ACS SyntheticBiology

Research Article

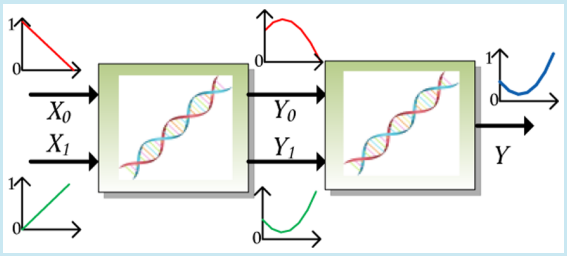
pubs.acs.org/synthbio

Chemical Reaction Networks for Computing Polynomials

Sayed Ahmad Salehi,* Keshab K. Parhi, and Marc D. Riedel

Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, Minnesota 55455, United States

ABSTRACT: Chemical reaction networks (CRNs) provide a fundamental model in the study of molecular systems. Widely used as formalism for the analysis of chemical and biochemical systems, CRNs have received renewed attention as a model for molecular computation. This paper demonstrates that, with a new encoding, CRNs can compute any set of polynomial functions subject only to the limitation that these functions must map the unit interval to itself. These polynomials can be expressed as linear combinations of Bernstein basis polynomials with positive coefficients less than or equal to 1. In the proposed encoding approach, each variable is represented using two molecular types: a type-0 and a type-1. The value is the ratio of the concentration of type-1 molecules to the sum of the concentrations of type-0 and type-1 molecules. The proposed encoding naturally exploits the expansion of a power-form polynomial into a Bernstein polynomial. Molecular encoders for converting any input in a standard representation to the fractional representation as well as decoders for converting the computed output from the fractional to a standard representation are presented. The method is illustrated first for generic CRNs; then, an example is mapped to DNA strand-displacement reactions.



KEYWORDS: *molecular computing, polynomials, DNA strand-displacement reaction, mass-action kinetics*

It has long been recognized that, viewed from a mathematical standpoint, a set of chemical reactions can exhibit rich dynamical behavior.[1] On the computational front, there has been a wealth of research into efficient methods for simulating chemical reactions, ranging from ordinary differential equations (ODEs)[2] to stochastic simulation.[3] On the mathematical front, entirely new branches of theory have been developed to characterize chemical dynamics.[4] The idea of computation directly with chemical reactions, as opposed to writing computer programs to analyze chemical systems, dates back to the seminal work of Adleman.[5] In this context, a chemical reaction network (CRN) transforms input concentrations of molecular types into output concentrations and thus implements computation. It should be noted that the equilibrium concentrations of the output molecules are considered as the computed output of the system.

The question of the computational power of chemical reactions has been considered by several authors. Magnasco demonstrated that chemical reactions can compute anything that digital circuits can compute.[6] Soloveichik et al. demonstrated that chemical reactions are Turing Universal, meaning that they can compute anything that a computer algorithm can compute.[7] This work was applicable to a discrete, stochastic model of chemical kinetics. The computation is probabilistic; the total probability of error of the computation can be made arbitrarily small (but not zero).

Either explicitly or implicitly, prior work has considered two types of encodings for the input and output variables of CRNs:[8,9]

1. The value of each variable corresponds to the concentration of a specific molecular type; we will call this the direct representation.
2. The value of each variable is represented by the difference between the concentrations of a pair of molecular types; we will call this the dual-rail representation.[9]

In this paper, we introduce a new representation that we call the fractional representation. A pair of molecular types is assigned to each variable, e.g., (X_0,X_1) for a variable x. The value of the variable is determined by the ratio

$$x = \frac{[X_1]}{[X_0] + [X_1]} \quad (1)$$

Evidently, the value is confined to the unit interval $[0, 1]$. The proposed encoding method is inspired by prior work in designing stochastic circuits.[10−12,15] Such circuits operate on randomized bit streams with the values of variables represented as the fraction of 1s versus 0s in the streams. In a sense, the main contribution of this paper is the application of this theory from stochastic circuit design to CRNs.

On the basis of the fractional representation in eq 1, we propose a CRN framework for computing univariate polynomials that map the unit interval $[0,1]$ to itself. We demonstrate that a CRN exists that computes any such polynomial. The full system consists of an encoder, the computation CRNs, and a decoder, as shown in Figure 1. The encoder converts the input molecular type, X (for $0 \le [X] \le 1$), into two molecular types, X_0 and X_1, such that

Received: September 24, 2015

DOI: 10.1021/acssynbio.5b00163
ACS Synth. Biol. XXXX, XXX, XXX−XXX

Figure 1. Whole system performing computation in fractional representation.

$$[X] = \frac{[X_1]}{[X_0] + [X_1]}$$

The decoder converts the ratio of two molecular types, Y_0 and Y_1, into a single molecular type, Y, as the final output such that

$$[Y] = \frac{[Y_1]}{[Y_0] + [Y_1]}$$

We describe the design of the encoder and decoder in the Encoding and Decoding section.

We first illustrate the computation CRN block with a simple example. Consider the following CRN

$$\begin{aligned} X_0 + X_0 &\rightarrow S_0 + X_0 + X_0 \\ X_0 + X_1 &\rightarrow 2S_1 + X_0 + X_1 \\ X_1 + X_1 &\rightarrow S_2 + X_1 + X_1 \end{aligned} \quad \text{(a)}$$

$$\begin{aligned} S_0 + B_{0,0} &\rightarrow Y_0 + B_{0,0} \\ S_0 + B_{0,1} &\rightarrow Y_1 + B_{0,1} \\ S_1 + B_{1,0} &\rightarrow Y_0 + B_{1,0} \\ S_1 + B_{1,1} &\rightarrow Y_1 + B_{1,1} \\ S_2 + B_{2,0} &\rightarrow Y_0 + B_{2,0} \\ S_2 + B_{2,1} &\rightarrow Y_1 + B_{2,1} \\ Y_0 &\rightarrow \varnothing \\ Y_1 &\rightarrow \varnothing \end{aligned} \quad \text{(b)}$$

Set the initial concentrations as

$$\left.\begin{aligned}[B_{0,0}] &= 0.25nM \\ [B_{0,1}] &= 0.75nM\end{aligned}\right\} \Rightarrow b_0 = \frac{[B_{0,1}]}{[B_{0,0}] + [B_{0,1}]} = \frac{0.75}{0.25 + 0.75} = \frac{3}{4}$$

$$\left.\begin{aligned}[B_{1,0}] &= 0.75nM \\ [B_{1,1}] &= 0.25nM\end{aligned}\right\} \Rightarrow b_1 = \frac{[B_{1,1}]}{[B_{1,0}] + [B_{1,1}]} = \frac{0.25}{0.75 + 0.25} = \frac{1}{4}$$

$$\left.\begin{aligned}[B_{2,0}] &= 0.50nM \\ [B_{2,1}] &= 0.50nM\end{aligned}\right\} \Rightarrow b_2 = \frac{[B_{2,1}]}{[B_{2,0}] + [B_{2,1}]} = \frac{0.50}{0.50 + 0.50} = \frac{1}{2}$$

Although not obvious, it may be shown that this CRN computes the function

$$y(x) = \frac{3}{4}x^2 - x + \frac{3}{4} \quad (2)$$

where $0 \leq x \leq 1$.

Note that any unit could have been used in this paper for the molecular concentrations; nM has been used due to the practical utility.

The CRN is composed of two sets of reactions: the three reactions in group (a) are referred as control generating reactions, and the six reactions in group (b) represent the transferring reactions. The control generating reactions generate the molecules that control the transferring reactions (similar to the way that the control bits select outputs from inputs with multiplexors in electronic circuits). However, the control molecules represent analog values and transfer inputs to outputs proportionally. We note that the transferring reactions are conceptually similar to the molecular reactions proposed in ref 13 for implementing Markov Chains.

We provide details regarding the synthesis method in the Synthesizing CRNs for Computing Polynomials section. Here, we simply note that, given a polynomial $y(x)$, the first step is to convert it to its Bernstein polynomial equivalent $g(x)$. For the polynomial $y(x)$ in eq 2

$$g(x) = \frac{3}{4}[(1 - x)^2] + \frac{1}{4}[2x(1 - x)] + \frac{1}{2}x^2 \quad (3)$$

(A discussion of the math behind this is given in the Proof Based on the Mass-Action Kinetics section.)

Note that the coefficients of the Bernstein polynomial correspond to the values of b_i for $i = 0,1,2$. These values are used to initialize the molecular types $B_{i,0}$ and $B_{i,1}$ for $i = 0,1,2$. In fact, computing with chemical reaction networks consists of two parts. First, choose a CRN as a means of building the dynamical system. Second, simulate a purposefully chosen dynamical system to equilibrium. By introducing the $B_{i,0}$ and $B_{i,1}$ species, the concentrations of which are time-invariant and fixed to what would have been rate constants, we propose changes to the first part that result in the same dynamical system simulated in the second part.

Suppose we want to evaluate $y(x)$ at $x = 0.5$. We would initialize $X_0 = X_1 = 0.5$ nM such that

$$x = \frac{[X_1]}{[X_0] + [X_1]} = 0.5 \quad (4)$$

We would set the initial concentration of the other types to zero. The control generating reactions use X_0 and X_1 to produce the control molecules S_0, S_1, and S_2, and transferring reactions use control molecules to compute the output. The output value, $y(x)$, is computed as the ratio of the final concentrations of Y_0 and Y_1, i.e.,

$$y(x) = \frac{[Y_1]}{[Y_0] + [Y_1]} \quad (5)$$

The simulation results for evaluating this example at $x = 0.5$ using a continuous mass-action kinetics model are shown in Figure 2. As time $t \rightarrow \infty$, the ratio

$$\frac{[Y_1(t)]}{[Y_0(t)] + [Y_1(t)]} \quad (6)$$

approaches the correct value of $y(0.5) = 0.4375$.

RESULTS AND DISCUSSION

Representation. In our method, the Bernstein representation of a polynomial is a key element. We briefly describe the relevant mathematics. The family of $n + 1$ polynomials of the form

$$B_{i,n}(x) = \binom{n}{i} x^i (1 - x)^{n-i}, \quad i = 0, \ldots, n \quad (7)$$

are called Bernstein basis polynomials of degree n. A linear combination of Bernstein basis polynomials of degree n,

DOI: 10.1021/acssynbio.5b00163
ACS Synth. Biol. XXXX, XXX, XXX−XXX

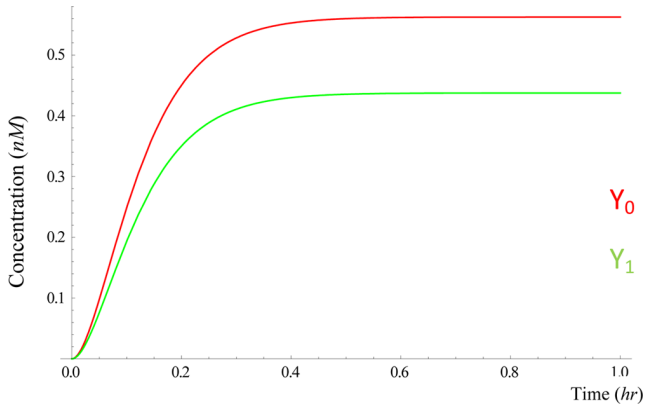


Figure 2. Simulation results for the CRN implementing the polynomial $y(x) = \frac{3}{4}x^2 - x + \frac{3}{4}$ at $x = 0.5$. These were obtained from an ODE simulation of the mass-action kinetics.

$$g(x) = \sum_{i=0}^{n} b_{i,n} B_{i,n}(x) \tag{8}$$

is a Bernstein polynomial of degree n. The $b_{i,n}$s are called Bernstein coefficients.

Polynomials are usually represented in power form, i.e.,

$$y(x) = \sum_{i=0}^{n} a_{i,n} x^i \tag{9}$$

We can convert such a power-form polynomial of degree n into a Bernstein polynomial of degree n. The conversion from the power-form coefficients, $a_{i,n}$, to the Bernstein coefficients, $b_{i,n}$, is a closed-form expression

$$b_{i,n} = \sum_{j=0}^{i} \frac{\binom{i}{j}}{\binom{n}{j}} a_{j,n}, \qquad 0 \le i \le n \tag{10}$$

For a proof of this, the reader is referred to ref 14.

Generally speaking, a power-form polynomial of degree n can be converted into an equivalent Bernstein polynomial of degree greater than or equal to n. The coefficients of a Bernstein polynomial of degree $m+1$ ($m \ge n$) can be derived from the Bernstein coefficients of an equivalent Bernstein polynomial of degree m as

$$b_{i,m+1} = \begin{cases} b_{0,m} & i = 0 \\ \left(1 - \frac{i}{m+1}\right) b_{i,m} + \frac{i}{m+1} b_{i-1,m} & 1 \le i \le m \\ b_{m,m} & i = m+1 \end{cases} \tag{11}$$

Again, for a proof, the reader is referred to ref 14.

By encoding the values of variables as the ratio of the concentrations of two molecular types,

$$x = \frac{[X_1]}{[X_0] + [X_1]}$$

we can only represent numbers between 0 and 1. Accordingly, our method synthesizes functions that map the unit interval $[0,1]$ onto itself. The method can also synthesize functions that map the unit interval to the negative unit interval $[-1,0]$. This computes the negative of a function that maps the unit interval to itself. As was shown in Example 1, the coefficients of the polynomials that we compute are also represented in this fractional form. Fortunately, Qian et al. proved that any polynomial that maps the unit interval onto the unit interval can be converted into a Bernstein polynomial with all coefficients in the unit interval.[15]

Synthesizing CRNs for Computing Polynomials. In this section, we present a systematic methodology for synthesizing CRNs that can compute polynomials. As discussed in the previous section, we assume that the target polynomial is given in Bernstein form with all coefficients in the unit interval. The method is composed of two parts, designing the CRN and initializing certain types to specific values, as discussed in the following section.

Designing the CRN. The CRN reactions consist of two sets of reactions that we call the control generating reactions and the transferring reactions.

First, consider the control generating reactions. When our proposed CRN is computing a polynomial of degree m, each control generating reaction should have m reactants. The reactions consist of all possible combinations of m molecules chosen from X_0 and X_1. These $(m+1)$ reactions are listed in eq 12. In the first reaction of eq 12, all reactants are chosen from molecules of X_0 and produce molecules of S_0. In the second, $(m-1)$ molecules of X_0 and one molecule of X_1 are combined to produce molecules of S_1. Similarly, the $(i+1)$st reaction contains i molecules of X_1 and $(m-i)$ molecules of X_0. The total number of possible reactions, as shown in eq 12, is $(m+1)$.

$$\begin{aligned} mX_0 &\rightarrow S_0 + mX_0 \\ X_1 + (m-1)X_0 &\rightarrow mS_1 + X_1 + (m-1)X_0 \\ 2X_1 + (m-2)X_0 &\rightarrow \binom{m}{2} S_2 + 2X_1 + (m-2)X_0 \\ &\vdots \\ iX_1 + (m-i)X_0 &\rightarrow \binom{m}{i} S_i + X_{0,1} + iX_1 + (m-i)X_0 \\ &\vdots \\ mX_1 &\rightarrow S_m + mX_1 \end{aligned} \tag{12}$$

A degree m Bernstein polynomial has $(m+1)$ Bernstein coefficients. We consider $(m+1)$ pairs of types $(B_{j,0}, B_{j,1})$ for $j = 0,1, \ldots, m$ to represent these coefficients. The transferring reactions produce the final output, Y_0 or Y_1, from the products of the control generating reactions, the S_js. They do so proportionally to the Bernstein coefficients. S_j goes to Y_0 if it combines with $B_{j,0}$ and goes to Y_1 if it combines with $B_{j,1}$. Accordingly, there are $2(m+1)$ transferring reactions as listed in eq 13.

DOI: 10.1021/acssynbio.5b00163
ACS Synth. Biol. XXXX, XXX, XXX−XXX

$$
\begin{aligned}
S_0 + B_{0,0} &\rightarrow Y_0 + B_{0,0}\\
S_0 + B_{0,1} &\rightarrow Y_1 + B_{0,1}\\
S_1 + B_{1,0} &\rightarrow Y_0 + B_{1,0}\\
S_1 + B_{1,1} &\rightarrow Y_1 + B_{1,1}\\
&\vdots\\
S_m + B_{m,0} &\rightarrow Y_0 + B_{m,0}\\
S_m + B_{m,1} &\rightarrow Y_1 + B_{m,1}\\
Y_0 &\rightarrow \varnothing\\
Y_1 &\rightarrow \varnothing
\end{aligned}
\tag{13}
$$

The number of required reactions for the implementation of a Bernstein polynomial of degree m is equal to $3m + 5$. We also need $3m + 7$ molecular types listed in Table 1.

Table 1. Number of Required Molecular Types in the Proposed CRN for a Polynomial of Degree m

represented molecular type	number of molecular types
X_0, X_1	2
S_j	$m + 1$
$B_{i,0}$, $B_{i,1}$	$2m + 2$
Y_0, Y_1	2
total	$3m + 7$

Initialization. We initialize the pair $(B_{j,0},B_{j,1})$ according to the Bernstein coefficients $b_{j,m}$, i.e., we have

$$
b_{j,m} = \frac{[B_{j,1}]}{[B_{j,0}] + [B_{j,1}]} \tag{14}
$$

For simplicity, we initialize $B_{j,0}$ and $B_{j,1}$ such that the sum $[B_{j,0}] + [B_{j,1}]$ is the same arbitrary value for all js. Call the sum $[B_{j,0}] + [B_{j,1}] = B$ for all js. In fact, we first calculate the values of Bernstein coefficients using eq 10 and then initialize $B_{j,1}$ and $B_{j,0}$ as $[B_{j,1}] = B \times b_{j,m}$ and $[B_{j,0}] = B-[B_{j,1}]$. (For the example in the introduction, we considered $B = 1$ nM.)

We initialize the corresponding molecular type in the input pair (X_0, X_1) based on the value x_{in} at which the polynomial is to be evaluated, i.e.,

$$
x_{in} = \frac{[X_1]}{[X_0] + [X_1]} \tag{15}
$$

All of the other intermediate types, i.e., the S_js as well as the output types Y_0 and Y_1, are initialized to zero.

Proof Based on the Mass-Action Kinetics. We use an ordinary differential model of the mass-action kinetics to prove the correctness of our proposed CRN design.

The control generating reactions (eq 12) produce type S_j whereas the transferring reactions (eq 13) consume them. Therefore, the ODEs for type S_j are

$$
\begin{aligned}
\frac{d[S_0]}{dt} &= [X_0]^m - [B_{0,0}][S_0] - [B_{0,1}][S_0]\\
&= [X_0]^m - [S_0]([B_{0,0}] + [B_{0,1}])\\
\frac{d[S_1]}{dt} &= m[X_0]^{m-1}[X_1] - [B_{1,0}][S_1] - [B_{1,1}][S_1]\\
&= m[X_0]^{m-1}[X_1] - [S_1]([B_{1,0}] + [B_{1,1}])\\
&\vdots\\
\frac{d[S_k]}{dt} &= \binom{m}{k}[X_0]^{m-k}[X_1]^k - [B_{k,0}][S_k] - [B_{k,1}][S_k]\\
&= \binom{m}{k}[X_0]^{m-k}[X_1]^k - [S_k]([B_{k,0}] + [B_{k,1}])\\
&\vdots\\
\frac{d[S_m]}{dt} &= [X_1]^m - [B_{m,0}][S_m] - [B_{m,1}][S_m]\\
&= [X_1]^m - [S_m]([B_{m,0}] + [B_{m,1}])
\end{aligned}
$$

At equilibrium, $\frac{d[S_j]}{dt} = 0$ for all js. Accordingly, we can compute the S_js as

$$
[S_j] = \frac{\binom{m}{j}[X_0]^{m-j}[X_1]^j}{[B_{j,0}] + [B_{j,1}]} \quad 0 \le j \le m \tag{16}
$$

Now, we write the ODEs for the output types Y_0 and Y_1. On the basis of the transferring reactions (eq 13), we have

$$
\begin{aligned}
\frac{d[Y_0]}{dt} &= [B_{0,0}][S_0] + [B_{1,0}][S_1] + \cdots + [B_{m,0}][S_m] - [Y_0]\\
\frac{d[Y_1]}{dt} &= [B_{0,1}][S_0] + [B_{1,1}][S_1] + \cdots + [B_{m,1}][S_m] - [Y_1]
\end{aligned}
\tag{17}
$$

At equilibrium, $\frac{d[Y_0]}{dt} = \frac{d[Y_1]}{dt} = 0$ and

$$
\begin{aligned}
[Y_0] &= [B_{0,0}][S_0] + [B_{1,0}][S_1] + \cdots + [B_{m,0}][S_m]\\
[Y_1] &= [B_{0,1}][S_0] + [B_{1,1}][S_1] + \cdots + [B_{m,1}][S_m]
\end{aligned}
\tag{18}
$$

According to the fractional encoding, the output value y is calculated as

$$
\begin{aligned}
y &= \frac{[Y_1]}{[Y_0] + [Y_1]}\\
&= \{[B_{0,1}][S_0] + [B_{1,1}][S_1] + \ldots + [B_{m,1}][S_m]\}\\
&\quad /\{([B_{0,0}][S_0] + [B_{1,0}][S_1] + \ldots + [B_{m,0}][S_m])\\
&\quad + ([B_{0,1}][S_0] + [B_{1,1}][S_1] + \cdots + [B_{m,1}][S_m])\}
\end{aligned}
\tag{19}
$$

With the assumption that $([B_{j,0}] + [B_{j,1}]) = B$ for all js, we have

DOI: 10.1021/acssynbio.5b00163
ACS Synth. Biol. XXXX, XXX, XXX−XXX

$$
\begin{aligned}
y &= \{[B_{0,1}][S_0] + [B_{1,1}][S_1] + \cdots + [B_{m,1}][S_m]\} \\
&\quad /\{([B_{0,0}] + [B_{0,1}])[S_0] + ([B_{1,0}] + [B_{1,1}])[S_1] + \cdots \\
&\quad + ([B_{m,0}] + [B_{m,1}])[S_m]\} \\
&= \frac{[B_{0,1}][S_0] + [B_{1,1}][S_1] + \cdots + [B_{m,1}][S_m]}{B([S_0] + [S_1] + \cdots + [S_m])} \\
&= \frac{\sum_{j=0}^{m} [B_{j,1}][S_j]}{B(\sum_{j=0}^{m} [S_j])}
\end{aligned} \tag{20}
$$

By substituting $[S_i]$ from eq 16,

$$
y = \frac{\sum_{j=0}^{m} [B_{j,1}] \frac{\binom{m}{j}[X_0]^{m-j}[X_1]^j}{B}}{B\left(\sum_{j=0}^{m} \frac{\binom{m}{j}[X_0]^{m-j}[X_1]^j}{B}\right)} \tag{21}
$$

We know that $\sum_{j=0}^{m} \binom{m}{j}[X_0]^{m-j}[X_1]^j = ([X_0] + [X_1])^m$ due to binomial theorem; therefore, the denominator can be replaced by $([X_0] + [X_1])^m$.

$$
\begin{aligned}
y &= \frac{\sum_{j=0}^{m} [B_{j,1}] \frac{\binom{m}{j}[X_0]^{m-j}[X_1]^j}{B}}{([X_0] + [X_1])^m} \\
&= \sum_{j=0}^{m} \frac{[B_{j,1}]}{B} \binom{m}{j} \frac{[X_0]^{m-j}[X_1]^j}{([X_0] + [X_1])^m} \\
&= \sum_{j=0}^{m} b_{j,m} \binom{m}{j} (1 - x)^{m-j} x^j
\end{aligned} \tag{22}
$$

Eq 22 is exactly the expression for a Bernstein polynomial representation of degree m for $y(x)$. Thus, this CRN computes $y(x)$. Note that y is finite because $0 \le [X_0] \le 1$ and $0 \le [X_1] \le 1$. Therefore, for every initial state of interest, our proposed CRN computes a stable equilibrium state.

Note that, in general, all the rate constants in our CRNs are assumed to be equal to each other. More precisely, on the basis of the proof, there are three categories of reactions with respect to the rate constants: the control generating reactions, the transferring reactions, and the last two annihilation reactions of the transferring reactions. All reactions in each of these categories are required to have the same rate constant.

Encoding and Decoding. Our proposed CRNs perform computations on the fractional representation in eq 1. In this section, we present chemical reactions that convert between this representation and a "direct representation", where the value of each variable is represented directly the concentration of a molecular type.

Encoding. Let a molecular type X denote the direct representation of the input value x and (X_0,X_1) denote the molecular pair for its fractional representation. Assume that the total concentration of X_0 and X_1 is 1 nM. Then, we have

$$
\left.\begin{aligned}
[X] &= \frac{[X_1]}{[X_0] + [X_1]} \\
[X_0] + [X_1] &= 1 \text{ nM}
\end{aligned}\right\} \Rightarrow \begin{cases} [X_1] = [X] \\ [X_0] = 1 - [X_1] \end{cases} \tag{23}
$$

Because the concentration values for X_1 and X are the same and subsequent stages do not consume them, type X can be directly used as type X_1 in the fractional representation.

For generating X_0, we must implement subtraction, which is a little tricky. We designed the following reactions (eq 24) for this task. T is initialized to 1 nM, and B is an intermediate molecular type with an initial value of zero.

$$
\begin{aligned}
T &\rightarrow X_0 + T \\
B + X_0 &\rightarrow \varnothing \\
X_1 &\rightarrow X_1 + B \\
X_0 &\rightarrow \varnothing
\end{aligned} \tag{24}
$$

For these reactions, the ODEs are

$$
\begin{aligned}
\frac{d[X_0]}{dt} &= [T] - [B][X_0] - [X_0] \\
\frac{d[B]}{dt} &= [X_1] - [B][X_0]
\end{aligned} \tag{25}
$$

and at equilibrium, we have

$$
\frac{d[X_0]}{dt} = 0 \Rightarrow [X_0] = [T] - [B][X_0] \tag{26}
$$

$$
\frac{d[B]}{dt} = 0 \Rightarrow [X_1] = [B][X_0] \tag{27}
$$

By substituting $[B][X_0]$ from eq 27 to 26, we have

$$
[X_0] = [T] - [X_1] \tag{28}
$$

Eq 28 is valid when $[T] \ge [X_1]$. Because $[X_0]$ cannot be negative, for $[T] \le [X_1]$, $[X_0] = 0$. Thus, the equilibrium ODE solution for these reactions is

$$
[X_0] = \begin{cases} [T] - [X_1] & \text{if } [T] \ge [X_1] \\ 0 & \text{if } [T] \le [X_1] \end{cases} \tag{29}
$$

If T is initialized to 1 nM, the reactions in 24 compute $[X_0] = 1 - [X_1]$.

Thus, the reactions in 24 encode the input concentration of X as a pair of concentrations (X_0,X_1) in a fractional representation. Here, in fact, X_1 can substitute for X, as discussed above. Note that the concentration of X_0 is initialized to zero at the outset.

Decoding. For the output of our molecular computing system, we convert the fractional representation back to a direct representation. If the fractional output is represented by the pair of molecules (Y_0,Y_1) and the direct output by Y, we have

$$
[Y] = \frac{[Y_1]}{[Y_0] + [Y_1]} \tag{30}
$$

In other words, we need to compute the summation of $[Y_0]$ and $[Y_1]$ and then the ratio of $[Y_1]$ over this summation. For this computation, we use the reactions proposed in ref 16. We will show that the reactions in 31 compute $[Y'] = [Y_0] + [Y_1]$

DOI: 10.1021/acssynbio.5b00163
ACS Synth. Biol. XXXX, XXX, XXX−XXX

and the reactions in 32 compute the final output $[Y] = \frac{[Y_1]}{[Y']} = \frac{[Y_1]}{[Y_0] + [Y_1]}$.

$$\begin{aligned} Y_0 &\rightarrow Y_0 + Y' \\ Y_1 &\rightarrow Y_1 + Y' \\ Y' &\rightarrow \varnothing \end{aligned} \tag{31}$$

$$\begin{aligned} Y_1 &\rightarrow Y_1 + Y \\ Y' + Y &\rightarrow Y' \end{aligned} \tag{32}$$

According to the ODEs of the reactions in 31, we have

$$\frac{d[Y']}{dt} = [Y_0] + [Y_1] - [Y']$$

and at equilibrium

$$\frac{d[Y']}{dt} = 0 \Rightarrow [Y'] = [Y_0] + [Y_1] \tag{33}$$

Similarly, for the reactions in 32, we have

$$\frac{d[Y]}{dt} = [Y_1] - [Y][Y']$$

and the equilibrium value of $[Y]$ is

$$\frac{d[Y]}{dt} = 0 \Rightarrow [Y] = \frac{[Y_1]}{[Y']} = \frac{[Y_1]}{[Y_0] + [Y_1]} \tag{34}$$

Therefore, the set of reactions in 31 and 32 implement the decoding of the output.

METHODS AND MATERIALS

DNA Implementation. The proposed CRN for computing polynomials is general in the sense that it can be implemented by any chemical or biochemical system with mass-action kinetics. As a practical medium, we choose DNA strand-displacement reactions. Indeed, Soleveichik et al. demonstrated that DNA strand-displacement reactions can emulate the kinetics of any CRN.[17] They presented a software tool that maps chemical CRNs to DNA reactions.

We illustrate with the following target function

$$y(x) = \frac{1}{4} + \frac{9}{8}x - \frac{15}{8}x^2 + \frac{5}{4}x^3 \tag{35}$$

The CRN includes reactions for the encoder, computation, and decoder parts. The Bernstein polynomial for $y(x)$ is

$$g(x) = \frac{2}{8}[(1-x)^3] + \frac{5}{8}[3x(1-x)^2] + \frac{3}{8}[3x^2(1-x)] + \frac{6}{8}x^3 \tag{36}$$

From the Bernstein coefficients, we initialize the types $(B_{i,0}, B_{i,1})$ for $i = 0,1,2,3$ as

$$\left.\begin{aligned}[B_{0,0}] &= 0.6\ \text{nM} \\ [B_{0,1}] &= 0.2\ \text{nM}\end{aligned}\right\} \Rightarrow \frac{0.2}{0.6 + 0.2} = \frac{2}{8}$$

$$\left.\begin{aligned}[B_{1,0}] &= 0.3\ \text{nM} \\ [B_{1,1}] &= 0.5\ \text{nM}\end{aligned}\right\} \Rightarrow \frac{0.5}{0.3 + 0.5} = \frac{5}{8}$$

$$\left.\begin{aligned}[B_{2,0}] &= 0.5\ \text{nM} \\ [B_{2,1}] &= 0.3\ \text{nM}\end{aligned}\right\} \Rightarrow \frac{0.3}{0.5 + 0.3} = \frac{3}{8}$$

$$\left.\begin{aligned}[B_{3,0}] &= 0.2\ \text{nM} \\ [B_{3,1}] &= 0.6\ \text{nM}\end{aligned}\right\} \Rightarrow \frac{0.6}{0.2 + 0.6} = \frac{6}{8} \tag{37}$$

We map our design to DNA strand-displacement reactions and evaluate it for 11 different input values between 0 and 1. The values of y computed by these CRNs are plotted against x and shown with the target polynomial $y(x)$ in Figure 3. Table 2 tabulates the computed values of $y(x)$ and the corresponding errors.

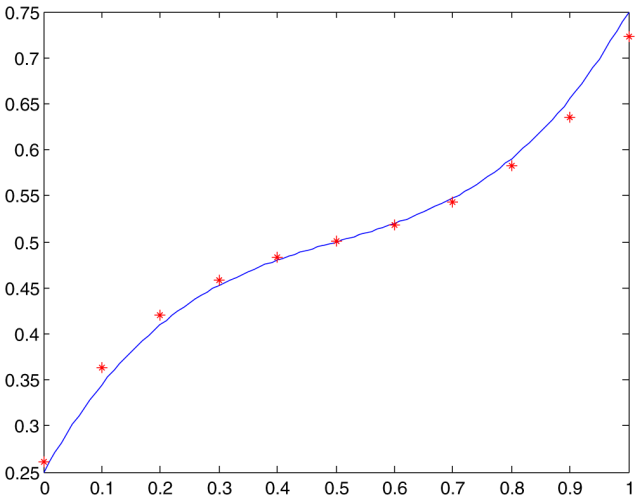


Figure 3. Values of $y(x)$ computed by a DNA implementation of proposed CRN. Blue line: target $y(x)$. Red stars: computed by DNA reactions.

For the DNA implementation, we used the parameters based on the examples in ref 17. The maximum strand displacement rate constant is $q_{max} = 10^6\ M^{-1}\ s^{-1}$, and the initial concentration of auxiliary complexes is set to $C_{max} = 10^{-5}$ M. If the concentration of auxiliary species, C_{max}, is much larger than the maximum concentration of other species (i.e., in proposed CRNs, $C_{max} \gg 1$ nM), then, as described in ref 17, we can assume that over the simulation time the auxiliary concentrations remain effectively constant. Therefore, DNA reactions correctly emulate the CRN independent of the auxiliary concentrations. Note that, for this assumption, the simulation time and reaction rates should not be very large values.[17] Although these requirements have been met in our simulations, errors exist.

As we describe later, the error stems from the fact that each molecular reaction is implemented by a sequence of DNA

DOI: 10.1021/acssynbio.5b00163
ACS Synth. Biol. XXXX, XXX, XXX−XXX

Table 2. Accuracy of a DNA Strand Displacement Implementation of a CRN Computing $y(x) = \frac{1}{4} + \frac{9}{8}x - \frac{15}{8}x^2 + \frac{5}{4}x^3$ Using the Proposed Method

x_{in}	computed $y(x)$	error (%)
0	0.261	4.4
0.1	0.3626	5
0.2	0.4207	2.5
0.3	0.4588	1.4
0.4	0.4838	0.8
0.5	0.5010	0.2
0.6	0.5180	0.4
0.7	0.5426	0.9
0.8	0.5823	1.3
0.9	0.6356	3
1	0.723	4

strand displacement reactions; the concentrations of auxiliary molecules, C_{max}, is bounded. In fact, if $C_{max} \to \infty$, the DNA simulation results converge to ODE simulation results. Further details concerning the analysis of errors when implementing CRNs with DNA strand displacement reactions, as well as a proof of convergence of a DNA implementation to the target CRN, can be found in the Supporting Information of refs 17 and 18.

Using the method presented in ref 17, each chemical reaction with m reactants and nonzero products can be emulated by $m + 1$ DNA strand displacement reactions. For example, bimolecular reactions are mapped to three DNA strand displacement reactions. To illustrate this, we present a sequence of DNA strand displacement reactions that are used to simulate a bimolecular reaction with three products.

As described in refs 17 and 18, three DNA reactions, $R1-R3$ shown in Figure 4, implement the molecular reaction $A + B \xrightarrow{k_i} A + B + C$. Unimolecular reactions without product, e.g., $Y \to \varnothing$, can be implemented by a single DNA strand displacement reaction. The DNA reaction shown in Figure 5 emulates the reaction $A \xrightarrow{k_i} \varnothing$. The toehold of strand A binds to its complementary part of gate molecule G and produces double strand W_1 and single strand W_2. Because W_1 and W_2 cannot bind together, the reaction is unidirectional.

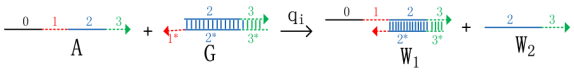


Figure 5. DNA strand displacement reaction that emulates reaction $A \xrightarrow{k_i} \varnothing$.

Table 3 summarizes the number of chemical and DNA strand displacement reactions for each group in our proposed method for computing the polynomial of degree m.

Table 3. Number of Chemical and DNA Strand-Displacement Reactions for Each Group of the Proposed CRN for Computation of a Bernstein Polynomial of Degree m

group of reactions	type of chemical reaction	number of chemical reactions	number of DNA reactions
control generating	reactions with m reactants	$m + 1$	$(m + 1) \times (m + 1)$
transferring	bimolecular	$2m + 2$	$(2m + 2) \times 3$
	unimolecular without product	2	2×1
total		$3m + 5$	$m^2 + 8m + 9$

CONCLUSIONS

We have introduced a new encoding for computation with CRNs: the value corresponding to each variable consists of the ratio of the concentration of a molecular type to the sum of two types. On the basis of this fractional representation, we proposed a method for computing arbitrary polynomials that map the unit interval [0,1] to itself or to [−1,0]. This is a rich class of functions.

Computation of polynomials with chemical kinetics has been attempted before by Buisman et al.[16] Compared to our method, their method requires fewer molecular types and fewer reactions (m molecular types and $3m$ molecular reactions for

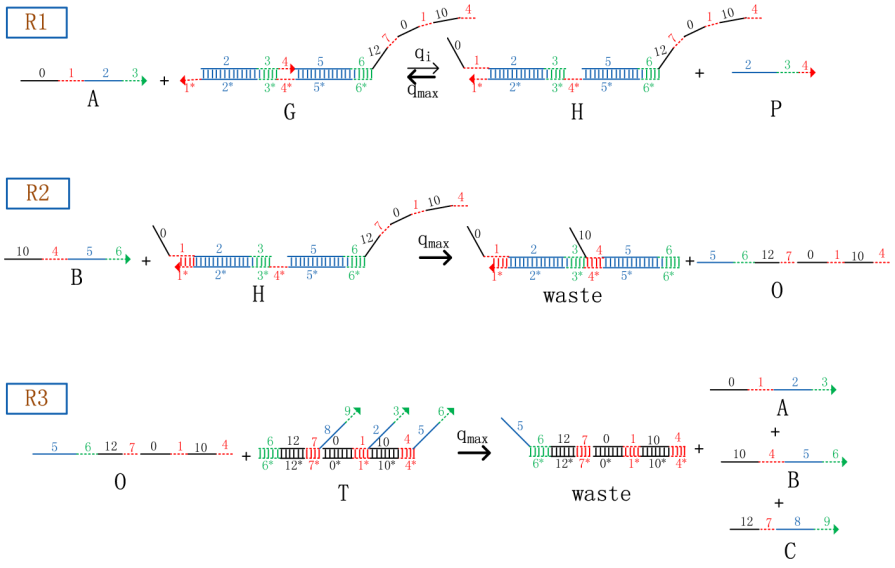


Figure 4. DNA strand displacement reactions that emulate reaction $A + B \xrightarrow{k_i} A + B + C$.

DOI: 10.1021/acssynbio.5b00163
ACS Synth. Biol. XXXX, XXX, XXX−XXX

a complete polynomial of degree m). However, unlike our approach, their CRNs are dependent on reaction rates. In fact, for each coefficient of the desired polynomial, they need a distinct reaction rate. This is unrealistic. Note that our approach only requires a single rate.

Soloveichik et al.,[7] as well as earlier work,[6,19,20] attempted to achieve Turing universality with chemical reactions. Although it is possible to compute polynomials with their CRNs, they did not provide a systematic framework for doing so.

The fractional representation that we propose is a non-standard representation. However, we note that it is similar to encodings found in nature. Many biological systems have species with two distinct states. For example, it is common for an enzyme to have active and inactive states. The ratio of the concentrations of the two states is a meaningful value. This is quite analogous to our representation.

Clearly, the primary interest of this work is theoretical. CRNs are a fundamental model of computation, abstract yet conforming to the physical behavior of chemical systems. Delineating the range of behaviors of such systems has intellectual merit. These results may also have practical applications.

Control theory has played a remarkable role in mathematical biology, providing a framework for modeling and designing the dynamic behavior of systems such as biological oscillators.[21−23] Polynomials play a central role in control and oscillation. In fact, the transfer function of a control system, which is the ratio of its output to its input in the Laplace domain, is the ratio of two polynomials, i.e., $H(z) = \frac{A(z)}{B(z)} = \frac{a_0 + a_1 z + \cdots + a_n z^n}{b_0 + b_1 z + \cdots + b_m z^m}$.[24] Nonlinear feedback in oscillators can be implemented by polynomials.[25,26]

Practitioners in synthetic biology are striving to create "embedded controllers", viruses and bacteria that are engineered to perform useful molecular computation in situ where needed, for instance, for drug delivery and biochemical sensing. Such embedded controllers may be called upon to perform computation such as filtering or signal processing. Computing polynomial functions is at the core of many of these computational tasks.

In future work, we will attempt to optimize the CRNs that we propose for computing polynomials, reducing the number of molecular types as well as the number of reactions. We will also attempt to generalize the method to compute a wider class of operations.

AUTHOR INFORMATION

Corresponding Author

*E-mail: saleh022@umn.edu.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

The authors gratefully acknowledge numerous constructive comments of the reviewers. This research is supported by the National Science Foundation Grant CCF-14234707.

REFERENCES

(1) Horn, F., and Jackson, R. (1972) General Mass Action Kinetics. *Arch. Ration. Mech. Anal. 47*, 81−116.

(2) Érdi, P., and Tóth, J. (1989) *Mathematical Models of Chemical Reactions: Theory and Applications of Deterministic and Stochastic Models*, Manchester University Press.

(3) Gillespie, D. (1977) Exact Stochastic Simulation of Coupled Chemical Reactions. *J. Phys. Chem. 81* (25), 2340−2361.

(4) Strogatz, S. (1994) *Nonlinear Dynamics and Chaos with Applications to Physics, Biology, Chemistry, and Engineering*, Perseus Books.

(5) Adleman, L. (1994) Molecular Computation of Solutions to Combinatorial Problems. *Science 266*, 1021−1024.

(6) Magnasco, M. O. (1997) Chemical Kinetics is Turing Universal. *Phys. Rev. Lett. 78* (6), 1190−1193.

(7) Soloveichik, D., Cook, M., Winfree, E., and Bruck, J. (2008) Computation with Finite Stochastic Chemical Reaction Networks. *Nat. Comput. 7* (4), 615−633.

(8) Chen, H., Doty, D., and Soloveichik, D. (2012) Deterministic Function Computation with Chemical Reaction Networks. *DNA Computing and Molecular Programming, LNCS*, Springer, Vol. *7433*, pp 24−42.

(9) Chen, H., Doty, D., and Soloveichik, D. (2014) Rate-Independent Computation in Continuous Chemical Reaction Networks. *Conference on Innovations in Theoretical Computer Science*, 313−326.

(10) Gaines, B. R. (1967) Stochastic Computing. In *Proceedings of AFIP spring join computer conference*, ACM, pp 149−156.

(11) Qian, W., and Riedel, M. D. (2008) The Synthesis of Robust Polynomial Arithmetic with Stochastic Logic. *Design Automation Conference*, 648−653.

(12) Qian, W., Li, X., Riedel, M. D., Bazargan, K., and Lilja, D. J. (2011) An Architecture for Fault-Tolerant Computation with Stochastic Logic. *IEEE Trans. Comput. 60* (No. 1), 93−105.

(13) Salehi, S. A., Riedel, M. D., and Parhi, K. K. (2015) Markov Chain Computations using Molecular Reactions. *Proc. IEEE International Conference on Digital Signal Processing (DSP)*, 689−693.

(14) Farouki, R., and Rajan, V. (1987) On the Numerical Condition of Polynomials in Bernstein Form. *Computer-Aided Geometric Design 4* (3), 191−216.

(15) Qian, W., Riedel, M. D., and Rosenberg, I. (2011) Uniform Approximation and Bernstein Polynomials with Coefficients in the Unit Interval. *European J. Comb. 32* (3), 448−463.

(16) Buisman, H. J., ten Eikelder, H. M. M., Hilbers, P. A. J., and Liekens, A. M. L. (2009) Computing Algebraic Functions with Biochemical Reaction Networks. *Artif. Life. 15* (1), 5−19.

(17) Soloveichik, D., Seelig, G., and Winfree, E. (2010) DNA as a Universal Substrate for Chemical Kinetics. *Proc. Natl. Acad. Sci. U. S. A. 107*, 5393−5398.

(18) Chen, Y., Dalchau, N., Srinivas, N., Phillips, A., Cardelli, L., Soloveichik, D., and Seelig, G. (2013) Programmable chemical controllers made from DNA. *Nat. Nanotechnol. 8*, 755−762.

(19) Liekens, A. M. L., and Fernando, C. T. (2007) Turing complete catalytic particle computers. *In Proceedings of Unconventional Computing Conference 4648*, 1202.

(20) Angluin, D., Aspnes, J., and Eisenstat, D. (2006) *Fast computation by population protocols with a leader. Technical Report YALEU/DCS/TR-1358*, Yale University Department of Computer Science.

(21) IMA Thematic Year on Control Theory and its Applications; *Workshop: Biological Systems and Networks*, November 16−20 (2015) http://www.ima.umn.edu/2015-2016/W11.16-20.15/abstracts.html.

(22) Chandra, F. A., Buzi, G., and Doyle, J. C. (2011) Glycolytic Oscillations and Limits on Robust Efficiency. *Science 333* (6039), 187−192.

(23) Iglesias, P. A., and Ingalls, B. P. (2010) *Control Theory and Systems Biology*, MIT Press.

(24) Dorf, R. C., and Bishop, R. H. (2001) *Modern Control Systems*, 9th ed., Prentice Hall.

(25) Stan, G., and Sepulchre, R. (2007) Analysis of interconnected oscillators by dissipativity theory. *IEEE Trans. Autom. Control 52* (2), 256−270.

(26) Agrawal, D. K., Franco, E., and Schulman, R. (2015) A self-regulating biomolecular comparator for processing oscillatory signals. *J. R. Soc., Interface 12* (111), 20150586.

DOI: 10.1021/acssynbio.5b00163
ACS Synth. Biol. XXXX, XXX, XXX−XXX