure 7 depicts the robot's color actuator and an enzyme emitting a digital molecule product with votes for either the Food color or Home color being displayed by the robot in its next move. An internal tally is maintained for each possible action of an actuator. When a molecule is emitted to the actuator, the bits set within this molecule increment their associated action's tally. If the actuator membrane's size limit has been reached, the bits set within the molecule being replaced cause their associated action's count to be decremented. Each actuator's response is decided by majority vote at the end of the controller's execution.

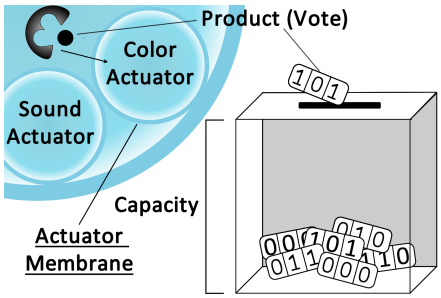

Figure 7: Actuator membrane structure aggregating "votes" from digital molecules for color actuator's next response.

Enzymes can produce multiple votes by either (1) setting more than one bit within an emitted molecule or (2) emitting more than one molecule to an actuator's membrane box. Although democracy is not enforced, the collection of enzymes within a robotic controller is the unit of selection during evolution, imposing a selection pressure on cooperation between enzymes to produce beneficial response behaviors for the robot.

3.4 Summary: Executing the Controller

During simulation, a robot is permitted a user-defined number of controller "cycles" where each cycle is one progression through the three phases previously described: (1) sensing of the robot's environment, (2) digital enzyme interactions, and (3) producing the robot's response.

In the **sensing** phase, the simulated robot uses its digital receptors to detect environmental stimuli occurring within detection range of the robot. Each stimulus detected by a receptor is (1) translated into the bit of a digital molecule based on the receptor's type (color, sound, or obstacle) and (2) associated to a compass direction using the angle between the position of the stimulus relative to the robot. The direction is used to index into the receptor's array of digital molecules and the stimulus's associated bit is set within the indexed molecule. At the completion of this phase, each stimulus receptor contains the digital molecule representations for all stimuli currently detected by the robot.

In the **digital enzyme interactions** phase, each digital molecule in the collection of receptors is "serviced" by *one* enzyme throughout the robot's evaluation. The type of receptor in which the digital molecule resides determines which program a digital enzyme executes from. For example, the digital molecule associated with stimuli occurring in the Northeast direction of the sound receptor will be serviced by a digital enzyme executing from the sound program. Since there are three digital receptors (color, sound, and obstacle) and each receptor stores one molecule for each of eight compass directions, a total of twenty-four enzymes are executing in parallel within a controller. Prior to the beginning of a cycle, the direction being serviced by an enzyme is encoded in a digital molecule and loaded into the enzyme's AX register while the digital molecule storing all detected stimuli from that direction is loaded into the BX register. The CX register and stacks persist across cycles of the controller to give the enzymes an additional means of storing information locally for subsequent cycles. No information persists across generations of evolution since only the three programs (and not their state) are "reproduced." The digital enzymes are located on a run queue within the controller and executed in by a fair scheduling algorithm until either (1) a halt command is executed by the enzyme, stopping it from further executing in this cycle or (2) a user-defined parameter for the maximum number of instructions to be executed is reached. At the end of a cycle, each enzyme's IP head and Flow head are reset to the beginning of the enzyme's program.

Finally, in the **response** phase, the bits set within the digital molecules emitted to the actuator membranes are counted. The majority bit vote according to the bit mapping in Figure 4 is used to decide the next behavior of that membrane's associated actuator, with ties broken randomly. A digital molecule with *none* if its bits set (zero) corresponds to a special vote of No Behavior by that actuator.

4. EXPERIMENTAL SETUP

Figure 8 depicts the problem and context for the experimental setup. The overall objective was to determine whether a design based on interactions amongst cooperating parallel processes within a controller was able to evolve foraging strategies. The experiments in this paper address the **central-place foraging** problem, where the goal for a homogeneous "colony" of six simulated robots is to find, return, and drop eight pieces of food to a 5x5 square of cells marked as "home" in the center of the world. Home

cells emitted a color called Home (red) while food emitted a color called Food (blue).

4.1 World Setup

A simple version of the problem was initially used as a baseline experiment to evaluate whether successful strategies could be evolved in a bounded world of size 26x26, where the outer cells formed a wall. The results from these preliminary experiments revealed that although successful foraging strategies did evolve, the selective pressure for cooperative group behaviors was weakened due to exploitation of the walls in search strategies that did not require coordination among robots. Hence, a second *unbounded* world setup is used as the basis for the results described in this paper. The unbounded world is more akin to biological foraging settings, where organisms that do not cooperate risk becoming lost and facing predation or starvation.

Digital "food" was placed at varying distances from the home region, indicating the difficulty level (and also the reward for) the foraging task. For example, using Figure 8, the genome being evaluated in this world has a difficulty level of six, indicating that the closest piece of food is six units from the edge of the home area. Two rules are used to place the remaining pieces of food: (1) no food can be placed closer than the specified difficulty level and (2) all food pieces must be in a contiguous "clump." A genome's difficulty level increased when the simulated robots returned all eight pieces and decreased if fewer than eight pieces were returned.

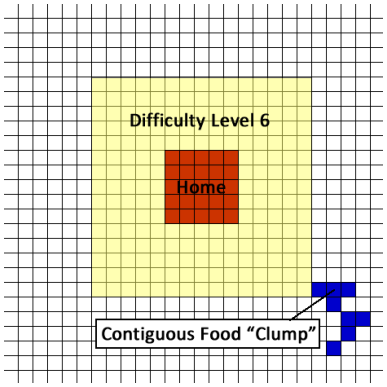

Figure 8: An unbounded world with a difficulty level of 6.

4.2 Robot Setup

In order to maintain consistency with the preliminary bounded experiment, each robot was allowed a maximum execution of 676 cycles, equivalent to traversing each open cell in a 26x26 bounded world. To avoid situations where the robots exploited their initial conditions, we randomly selected the positions (and orientations) of both the robots and food for each difficulty level. Since offspring inherit their parent's difficulty level during reproduction, an unfavorable random placement during a generation meant all individuals using an exploitative strategy did poorly.

During an evaluation, we used a fair scheduling algorithm to simulate equal operating speeds until either (1) the forage succeeded or (2) the maximum number of cycles per robot was exceeded. Each robot in simulation is able to sense its environment within specified ranges. Obstacle receptors extended to a distance of only 1 (adjacent to the robot), while sound and color receptors extended to a distance of 4. A robot's color actuator could emit the color of Home (red), Food (blue), or a neutral color (green). No other objects in the world aside from robots emit sound, providing one undisturbed medium to communicate information. The direction actuator could orient the robot in any of the eight compass directions; and the movement actuator either moved the robot forward, in reverse, or actively dropped a piece of food it was carrying. Food is automatically acquired when a robot drives over it, but once a robot is in possession of food, any subsequently encountered food becomes an obstacle that must be circumvented. The food is held until a Drop command is given by the controller, releasing the food in the cell the robot is facing. The Drop command provides no movement and therefore requires judicious use to forage effectively. If a piece of food is successfully dropped on a home cell, then it is tallied and removed from the world.

4.3 Evolution Setup

A population of 300 controller genomes evolved for 1000 generations using a tournament selection protocol where the genome with the highest fitness among 6 randomly selected genomes "reproduced" one offspring into the next generation's population. One generation of evolution entailed (1) evaluating each controller genome in a colony of six robots, (2) selecting a controller genome using the tournament selection protocol, (3) generating an offspring from the selected genome by exposing the offspring to insertion, deletion, and point mutations, and (4) placing the offspring into the next generation. Since each genome consisted of three genes (programs) evolving in concert with one another, mutation rates were tested at three different rates: 0.8%, 1.0%, and 1.2%. For each mutation rate, we also varied the maximum number of instructions an enzyme could execute during one cycle at 40 and 80 instructions for a total of 6 experimental combinations. Replicate runs for each experiment were seeded with 10 randomly generated integers for a total of 60 different runs.

4.4 Fitness

Fitness was assigned to each genome at the end of evaluation in the world based on the four components shown in Figure 9. The first component rewards for each item of food successfully returned with its "value" (genome's difficulty level) multiplied by the maximum number of cycles in a generation (676). Second, if the forage succeeded, then the remaining fraction of time was added to the genome's fitness to distinguish genomes of the same difficulty level, rewarding those that completed more quickly. The remaining terms account for how "close" the food was to the center of home when the evaluation ended. A dropped item of food generated the full reward while food still being carried generated half the reward value. Since food was undetectable while it was being carried, we rewarded strategies that increased the likelihood that 'lost' members with food could be detected in an effort to better smooth the fitness landscape.

5. RESULTS AND DISCUSSION

The dominant genome (that with the highest fitness) was saved and tested on 100 random food placements at increasing distances from home, until the colony failed to bring back any food for all 100 food placements at a particular distance. The average number of food items successfully foraged and returned to home for each treatment combination across the 10 runs is shown in Figure 10. At first glance, the evaluation results suggested that "good" foraging behaviors were not present in the dominant genomes, given that foraging success reduced to 50% when the radius from home reached 5 units. We suspect this drop occurs for one or more reasons: (1) higher difficulty levels increase the search area, requiring more execution cycles before more efficient strategies can be evolved, (2)

$$\boxed{fitness = Food_{foraged} + Speed_{success} + Food_{dropped} + Food_{carried}}$$

$$Food_{foraged} = num\_foraged \times (\max\_cycles \times difficulty)$$

$$Speed_{success} = 1.0 - (cycle_{finish} / \max\_cycles)$$

$$Food_{dropped} = \sum_{i=0}^{num\_remaining} \left((\max\_cycles - dist(food_i)) \times difficulty \right)$$

$$Food_{carried} = \sum_{i=0}^{num\_remaining} \left((\max\_cycles - dist(food_i)) \times difficulty \right) \times 0.5$$

Figure 9: The components of a genome's fitness determined by a homogeneous robot colony's forage.

more generations of evolution may be necessary, and (3) food extending (on average) 4 units further into the world raises the difficulty level of the forage if the first piece is moved. The sharp decrease at one radius unit outside of the robot's detection range may also signify that additional or alternate memory structures are necessary either within each enzyme or the controller itself to "remember" routes to food at further distances. More complex memory structures are a subject of our ongoing research. However, further analysis revealed several interesting behaviors, as described below.

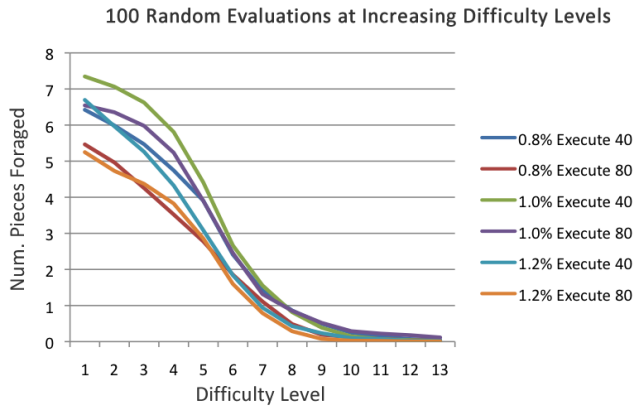

Figure 10: Average fitness of the dominant genomes evaluated using 100 random evaluations at increasing difficulty levels.

To better assess whether a foraging "strategy" was present in the genomes, we visually inspected 100 forage attempts made by each of the top 6 evolved genomes during the ecological period and found the results surprising. Each of the 6 genomes evolved a separate, *unique* strategy that included one or more of the following behaviors: (1) swarming, (2) coordinated movement, (3) communication of concepts using a primitive language based on sound and/or color, (4) cooperation, (5) division of labor, and behaviors termed as (6) "smart" behaviors. Examples of "smart" behaviors include extending the notion of "home" through color, increasing the detection range for food by dispersing part of the clump outward, and brief dashes off home in search of lost group members. Three of these behaviors are given a descriptive name and described below. In the following images, an arrow represents a robot and the direction it is facing. The color of the arrow indicates the current color the robot is emitting, while a circle around an arrow indicates the robot is emitting sound. Videos of these foraging strategies are available online at: http://www.cse.msu.edu/thinktank/foraging.

Foraging Strategy I.

In the first strategy, termed **"The Red-Armed Blinker,"** each robot continuously emitted Sound 1 and the Home color while the robot did not possess food, as shown in the top-left of Figure 11. The colony then exhibited "swarm-like" behavior by remaining within color detection range of one another and extending out into the world as a "red arm." Eventually, the red arm's reach enabled one robot to detect food. At this point, the robot that made the discovery moved toward the food and out of detection range from the retracting arm of robots. Upon acquiring a piece of food, the robot immediately changed its behavior to remain stationary and blinked rapidly back and forth between red and the neutral color (green) shown in the top-right of Figure 11. Over time, the red arm of tethered robots entered the detection range of the "blinker," discovered the individual, and detached completely from home (bottom-left). Although one individual was left continually roaming the perimeter making small dashes off home in search of others, its efforts were not necessary for this particular forage. The group progressively moved the remaining food items in the general direction of home and completed the forage.

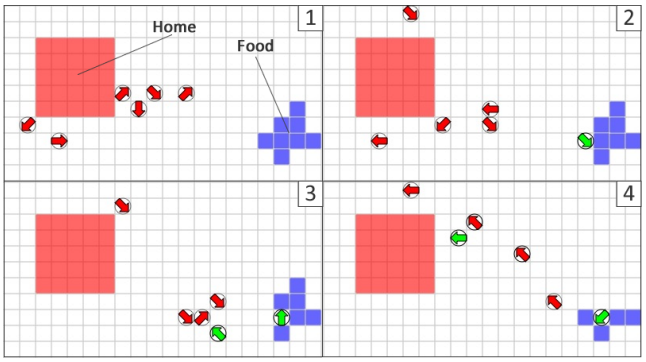

Figure 11: The "Red-Armed Blinker" Strategy

Foraging Strategy II.

A second evolved strategy was a variation of the first, termed **"The Green Arm, Blue Thumb."** Here, each robot constantly emitted the neutral color (green) until a robot discovered food, at which point it changed its behavior to emit the Food color (blue) and remained blue until the food was dropped, as shown on the left of Figure 12. In this strategy, rather than extending the notion of "home" by emitting the Home color, the robots instead extended the notion of "food." Similar to the previous approach, the robots swarmed in a "green arm" structure, extending out into the world from home. The "thumb" of the green arm soon discovered food and left the detection range of the remaining colony of robots. Surprisingly, the lone explorer began foraging by picking up different pieces of food and dispersing them to broaden the detection range of food for the green arm to discover. The green arm quickly detected the explorer emitting the Food color, detached completely from home, and progressively moved ("shoveled") each piece of food in the general direction of home. Similar to the previous strategy, *two* individuals remained roaming home's perimeter making quick dashes off home on "lookout" for the remainder of the group,

as shown on the right of Figure 12. These lookouts eventually detected the lost group members and led to the forage's success.

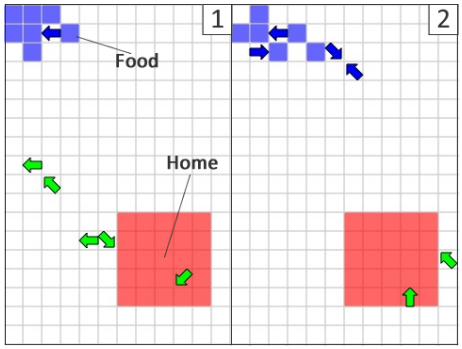

Figure 12: The "Green-Arm, Blue Thumb" Strategy

Foraging Strategy III .

The most coordinated of the six evolved strategies, termed **"The Sink,"** is depicted in Figure 13. Individuals aligned themselves in parallel and used the neutral color to signal the initiation of continual broadening and overlapping circular sweeps. These circular sweeps formed a "whirlpool" or "sink" pattern where the "drain" of the sink was slightly off-center from home. When a robot collected a piece of food, it would tighten its circle momentarily and then broaden its path, altering which "ring" relative to home it was on. As the robot's circular path crossed over home, shown on the right of Figure 13, it deposited the food and continued on its outward circular trajectory until all the food was gathered.

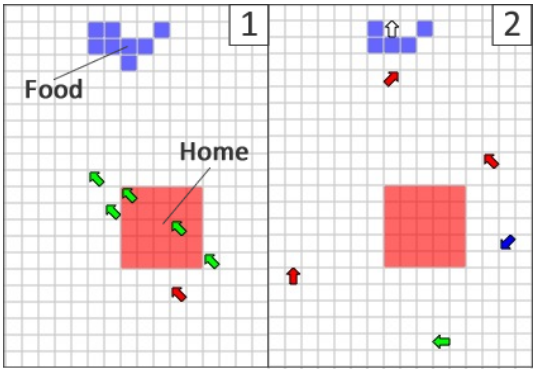

Figure 13: The "Sink" Strategy

In summary, these behaviors arose by providing evolution with only a universal material (digital molecules) and a way for reactions to take place through the cooperation of parallel-executing entities (digital enzymes) altering and working to produce a response. The interactions occurring within each controller were able to (1) *give meaning* to colors, sounds, and movements, (2) *transmit concepts* such as the location of food to colony members, (3) *maintain cooperation* within the controller and at a higher-level between individuals of a colony, and (4) *provide roles* to a subset of individuals within the colony. The shortest program (gene) contained within these three strategies consisted of 24 instructions while the longest program contained only 133 instructions. These results support the premise that internal, parallel entities may provide evolution with a means of storing genetic information in complex interactions.

6. RELATED WORK

Previous approaches to the foraging problem have used a range of techniques, such as genetic programming [2, 6, 20, 9], cellular automata and Finite State Machines (FSMs) [4, 17, 11] and extensions in P-Systems [3, 8, 15]. Our approach exploits the benefits of complex interactions among parallel controller entities that evolve concurrently. In contrast to previous state-based work using FSMs or cellular automata to study foraging, the entities in our system are dynamically executing programs containing their own virtual CPUs and using a Turing-complete language. Of those techniques previously listed, our approach is most similar to extensions within the domain of P-systems involving quorum [3] and X-machines [8]. Gheorghe et. al [8] has designed a computational model to *verify* honeybee foraging using a membrane-based architecture, where each agent contains a state-based machine, termed an X-machine, mapping sequences of input symbols to output symbols based on internal functions. Each agent's X-machine contained a predefined set of states and rules that specify how input symbols read in are transformed into output and the agent's next state. In our approach, *many* machines (processes) execute in parallel to interact and cooperate with one another to produce the agent's response. Agents do not have a predefined set of internal states but rather their behavior or "state" is constantly redefined by (1) the interactions between agents and (2) a changing internal environment. Basing our computational entities on virtual CPUs and hardware defined through *operations*, rather than rules, enables complex interactions and behavior to be explored using a minimal number of elements in the system's encoding. Moreover, we do not *design* an agent's program but instead take an open-ended approach to evolve solutions where cooperation *within* an individual agent's parallel machines and *among* individual agents of a colony work together to provide successful foraging strategies.

7. CONCLUSIONS

In this paper, we presented the design of a bottom-up, reactive controller based on digital models of signal transduction and enzymatic parallelism. Evaluation of our results revealed the successful evolution of complex behaviors including: (1) swarming, (2) assignment of meaning to color, sound, and movement, (3) communication of "concepts" via simulated actuators, (4) cooperation both within the entities of a controller as well as among the members of a colony, and (5) assignment of roles within a colony. These results reveal that "genetic information" may be captured in a "hidden" layer of parallel interactions, in addition to the instructions found within the genome. Future research will continue to explore the mechanisms and models *driving* the complexities in this work to (1) gain important biological insights and (2) better understand the role of bio-inspired parallelism in real-world systems.

8. ACKNOWLEDGMENTS

This work has been supported in part by NSF grants CCF-0541131, CNS-0751155, IIP-0700329, CCF-0750787, CCF-0820220, DBI-0939454, CNS-0854931, CNS-0915855 Army Research Office grant W911NF-08-1-0495, Ford Motor Company, and a Quality Fund Program grant from Michigan State University. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, U.S. Army, Ford, or other research sponsors.

9. REFERENCES

[1] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Molecular Biology of the Cell, Fourth Edition*. Garland Science, 4 edition, 2002.

[2] F. H. Bennett III. Automatic creation of an efficient multi-agent architecture using genetic programming with architecture-altering operations. In *Proceedings of the First Annual Conference on Genetic Programming*, pages 30–38. MIT Press, 1996.

[3] F. Bernardini, M. Gheorghe, and N. Krasnogor. Quorum sensing P systems. *Theoretical Computer Science*, 371:20–33, February 2007.

[4] A. Campo and M. Dorigo. Efficient multi-foraging in swarm robotics. In *Proceedings of the 9th European Conference on Advances in Artificial Life*, ECAL'07, pages 696–705, Lisbon, Portugal, 2007. Springer-Verlag.

[5] D. Christian. *Maps of Time: An Introduction to Big History (California World History Library, 2)*. University of California Press, February 2004.

[6] C. Di Chio and P. Di Chio. Group-foraging with particle swarms and genetic programming. In *Proceedings of the 10th European Conference on Genetic Programming*, EuroGP'07, pages 331–340, Valencia, Spain, 2007. Springer-Verlag.

[7] J. J. Falke, R. B. Bass, S. L. Butler, S. A. Chervitz, and M. A. Danielson. The two-component signaling pathway of bacterial chemotaxis: a molecular view of signal transduction by receptors, kinases, and adaptation enzymes. *Annual Review of Cell and Developmental Biology*, 13:457–512, 1997.

[8] M. Gheorghe, C. Martín-Vide, V. Mitrana, and M. J. Pérez Jiménez. An agent based approach of collective foraging. In *Proceedings of the Artificial and Natural Neural Networks 7th International Conference on Computational Methods in Neural Modeling - Volume 1*, IWANN'03, pages 638–645, Maó, Menorca, Spain, 2003. Springer-Verlag.

[9] J. R. Koza, J. Roughgarden, and J. P. Rice. Evolution of food-foraging strategies for the caribbean anolis lizard using genetic programming. *Adaptive Behavior*, 1:171–199, October 1992.

[10] D. Leja. *The National Human Genome Research Institute*, 2009. http://www.accessexcellence.org/RC/VL/GG/enzyme.php.

[11] W. Liu and A. F. T. Winfield. Modeling and optimization of adaptive foraging in swarm robotic systems. *International Journal of Robotics Research*, 29:1743–1760, December 2010.

[12] C. Ofria and C. O. Wilke. Avida: a software platform for research in computational evolutionary biology. *Artificial Life*, 10(2):191–229, March 2004.

[13] J. S. Parkinson. Signal transduction schemes of bacteria. *Cell*, 73(5):857–871, June 1993.

[14] E. Pennisi. Jumping genes hop into the evolutionary limelight. *Science*, 317(5840):894–895, 2007.

[15] G. Păun and F. J. Romero-Campero. Membrane computing as a modeling framework: cellular systems case studies. In *Proceedings of the Formal Methods for the Design of Computer, Communication, and Software Systems*, SFM'08, pages 168–214, Bertinoro, Italy, 2008. Springer-Verlag.

[16] P. G. Saffman and M. Delbrück. Brownian motion in biological membranes. *Proceedings of the National Academy of Sciences of the United States of America*, 72(8):3111–3113, Aug. 1975.

[17] M. Sipper. The evolution of parallel cellular machines: toward evolware. *Biosystems*, 42(1):29–43, March 1997.

[18] I. Steffan-Dewenter and A. Kuhn. Honeybee foraging in differentially structured landscapes. *Proceedings of the Royal Society of London - Biological Sciences*, 270:569–575, 2003.

[19] D. J. T. Sumpter and M. Beekman. From nonlinearity to optimality: pheromone trail foraging by ants. *Animal Behavior*, 66:273–280, 2003.

[20] J. A. Walker, K. Völk, S. L. Smith, and J. F. Miller. Parallel evolution using multi-chromosome cartesian genetic programming. *Genetic Programming and Evolvable Machines*, 10:417–445, December 2009.