Evolution of Resistance to Quorum Quenching in Digital Organisms

Benjamin E. Beckmann[**]
General Electric Global Research

David B. Knoester[†]
Michigan State University

Brian D. Connelly[†]
Michigan State University

Christopher M. Waters[‡]
Michigan State University

Philip K. McKinley[*,†]
Michigan State University

Abstract Quorum sensing (QS) is a collective behavior whereby actions of individuals depend on the density of the surrounding population. Bacteria use QS to trigger secretion of digestive enzymes, formation and destruction of biofilms, and, in the case of pathogenic organisms, expression of virulence factors that cause disease. Investigations of mechanisms that prevent or disrupt QS, referred to as *quorum quenching*, are of interest because they provide a new alternative to antibiotics for treating bacterial infections. Traditional antibiotics either kill bacteria or inhibit their growth, producing selective pressures that promote resistant strains. In contrast, quorum quenching and other so-called *anti-infective* strategies focus on altering behavior. In this article we evolve QS in populations of *digital* organisms, a type of self-replicating computer program, and investigate the effects of quorum quenching on these populations. Specifically, we injected the populations with mutant organisms that were impaired in selected ways to disrupt the QS process. The experimental results indicate that the rate at which these mutants are introduced into a population influences both the evolvability of QS and the persistence of an existing QS behavior. Surprisingly, we also observed resistance to quorum quenching: Effectively, populations evolved resistance by reaching quorum at lower cell densities than did the parent strain. Moreover, the level of resistance was highest when the rate of mutant introduction increased over time. These results show that digital organisms can serve as a model to study the evolution and disruption of QS, potentially informing wet-lab studies aimed at identifying targets for anti-infective development.

Keywords
Digital evolution, quorum sensing, quorum quenching, anti-infective, bacteria, resistance

A version of this paper with color figures is available online at http://dx.doi.org/10.1162/artl_a_00066. Subscription required.

1 Introduction

Cooperative behavior within and among living organisms is one of the most pervasive and important phenomena found on Earth. Even the simplest organisms, bacteria, exhibit collective behaviors that provide the foundation for more complex life forms. For example, many bacteria engage in *quorum sensing* (QS) [24, 56], a collective signaling behavior where actions of individuals depend on the density

* Contact author.
** General Electric Global Research, Niskayuna, New York. This research was conducted while he was a graduate student at Michigan State University.
† Department of Computer Science and Engineering, Michigan State University, 3115 Engineering Building, East Lansing, MI 48824-1027. E-mail: mckinley@cse.msu.edu (P.K.M.)
‡ Department of Microbiology and Molecular Genetics, Michigan State University, East Lansing, Michigan.

of the surrounding population. Bacteria use QS for a variety of purposes, including secretion of digestive enzymes in the gastrointestinal tract [10], bioluminescence and phototrophy in marine environments [8, 43], and, in the case of pathogenic bacteria, release of toxins or other virulence factors [15, 21, 39]. In addition, QS is closely related to other multicellular behaviors, such as the formation of *biofilms* [29], where communities of bacteria secrete, and are encased in, a protective extrapolymeric substance (EPS) [40]. Biofilms are an important element of natural food webs, but their shielding properties make them a serious problem in human health [14, 17, 37, 58].

One of the most pressing issues driving the study of QS is the evolution of antibiotic resistance. Traditional antibiotics either kill bacteria or inhibit their growth, producing selective pressures that promote resistant strains. Since the introduction of penicillin as an antibiotic during World War II, each successive deployment of a new antibiotic has been followed (in some cases, less than a year later) by the evolution of resistance to that antibiotic [13]. In an attempt to gain the advantage in this "arms race," the research community has started to explore a fundamentally different approach to treating bacterial infections. These strategies, referred to as *anti-infectives* [1, 13], attempt to modify virulent behavior without killing the bacteria. The philosophy behind anti-infective treatments is that if the bacteria can be manipulated so they no longer cause disease, the host will eventually be able to manage the infection. Moreover, these therapies would presumably exert less selective pressure on bacterial populations than antibiotics, thereby limiting the development of resistance. Since QS is essential for disease progression in many pathogenic bacteria, it is a potential target for the treatment of infections. Disrupting QS behavior, referred to as *quorum quenching* [11, 32, 45, 48, 50, 52, 61], has shown promise as an anti-infective strategy. For example, Rumbaugh et al. [52] demonstrated that a particular form of quorum quenching, discussed later, reduced the virulence of infections in mice and led to an decrease in mortality rate.

However, a key question about the longer term remains unanswered: Will bacteria evolve resistance to quorum quenching? While many evolutionary models suggest that anti-infectives should produce little resistance, researchers have also identified several ways that quorum quenching might indeed pose selective pressure on bacteria [18]; indeed, some argue that selection will always act to promote adaptations that confer increases in survival and growth. Unfortunately, testing these predictions using traditional microbiology methodologies is challenging. For example, to quantify the development of resistance to quorum quenching requires somehow isolating resistant organisms from the total population. However, unlike treatment with traditional antibiotics, in which only resistant mutants survive and are easily identified, both resistant and sensitive organisms continue to grow in the presence of quorum-quenching therapies.

In this article we investigate possible outcomes of quorum-quenching treatments through the evolution of *digital* organisms in the Avida system [46]. Digital organisms are a type of self-replicating computer program, subject to mutations and natural selection, that exist in a computational environment. In an earlier study [5], we demonstrated the evolution of QS behavior in Avida populations. Specifically, we showed that populations are capable of evolving a strategy to collectively suppress self-replication when the population density reaches an evolved threshold. In the study reported here, we injected Avida populations with mutant organisms whose communication capabilities were impaired in ways intended to disrupt the quorum-sensing process. Our results show that the rate at which these mutants were introduced into a population influenced both the evolvability of quorum sensing and the persistence of an existing quorum-sensing behavior. Most importantly, we also observed resistance to this form of quorum quenching: Effectively, populations evolved resistance by reaching quorum at lower cell densities than the parent strain. The level of resistance was highest when the rate of mutant introduction increased over time. Although other researchers have investigated QS in computational systems [9, 16, 41, 42, 51, 55, 62], to the authors' knowledge this is the first study to show the development of resistance to interventions intended to disrupt the QS process.

The remainder of this article is organized as follows. Section 2 provides background on quorum sensing in natural and artificial systems. Section 3 discusses the Avida digital evolution platform, including extensions introduced for this study. Section 4 reviews the evolution of QS in Avida and analyzes the genome of an evolved organism. Section 5 defines the mechanisms used to disrupt quorum sensing and presents the results of quorum-quenching experiments, including conditions leading to the evolution of resistance. Conclusions and possible future directions for this work are provided in Section 6.

2 Background

2.1 Quorum Sensing

Quorum sensing in bacteria was first reported in 1970 by Nealson et al. [44], dispelling a view held for 300 years that bacteria were asocial organisms. This discovery spawned a new branch of research to explore such interactions and assess the consequences of these behaviors [2, 21, 24, 54, 56]. QS has since been observed in many species of bacteria, which use it for a wide variety of purposes. Indeed, many key functions associated with bacteria (enzyme production, biofilm formation, and pathogenesis) have been linked to QS. To engage in QS, bacteria continually secrete and detect small-molecule signals called *autoinducers* (AIs) [44]. Under low-density conditions, the concentration of AI molecules in the environment is below the level of detection of the bacteria. However, as the cell density increases, so does the concentration of AI. When it exceeds a specific threshold, the bacteria detect the AIs and switch behaviors to a high-cell-density state, which in turn triggers the upregulation of a variety of genes, including those that encode enzymes that produce the AIs themselves. This transition floods the environment with AIs, creating a positive feedback loop that causes all the bacteria nearby to switch to the high-cell-density state. Hence, the bacteria alter gene expression in unison, changing the behavior of the entire population once a quorum has been reached.

QS is closely associated with disease and has been shown to be a key regulatory mechanism for virulence expression [21, 45]. Moreover, many pathogenic bacteria, including *Staphylococcus aureus*, *Streptococcus epidermidis*, and *Pseudomonas aeruginosa*, employ QS to construct biofilms at high density [19], adhering to host tissue as well as inanimate objects, such as catheters and other implanted medical devices [32]. Biofilm formation is a key virulence property of many bacteria, as it can render bacteria 1000 times less likely to be affected by antibiotics [17] and helps to protect the organisms against the host's immune system [17, 26]. Other bacteria use quorum sensing to *repress* biofilm formation. *Vibrio cholerae*, for instance, alternates between expression of virulence traits while colonizing the intestine of a host, and expression of traits such as biofilm formation that help it to survive in marine environments after leaving the host [28, 57, 63].

2.2 Quorum Quenching

Quorum quenching is the process whereby a quorum is disrupted by degradation of the AI molecules or inhibition of QS signaling. The intimate connection between QS and virulence in bacterial pathogens has led to extensive research on anti-infective quorum-quenching measures [11, 13, 32, 45, 48, 50, 52, 61]. One approach is to introduce into the population mutant strains incapable of producing AIs (*signal-negative*) or of detecting AIs (*signal-blind*) [52]. Here, the mutants typically act as *cheaters*, exploiting the cooperative production of virulence factors, but not fully participating in the underlying QS behavior. Recently, Rumbaugh et al. [52] studied the disruption of QS in mice infected with *P. aeruginosa* by introducing different types of mutants. The authors found that mice co-infected with wild-type bacteria and signal-negative or signal-blind mutants had lower mortality rates than those infected with the wild-type bacteria alone. Furthermore, it was shown that signal-blind mutants were more effective at reducing the mortality rate than signal-negative mutants, suggesting that different impairments disrupt QS activity to varying degrees.

Despite such advances related to QS *behavior*, less is understood about the *evolution* of QS in bacteria. While researchers have determined that cheater cells can naturally arise and become predominant members of the population [20, 53], it is impossible to study the evolution of QS *de novo* in a bacterial system, due to its complexity. Yet, understanding how these behaviors evolved is critical to revealing additional ways in which they can be modified, exploited, and disrupted in order to bring about desired changes in bacterial communities. Furthermore, such knowledge is essential to characterizing the development of resistance to different anti-infective strategies.

2.3 Computational Studies

To overcome the challenges of utilizing living organisms, researchers have gained insight into QS behavior by constructing models describing gene expression in QS bacteria. Several studies have used P systems [49] to abstract cell structure and operation [9, 51, 55]. These studies not only have increased our understanding of QS in biological organisms, but also provide a foundation for the development of novel, QS-based computational methods. In addition, Dockery and Keener [22] developed a model of QS in *P. aeruginosa* based on differential equations and used this model to help explain the molecular basis for switching between autoinduction states at different bacterial densities. More recently, Kambam et al. [31] used differential equations to model quorum-sensing behavior as an integral mechanism in a small two-species symbiotic ecosystem, enabling the authors to identify conditions necessary for steady state coexistence, and providing insight into potential molecular targets for bioengineering applications such as industrial fermentation.

Computational modeling has also been used to explore the *evolution* of collective behavior in bacteria. For example, Foster and colleagues [62] developed a computational model simulating biofilm development that approximates bacterial cells as small spheres. Using this model, they have demonstrated several key processes, including the selective advantage of EPS secretion [62], the roles of quorum sensing in biofilm formation and dispersal [42], and the effects of spatial structure on biofilm development [41]. In addition, Czárán and Hoekstra [16] developed a cellular automaton system and applied it to modeling the evolution of QS. That study is particularly intriguing, as a wide diversity of strategies evolved, depending on factors such as the cost of QS, the benefits of cooperation, and the amount of dispersal in the population. The study described herein complements the investigations described above by evolving QS behavior in digital organisms and then analyzing the responses when QS populations were exposed to different quorum-quenching treatments. These responses included the evolution of resistance to the treatments, where populations evolved to reach quorum at lower densities than they had previously.

3 Methods

The experiments were conducted using the Avida digital evolution platform [46]. Digital organisms in Avida replicate asynchronously and can interact with one another and with their environment. As in the natural world, whether a given organism's genes propagate to future generations depends on the organism's ability to replicate. However, Avida is not intended to simulate the physical and chemical processes found in natural organisms. Rather, experiments in the Avida "digital Petri dish" enable the researcher to distill complex interactions into an abstract form that permits direct observation and analysis, while still capturing key aspects of the evolutionary process [12, 25, 36, 47, 60].

3.1 Basic Operation

In Avida, individual organisms compete for space within a fixed-size two-dimensional collection of *cells*. Each cell can contain at most one organism, which comprises a circular list of instructions (its genome) and a virtual CPU that executes those instructions, as shown at the top of Figure 1. Instructions perform simple arithmetic operations (addition, bit shift, increment, and so on), control the execution flow, and enable organisms to signal one another and sense their environment. An organism executes instructions on its virtual CPU, which contains three general-purpose registers (AX, BX, CX), two general-purpose stacks, and special-purpose heads that point to locations within an organism's genome. Similar to a traditional program counter and stack pointer, heads are used to control the flow of execution. The execution of an instruction costs both virtual CPU cycles and energy. Different instructions can be assigned different virtual CPU cycle and energy costs.

Avida organisms are self-replicating, that is, their genomes must contain instructions to create offspring. An offspring is placed in a randomly selected cell, terminating any previous inhabitant. Typically, an Avida population starts with a single ancestral seed organism capable only of replication. As organisms replicate, instruction-level mutations produce variation within the population. In this study, the

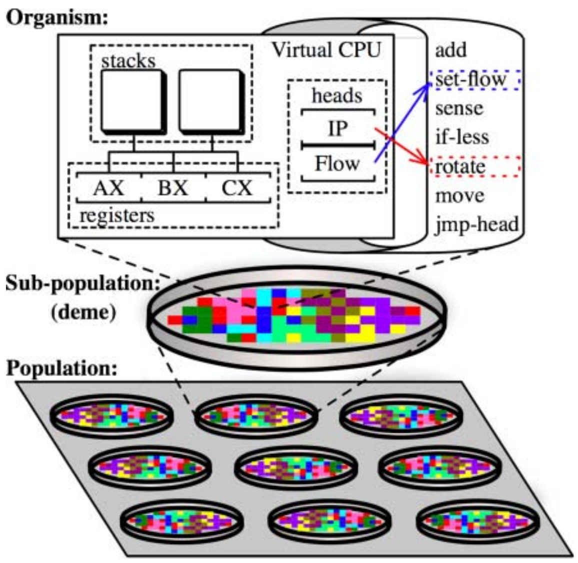


Figure 1. Population (bottom), subpopulation (middle), and composition of a digital organism: genome (top right), virtual CPU (top left) with heads pointing to locations within the genome [4].

ancestral organism contains 49 no-operation (nop-C) instructions and a single repro instruction, which performs self-replication. The nop instructions have no effect on the ancestral organism's observed behavior, or *phenotype*, excluding its gestation time. However, they do provide the evolutionary process with a "blank tape," as mutations during replication transform these nop instructions into other instructions through replacement, insertion, and deletion. Selective pressures, discussed below, favor sequences of instructions realizing behaviors that benefit the organism and its offspring. When conducting an Avida experiment, we execute several batches of experiments, each with a different configuration, and analyze the evolutionary process and resulting behaviors. A batch typically contains 20 replicate populations (runs), each of which starts with the same ancestral organism but a different random number seed, enabling the populations to follow different evolutionary paths.

In the version of Avida used in this study, each organism has an energy store, E_S. An organism's metabolic rate, M, is calculated as follows [3, 7]:

$$M = \frac{E_S}{I_{MAX}}, \tag{1}$$

where I_{MAX} is a user-defined limit on the total number of instructions an organism can execute before its energy is depleted, assuming no new energy influx and that all instructions cost 1 energy unit. An organism with a higher metabolic rate will be granted more virtual CPU cycles to execute instructions than an organism with a lower metabolic rate. However, an organism with a higher metabolic rate will also pay a higher energy cost to execute each instruction. The actual energy cost per instruction, E_C, is calculated as follows:

$$E_C = M \times I_C, \tag{2}$$

where I_C is the default energy cost per instruction. An organism can increase its energy store, either individually or as part of a group, by exhibiting behaviors prescribed by the user, as described later.

3.2 Messaging and Interrupt Handling

Avida organisms communicate by sending messages to one another. A message consists of a single packet containing the values from two of the sending organism's virtual CPU registers. The send-msg instruction delivers a message to an organism residing in the currently *faced* cell; each cell has eight neighboring cells. If the cell is unoccupied, then the message is dropped. An organism can change its facing by executing one of several rotate instructions.

In most prior Avida studies using messages [6, 33, 38], the receiving organism must explicitly *retrieve* the message from its input buffer in order to process it. However, for this study we extended Avida with an interrupt model similar to the execution model used in TinyOS [30], an operating system for wireless sensor nodes. In this model, an organism's main execution thread can be interrupted by an *event*, such as receiving a message. To enable the evolution of an interrupt handler, we introduced two instructions that denote the beginning (msg-handler) and end (end-handler) of an interrupt handler. We emphasize that these instructions have simply been added to the set of instructions available for mutation into an organism's genome. Whether they are used is solely a result of evolution.

Figure 2 depicts the semantics of these instructions within a genome. Assuming no messages have been received, when the sequential execution reaches a msg-handler instruction, the handler code is skipped, and execution jumps past the next end-handler instruction; if no end-handler exists in the genome, execution continues following the msg-handler instruction. However, when an organism receives a message, its genome is searched in the forward direction for the nearest instance of a msg-handler instruction. If none is found, the message is ignored. If a msg-handler instruction exists, then the organism's context is saved (including registers and heads), the contents of the message are placed in the organism's registers, and the instruction pointer is moved to the instruction following the msg-handler instruction. An interrupt handler's execution is terminated when the end-handler instruction is executed, at which time the original context is restored. If no end-handler instruction exists, execution continues sequentially through the genome in the interrupted state.

In this model, as in TinyOS, an interrupt handler cannot be preempted; therefore, all interrupts are handled atomically. Specifically, if a message is received while an organism is interrupted, the message is queued until the handler has finished, at which time the handler is reentered and the next message in the input buffer is processed. The interrupted context is restored only when all received messages have been processed. In this study, the input buffer is limited to 20 messages. Messages received when the buffer is full are dropped.

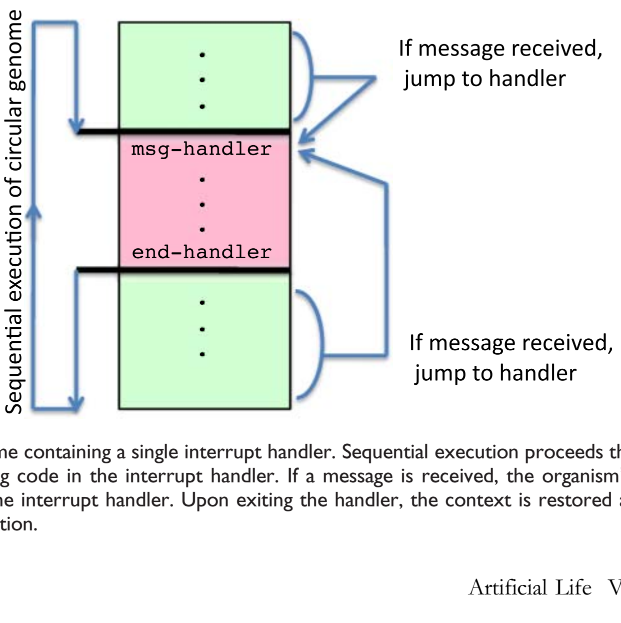

Sequential execution of circular genome
msg-handler
end-handler
If message received, jump to handler
If message received, jump to handler

Figure 2. Generic genome containing a single interrupt handler. Sequential execution proceeds through the genome in a circular fashion, skipping code in the interrupt handler. If a message is received, the organism's context is saved and execution jumps into the interrupt handler. Upon exiting the handler, the context is restored and execution resumes at the point of interruption.

3.3 Demes and Group-Level Selection

In some Avida studies, all organisms are treated as part of a single population, in which case individual organisms compete against each other for space. However, in other studies, especially those involving the evolution of cooperative behavior, it is useful to subdivide a population and have groups of organisms compete [33–35]. As shown at the bottom of Figure 1, a population of organisms can be partitioned into multiple independent subpopulations, referred to as *demes*. All demes have identical environments and initial configurations, and typically an organism can interact only with other organisms in its deme. Subdividing the population this way is akin to the island model [59] commonly used in evolutionary computation, and enables the detection and selection of demes that perform group-level behaviors. Demes are periodically competed against one another to determine which demes move to the next competition period. After a deme is selected for replication, mutations are applied to the genome that was used to seed that deme. The newly created genome is then used to seed the offspring deme. A sequence of seed genomes is referred to as a *germline* [35]. At the beginning of a run, all demes are seeded with the same ancestral organism. As genomes are mutated and used to seed other demes in subsequent competition periods, a tree of germlines is produced. In this work, individual organisms within a deme are able to self-replicate; however, those replications do not involve mutations to the genome. A consequence of this process is that all organisms within a deme are genetically identical. This approach has been theoretically [27] and experimentally [23] shown to be effective in evolving cooperative behavior.

3.4 Experimental Setup

The key instructions used in this study are listed in Table 1. The instruction set includes send-msg, msg-handler, and end-handler, whose functionality was discussed earlier. Various types of rotate instructions were tested [5], but the most effective were the single-step rotations, rotate-right and rotate-left, which enable an organism to rotate one cell to its right or left, respectively. Four different nop instructions were included (nop-A, nop-B, nop-C, and nop-X). Sequences of nops in the genome form *labels* that modify instruction behavior (for example, which registers are used in an operation) as well as affect control flow by identifying the destinations of jump instructions. Additional details of labels

Table 1. Instructions used in study.

Instruction	Description
send-msg	Send message to faced neighbor
msg-handler	Delimit beginning of an interrupt handler
end-handler	Delimit end of an interrupt handler
rotate-left	Rotate organism facing to left
rotate-right	Rotate organism facing to right
nop-A	Nop that can be used as part of a label
nop-B	Nop that can be used as part of a label
nop-C	Nop that can be used as part of a label
nop-X	Nop that cannot be used as part of a label
repro	Replicate organism

and nop-modifiable instructions are described elsewhere [46]. Finally, the repro instruction is used for replication. It should be noted that this instruction set contains fewer instructions than in many prior Avida studies [36]. While other instructions could be included in the set of instructions available for mutation, our results show this set is sufficient for evolution of QS-like behavior.

Populations were divided into 400 demes, each with a 5 × 5 torus topology. Each deme was seeded with a single organism at the beginning of each competition period. A competition period lasted for 20 *updates*, a unit of time in Avida where the average organism in the entire population will probabilistically execute 50 instructions per update. Following deme competitions and prior to reseeding, mutations to the germlines were applied as follows: Each instruction in the genome was subject to a 0.75% chance of being mutated to a random instruction, plus there was a 5% chance for a random instruction to be inserted at (or deleted from) a random location in the genome. When seeding a new deme, the seed organism was placed in a random cell and rotated to a random facing, so that its initial position and facing could not be "learned" through the evolutionary process. In addition, deme-level populations were *well mixed*, meaning that during replication, the offspring was placed in a random cell within a deme, with a preference for empty cells.

In this study populations were subject to a deme-level pressure to conserve energy. Specifically, in deme competitions, demes that conserved more energy were probabilistically more likely to be selected for participation in the next competition period (fitness-proportional selection). At the beginning of a competition period, each deme was provided with a fixed amount of energy. Initially, all the energy of a deme belonged to the single seed organism, but every replication split the energy of the parent evenly with its offspring. At the end of the competition period, each deme's fitness was evaluated using

$$fitness_i = \begin{cases} 1 & \text{if } i \text{ is sterile,} \\ \frac{RemaingingEnergy_i}{InitialEnergy_i} + 1 & \text{otherwise:} \end{cases} \tag{3}$$

a deme's fitness value was proportional to the amount of energy it conserved. To avoid evolving sterile demes (demes within which no organisms are born, for example, on account of evolving away the repro instruction), at least one birth was required for a deme to achieve a fitness above 1.

We note that there are only two ways for a deme to lose energy: First, the organisms in the deme burn energy as they execute individual instructions. Second, the energy remaining in an organism is purged when that organism is replaced by a newly replicated organism. In this study, all instructions had the same energy cost, so different instruction execution sequences of the same length all had the same explicit energy cost. Therefore, the only way organisms in a deme could conserve energy, without becoming sterile, was to evolve some other means to limit self-replication.

4 Evolution of Quorum Sensing

Using the system described above, we conducted a series of experiments that demonstrated the evolution of QS in Avida populations; effectively, organisms used QS to limit replication and thereby conserve energy. We executed 20 runs, each lasting for 2500 deme competition periods (generations). The amount of energy conserved by a deme determined its fitness, as defined in Equation 3. In this section, we review the results of those experiments, including analysis of an evolved genome; additional details can be found in [5]. Of the 20 runs, 15 exhibited energy conservation, so the discussion focuses on those populations. The most abundant, or *dominant*, genomes from these 15 populations were then used as seed organisms for the quorum-quenching experiments described in Section 5.

4.1 Organism Density

As described earlier, QS behavior in bacteria exhibits two key features. First, a density-based change in behavior can be observed, and second, after a density threshold has been reached, a positive feedback loop is created that causes more AI to be emitted by the organisms. In Avida, the organism density is the number of organisms per cell; therefore, a density of 1.0 can be achieved only if every cell in the environment contains an organism. The dominant genome was extracted from each run at its conclusion and used to seed 400 demes, which were then executed for one competition period. During execution, we recorded for each deme the organism density, total births, and number of organisms running in an interrupted state. Figure 3a plots the organism density and total births per deme, averaged over all dominant genomes. As shown, both the organism density and total births per deme reached a plateau after update 4. We conclude that replication, which was present before update 4, was suppressed when the organism density reached approximately 0.8 organisms per cell, indicating a quorum had been reached.

Figure 3b plots the average number of organisms per deme that were executing in an interrupt handler during the competition period. These data closely mirror Figure 3a, providing an indication that interrupts were being used to suppress organism self-replication. In addition, when we disabled messaging so that organisms could no longer be interrupted, the average organism density within a deme quickly increased to 1.0, and all demes died out due to energy depletion (data not shown). This behavior provided further evidence that interrupt-causing messages were an important feature of the evolved genomes. Next, let us focus on the genome of an organism that realized this behavior.

4.2 Genome Analysis

Figure 4 depicts the genome of the dominant organism from the Avida run that produced the lowest average organism density, 0.67. In this evolved genome, an interrupt handler happens to wrap around the bottom and top of the genome, as shown. Hence, when a new organism begins execution at the top of the genome, it is in the interrupt handler code, but is not actually serving an interrupt. According to the semantics defined earlier, execution simply falls through to the next instruction, which happens to be a redundant end-handler instruction. (This redundancy is itself interesting; if one end-handler instruction were lost to mutation, the other would ensure that interrupts are still handled.) Thus, in the absence of messages, the organism will execute the first 16 instructions in the genome, with repro being the last instruction executed. (We note the presence of a redundant repro instruction as well.) During this sequence, the organism sends a message to its initially faced cell, then to the cell two rotations to its left, and then to the cell one rotation to its left. Finally, the organism replicates. In short, the organism will send 3 messages for every 16 executed instructions during normal execution and then replicate. Upon replication, the organism's state is reset, causing the genome to be executed from the beginning.

The above sequence will be repeated until the organism either runs out of energy, is replaced by the offspring of another organism, or is interrupted due to arrival of a message. If interrupted, the current context of the organism is saved, and the interrupt-causing message is processed in the inter-

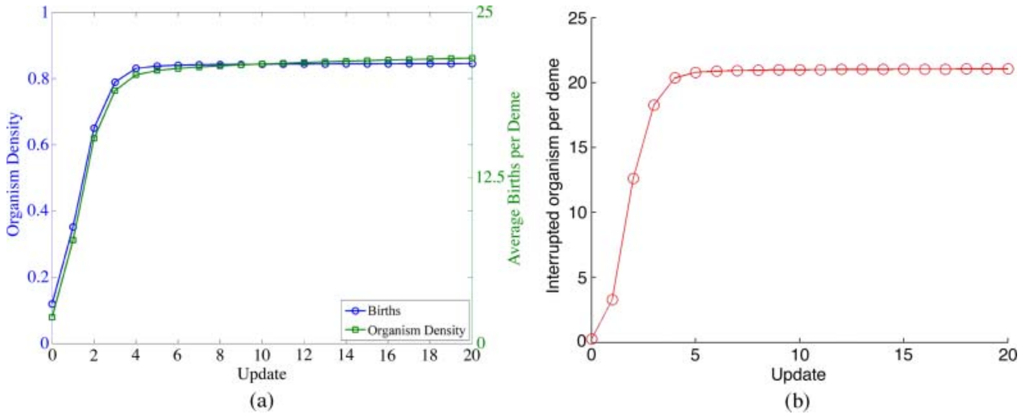
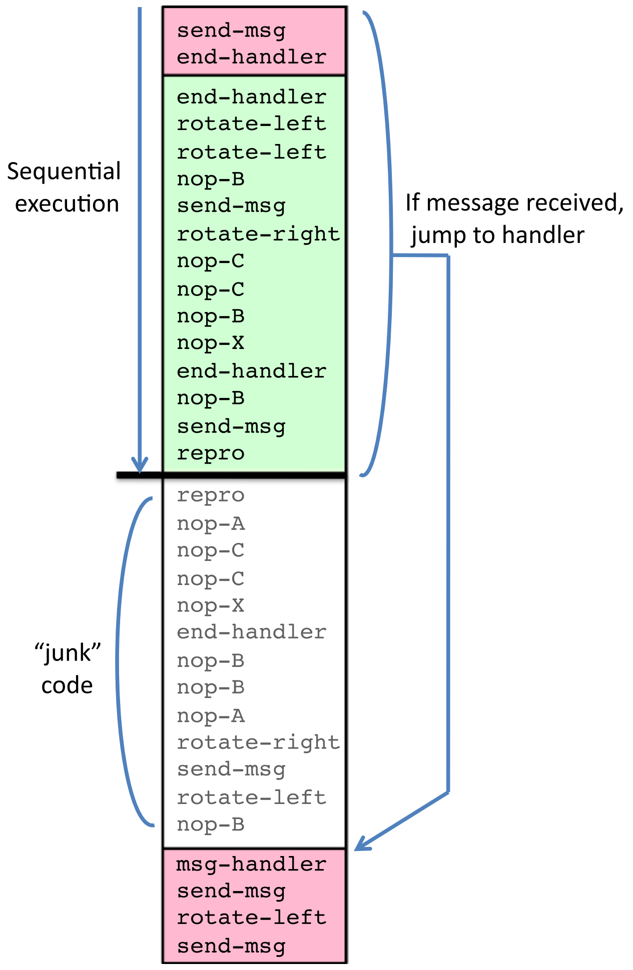


Figure 3. Characteristics of dominant organisms when executed for one competition period after evolution: (a) organism density and total births per deme; (b) mean number of organisms interrupted per deme [5].



```
send-msg
end-handler
end-handler
rotate-left
rotate-left
nop-B
send-msg
rotate-right
nop-C
nop-C
nop-B
nop-X
end-handler
nop-B
send-msg
repro
repro
nop-A
nop-C
nop-C
nop-X
end-handler
nop-B
nop-B
nop-A
rotate-right
send-msg
rotate-left
nop-B
msg-handler
send-msg
rotate-left
send-msg
```

Figure 4. Dominant genome from a population that produced the lowest average organism density.

rupt handler. While interrupted, the organism will send one message to the cell it currently faces and two messages to the cell left of its initial facing. During this time, the organism might receive additional messages. The organism will remain in the interrupt handler, and therefore not replicate, until all received messages have been processed. Effectively, for every entrance into the interrupt handler due to receipt of one message, the organism will produce three messages and delay sequential execution by the time to execute the five instructions following the msg-handler instruction. An organism that receives 1 message during its sequential execution will send a total of 6 messages (3 during sequential execution and 3 at interrupt level). An organism that receives 2 messages during its sequential execution will send 9 messages, and so on. In turn, the messages sent by the organism are likely to interrupt neighboring organisms (which execute the same genome), causing them to send more messages, some of which will be directed back to the original organism.

Hence, execution of the interrupt handler produces a positive feedback loop reminiscent of QS in bacteria, where the level of messaging (AI) in the system is increased, in turn increasing the chances than an organism will become interrupted. Therefore, the expression of this individual behavior in a dense group of digital organisms will cause an organism to remain in a state of perpetual interruption and never self-replicate. The organisms continue to burn energy while executing in their interrupt handlers, but no energy is lost due to organism replacement. Although this behavior might appear unproductive from a computing perspective, we note that QS in bacteria is often irreversible, in that once the bacteria reach quorum, the high-cell-density state is maintained until an external mechanism intervenes to change the organism density (for example, being flushed from a host organism).

4.3 Experiments with Larger Demes

To investigate whether the behavior scales, we seeded demes many times larger than the original 5 × 5 demes in which the behavior evolved. For each deme size, we conducted 15 different runs, one for each of the dominant organisms from the 15 (out of 20) earlier runs that evolved population control behavior. Each deme was allowed to execute for a single competition period. Figure 5 plots the average organism density for different-sized demes, over the duration of the competition period. Although it necessarily takes longer for the larger demes to populate, the final average densities all plateau at just over 80%. We conclude that the quorum-sensing behavior that evolved in 5 × 5 demes scales to the larger populations, reaching quorum at similar organism densities.

Finally, we seeded a single 100 × 100 deme with an organism containing the genome in Figure 4. We allowed the deme to execute for one competition period as we tracked the constituent organisms' execution behavior. Figure 6 shows snapshots of the resulting behavior, where each (*x*, *y*) corresponds to a single cell in the deme. Each cell is colored according to the current activity within that cell. If a cell does not contain an organism, it is colored white. A cell that contains an uninterrupted organism is colored black, and a red (shaded if printed in gray scale) cell denotes an organism in an interrupted state. As depicted in Figure 6, the population quickly switches behaviors as density increases. Indeed, the time difference between Figures 6(c) and 6(e) is less than two updates. The rapid change in behavior is a hallmark of QS and has been observed in natural and, now, digital organisms. Having evolved QS behavior in Avida and with a collection of QS populations in hand, we were now ready to explore the effect of quorum-quenching strategies on these populations.

5 Quorum Quenching

As described in Section 2, methods of disrupting quorum-sensing behavior are of considerable interest as possible treatments for bacterial infections. One approach is to inject the QS population with mutant organisms that are incapable of either producing (*signal-negative*) or detecting (*signal-blind*) AI molecules [52]. In this section we describe experiments to determine whether similar methods are effective against the QS behavior that evolved in Avida populations. Specifically, we observed how digital organisms fare in environments where offspring are probabilistically impaired at birth so they cannot send or receive messages, which organisms treat analogously to AI molecules.

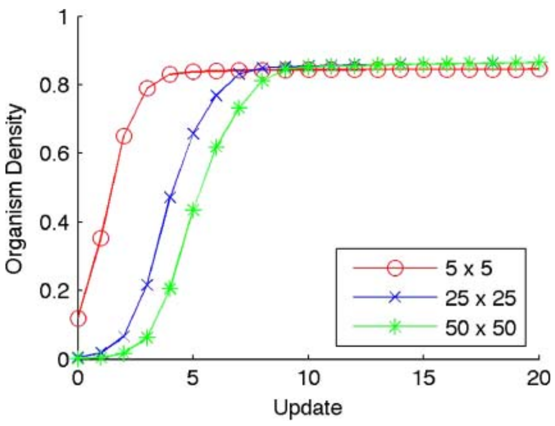
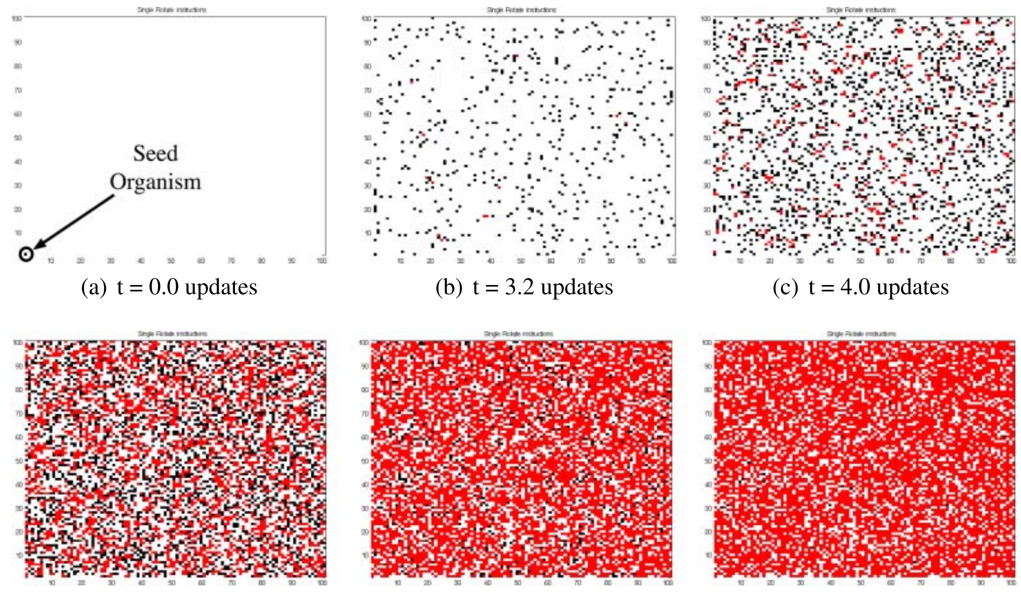


Figure 5. Average organism density under multiple deme sizes [5].



(a) t = 0.0 updates (b) t = 3.2 updates (c) t = 4.0 updates

(d) t = 4.8 updates (e) t = 5.6 updates (f) t = 8.0 updates

Figure 6. Snapshots of a 100 × 100 deme initially seeded with the genome shown in Figure 4: (a) the initial state of the deme with a single uninterrupted organism in the lowest, leftmost cell; (b,c) steady exponential increase in population size where the majority of organisms are executing sequentially; (d,e) rapid behavioral change from self-replication to suppression of self-replication; (f) state of the deme population 40% of the way through a competition period [5].

In all experiments described, the populations were again divided into 400 5 × 5 toroidal demes. Each deme was seeded with an organism at the beginning of each competition period, which lasted for 20 updates. At the end of a competition period, each deme's fitness was evaluated using Equation 3. The genomes used for the initial seed organisms varied by treatment, and will be described below.

5.1 Evolving QS in the Presence of Mutants

First, we investigated whether Avidians can evolve QS behavior despite the presence of mutants in the population. In this treatment, mutants were introduced at five different rates: 0, 1, 2, 5, or 10 percent of all births. Signal-negative (send-impaired) mutants were constructed by disabling the send-msg instruction. Signal-blind (receive-impaired) mutants were constructed by disabling message receive interrupt handlers, causing all messages to remain buffered. We conducted a set of experiments for each of the 25 resulting pairs of send and receive mutant introduction rates. Once a mutant was introduced, it remained a mutant for its lifetime, and its impaired abilities were inherited by all its descendants. As in the QS experiments described in Section 4, the default ancestral organism contained 49 nop-C instructions followed by a single repro. A copy of this organism was used to seed each deme, in each of 20 runs, for each of the 25 mutant rate pairs. Each run was also seeded with a different random number and allowed to evolve for 5000 generations (competition periods). Compared to the QS experiments in Section 4, the additional 2500 generations facilitated later comparisons between treatments that were seeded with the default ancestral organism and those seeded with the dominant organisms produced by the QS experiments.

Figure 7 plots the fraction of runs for each mutant rate pair that evolved QS behaviors. From this figure, we observe that the introduction of receive-impaired mutants is more effective at preventing QS than that of send-impaired mutants. As the fraction of receive-impaired mutants increases, the number of runs that evolve QS behavior approaches zero. This result suggests that an environment with a high concentration of mutants incapable of receiving messages produces a barrier to the evolution of QS. Apparently, to evolve QS the fraction of organisms unable to receive messages must remain small, less than 5%.

A possible explanation for the disparity between treatments with send- and receive-impaired mutants is that sending a message is less costly than receiving one, considering the time needed to execute interrupt handler code. Moreover, if the evolved behavior is similar to that of the organism in Figure 4, the number of messages generated is greater than the number that can be processed. Therefore, replacing some percentage of organisms with send-impaired mutants would likely produce less effect on the overall process than would the introduction of receive-impaired mutants. To further examine this issue, we next tested the resistance of organisms that had *already* evolved QS, to the introduction of mutants.

5.2 Resistance in Quorum-Sensing Organisms

In this treatment, we extended the 20 runs described in Section 4 from 2500 deme generations to 5000, allowing evolution to further optimize the genomes in the absence of artificially introduced mutants. We observed during this extended trial that 16 runs evolved QS, one more than with 2500 generation. We then tested the robustness of the 16 dominant genomes to the introduction of mutants. We note that the energy consumed by organisms with any of these 16 genomes will eventually deplete the available energy of the deme containing that organism, causing deme death, unless the birth rate is limited and many organisms are locked in an interrupted state. As noted earlier, although interrupted organisms continue to execute and consume energy, the consumption rate is far less than for populations in which organisms are replicating (and others are being replaced). Using the same mutant introduction rates as before, we subjected each of these 16 dominant genomes to all 25 pairs of mutant introduction rates for one competition period. In each of the 16 tests, we injected into all 400 demes a copy of a seed organism containing the selected dominant genome. We then measured the deme survival rate, that is, the fraction of demes that contain living organisms after one competition period. This metric provides an indication of how successful the organisms were at using QS to conserve energy.

Figure 8 shows the mean fraction of demes surviving at the conclusion of a competition period, interpolated over the range of 0 to 10 mutants per 100 births for both mutant types. As expected, when mutants were not present (the conditions in Section 4), all of the demes contained living organisms at the conclusion of the experiment, denoted by point *A* in Figure 8. Moreover, the introduction of only send-impaired mutants had minimal effect on the number of demes that survived for a single competition period. Across the 16 runs, only 549 out of 6400 demes (16 runs × 400 demes each) succumbed to the highest introduction rate of send-impaired mutants (10%) and the lowest introduction rate of receive-impaired mutants (0%), denoted by point *B* in Figure 8. However, demes that were subjected to the introduction of receive-impaired mutants exhibited a dramatic increase in deme mortality. At an

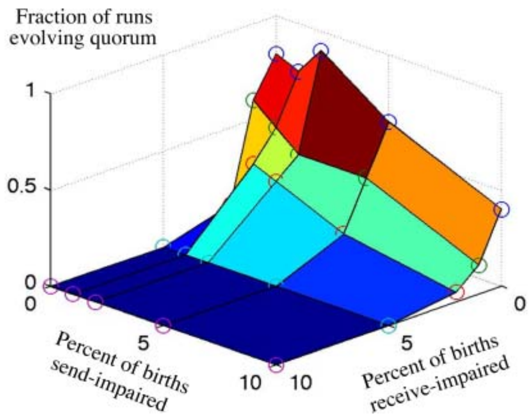


Figure 7. Fraction of runs that evolved quorum-sensing behavior under constant mutant introduction rates. Results are the mean of 20 runs sampled at each of the 25 configurations, marked by a circle.

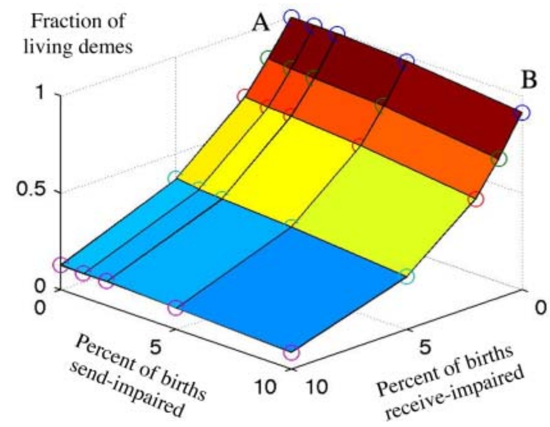


Figure 8. Fraction of living demes exposed to mutants after a competition period. Results are the mean of 16 runs, one for each dominant exhibiting QS, sampled at each of the 25 configurations, marked by a circle.

introduction rate of 10%, such mutants killed 88.4% of all demes tested, specifically 28,297 out of 32,000 across the 5 send-impaired mutant birth rates (5 impairment rates × 16 runs × 400 demes). Clearly, the evolved genomes were more susceptible to disruption by receive-impaired mutants than by send-impaired mutants. Based on these data, a logical quorum-quenching treatment would be to introduce receive-impaired mutants into the demes.

We observe that while the dependence trends in Figure 8 are similar to those in Figure 7, a higher resistance to receive-impaired mutants was exhibited. Specifically, a 10% introduction rate did not inhibit QS in all demes. The presence of this resistance is intriguing, since the dominant genomes evolved under conditions free of explicitly introduced mutants, limiting selective pressures that might drive a population toward a resistant solution. However, we note that a pressure to build up resistance to mutants could be supplied during the evolution of QS by the deme-level fitness function, which rewards demes that minimize their energy consumption. Specifically, QS populations must overcome genetic mutations that, during the evolutionary process, regularly produce variants of QS behavior that is disrupted in some way. We hypothesize that this evolutionary pressure produces a more robust algorithm, regardless of whether the misbehaving organisms are produced through natural genetic mutation or are artificially impaired and placed in the population. Extending this line of exploration, we next investigated whether even more resistant organisms might evolve under environmental conditions where the introduction of mutants was gradual.

5.3 Evolving Resistant Organisms—Staged Method

In this set of experiments, we used as seed organisms the QS-performing dominant genomes from the sixteen 5000-generation runs (Section 5.2) and evolved them for an additional 5000 generations. During the runs, the populations were subjected to a staged increase in the level of receive- or send-impaired mutants. Specifically, the impaired mutant introduction rate started at 1% and was increased by 1% every 500 competition periods. At the end of each such stage, we extracted the dominant genomes and evaluated them during one competition period at the same mutant introduction rate. We carried out 10 replicate runs (with different random number seeds) for each of the 16 dominants; each had 400 demes initialized with the corresponding dominant. All other configuration parameters were identical to those of the experiments described in Section 5.2.

Figure 9 plots the results for the staged introduction method. Consistent with the two previous experiments, receive-impaired mutants had a greater effect on a deme's ability to conserve energy than did send-impaired mutants. However, these data show that the organisms developed a stronger resistance to both mutant types. Indeed, for the case of send-impaired mutants, all of the 160 runs exhibited energy-conserving QS behavior at the end of every stage. Even at the 10% introduction

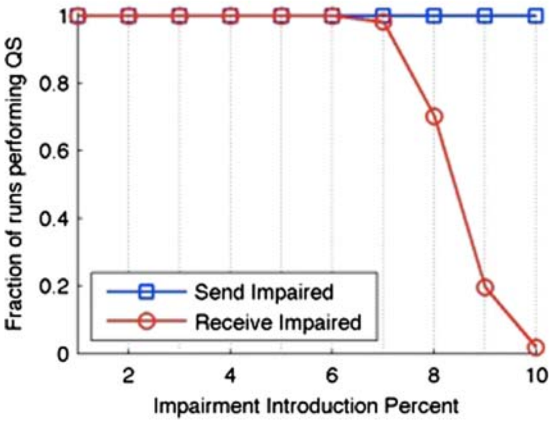


Figure 9. Fraction of runs performing QS when subjected to staged introduction of mutants. Each data point is the mean of 160 runs (16 dominants, 10 runs each).

rate, all demes survived, compared to approximately 90% in the previous experiment (point *B* in Figure 8). Apparently, as the environment became increasingly adverse with respect to the introduction of send-impaired mutants, the evolutionary process produced organisms that were more robust to these mutants. For the case of receive-impaired mutants, the last three stages demonstrate a rapid decline in the fraction of populations that survived; by the last stage, only 6 of the 160 runs were viable. However, at introduction rates from 1% to 6%, all of the 160 dominants were immune to receive-impaired mutants, a considerable increase compared to the results in Figure 8.

Despite the clear effect of mutants on deme mortality rate, we note that these data alone do not reveal whether QS was actually *suppressed* by the introduction of mutants, or whether the code for the behavior simply evolved away. To help answer this question, we analyzed the behavior of individual organisms. The strategy evolved in all initial QS dominant seed organisms was to repeatedly interrupt their neighbors at quorum, preventing self-replication. By causing interrupts, an organism can effectively extend the time until its neighbor replicates, assuming replication does not occur within the interrupt handler. Figure 10 shows the mean number of organisms interrupted per deme during evolution in the staged environment. For the case of send-impaired mutants, we observe only a small decline in the number of interrupted organisms, indicating that the genomes subjected to these mutants did not lose the genetic code required to handle an interrupt. However, we observe a larger decline in the

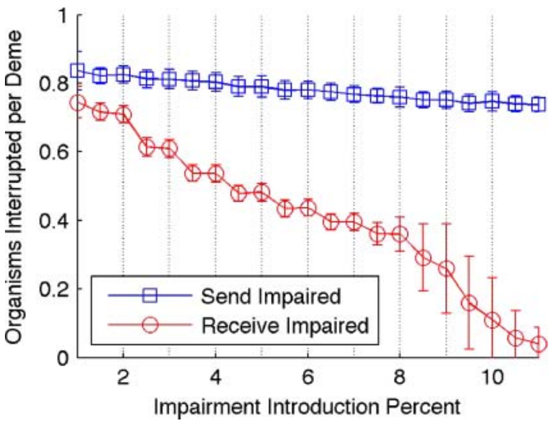


Figure 10. Fraction of organisms interrupted per deme during evolution in the staged environment. Error bars denote one standard deviation from mean. Each data point is a composite of 160 runs.

treatment where receive-impaired mutants were introduced. In the early stages, the interrupt code is apparently still present in the genome. However, by the last few stages, the decline to near zero indicates that either the mutants are disrupting QS to the extent that organisms are rarely interrupted, or that the genetic code to handle interrupts has evolved away.

To elucidate changes in genomes (and hence behavior) over evolutionary time, we extracted dominant genomes at time points 0, 25, 50, 75, and 100 percent through each of the staged environment runs. We then tested these dominant genomes for QS behavior in the absence of mutants. Specifically, each dominant genome was used to construct a seed organism that was then injected into each of 400 demes. The demes were allowed to execute for one competition period, and we tracked the mean organism density (fraction of occupied cells per deme) of all demes. We then identified the genomes that were performing QS and plotted the mean organism density of all demes for each of the dominant genomes over the course of a single competition period.

Figure 11a plots the mean organism density of the dominant genomes extracted at different points during the staged introduction runs. These data show that as the runs progressed (and the percentage of mutants increased), the populations evolved to reach quorum at a lower organism density, concluding with a mean quorum-triggering density of approximately 0.65 for organisms extracted 100% of the way through the runs. In addition, we note that all pairwise comparisons of organism density at the end of a competition period, excluding the 0% and 25% pair, are significantly different according to the Wilcoxon rank sum test for equal medians using an $\alpha = 0.01$. The significant decline in organism density required to reach a quorum illustrates the effect of evolution with increasing numbers of send-impaired mutants. Specifically, a quorum is sensed, triggering behavioral change, with fewer organisms, thereby producing a fitter deme.

Figure 11b shows that receive-impaired mutants had a similar effect, reducing the number of organisms required to achieve a quorum. However, unlike send-impaired mutants, as the introduction of receive-impaired mutants increased, the time required to achieve a quorum also increased, which had a negative effect on the ability of a deme to sense a quorum. Eventually, only 6 of the 160 dominant genomes present at the end of the staged environment runs performed QS in the absence of mutants. These are the same 6 runs that perform QS in the presence of receive-impaired mutants introduced at a rate of 10%. This fact, in conjunction with the data presented in Figures 9 and 10, demonstrates that receive-impaired mutants introduced at a rate of 10% of all births was effective at disrupting QS. Furthermore, their presence caused QS behavior to evolve away in more that 96% of the runs. Finally, we note that the 100% populations actually reach a higher quorum level than that of the 25%, 50%, and 75% populations. This result could be due to a lucky mutation that produced more

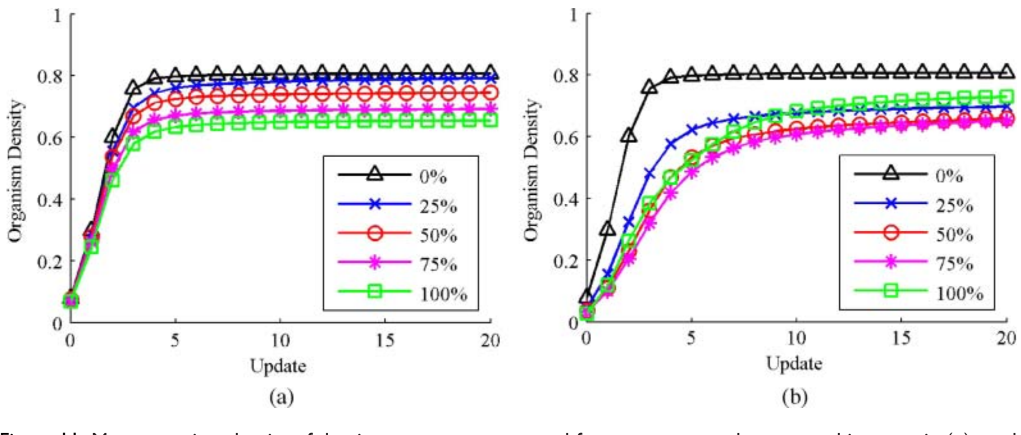


Figure 11. Mean organism density of dominant genomes extracted from runs exposed to a staged increase in (a) send-impaired mutants or (b) receive-impaired mutants. Different curves (0%, 25%, 50%, 75%, 100%) refer to the point in the run when the dominant genomes were extracted. Error bars are omitted for clarity. Data is a composite of 160 samples per line, 16 runs (one per dominant genome) × 10 replicates.

robust QS behavior near the end of the run, possibly combined with the effects of the small sample size for this data point (6 runs).

6 Conclusions and Future Work

Quorum sensing is a collective signaling behavior used by bacteria for a wide variety of purposes, both beneficial and harmful. Enhancing our understanding of cooperation in this process, and how it can be disrupted, can directly advance many fields, including ecology, agriculture, and medicine. In particular, quorum quenching therapies are viewed as a potential alternative to traditional antibiotics, and recent in vivo experiments show promise [52]. However, while evolutionary models predict such treatments are less likely to lead to the development of resistance, this hypothesis is difficult to test in living organisms in a timely manner. The complexity of these biological processes demands novel approaches, such as computational evolution, to tease apart the interrelated factors that produce and sustain them.

In this article we have explored both quorum sensing and quorum quenching in digital organisms. Our experiments demonstrated that Avida populations, when provided with the instructions capable of handling interrupts, evolved a density-dependent signaling protocol reminiscent of QS in bacteria. Effectively, the Avidians evolved to treat messages as autoinducer molecules: For every message received, the organism jumped to its interrupt handler and sent three messages, producing a positive feedback loop characteristic of QS in natural systems. Avidians used this protocol to suppress population growth, thereby conserving energy and achieving higher fitness.

We then carried out several treatments where we introduced either send-impaired or receive-impaired mutant organisms into Avida populations. The results indicated that the introduction of send-impaired mutants had a relatively small effect on populations that had already evolved QS, but receive-impaired mutants had a large effect. In a subsequent set of experiments, these mutants were introduced at increasing rates over time, from 1% to 10%, and a higher level of resistance was observed for both types of mutants. Analysis of the dominant genomes showed that organisms had evolved resistance by reaching quorum at lower cell densities than the parent strain; effectively, the populations responded to disruption by altering the definition of quorum.

Our ongoing studies address both computational and biological QS experiments. In addition to providing insight into general principles regarding the evolutionary process, targeted Avida experiments can generate testable hypotheses for specific wet-bench experiments. We are currently conducting Avida-inspired studies with the bacteria *Vibrio harveyi*, *Vibrio cholerae*, and *Pseudomonas aeruginosa*; these species are genetically tractable and have well-established QS and biofilm formation systems. For instance, we are testing the predictions generated by the Avida study described in this article by examining the ability of send-impaired and receive-impaired *V. harveyi* mutants to inhibit QS of the wild-type strain. We expect the results of those experiments to feed back into the computational models, leading to a deeper understanding of not only collective behaviors, but also the selective pressures and environmental conditions that can lead to their evolution. It is our goal that these two distinct but related methods will inform and direct each other in a reciprocal manner, accelerating any progress that either approach could make independently.

7 Further Information

Information on evolving adaptive and cooperative behavior in Avida can be found at http://www.cse.msu.edu/thinktank. Papers on other applications of digital evolution and the Avida software are available at http://devolab.msu.edu. The source code for Avida is free software available under the GNU General Public Licence and is available for download via SourceForge (http://sourceforge.net/projects/avida). All data reported in this article were produced using version 3204 of the Avida source code, also publicly available in a subversion repository located at https://avida.devosoft.org/svn/branches/interrupt.

Acknowledgments

The authors gratefully acknowledge the contributions and feedback on the work provided by members of the Software Engineering and Network Systems Laboratory and the Digital Evolution Laboratory at Michigan State University. This work was supported by NSF grants DBI-0939454, CNS-0915855, CCF-0820220, CNS-0751155, and by U.S. Army Grant W911NF-08-1-0495.

References

1. André, J.-B., & Godelle, B. (2005). Multicellular organization in bacteria as a target for drug therapy. *Ecology Letters*, *8*(June), 800–810.
2. Bassler, B. L., Wright, M., Showalter, R. E., & Silverman, M. R. (1993). Intercellular signalling in *Vibrio harveyi*: Sequence and function of genes regulating expression of luminescence. *Molecular Microbiology*, *9*(4), 773–786.
3. Beckmann, B. E. (2010). *Evolving cooperative, energy-conserving agent-based systems*. Ph.D. thesis, Department of Computer Science and Engineering, Michigan State University, East Lansing, MI.
4. Beckmann, B. E., & McKinley, P. K. (2008). Evolution of adaptive population control in multi-agent systems. In *Proceedings of the Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems* (pp. 181–190).
5. Beckmann, B. E., & McKinley, P. K. (2009). Evolving quorum sensing in digital organisms. In *Proceedings of the ACM Genetic and Evolutionary Computation Conference (GECCO-2009)* (pp. 97–104).
6. Beckmann, B. E., McKinley, P. K., Knoester, D. B., & Ofria, C. (2007). Evolution of cooperative information gathering in self-replicating digital organisms. In *Proceedings of 1st International Conference on Self-Adaptive and Self-Organizing Systems (SASO)* (pp. 65–76). IEEE Computer Society.
7. Beckmann, B., McKinley, P. K., & Ofria, C. A. (2007). Evolution of adaptive sleep response in digital organisms. In *Advances in artificial life (Proceedings of the 9th European Conference on Artificial Life)* (pp. 233–242).
8. Béjà, O., Spudich, E. N., Spudich, J. L., Leclerc, M., & DeLong, E. F. (2001). Proteorhodopsin phototrophy in the ocean. *Nature*, *411*, 786–789.
9. Bernardini, F., Gheorghe, M., & Krasnogor, N. (2007). Quorum sensing P systems. *Theoretical Computer Science*, *371*(1–2), 20–33.
10. Brown, S. P., & Johnstone, R. A. (2001). Cooperation in the dark: Signalling and collective action in quorum-sensing bacteria. *Proceedings of the Royal Society of London B: Biological Sciences*, *268*(1470), 961–965.
11. Brown, S. P., West, S. A., Diggle, S. P., & Griffin, A. S. (2009). Social evolution in micro-organisms and a Trojan horse approach to medical intervention strategies. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *364*, 3157–3168.
12. Chow, S., Wilke, C. O., Ofria, C., Lenski, R. E., & Adami, C. (2004). Adaptive radiation from resource competition in digital organisms. *Science*, *305*, 84–86.
13. Clatworthy, A. E., Pierson, E., & Hung, D. T. N. (2007). Targeting virulence: A new paradigm for antimicrobial therapy. *Nature Chemical Biology*, *3*(September), 541–548.
14. Costerton, J. W., & Stewart, P. S. (2010). Battling biofilms. *Scientific American*, *285*(July), 60–67.
15. Crespi, B. J. (2001). The evolution of social behavior in microorganisms. *Trends in Ecology and Evolution*, *16*(April), 178–183.
16. Czárán, T., & Hoekstra, R. F. (2009). Microbial communication, cooperation and cheating: Quorum sensing drives the evolution of cooperation in bacteria. *PLoS ONE*, *4*(8), e6655.
17. Davies, D. (2003). Understanding biofilm resistance to antibacterial agents. *Nature Reviews Drug Discovery*, *2*, 114–122.
18. Defroirdt, T., Boon, N., & Bossier, P. (2010). Can bacteria evolve resistance to quorum sensing disruption? *PloS Pathogens*, *6*(July), 73–87.
19. Dickschat, J. S. (2010). Quorum sensing and bacterial biofilms. *Natural Product Reports*, *27*, 343–369.
20. Diggle, S. P., Griffin, A. S., Campbell, G. S., & West, S. A. (2007). Cooperation and conflict in quorum-sensing bacterial populations. *Nature*, *450*, 411–414.

21. de Kievit, T. R., & Iglewski, B. H. (2000). Bacterial quorum sensing in pathogenic relationships. *Infection and Immunity*, *68*(September), 4839–4849.

22. Dockery, J., & Keener, J. (2001). A mathematical model for quorum sensing in *Pseudomonas aeruginosa*. *Bulletin of Mathematical Biology*, *63*(January), 95–116.

23. Floreano, D., Mitri, S., Magnenat, S., & Keller, L. (2007). Evolutionary conditions for the emergence of communication in robots. *Current Biology*, *17*(March), 514–519.

24. Fuqua, C., Winans, S., & Greenberg, E. (1996). Census and consensus in bacterial ecosystems: The LuxR-LuxI family of quorum-sensing transcriptional regulators. *Annual Review of Microbiology*, *50*, 727–751.

25. Goings, S., Clune, J., Ofria, C., & Pennock, R. (2004). Kin selection: The rise and fall of kin-cheaters. In *Proceedings of the Ninth International Conference on Artificial Life* (pp. 303–308).

26. Hall-Stoodley, L., Costerton, J. W., & Stoodley, P. (2004). Bacterial biofilms: From the environment to infectious disease. *Nature Reviews Microbiology*, *2*, 95–108.

27. Hamilton, W. D. (1963). The evolution of altruistic behavior. *The American Naturalist*, *97*, 354–356.

28. Hammer, B. K., & Bassler, B. L. (2003). Quorum sensing controls biofilm formation in *Vibrio cholerae*. *Molecular Microbiology*, *50*, 101–104.

29. Hansen, S. K., Rainey, P. B., Haagensen, J. A. J., & Molin, S. (2007). Evolution of species interactions in a biofilm community. *Nature*, *445*, 533–536.

30. Hill, J. L. (2003). *System architecture for wireless sensor networks*. Ph.D. thesis, University of California at Berkeley.

31. Kambam, P. K. R., Henson, M. A., & Sun, L. (2008). Design and mathematical modelling of a synthetic symbiotic ecosystem. *IET Systems Biology*, *2*(January), 33–38.

32. Kiran, M. D., Giacometti, A., Cirioni, O., & Balaban, N. (2008). Suppression of biofilm related, device-associated infections by staphylococcal quorum sensing inhibitors. *International Journal of Artificial Organs*, *31*, 761–770.

33. Knoester, D. B., & McKinley, P. K. (2009). Evolution of probabilistic consensus in digital organisms. In *Proceedings of the Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems* (pp. 223–232).

34. Knoester, D. B., & McKinley, P. K. (2011). Evolution of synchronization and desynchronization in digital organisms. *Artificial Life*, *17*(1), 1–20.

35. Knoester, D. B., McKinley, P. K., & Ofria, C. (2008). Cooperative network construction using digital germlines. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*.

36. Lenski, R. E., Ofria, C., Pennock, R. T., & Adami, C. (2003). The evolutionary origin of complex features. *Nature*, *423*, 139–144.

37. Lewandowski, Z. (2000). Structure and function of biofilms. In L. Evans (Ed.), *Biofilms: Recent Advances in Their Study and Control* (pp. 1–17). Harwood Academic Publishers.

38. McKinley, P., Cheng, B., Ofria, C., Knoester, D., Beckmann, B., & Goldsby, H. (2008). Harnessing digital evolution. *IEEE Computer*, *41*(January), 54–63.

39. Miller, M. B., & Bassler, B. L. (2001). Quorum sensing in bacteria. *Annual Review of Microbiology*, *55*, 165–199.

40. Monds, R. D., & O'Toole, G. A. (2009). The developmental model of microbial biofilms: Ten years of a paradigm up for review. *Trends in Microbiology*, *17*, 73–87.

41. Nadell, C. D., Foster, K. R., & Xavier, J. B. (2010). Emergence of spatial structure in cell groups and the evolution of cooperation. *PLoS Computational Biology*, *6*(3). **Q2**

42. Nadell, C. D., Xavier, J. B., Levin, S. A., & Foster, K. R. (2008). The evolution of quorum sensing in bacterial biofilms. *PLoS Biology*, *6*(January), 0171–0179.

43. Nealson, K., & Hastings, J. (1979). Bacterial bioluminescence: Its control and ecological significance. *Microbiological Reviews*, *43*(4), 496–518.

44. Nealson, K. H., Platt, T., & Hastings, J. W. (1970). Cellular control of the synthesis and activity of the bacterial luminescent system. *Journal of Bacteriology*, *104*(1), 313–322.

45. Njoroge, J., & Sperandio, V. (2009). Jamming bacterial communication: New approaches for the treatment of infectious diseases. *EMBO Molecular Medicine*, *1*, 201–210.

46. Ofria, C., & Wilke, C. O. (2004). Avida: A software platform for research in computational evolutionary biology. *Artificial Life*, *10*(March), 191–229.

47. Ostrowski, E., Ofria, C., & Lenski, R. E. (2007). Ecological specialization and adaptive decay in digital organisms. *The American Naturalist*, *169*, E1–E20.

48. Pan, J., & Ren, D. (2009). Quorum sensing inhibitors: A patent overview. *Expert Opinion on Therapeutic Patents*, *19*, 1581–1601.

49. Păun, G., & Rozenberg, G. (2002). A guide to membrane computing. *Theoretical Computer Science*, *287*(September), 73–100.

50. Raina, S., De Vizio, D., Odell, M., Clements, M., Vanhulle, S., & Keshavarz, T. (2009). Microbial quorum sensing: A tool or a target for antimicrobial therapy? *Biotechnology and Applied Biochemistry*, *54*, 65–84.

51. Romero-Campero, F. J., & Pérez-Jiménez, M. J. (2008). A model of the quorum sensing system in Vibrio fischeri using P systems. *Artificial Life*, *14*(1), 95–109.

52. Rumbaugh, K. P., Diggle, S. P., Watters, C. M., Ross-Gillespie, A., Griffin, A. S., & West, S. A. (2009). Quorum sensing and the social evolution of bacterial virulence. *Current Biology*, *19*(January), 341–345.

53. Sandoz, K. M., Mitzimberg, S. M., & Schuster, M. (2007). Social cheating in *Pseudomonas aeruginosa* quorum sensing. *Proceedings of the National Academy of Sciences*, *104*, 15876–15881.

54. Schauder, S., Shokat, K., Surette, M., & Bassler, B. (2001). The LuxS family of bacterial autoinducers: Biosynthesis of a novel quorum-sensing signal molecule. *Molecular Microbiology*, *41*(July), 463–476.

55. Terrazas, G., Krasnogor, N., Gheorghe, M., Bernardini, F., Diggle, S., & Cámara, M. (2005). An environment aware P-system model of quorum sensing. In S. B. Cooper, B. Löwe, & L. Torenvliet (Eds.), *New Computational Paradigms, First Conference on Computability in Europe (CiE)* (pp. 479–485). Berlin, Heidelberg: Springer.

56. Waters, C. M., & Bassler, B. L. (2005). Quorum sensing: Cell-to-cell communication in bacteria. *Annual Review of Cell and Developmental Biology*, *21*, 319–346.

57. Waters, C., Lu, W., Rabinowitz, J. D., & Bassler, B. L. (2008). Quorum sensing controls biofilm formation in *Vibrio cholerae* through modulation of cyclic di-GMP levels and repression of vpsT. *Journal of Bacteriology*,
Q3 *190*(7), 2527–2536.

58. Watnick, P., & Kolter, R. (2000). Biofilm, city of microbes. *Journal of Bacteriology*, *182*(May), 2675–2679.

59. Whitley, W. D., Rana, S. B., & Heckendorn, R. B. (1997). Island model genetic algorithms and linearly separable problems. In *Selected Papers from AISB Workshop on Evolutionary Computing* (pp. 109–125). Springer-Verlag.

60. Wilke, C. O., Wang, J., Ofria, C., Adami, C., & Lenski, R. E. (2001). Evolution of digital organisms at high mutation rate leads to survival of the flattest. *Nature*, *412*, 331–333.

61. Xavier, K. B., & Bassler, B. L. (2005). Interference with AI-2-mediated bacterial cell-cell communication. *Nature*, *437*, 750–753.

62. Xavier, J. B., & Foster, K. R. (2007). Cooperation and conflict in microbial biofilms. *Proceedings of the National Academy of Sciences*, *104*, 876–881.

63. Zhu, J., & Mekalanos, J. J. (2003). Quorum sensing-dependent biofilms enhance colonization in *Vibrio cholerae*. *Developmental Cell*, *5*, 647–656.

AUTHOR QUERIES

AUTHOR PLEASE ANSWER ALL QUERIES

During the preparation of your manuscript, the questions listed below arose. Kindly supply the necessary information.

1. Please check if the proposed running head is correct.
2. Please provide page range.
3. Please check if okay to change 2527–36 to 2527–2536.

END OF ALL QUERIES