Amplitude-Width Encoding for Error Correction in Bacterial Communication Networks

Bhuvana Krishnaswamy
Georgia Institute of Technology
bhuvana@ece.gatech.edu

Raghupathy Sivakumar
Georgia Institute of Technology
siva@ece.gatech.edu

ABSTRACT

In this work, we consider the problem of forward error correction in a multiple access molecular communication network with bacteria as transceivers. A number of forward error correction techniques have been developed to maximize throughput and achieve the lower bound on the bit error rate performance. All existing codes were developed for traditional networks and hence the constraints on computational complexity do not match that of bio-circuits. Designing reliable and accurate bio-circuits for operations like polynomial multiplication that are basic to FEC is extremely challenging.

We propose and design Amplitude-Width Forward Error Correction, a simple and efficient FEC mechanism that can be implemented using bio-circuits reliably for real-time application. FEC techniques allow the receiver to detect and correct for errors by introducing redundancy in the message transmitted. EEC introduces redundancy by varying the on-period of the signal transmitted across senders. Senders with the same on-period are assigned amplitudes with maximum distance. Increasing the distance between amplitudes of senders with the same on-period increases the error resilience of the receiver. Bit error rate of the order of 10^{-2} is achieved using the proposed error correction mechanism.

KEYWORDS

Molecular communication, On-Off Keying, Embedded Addressing, Multiple Access, Forward Error Correction

ACM Reference Format:

Bhuvana Krishnaswamy and Raghupathy Sivakumar. 2018. Amplitude-Width Encoding for Error Correction in Bacterial Communication Networks. In *NANOCOM '18: NANOCOM '18: ACM The Fifth Annual International Conference on Nanoscale Computing and Communication, September 5–7, 2018, Reykjavik, Iceland*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3233188.3233212

1 INTRODUCTION

Advancements in synthetic biology and bio-engineering [3] have enabled bacteria as emerging candidates for processing, transmission, and reception in a Molecular Communication (MC) System. [18] presents a modular approach to building genetic circuits from basic building blocks. Genetically engineered bacteria as sensors have been used in different applications such as detection of heavy metal toxicity in liquid [6], quality of drinking water, [9], food pathogen detection [10]. We consider a pathogen detection system, where bacterial populations that are genetically engineered to detect specific pathogens are deployed in the field. The sensor reports the information to a receiver (sink) continuously. The sink monitors the information received and notifies the user (human client) about the presence and intensity of each of these pathogens. Such a pathogen detection system can avoid the onset of an epidemic by identifying disease-causing pathogens at an early stage. A successful implementation of the above architecture can, therefore, open more applications of autonomous biosensor networks. Building a complex network of biological sensors is a challenging problem. Working towards this goal, we consider a simple star topology as shown in Figure 1. Each sender is a genetically engineered bacterial population that senses a particular pathogen and modulates a signaling chemical (Acyl HomoserineLactone molecule) that carries the information to a local receiver or sink. The sink then uploads information to a centralized server. In a network of bacterial sensors and processors, communication algorithms that can be implemented using simple and feasible bio-circuits is an essential requirement.

The focus of this work is to ensure reliable transfer of information in an MC system. Specifically, we focus on designing low complexity error correction codes that can be implemented using genetically engineered bacterial populations. The existing error correction codes are adapted from traditional coding techniques and do not consider the practicality of implementing these codes in a real-time, live bacterial system. The complexity of the code and its implementation is a crucial factor in the practical realization of the reliability mechanism. In this work, we make the following contributions

(1) We identify the challenges in implementing a Forward Error Correction (FEC) in a bacterial communication system
(2) We propose a forward error correction mechanism that introduces redundancy in the duty-cycle of the transmitted signal, Amplitude-Width Error Correction (AWEC). Each sender is assigned a unique 2-tuple identification $<$ **amplitude, on $-$ period** $>$.
(3) We design a practical encoder and decoder to assign and decode the amplitude and on-period that maximizes the decoding efficiency.
(4) We implement the proposed error correction mechanism in a python based custom built MC simulator.

The remainder of the paper is organized as follows; in Section 2, we present the application considered, the communication algorithms required to build the application, and an account of the types of errors and related work on reliability. In Section 3, we describe the proposed error correction codes, the design goals and challenges of the encoder and in Section 4, we discuss in detail the decoder architecture. We present the performance evaluation of AWEC in Section 5 and conclude the paper in Section 6.

NANOCOM '18, September 5–7, 2018, Reykjavik, Iceland

ACM ISBN 978-1-4503-5711-1/18/09…$15.00
https://doi.org/10.1145/3233188.3233212

2 BACKGROUND AND MOTIVATION

2.1 System Architecture

We consider a pathogen detection system consisting of genetically engineered sensor bacteria that detects a specific pathogen and communicates the intensity of the pathogen to a receiver bacteria. Specifically, in this work, we consider *Escherichia coli (E. coli)* bacteria genetically engineered to exhibit fluorescence upon the receipt of a specific signal molecule (N-(3-Oxyhexanoyl)-L-homoserine lactone, or C6-HSL). Fluorescent images were captured once every 10 minutes and post-processed using MATLAB. The intensity of the pixels within the bacteria chamber was averaged and the background fluorescence was subtracted, yielding relative fluorescence (arbitrary units, or AU). Methods and functionality of the bacteria and the microfluidic device fabrication and specifications can be seen in [1, 13].

Multiple senders, each detecting a unique pathogen communicate using the same signaling molecule to the single receiver as shown in Figure 1. The above architecture is typical of a biosensing network and thus algorithms developed for this system is applicable to other bio-networks as well. We identify three fundamental communication problems needs to be solved to realize a bio-network viz., modulation, multiple access control (MAC), and reliability.

Modulation. The transmitter conveys the intensity of the pathogens detected using signaling molecules. A modulation technique to encode information in the molecules is required. Here, we consider a simple modulation technique, On-Off-Keying (OOK), where bit 1 is represented by a rectangular pulse of amplitude A and period T_b and bit 0 by an absence of signal for T_b. Concentration Shift Keying [16], Molecule Shift Keying [12], Time Elapse Communication [13], have been designed specifically for MC to maximize information transfer. Even though we consider OOK as the modulation technique, the proposed reliability mechanism can operate with the above modulation techniques with minor alterations.

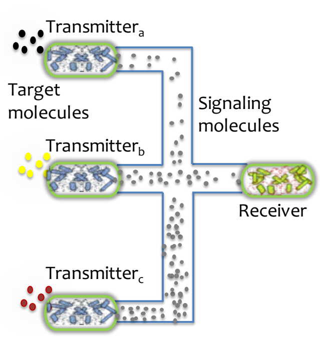


Figure 1: Pathogen Detection System

Multiple Access Control. In a star topology with multiple senders, a MAC protocol to let the senders share the channel and the resources at the receiver is required. In this work, we consider ADMA (Amplitude Division Multiple Access), an amplitude based medium access control proposed in [14]. ADMA assigns a unique amplitude to each sender, where each sender uses OOK as the modulation technique. The receiver decodes the received signal and maps the received amplitude to the sender, thus eliminating address overheads. ADMA is a simple to implement and practical MAC designed specifically for MC system with very high delays and low computational complexity. An implementation for amplitude assignment

Commonly used MAC protocols such as random access control, token-based, time division multiplexing and frequency division multiplexing are unsuitable for the unique delay and complexity constraints of the MC system. Multiple senders transmit rectangular signals with an amplitude that is assigned uniquely to the sender. The receiver receives the sum of amplitudes when multiple senders transmit simultaneously. ADMA proposes an amplitude address assignment algorithm, which, along with an optimal decoder can resolve the addresses from the received summation in the absence of any amplitude errors in the channel.

2.2 Errors in an MC System

The amplitude of a molecular signal can be affected by the channel and the receiver design. We identify three types of errors that affect the the received amplitude viz., 1) channel induced errors, 2) receiver induced errors, and 3) collision-induced errors.

2.2.1 **Channel error.** Channel noise models [1, 4, 21] and capacity analysis [2, 19] have been developed for diffusive channel. Though the channel noise model depends on the geometry, type of molecule, rate of flow and varies with the application, we observe that in each of these models, the channel noise is proportional to the concentration (amplitude) of the signal being transmitted. In a closed-loop system as shown in Figure 1, the molecules reach the receiver only when transmitted by the sender and hence we assume bounded channel error. The uniform distribution has maximum entropy in a bounded noise case and therefore has the worst case performance. We analyze and evaluate our proposed solution for the worst case channel noise and hence assume uniformly distributed channel noise.

2.2.2 **Receiver error.** In a molecular communication system with a bacterial receiver, the response of the receiver bacteria is a non-linear function of the input amplitude as modeled and experimentally validated in [1]. The stochastic nature of receiver bacteria leads to variations in the fluorescent response of the receiver which in turn leads to receiver induced demodulation errors. The receiver error on the amplitude is independent of the channel error and collision error and depends only on the circuit design of the bacterial receiver. The receiver noise model is not well-defined. For simplicity of analysis, we assume a bounded, uniform distribution.

2.2.3 **Collision error.** When senders use OOK to transmit information, bit 0 is communicated by an absence of signal for a period T and therefore does not collide with signals from other senders at the receiver. Collision errors are caused by the transmission of bit 1 from multiple senders. The fewer the number of bit 1s to be transmitted, fewer is the number of collisions and hence few errors due to collisions. It must be observed that even though bit 0 from different sender does not collide, its reception can be affected by collision of bit 1.

2.3 Problem Definition and Design Challenges

To ensure reliable reception of the amplitude conveying both the information and the address, an error correction mechanism is required. Due to the high latency of an MC system [13], feedback based error correction mechanism will negatively impact the throughput performance and complexity of system design. Capacity approaching forward error correction (FEC) codes such as convolutional codes, LDPC [7] to detect and correct errors have been

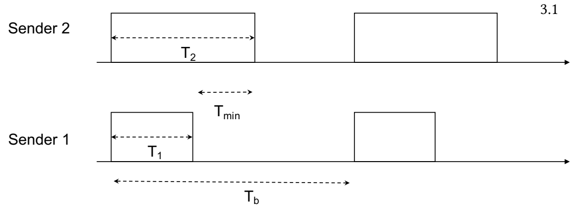


Figure 2: Redundancy in Duty-cycle

widely used in traditional networks. Implementing these FEC codes using biological circuits is highly challenging and the accuracy and consistency of the circuit design deteriorate with increasing complexity [11].

A number of research works have modified traditional FEC codes for MC networks without considering the practical constraints of an MC system [5, 8, 17]. [20] develops a family of ISI-free (Inter-Symbol Interference) codes that are simple and practical. The ISI free codes increase the Hamming distance between codewords and assign unique Hamming weight codewords to detect and correct codeword errors. Even though [20] provides a practical error correction code, it relies on a MAC protocol to handle collision. Any error caused by channel collisions will result in packet drop at the receiver.

An FEC that ensures reliable reception in a multiple access molecular communication network is still an open challenge. The focus of this work is in the design of a practical, low-complexity error correction code for an MC system with high latency.

Design of an efficient reliability mechanism has the following challenges

(1) Low complexity : should not require additional modules to implement reliability mechanisms
(2) High accuracy : correct amplitude errors induced by the channel and receiver
(3) High coding gain : should not affect the network throughput
(4) Multiple Access : handle collisions in a multiple access network.

In this work, we develop a reliable molecular communication system that corrects for amplitude errors in a multiple access, single-hop network.

3 AMPLITUDE-WIDTH FORWARD ERROR CORRECTION

Based on the insights from the analysis of amplitude errors, we develop AWEC, a forward error correction mechanism that embeds redundancy in the on-period and amplitude of the transmitted signal. The characteristics of a rectangular molecular signal as shown in Equation 1 are, amplitude A, on-period T_{ON}, bit period T_b, and molecule type (AHL).

$$m(t) = \begin{cases} A, & 0 \leq t \leq T_{ON} \\ 0, & T_{ON} < t \leq T_b. \end{cases} \tag{1}$$

3.1 Receiver error correction

From the analysis of receiver induced error, we infer that the response of the receiver is a linear function of the duration of the signal. AWEC assigns a distinct on-period to different senders, thereby allowing accurate decoding of the sender at the receiver. Figure 2 is an illustration of distinct on-periods being assigned to two senders. On receiving the signal, the decoder maps the received T_{ON} to the closest on-periods that are assigned to the senders.

Algorithm 1 describes the AWEC encoder design. In lines 6-9, AWEC cyclically assigns the unique on-periods to each sender. The on-periods assigned are determined two system constraints

Algorithm 1 Encoder : Width and Amplitude Assignment

1: $N \leftarrow$ Number of senders
2: $N_w \leftarrow$ Number of unique on-periods
3: $(-\frac{w_e}{2}, +\frac{w_e}{2}) \leftarrow$ Estimated width error
4: $W \leftarrow \{T_1, T_2, \ldots, T_N\}$, on-period assigned
5: $A \leftarrow \{A_1, A_2, \ldots, A_N\}$, amplitudes
▷ Width assignment
6: **for** i := 1 to N **do**
7: bit_i ← i(mod N_w)
8: $T_\text{i} \leftarrow T_{ON} + \text{bit\_i} \cdot w_e$
9: **end for**
▷ Amplitude assignment
10: **for** i := 1 to N **do**
11: block_i ← i/N_w
12: $A_\text{i} \leftarrow 2^{\text{block\_i}}$
13: **end for**

specified by the receiver bacteria viz., T_{ON}, the minimum on-period required to decode a rectangular signal and w_e, an estimate of the width error at the receiver. The on-periods are chosen such that the minimum difference between two on-periods is $\geq \frac{w_e}{2}$ (line 8). A received on-period in the range $\left(T_1 - \frac{w_e}{2}, T_1 + \frac{w_e}{2}\right)$ is decoded as T_1 thus implying that the receiver can correct up-to $\frac{w_e}{2}$ error in the on-period of the transmitted signal. For a known bounded on-period error w_e, the receiver induced error can be corrected by increasing the distance between on-periods of senders.

However, assigning a unique on-period to each sender with a minimum distance of w_e increases the overall T_b, one bit-period. AWEC utilizes the amplitude of the rectangular signal to overcome this disadvantage of on-period based error correction. In line 7, AWEC limits the number of unique on-periods to N_w and assigns to N senders with repetition ($N_w < N$). The senders with the same on-period are differentiated using unique amplitudes i.e., AWEC assigns a unique 2-tuple id <**amplitude, on-period**> to each sender. The higher the number of on-periods, (N_w), lower is the intra-width collisions and higher the overall T_b. Here we present our solution to choose N_w that reduces the probability of intra-width collisions without reducing throughput significantly. The probability of k senders colliding at a given time is given by,

$$\Pr(k\ collisions) = \binom{N}{k} p^k \cdot (1-p)^{(N-k)} \tag{2}$$

where, N is the total number of senders in the network and p is the probability of each sender transmitting bit 1. The probability of intra-width collisions is calculated by replacing N with N_w in Equation 2. Choosing very high values of N_w affects the throughput performance while very low values of N_w leads to increase in intra-width collisions.[1] We determine N_w that satisfies the condition of $\{\Pr(\textit{Intra-width collisions} \geq 2) \leq 0.2\}$ give p and N. The threshold can vary with application and the system in use. N_w that

[1] To achieve fewer collisions and small values of N_w, p_1, the probability of each sender transmitting bit 1 must be small. In this work, we focus on a system with small p_1. We consider encoding techniques [13] that achieve a low probability of bit 1 being transmitted in the channel such that p is low.

satisfies the above condition is then used in Algorithm 1 to assign on-periods.

3.2 Channel error correction

For a network with N sender and N_w unique on-periods, up-to $A_N = \lfloor \frac{N}{N_w} \rfloor$ senders have the same on-period (for each on-period). The choice of amplitudes assigned affect the error correction capability of AWEC due to channel and collision induced amplitude errors discussed in Section 2. It has been proved in [15] that a *binary set* of amplitudes, the set of increasing powers of 2 (example $S : \{1, 2, 4, 8\}$), is an optimal amplitude assignment to recover from collision errors. This is because the sum of any combination of powers of 2 is a unique value. For example, let four senders be assigned amplitudes $1, 2, 4, 8$ and an on-period T_1. On receiving an amplitude 5 and on-period T_1, the receiver can identify that senders with amplitudes 1 and 4 transmitted bit 1 while others transmitted bit 0, as there is only one possible way to arrive at this sum. Lines 10 to 13 in Algorithm 1 assigns powers of 2 amplitudes to the senders with the same on-period. Similar to on-period, the amplitudes are repeated across senders i.e., senders with different on-periods have same amplitude. Amplitude repetition is designed to allow for maximum distance between adjacent amplitudes.

A binary amplitude assignment can correct for collision errors but assumes that the amplitude received is accurate. In the presence of channel errors, the received amplitude can be greater than or less than the sum of amplitudes transmitted leading to an error in decoded amplitudes. The choice of N_w such that the probability of intra-width collisions is less than 0.2 implies that the probability of the received amplitude being the result of the sum of two amplitudes is very small. By choosing N_w that minimizes collisions, the absolute difference between amplitudes is used by the decoder to correct for channel-induced amplitude errors. In the example above, the probability of receiving an amplitude as the sum of 1 and 4 is very small. Therefore, the decoder finds the amplitude closest to 5 (4) and decodes bit 1 for sender with amplitude 4 and on-period T_1 and bit 0 for others. The exponentially increasing amplitudes is suitable to correct for amplitude dependent channel errors. In the binary set of amplitudes, as the amplitudes increase, the difference between adjacent amplitudes increases i.e., the minimum distance between adjacent amplitudes increases and therefore the error correction capability is unaffected. The maximum amplitude that can be assigned to a sender is limited by the saturation amplitude at the receiver. Let A_{max} be the maximum decodable amplitude i.e., any amplitude greater than A_{max} is received as A_{max}. Therefore, N_w must be chosen such that $2^{(N_w-1)} \leq A_{max}$.

Thus, AWEC performs error correction in two steps.

(1) Inter-width errors : Receiver induced errors of upto $\frac{w_e}{2}$ that affects the on-period of the signal is corrected by increasing the distance between adjacent on-periods.
(2) Intra-width errors : Channel induced amplitude errors of upto $\frac{A_{min}}{2}$ that affects the amplitude of the received signals is corrected by increasing the minimum distance between amplitudes that share the same on-period.

3.3 AWEC Codewords

To this end, we presented AWEC embedding redundant information in the on-period and amplitude of the signal to achieve reliability. Embedding redundancy in the transmitted signal offers a practical implementation of codeword generation in an MC system. The discrete samples of the transmitted signal with a given $<\mathbf{amplitude, on-period}>$ is the codeword while the samples with $< 1, T_{ON} >$ is the actual signal. By increasing the on-period and the amplitude values, AWEC can generate codewords without any need for complex mathematical operation. Senders with different on-periods thus transmit rectangular signals whose samples are "1" for the assigned T_{ON}. For example, consider $T_b = 5$ min, a sample period of 1 minute and $T_1 = 3$ min and $T_2 = 5$ min represent codewords $\{1, 1, 1, 0, 0\}$ and $\{1, 1, 1, 1, 1\}$ of bit 1 for sender 1 and sender 2 respectively; $\{0, 0, 0, 0, 0\}$ represent bit 0 for every sender. The hamming distance between two codewords is the number of samples to represent w_e (here, $w_2 = 2$). Similarly, two senders with same T_{ON} are assigned unique amplitudes. For example, assigning amplitudes 2 and 4 to senders with $T_{ON} = 3$ implies their respective bit 1 codewords are $\{2, 2, 2, 0, 0\}$ and $\{4, 4, 4, 0, 0\}$. The absolute difference between the amplitudes determines intra-width error correction capability of AWEC.

4 AMPLITUDE-WIDTH DECODER

The receiver samples, demodulates and decodes the receiving samples to identify the amplitude and on-period of the signal. In this section, we present the decoder architecture used by AWEC to correct for errors.

4.1 Sample and Demodulate

The first block of the decoder architecture is *sample and demodulate*. We utilize the inverse of the receiver response model proposed in [1] to perform sampling and demodulation. The model proposed in [1] is experimentally validated and output the response of a receiver bacteria to an input chemical signal. The inverse of the model takes as input the response of the bacteria and output the amplitude samples for a given sampling rate. The output of the inverse model is thus a time sequence of amplitude samples at the receiver. Each sample in this sequence is the sum of amplitudes from different senders at a particular time instant and the amplitude error introduced by the receiver and channel.

An AWEC decoder takes these samples and estimates the codewords that were transmitted by each sender and corrects for channel and receiver errors. These codewords are then used by the MAC decoder to detect and correct collision errors. The output of *sample and demodulate* module is the time sequence of samples. If T_{ON} is the minimum time difference between on-periods and to capture the on-periods, the sampling rate must be at least $\frac{1}{2T_{ON}}$. The sample and demodulate module uses this parameter to generate $\frac{1}{2T_{ON}}$ discrete samples per second.

4.2 On-period Decoder

The next step in the decoder architecture is to identify the on-period and correct for receiver errors. The *on-period decoder* deciphers the reception of a signal from the amplitude transitions between adjacent samples. Let $s_0, s_1, s_2, \ldots$ be the time sequence of received samples and $r_1, r_2, \ldots$ be the differences of the received sample sequence i.e., $r_1 = s_1 - s_0$. A positive value of r_i indicates the reception of bit 1 by one or more senders and the corresponding difference r_i is the received amplitude. The *on-period decoder* searches for a matching $-r_i$ within one bit period. The difference in time at which the positive and negative r_is were received is the received on-period. Identifying the corresponding negative transition is simpler when the senders are time synchronized and transmit bits only in predetermined slots.

The challenge in an unsynchronized system is in identifying the rise and fall of the rectangular signals when multiple senders are transmitting simultaneously. We present a decoder that considers each case of sender collisions and corrects for collision,, receiver and channel induced errors. We identify the scenarios where the decoder cannot detect or correct for errors.

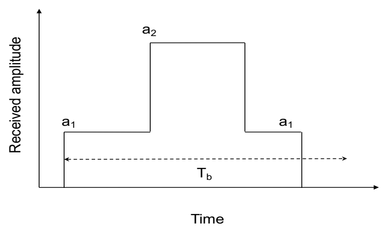


Figure 3: Rx Sample : Illustration

The bit period T_b remains constant across senders, and therefore, r_i, the rise in amplitude corresponding to bit 1 transmission from one(or more) sender must have a corresponding $-r_i$ that indicates the fall of the signal within T_b from r_i. Thus, for every positive rise in the sample amplitude difference, the decoder searches for a matching negative fall with the amplitude. If the rise and fall amplitudes match, the decoder finds the difference in location/time of the matching rise and fall samples to estimate the received on-period $\hat{T_{ON}}$. Figure 3 is an illustration of the rise and fall transition. The two positive transitions a_1 and a_2 are matched with the negative transitions $-a_1$ and $-a_2$ respectively in Figure 3.

When rise and fall of a signal do not collide with the rise and/or of another signal, the decoder can uniquely identify the on-periods and their corresponding amplitude transitions r_i by parsing through the received samples sequentially (Figure 3). In this case, the rise and fall are uniquely identifiable even if the samples in between collide with other signals. However, as the number of senders increases, the probability of two senders colliding with the rise and/or fall increases. We present a decoder algorithm that iterates through each case and estimates the rise and fall of a received signal. The proposed decoder algorithm is motivated by Maximum Aposteriori decoder that uses apriori knowledge of the transmitted symbols and channel transition probabilities. The proposed decoder algorithm iterates through each possibility of the transmitted symbols and decodes the most likely configuration.

For every positive difference $r_i > 0$, the decoder considers the next $2T_b$ samples(the samples corresponding to T_b) and checks for each of the following scenarios to search for the corresponding fall and decode the on-period. When one of the cases returns true, the on-period decoder exits the search and proceeds to decode the amplitude and then to the next rise transition.

Algorithm 2 is the pseudo-code for the decoder implementation. The algorithm updates the decoded bits queue maintained by the receiver for each transmitter. The receiver parses **dataRx**, the output of sample and demodulate and identifies amplitude transitions (line 41 to 43). The demodulator outputs integer amplitudes and therefore the individual elements of dataRx are always integers. For each positive transition or rise, a matching fall is identified and the on-period and amplitude are decoded. The function **FindMatchingFall()** searches for a fall within *stopPos*, T_b period from *startPos*, start of the transition. The decoder first searches for a fall that matches the rise amplitude (lines 17 to 22). If more than one fall occurs within T_b, the decoder check each of the fall to find a valid on-period (line 18). The function **OnPeriodDecoder** checks if the estimated on-period is valid by verifying the difference between estimated on-period $\hat{T_{ON}}$ and on-periods assigned to senders (line 1 to 5). The fall whose corresponding on-period (w) is valid is decoded as the matching fall and passed on to the *AmplitudeDecoder*. The amplitude decoder is similar to an on-period decoder. It compares the amplitude difference between the current sample and previous sample ($\hat{a}$) with the amplitudes assigned to sender with the decoded on-period (Amplitude[$\hat{w}$]). If the decoded amplitude is valid, the decoder updates the amplitude and on-period of the received samples for the corresponding sender.

When no such matching fall is identified, the decoder searches for rise collisions i.e., the decoder checks if more than one signal collides and starts at the same time while still having different fall positions(lines 23 to 29). In this case, no single fall will match the rise amplitude, but, a combination of falls that correspond to the colliding rise signals will match with the rise observed. To identify this case, the decoder generates a array *sumsOfFall* of the sum of subsets of falls in the range dataRx[startPos:stopPos] considered. If signals collided only at rise, and have a distinct falls, the rise will find a matching entry in *sumsOfFall* (line 23). The decoder loops through each fall value corresponding to their summation (*fallCombinations*) and estimates the on-period, verifies it validity and decodes the amplitude.

If the decoder cannot find any matching entry for the rise in sumsOfFall, it repeats the steps to identify for fall collisions (lines 30 to 36). If more than one sender ends or stops transmitting their signal at the same time, while still beginning at different times, then the above cases return false. A combination of rise amplitudes then match with a single fall. The decoder follows the same steps as that of fall collision check to find rise collision check by replacing sumsOfFall by sumsOfRise. The decoder returns if all three cases fail, the decider returns without being able to detect or correct any error and moves to the next step.

All collisions except the following can be corrected by the above AWEC decoder architecture.

(1) Two senders with the same amplitude transmit bit 1 one after the other such that the fall of sender 1 collides with the rise of sender 2. If the sum of on-periods is closer to another on-period, such a collision will be decoded incorrectly.
(2) Two senders with same amplitude rise and fall within T_b. If $a_1 = a_2$ in Figure 3, the decoder cannot match the rise and fall using amplitudes.

4.3 Amplitude Decoding and Error Correction

The output of the on-period decoder provides the estimate of the decoded on-period $\hat{T_{ON}}$, the time of rise and fall of the received signal and the corresponding r_i. The *amplitude decoder* resolves the transmitted amplitudes with prior knowledge of the amplitudes assigned to senders, estimated channel error and the received amplitude r_i. For each on-period, the amplitude decoder stores the list of amplitude assigned. The amplitude decoder finds the amplitude assigned (example $\{1, 2, 4, 8\}$) that is closest to the received amplitude. On receiving an amplitude 5, the amplitude decoder output is set to 4 for the positions corresponding to rise and fall time that is output by the on-period decoder. The output of the amplitude decoder is thus the decoded samples for each sender.

We make use of randomness in the asynchronous transmissions in decoding signals from multiple senders. Since the senders are not time synchronized, the probability of k collisions derived in Equation 2 is further reduced by the random delay in the start of a message. Inter-period collision errors are corrected by the redundancy introduced in duty-cycle and the duty-cycle decoder design. Amplitude decoding is performed assuming that the received amplitude (for a decoded on-period) is from a single sender and therefore the amplitude decoder corrects for channel errors.

To this end, we have discussed the system constraints and challenges in implementing an error correction mechanism in an MC system. We have presented AWEC, a practical and easy to implement error correction mechanism that embeds redundancy in the characteristics of the signal.

5 PERFORMANCE EVALUATION

We built a python based Bacterial Communication Simulator (BCS) to evaluate the performance of the error correction techniques presented here. We implement OOK as modulation technique where every sender transmits bit 1 with a probability p_t. BCS implements the encoder and decoder presented in Sections 3 and 4 respectively. We implement the model the receiver response model proposed and experimentally validated in [1]. We implement the inverse of the above model perform sampling and demodulation.

Unless otherwise mentioned, we use the following parameters in the simulations. A uniformly distributed, bounded amplitude error that is proportional to the amplitude of the transmitted signal is added randomly to the transmitted signal i.e., an amplitude a_i after passing through the channel and receiver, is received as $a_i + \epsilon a_i$, where ϵ is the percentage of error introduced by the channel. We assume that the error percentage is the same across senders without loss of generalization. AWEC implementation does not change for varying amplitude errors. We consider a 20% channel error in our evaluations. A_{max} is the receiver saturation amplitude. Any amplitude $\geq A_{max}$ is received as A_{max} by the receiver bacteria.

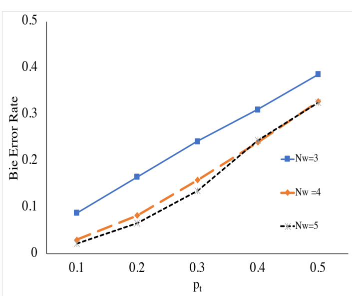


Figure 4: Bit Error Rate Performance of AWEC, N=15

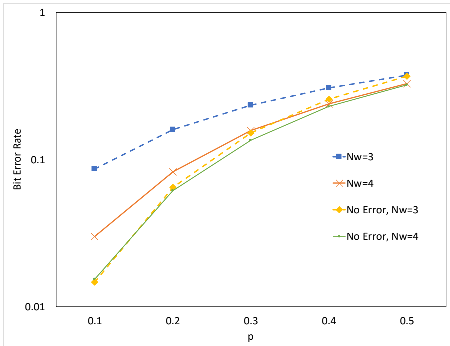
In Figure 4, we plot the bit error rate performance of AWEC as a function of p, the input load distribution for increasing values of N_w. It can be noted that the improvement in bit error rate does not increase linearly with N_w. When N_w reaches a threshold such that the intra-width collisions is very small, increasing it further does not add any value.

Figures 5(a) and 5(b) show the average bit error rate performance of the network AWEC in log-scale as a function of source load distribution p. We also plot the performance of AWEC with only collision error and zero amplitude errors due to channel and receiver error. The dotted lines in Figure 5(a) represent $N_w = 3$ and the solid lines represent $N_w = 4$. It can be noticed that as N_w increases, the performance of AWEC with error approaches *no error* indicating that AWEC can correct for channel and receiver induced errors. This is explained by the reduction in intra-width collisions

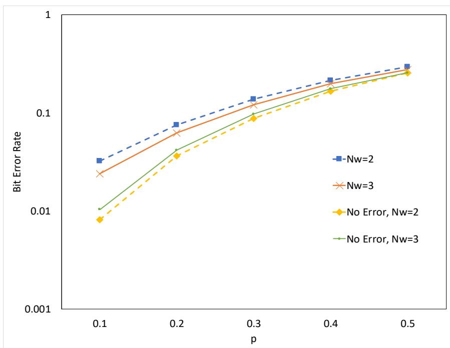
Algorithm 2 Decoder : On-period and Amplitude Estimation

width ← List of unique widths assigned
Amplitude[$\hat{w}$] ← List of unique amplitudes with width w
function OnPeriodDecoder(risePos,fallPos)
  $\hat{T_{ON}}$ ← **index**(fallPos) − **index**(risePos)
  closestWidth ← **min**(width − $\hat{T_{ON}}$)
  isValid(closestWidth)
end function
function AmplitudeDecoder(Amplitude[$\hat{w}$],$\hat{a}$)
  closestAmp ← **min**(Amplitude[$\hat{w}$] − $\hat{a}$)
  isValid(closestAmp)
end function
function FindMatchingFall(startPos,stopPos))
  diffRx ← **diff**(dataRx[startPos:stopPos])
  allRise ← diffRx[**where**($diffRx > 0$)]
  allFall ← diffRx[**where**($diffRx < 0$)]
  firstRise ← allRise[0], firstFall ← allFall[0]
  ▷ A distinct fall is found with T_b from firstRise
  if firstRise **in** allFall **then**
    for fallPos **in index**(allFall == firstRise) **do**
      OnPeriodDecoder(**index**(firstRise), fallPos)
      AmplitudeDecoder($Amplitude$[$\hat{w}$], $\hat{a}$)
    end for
    return
  ▷ Check for collisions at the rise of signals
  else if firstRise **in** sumsOfFall **then**
    fallCombinations ← **elements**(sumsOfFall)
    for fall **in** fallCombinations **do**
      OnPeriodDecoder(**index**(firstRise), fall)
      AmplitudeDecoder($Amplitude$[$\hat{w}$], $\hat{a}$)
    end for
    return
  ▷ Check for collisions at the fall of signals
  else if firstFall **in** sumsOfRise **then**
    riseCombinations ← **elements**(sumsOfRise)
    for rise **in** riseCombinations **do**
      OnPeriodDecoder(**index**(rise), firstFall)
      AmplitudeDecoder($Amplitude$[$\hat{w}$], $\hat{a}$)
    end for
    return
  else
    return
  end if
end function
▷ Main Function
for sample **in** dataRx **do**
  prevSample ← dataRx[**index**(sample) − 1]
  if sample − prevSample > 0 **then**
    startPos ← **index**(sample)
    stopPos ← startPos + T_b
    FindMatchingFall(startPos,stopPos)
  end if
end for

with increasing N_w. The higher the N_w, lesser is the number of senders with the same on-period. However, as N_w increases, the average throughput performance of a single sender compared to that of the maximum throughput using OOK decreases. As N_w increases, the bit-period to accommodate all on-periods increases,



(a) $N = 15$



(b) $N = 10$

Figure 5: Bit Error Rate Performance of AWEC

thus decreasing the overall throughput. Though by allowing multiple senders to transmit simultaneously, the network throughput is improved, individual throughput performance is traded off to improve bit error performance. We repeat the exercise for $N = 10$ in Figure 5(b). For the reduced number of senders, the minimum value of N_w is also reduced. For a fewer number of senders, the probability of collision is smaller and hence the overall bit error rate is smaller than $N = 15$. It can be noted that increasing N_w increases the minimum bit period T_b thus decreasing throughput. The absolute values of w_i is determined by the system parameters, w_e, T_{min} and N_w. Effective datarate achieved using AWEC is given by the ratio $\frac{T_{max}}{T_{min}}$. As p increases, collision errors dominate the overall error and hence increasing N_w has little improvement.

In both these cases, the error correction capability of AWEC decreases with increasing value of p_t i.e., the bit error rate increases with increasing p_t. This is attributed to collision errors that dominate overall errors at higher values of p_t. As discussed in Section 3, in a high latency, low complexity MC system, it is desirable to use codewords with low weight i.e., the choice of message encoding and modulation techniques such that the probability of bit 1 in the channel is very small. The higher the probability of collisions, higher is the value of N_w required to achieve a lower probability of intra-width collision. At $p_t = 0.1$, the bit error rate of AWEC is of the order of 10^{-2} for $N = 15, A_{max} = 45, N_w = 3$ and 10^{-3} for $N = 10, A_{max} = 45, N_w = 2$.

AWEC improves the bit error rate performance by an order of magnitude from simple ADMA by an order of magnitude. The improvement in the error correction capability is achieved at the cost of reduced throughput due to increased bit-period.

6 CONCLUSIONS AND FUTURE WORK

In this work, we developed EEC, an embedded forward error correction mechanism for super-slow, extremely-low complexity, multiple access molecular communication system. EEC embeds redundancy in the duty-cycle of the transmitted signal and achieves bit error of the order of 10^{-2}. We developed a heuristic encoder and decoder designs to reduce bit error. A theoretical analysis of the encoder and decoder designs is a part of our future work. The network throughput decreases due to the increased bit period. An in-depth analysis of the bit error rate throughput tradeoff is required to find the maximum achievable bit error rate for a given throughput. We plan to study the practical achievable values for T_{min}, N_w which in turn determine the achievable throughput.

REFERENCES

[1] Caitlin Austin, William Stoy, Peter Su, Marie Harber, Patrick Bardill, Brian Hammer, and Craig Forest. 2014. Modeling and validation of autoinducer-mediated bacterial gene expression in microfluidic environments. *Biomicrofluidics* (2014).

[2] Arash Einolghozati, Mohsen Sardari, Ahmad Beirami, and Faramarz Fekri. 2011. Capacity of discrete molecular diffusion channels. In *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on.* IEEE.

[3] D Ewen Cameron, Caleb J Bashor, and James J Collins. 2014. A brief history of synthetic biology. 12 (04 2014).

[4] Nariman Farsad, Na-Rae Kim, Andrew W Eckford, and Chan-Byoung Chae. 2014. Channel and noise models for nonlinear molecular communication systems. *IEEE Journal on Selected Areas in Communications* (2014).

[5] Taro Furubayashi, Tadashi Nakano, Andrew Eckford, and Tetsuya Yomo. [n. d.]. Reliable end-to-end molecular communication with packet replication and retransmission. In *Global Communications Conference (GLOBECOM), 2015 IEEE.*

[6] I. Gammoudi, H. Tarbague, A. Othmane, D. Moynet, D. RebiÃĺre, R. Kalfat, and C. Dejous. 2010. Love-wave bacteria-based sensor for the detection of heavy metal toxicity in liquid medium. *Biosensors and Bioelectronics* (2010). Selected Papers from the World Congress on Biosensors,Glasgow, Scotland, UK May 26-28, 2010.

[7] Matthew S Gast. [n. d.]. *802.11 ac: A Survival Guide: Wi-Fi at Gigabit and Beyond.*

[8] Peng He, Yuming Mao, Qiang Liu, and Kun Yang. 2016. Improving reliability performance of diffusion-based molecular communication with adaptive threshold variation algorithm. *International Journal of Communication Systems* (2016).

[9] Bo Højris, Sarah Christine Boesgaard Christensen, Hans-Jørgen Albrechtsen, Christian Smith, and Mathis Dahlqvist. 2016. A novel, optical, on-line bacteria sensor for monitoring drinking water quality. *Scientific reports* 6 (2016), 23935.

[10] http://www.sample6.com/. [n. d.]. *Sample6.*

[11] Evgeny Katz. 2013. *Biomolecular information processing: from logic systems to smart sensors and actuators.* John Wiley & Sons.

[12] Na-Rae Kim and Chan-Byoung Chae. 2013. Novel modulation techniques using isomers as messenger molecules for nano communication networks via diffusion. *IEEE Journal on Selected Areas in Communications* (2013).

[13] Bhuvana Krishnaswamy, Caitlin M Austin, J Patrick Bardill, Daniel Russakow, Gregory L Holst, Brian K Hammer, Craig R Forest, and Raghupathy Sivakumar. 2013. Time-elapse communication: Bacterial communication on a microfluidic chip. *Communications, IEEE Transactions on* (2013).

[14] Bhuvana Krishnaswamy, Yubing Jian, Caitlin M Austin, Jorge E Perdomo, Sagar C Patel, Brian K Hammer, Craig R Forest, and Raghupathy Sivakumar. 2018. ADMA: Amplitude-Division Multiple Access for Bacterial Communication Networks. *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* (2018).

[15] Bhuvana Krishnaswamy and Raghupathy Sivakumar. 2015. Source Addressing and Medium Access Control in Bacterial Communication Networks. In *Proceedings of the Second Annual International Conference on Nanoscale Computing and Communication.* ACM.

[16] Mehmet S Kuran, Huseyin Birkan Yilmaz, Tuna Tugcu, and Ian F Akyildiz. 2011. Modulation techniques for communication via diffusion in nanonetworks. In *2011 IEEE international conference on communications (ICC).* IEEE.

[17] Mark S Leeson and Matthew D Higgins. 2012. Forward error correction for molecular communications. *Nano Communication Networks* 3, 3 (2012), 161–167.

[18] Yinqing Li and Ron Weiss. 2017. *A Modular Approach to Building Complex Synthetic Circuits.* Springer New York.

[19] Massimiliano Pierobon and Ian F Akyildiz. 2013. Capacity of a diffusion-based molecular communication system with channel memory and molecular noise. *IEEE Transactions on Information Theory* 59, 2 (2013), 942–954.

[20] Po-Jen Shih, Chia-han Lee, and Ping-Cheng Yeh. 2012. Channel codes for mitigating intersymbol interference in diffusion-based molecular communications. In *Global Communications Conference (GLOBECOM), 2012 IEEE.* IEEE, 4228–4232.

[21] K. V. Srinivas, Andrew W. Eckford, and Raviraj S. Adve. 2012. Molecular Communication in Fluid Media: The Additive Inverse Gaussian Noise Channel. *IEEE Transactions on Information Theory* (2012).