COMPUTATION WITH FINITE STOCHASTIC CHEMICAL REACTION NETWORKS

DAVID SOLOVEICHIK*, MATTHEW COOK†, ERIK WINFREE‡, AND JEHOSHUA BRUCK§

Abstract. A highly desired part of the synthetic biology toolbox is an embedded chemical microcontroller, capable of autonomously following a logic program specified by a set of instructions, and interacting with its cellular environment. Strategies for incorporating logic in aqueous chemistry have focused primarily on implementing components, such as logic gates, that are composed into larger circuits, with each logic gate in the circuit corresponding to one or more molecular species. With this paradigm, designing and producing new molecular species is necessary to perform larger computations. An alternative approach begins by noticing that chemical systems on the small scale are fundamentally discrete and stochastic. In particular, the exact molecular counts of each molecular species present, is an intrinsically available form of information. This might appear to be a very weak form of information, perhaps quite difficult for computations to utilize. Indeed, it has been shown that error-free Turing universal computation is impossible in this setting. Nevertheless, we show a design of a chemical computer that achieves fast and reliable Turing-universal computation using molecular counts. Our scheme uses only a small number of different molecular species to do computation of arbitrary complexity. The total probability of error of the computation can be made arbitrarily small (but not zero) by adjusting the initial molecular counts of certain species. While physical implementations would be difficult, these results demonstrate that molecular counts can be a useful form of information for small molecular systems such as those operating within cellular environments.

Key words. stochastic chemical kinetics; molecular counts; Turing-universal computation; probabilistic computation

1. Introduction. Many ways to perform computation in a chemical system have been explored in the literature, both as theoretical proposals and as practical implementations. The most common and, at present, successful attempts involve simulating Boolean circuits [1, 2, 3, 4]. In such cases, information is generally encoded in the high or low concentrations of various signaling molecules. Since each binary variable used during the computation requires its own signaling molecule, this makes creating large circuits onerous. Computation has also been suggested via a Turing machine (TM) simulation on a polymer [5, 6], via cellular automaton simulation in self-assembly [7], or via compartmentalization of molecules into membrane compartments [8, 9]. These approaches rely on the geometrical arrangement of a fixed set of parts to encode information. This allows unbounded computation to be performed by molecular systems containing only a limited set of types of enzyme and basic information-carrying molecular components. It had been widely assumed, but never proven, that these two paradigms encompassed all ways to do computation in chemistry: either the spatial arrangement and geometric structure of molecules is used, so that an arbitrary amount of information can be stored and manipulated, allowing Turing-universal computation; or a finite number of molecular species react in a well-mixed solution, so that each boolean signal is carried by the concentration of a

*Department of CNS, California Institute of Technology, Pasadena, CA, USA dsolov@caltech.edu
†Institute of Neuroinformatics, Zurich, Switzerland cook@ini.phys.ethz.ch
‡Departments of CNS and CS, California Institute of Technology, Pasadena, CA, USA winfree@caltech.edu
§Departments of CNS and EE, California Institute of Technology, Pasadena, CA, USA bruck@caltech.edu

dedicated species, and only finite circuit computations can be performed.

Here we show that this assumption is incorrect: well-mixed finite stochastic chemical reaction networks with a fixed number of species can perform Turing-universal computation with an arbitrarily low error probability. This result illuminates the computational power of stochastic chemical kinetics: error-free Turing universal computation is provably impossible, but once any non-zero probability of error is allowed, no matter how small, stochastic chemical reaction networks become Turing universal. This dichotomy implies that the question of whether a stochastic chemical system *can* eventually reach a certain state is always decidable, the question of whether this is *likely* to occur is uncomputable in general.

To achieve Turing universality, a system must not require a priori knowledge of how long the computation will be, or how much information will need to be stored. For instance, a system that maintains some fixed error probability per computational step cannot be Turing universal because after sufficiently many steps, the total error probability will become large enough to invalidate the computation. We avoid this problem by devising a reaction scheme in which the probability of error, according to stochastic chemical kinetics, is reduced at each step indefinitely. While the chance of error cannot be eliminated, it does not grow arbitrarily large with the length of the computation, and can in fact be made arbitrarily small without modifying any of the reactions but simply by increasing the initial molecular count of an "accuracy" species.

We view stochastic chemical kinetics as a model of computation in which information is stored and processed in the integer counts of molecules in a well-mixed solution, as discussed in [10] and [11] (see Sec. 5 for a comparison with our results). This type of information storage is effectively unary and thus it may seem inappropriate for fast computation. It is thus surprising that our construction achieves computation speed only polynomially slower than achievable by physical processes making use of spatial and geometrical structure. The total molecular count necessarily scales exponentially with the memory requirements of the entire computation. This is unavoidable if the memory requirements are allowed to grow while the number of species is bounded. However, for many problems of interest memory requirements may be small. Further, our scheme may be appropriate for certain problems naturally conceived as manipulation of molecular "counts", and may allow the implementation of such algorithms more directly than previously proposed. Likewise, engineering exquisite sensitivity of a cell to the environment may effectively require determining the exact intracellular molecular counts of the detectable species. Finally, it is possible that some natural processes can be better understood in terms of manipulating molecular counts as opposed to the prevailing regulatory circuits view.

The exponential trend in the complexity of engineered biochemical systems suggests that reaction networks on the scale of our construction may soon become feasible. The state of the art in synthetic biology progressed from the coupling of 2-3 genes in 2000 [12], to the implementation of over 100 deoxyribonuclease logic gates in vitro in 2006 [13]. Our construction is sufficiently simple that significant aspects of it may be implemented with the technology of synthetic biology of the near future.

2. Stochastic Model of Chemical Kinetics. The stochastic chemical reaction network (SCRN) model of chemical kinetics describes interactions involving integer number of molecules as Markov jump processes [14, 15, 16, 17]. It is used in domains where the traditional model of deterministic continuous mass action kinetics is invalid due to small molecular counts. When all molecular counts are large the model scales to the mass action law [18, 19]. Small molecular counts are prevalent in biology: for example, over 80% of the genes in the *E. coli* chromosome are expressed at fewer than a hundred copies per cell, with some key control factors present in quantities under a dozen [20, 21]. Experimental observations and computer simulations have confirmed that stochastic effects can be physiologically significant [22, 23, 24]. Consequently, the stochastic model is widely employed for modeling cellular processes (e.g. [25]) and is included in numerous software packages [26, 27, 28, 29]. The algorithms used for modeling stochastic kinetics are usually based on Gillespie's algorithm [30, 31, 32].

Consider a solution containing p species. Its state is a vector $\mathbf{z} \in \mathbb{N}^p$ (where $\mathbb{N} = \{0, 1, 2, \dots\}$) specifying the integral molecular counts of the species. A reaction α is a tuple $\langle \mathbf{l}, \mathbf{r}, k \rangle \in \mathbb{N}^p \times \mathbb{N}^p \times \mathbb{R}^+$ which specifies the stoichiometry of the reactants and products, and the rate constant k. We use capital letters to refer to the various species and standard chemical notation to describe a reaction (e.g. $A+C \xrightarrow{k} A+2B$). We write $\#_{\mathbf{z}} X$ to indicate the number of molecules of species X in state $\mathbf{z}$, omitting the subscript when the state is obvious. A *SCRN* $\mathcal{C}$ is a finite set of reactions. In state $\mathbf{z}$ a reaction α is possible if there are enough reactant molecules: $\forall i, \mathbf{z}_i - \mathbf{l}_i \geq 0$. The result of reaction α occurring in state $\mathbf{z}$ is to move the system into state $\mathbf{z} - \mathbf{l} + \mathbf{r}$. Given a fixed volume v and current state $\mathbf{z}$, the propensity of a unimolecular reaction $\alpha : X_i \xrightarrow{k} \dots$ is $\rho(\mathbf{z}, \alpha) = k\#_{\mathbf{z}} X_i$. The propensity of a bimolecular reaction $\alpha : X_i + X_j \xrightarrow{k} \dots$ where $X_i \neq X_j$ is $\rho(\mathbf{z}, \alpha) = k\frac{\#_{\mathbf{z}} X_i \#_{\mathbf{z}} X_j}{v}$. The propensity of a bimolecular reaction $\alpha : 2X_i \xrightarrow{k} \dots$ is $\rho(\mathbf{z}, \alpha) = \frac{k}{2}\frac{\#_{\mathbf{z}} X_i (\#_{\mathbf{z}} X_i - 1)}{v}$. Sometimes the model is extended to higher order reactions [15], but the merit of this is a matter of some controversy. We follow Gillespie and others and allow unary and bimolecular reactions only. The propensity function determines the kinetics of the system as follows. If the system is in state $\mathbf{z}$, no further reactions are possible if $\forall \alpha \in \mathcal{C}, \rho(\mathbf{z}, \alpha) = 0$. Otherwise, the time until the next reaction occurs is an exponential random variable with rate $\sum_{\alpha \in \mathcal{C}} \rho(\mathbf{z}, \alpha)$. The probability that next reaction will be a particular α_{next} is $\rho(\mathbf{z}, \alpha_{next}) / \sum_{\alpha \in \mathcal{C}} \rho(\mathbf{z}, \alpha)$.

While the model may be used to describe elementary chemical reactions, it is often used to specify higher level processes such as phosphorylation cascades, transcription, and genetic regulatory cascades, where complex multistep processes are approximated as single-step reactions. Molecules carrying mass and energy are assumed to be in abundant supply and are not modeled explicitly. This is the sense in which we use the model here because we allow reactions violating the conservation of energy and mass. While we will not find "atomic" reactions satisfying our proposed SCRNs, a reasonable approximation may be attained with complex organic molecules, assuming an implicit source of energy and raw materials. The existence of a formal SCRN with the given properties strongly suggests the existence of a real chemical system with the same properties. Thus, in order to implement various computations in real chemistry, first

we should be able to write down a set of chemical reactions (a SCRN), and then find a set of physical molecular species that behave accordingly. This approach is compatible with the philosophy of synthetic biology [4, 3]. Here we focus on the first step, reaction network design, and explore computation in SCRNs assuming arbitrary reactions can be used, and that they behave according to the above model of stochastic kinetics.

3. Time/Space-Bounded Algorithms. There is a rich literature on abstract models of computation that make use of integer counts, primarily because these are among the simplest Turing-universal machines known. Minsky's register machine (RM) [33] is the prototypical example. A RM is a finite state machine augmented with fixed number of registers that can each hold an arbitrary non-negative integer. An $inc(i, r, j)$ instruction specifies that when in state i, increment register r by 1, and move to state j. A $dec(i, r, j, k)$ instruction specifies that when in state i, decrement register r by 1 if it is nonzero and move to state j; otherwise, move to state k. There are two special states: start and halt. Computation initiates in the start state with the input count encoded in an input register, and the computation stops if the halt state is reached. The output is then taken to be encoded in the register values (e.g. the final value of the input register). While it may seem that a RM is a very weak model of computation, it is known that even two-register RMs are Turing-universal. Given any RM, our task is to come up with a SCRN that performs the same computation step by step. This SCRN is then said to simulate the RM.

For a given RM, we may construct a simple SCRN that simulates it with high probability as follows. We use a set of state species $\{S_i\}$, one for each state i of the RM, and set of register species $\{M_r\}$, one for each register. At any time there will be exactly one molecule of some species S_i corresponding to the current state i, and none of the other species S_j, for $j \neq i$. The molecular count of M_r represents the value of register r. For every $inc(i, r, j)$ instruction we add an *inc* reaction $S_i \to S_j + M_r$. For every $dec(i, r, j, k)$ instruction we add two reactions dec_1: $S_i + M_r \to S_j$ and dec_2: $S_i \to S_k$. We must ensure that a nonzero register decrements with high probability, which is the only source of error in this simulation. The probability of error per step is $\varepsilon = k_2/(k_1/v + k_2)$ in the worst case that the register holds the value 1, where k_1 is the rate constant for dec_1 and k_2 is the rate constant for dec_2. To decrease the error, we can increase k_1, decrease k_2, or decrease the volume v.

Decreasing the volume or changing the rate constants to modify the error rate is problematic. Changing the volume may be impossible (e.g. the volume is that of a cell). Further, a major assumption essential to maintain well-mixedness and justify the given kinetics is that the solution is dilute. The *finite density constraint* implies that the solution volume cannot be arbitrarily small and in fact must be at least proportional to the maximum molecular count attained during computation. Further, since developing new chemistry to perform larger computation is undesirable, improving the error rate of the chemical implementation of an RM without adjusting rate constants is essential.

In every construction to follow, the error probability is determined not by the volume or rate constants, but by the initial molecular count of an "accuracy species" which is easily changed.

A

	Rxn	Logical function
(*inc*)	$C + S_i \to S_j + M_r + C$	$inc(i,r,j)$
(*dec*$_1$)	$S_i + M_r \to S_j$	$dec(i,r,j,k)$
(*dec*$_2$)	$C_1 + S_i \to S_k + C_l$	

B

Rxn	Catalysts
$C_l \to C_{l-1}$	A^*
$C_{l-1} \to C_l$	A
$\vdots$	$\vdots$
$C_3 \to C_2$	A^*
$C_2 \to C_3$	A
$C_2 \to C_1$	A^*
$C_1 \to C_2$	A

FIG. 3.1. *(A) Bounded RM simulation. Species C ($\#C = 1$) acts a dummy catalyst to ensure that all reactions are bimolecular, simplifying the analysis of how the simulation scales with the volume. Initial molecular counts are: if $\hat{i}$ is the start state then $\#S_{\hat{i}} = 1$, $\#S_j = 0$ for $j \neq \hat{i}$, and $\#M_r$ is the initial value of register r. (B) Clock module for the RM and CTM simulations. Intuitively, the clock module maintains the average concentration of C_1 at approximately $(\#A^*)^l/(\#A)^{l-1}$. Initial molecular counts are: $\#C_l = 1$, $\#C_1 = \cdots = \#C_{l-1} = 0$. For the RM simulation $\#A^* = 1$, and $\#A = \Theta(1/\varepsilon^{1/(l-1)})$. In the RM simulation, A^* acts as a dummy catalyst to ensure that all reactions in the clock module are bimolecular and thus scale equivalently with the volume. This ensures that the error probability is independent of the volume. For the bounded CTM simulation, we use $\#A^* = \Theta((\frac{3^s ct}{s_{ct}})^{1/l})$, and $\#A = \Theta((\frac{1}{\varepsilon^{3/2}})^{1/(l-1)})$ (see Section A.3). Because constructions of Section 4 will require differing random walk lengths, we allow different values of l.*

In fact, we use exclusively bimolecular reactions[1] and all rate constants fixed at some arbitrary value k. Using exclusively bimolecular reactions simplifies the analysis of how the speed of the simulation scales with the volume and ensures that the error probability is independent of the volume. Further, working with the added restriction that all rate constants are equal forces us to design robust behavior that does not depend on the precise value of the rate constants.

We modify our first attempt at simulating an RM to allow the arbitrary decrease of error rates by increasing the initial molecular count of the accuracy species A. In the new construction, dec_2 is modified to take a molecule of a new species C_1 as reactant (see Fig 3.1(a)), so that decreasing the effective molecular count of C_1 is essentially equivalent to decreasing the rate constant of the original reaction. While we cannot arbitrarily decrease $\#C_1$ (at the bottom it is either 1 or 0), we can decrease the "average value" of $\#C_1$. Fig 3.1(b) shows a "clock module" that maintains the average value of $\#C_1$ at approximately $(1/\#A)^{l-1}$, where l is the length of the random walk in the clock module (see Lemma A.4 in the Appendix). Thus, to obtain error probability per step ε we use $\#A = \Theta(1/\varepsilon^{1/(l-1)})$ while keeping all rate constants fixed.[2]

How do we measure the speed of our simulation? We can make the simulation faster by decreasing the volume, finding a physical implementation with larger rate constants, or by increasing the error rate. Of course, there are limits to each of these: the volume may be set (i.e. operating in a cell), the chemistry is what's available, and, of course, the error cannot be increased too much or else computation is unreliable. As a function of the relevant parameters, the speed of the RM simulation is as given by the following theorem, whose proof is given in

[1] All unimolecular reactions can be turned into bimolecular reactions by adding a dummy catalyst.

[2] The asymptotic notation we use throughout this paper can be understood as follows. We write $f(x,y,\dots) = O(g(x,y,\dots))$ if $\exists c > 0$ such that $f(x,y,\dots) \leq c \cdot g(x,y,\dots)$ for all allowed values of $x, y, \dots$. The allowed range of the parameters will be given either explicitly, or implicitly (e.g. probabilities must be in the range $[0,1]$). Similarly, we write $f(x,y,\dots) = \Omega(g(x,y,\dots))$ if $\exists c > 0$ such that $f(x,y,\dots) \geq c \cdot g(x,y,\dots)$ for all allowed values of $x, y, \dots$. We say $f(x,y,\dots) = \Theta(g(x,y,\dots))$ if both $f(x,y,\dots) = O(g(x,y,\dots))$ and $f(x,y,\dots) = \Omega(g(x,y,\dots))$.

Section A.2.

THEOREM 3.1 (Bounded computation: RM simulation). *For any RM, there is an SCRN such that for any non-zero error probability δ, any input, and any bound on the number of RM steps t, there is an initial amount of the accuracy species A that allows simulation of the RM with cumulative error probability at most δ in expected time $O(\frac{vt^2}{k\delta})$, where v is the volume, and k is the rate constant.*

The major effort of the rest of this section is in speeding up the computation. The first problem is that while we are simulating an RM without much of a slowdown, the RM computation itself is very slow, at least when compared to a Turing machine (TM). For most algorithms t steps of a TM correspond to $\Omega(2^t)$ steps of a RM [33].[3] Thus, the first question is whether we can simulate a TM instead of the much slower RM? We achieve this in our next construction where we simulate an abstract machine called a clockwise TM (CTM)[34] which is only quadratically slower than a regular TM (Lemma A.9).

Our second question is whether it is possible to speed up computation by increasing the molecular counts of some species. After all, in bulk chemistry reactions can be sped up equivalently by decreasing the volume or increasing the amount of reactants. However, storing information in the exact molecular counts imposes a constraint since increasing the molecular counts to speed up the simulation would affect the information content. This issue is especially important if the volume is outside of our control (e.g. the volume is that of a cell).

A more essential reason for desiring a speedup with increasing molecular counts is the previously stated finite density constraint that the solution volume should be at least proportional to the maximum molecular count attained in the computation. Since information stored in molecular counts is unary, we require molecular counts exponential in the number of bits stored. Can we ensure that the speed increases with molecular counts enough to compensate for the volume that necessarily must increase as more information is stored?

We will show that the CTM can be simulated in such a manner that increasing the molecular counts of some species does not interfere with the logic of computation and yet yields a speedup. To get a sense of the speed-up possible, consider the reaction $X + Y \rightarrow Y + \ldots$ (i.e. Y is acting catalytically with products that don't concern us here) with both reactants initially having molecular counts m. This reaction completes (i.e. every molecule of X is used up) in expected time that scales with m as $O(\frac{\log m}{m})$ (Lemma A.5); intuitively, even though more X must be converted for larger m, this is an exponential decay process of X occurring at rate $O(\#Y) = O(m)$. Thus by increasing m we can speed it up almost linearly. By ensuring that all reactions in a step of the simulation are of this form, or complete just as quickly, we guarantee that by increasing m we can make the computation proceed faster. The almost linear speedup also adequately compensates for the volume increasing due to the finite density constraint.

For the purposes of this paper, a TM is a finite state machine augmented with a two-way infinite tape, with a single head pointing at the current bit on the tape. A TM instruction

[3] By the (extended) Church-Turing thesis, a TM, unlike a RM, is the best we can do, if we care only about super-polynomial distinctions in computing speed.

combines reading, writing, changing the state, and moving the head. Specifically, the instruction $op(i, j, k, z_j, z_k, D)$ specifies that if starting in state i, first read the current bit and change to either state j if it is 0 or state k if it is 1, overwrite the current bit with z_j or z_k respectively, and finally move the head left or right along the tape as indicated by the direction D. It is well known that a TM can be simulated by an "enhanced" RM in linear time if the operations of multiplication by a constant and division by a constant with remainder can also be done as one step operations. To do this, the content of the TM tape is represented in the binary expansion of two register values (one for the bits to the left of the head and one for the bits to the right, with low order bits representing tape symbols near the TM head, and high order bits representing symbols far from the head). Simulating the motion of the head involves division and multiplication by the number base (2 for a binary TM) of the respective registers because these operations correspond to shifting the bits right or left. In a SCRN, multiplication by 2 can be done by a reaction akin to $M \to 2M'$ catalyzed by a species of comparable number of molecules, which has the fast kinetics of the $X + Y \to Y + \ldots$ reaction above. However, performing division quickly enough seems difficult in a SCRN.[4] To avoid division, we use a variant of a TM defined as follows. A CTM is a TM-like automaton with a finite circular tape and instructions of the form $op(i, j, k, z_j, z_k)$. The instruction specifies behavior like a TM, except that the head always moves clockwise along the tape. Any TM with a two-way infinite tape using at most s_{tm} space and t_{tm} time can easily be converted to a clockwise TM using no more than $s_{ct} = 2s_{tm}$ space and $t_{ct} = O(t_{tm}s_{tm})$ time (Lemma A.9). The instruction $op(i, j, k, z_j, z_k)$ corresponds to: if starting in state i, first read the most significant digit and change to either state j if it is 0 or state k if it is 1, erase the most significant digit, shift all digits left via multiplying by the number base, and finally write a new least significant digit with the value z_j if the most significant digit was 0 or z_k if it was 1. Thus, instead of dividing to shift bits right, the circular tape allows arbitrary head movement using only the left bit shift operation (which corresponds to multiplication).

The reactions simulating a CTM are shown in Fig. 3.2. Tape contents are encoded in the base-3 digit expansion of $\#M$ using digit 1 to represent binary 0 and digit 2 to represent binary 1. This base-3 encoding ensures that reading the most significant bit is fast enough (see below). To read the most significant digit of $\#M$, it is compared with a known threshold quantity $\#T$ by the reaction $M + T \to \ldots$ (such that either T or M will be in sufficient excess, see below). We subdivide the CTM steps into microsteps for the purposes of our construction; there are four microsteps for a CTM step. The current state and microstate is indicated by which of the state species $\{S_{i,z}\}$ is present, with i indicating the state CTM finite control and $z \in \{1, 2, 3, 4\}$ indicating which of the four corresponding microstates we are in. The division into microsteps is necessary to prevent potentially conflicting reactions from occurring simultaneously as they are catalyzed by different state species and thus can occur only in different microsteps. Conflicting

[4]For example, the naive approach of dividing $\#M$ by 2 by doing $M+M \to M'$ takes $\Theta(1)$ time (independent of $\#M$) as a function of the initial amount of $\#M$. Note that the expected time for the last two remaining M's to react is a constant. Thus, if this were a step of our TM simulation we would not attain the desired speed up with increasing molecular count.

A

	Rxn	Catalysts	Logical function
State transitions	$C_1 \to C_l + I$	A^*	State transition initiation reaction
	$I + S_{i,4} \to S_{i,1}$ $S_{i,4} \to S_{i,1}$	$\emptyset$ $S_{i,1}$	State $(i,4) \to (i,1)$ transition
	$I + S_{i,1} \to S_{i,2}$ $S_{i,1} \to S_{i,2}$	$\emptyset$ $S_{i,2}$	State $(i,1) \to (i,2)$ transition
	$I + S_{i,2} \to S_{i,3}$ $S_{i,2} \to S_{i,3}$	$\emptyset$ $S_{i,3}$	State $(i,2) \to (i,3)$ transition
	$I \to I_T$ $I_T + S_{i,3} \to S_{j,4}$ $S_{i,3} \to S_{j,4}$	T $\emptyset$ $S_{j,4}$	State $(i,3) \to (j,4)$ transition if high order bit is 0
	$I \to I_M$ $I_M + S_{i,3} \to S_{k,4}$ $S_{i,3} \to S_{k,4}$	M $\emptyset$ $S_{k,4}$	State $(i,3) \to (k,4)$ transition if high order bit is 1
Memory operations	$M^\dagger \to M$	$S_{i,2}$	Restore memory
	$T^\dagger \to T$ $T + M \to T^* + M^*$	$S_{i,2}$ $\emptyset$	Compare memory to threshold
	$D^* \to D$ $B^* \to B$	$S_{i,2}$ $S_{i,2}$	Restore D and B
	$T^* \to T^\dagger$ $M^* \to M$	$S_{j,4}$ or $S_{k,4}$ $S_{j,4}$ or $S_{k,4}$	Restore memory and threshold
	$D \to K_1 + D^*$ $D \to K_2 + D^*$ $K_2 + M \to K_1$ $K_1 + M \to \emptyset$	$S_{j,4}$ $S_{k,4}$ $\emptyset$ $\emptyset$	Zero out high order bit
	$B \to (z_j + 1)M + B^*$ $B \to (z_k + 1)M + B^*$	$S_{j,4}$ $S_{k,4}$	Write new low order bit ($z_j, z_k \in \{0,1\}$)
	$M \to 3M^\dagger$	$S_{j,4}$ or $S_{k,4}$	Shift digits left

B

Initial molecular counts	Base 3 representation						
$\#M = (b_1+1)\cdot 3^{s-1} + (b_2+1)\cdot 3^{s-2} + \ldots$ $\cdots + (b_{s-1}+1)\cdot 3 + (b_s+1)$	(b_1+1)	$+(b_2+1)$	$+(b_3+1)$	$+(b_4+1)$	$+(b_5+1)$	$\ldots+$	(b_s+1)
$\#T = 2\cdot 3^{s-1} + 2\cdot 3^{s-3}$	2	0	2	0	0	…	0
$\#D = \#S_{\hat{i},1} = 3^{s-1}$	1	0	0	0	0	…	0
$\#B = 1$	0	0	0	0	0	…	1
				s			

FIG. 3.2. *Bounded CTM simulation: reactions and initial molecular counts. (A) Reactions for* $op(i,j,k,z_j,z_k)$. *The clock module is the same as for the RM simulation (Fig. 3.1(B)). Here* $\emptyset$ *indicates "nothing". (B) Letting* $s = s_{ct}$, *initial molecular counts for binary input* $b_1 b_2 \ldots b_s$. *Input is padded with zeros to be exactly* s *bits long. Here* $\hat{i}$ *is the start state of the CTM. All species not shown start at* 0.

reactions are separated by at least two microsteps since during the transition between two microsteps there is a time when both state species are present. A self-catalysis chain reaction is used to move from one microstep to the next. The transition is initiated by a reaction of a state species with a clock molecule C_1 to form the state species corresponding to the next microstep.

Now with $m = 3^{s_{ct}-1}$, Lemmas A.5–A.7 guarantee that all reactions in a microstep (excluding state transition initiation reactions) complete in expected time $O(\frac{v \log m}{km}) = O(\frac{v s_{ct}}{k 3^{s_{ct}}})$. Specifically, Lemma A.5 ensures that the memory operation reactions having a state species as

a catalyst complete in the required time. Lemma A.7 does the same for the self-catalytic state transition reactions. Finally, ensuring that either M or T is in excess of the other by $\Theta(m)$ allows us to use Lemma A.6 to prove that the reading of the most significant bit occurs quickly enough. The separation of $\#M$ or $\#T$ is established by using values of $\#M$ expressed in base 3 using just the digits 1 and 2. Then the threshold value $\#T$ as shown in Fig. 3.2 is $\Theta(3^{s_{ct}})$ larger than the largest possible s_{ct}-digit value of $\#M$ starting with 1 (base-3) and $\Theta(3^{s_{ct}})$ smaller than the smallest possible s_{ct}-digit value of $\#M$ starting with 2 (base-3), implying that either T or M will be in sufficient excess.

The only source of error is if not all reactions in a microstep finish before a state transition initiation reaction occurs. This error is controlled in an analogous manner to the RM simulation: state transition initiation reactions work on the same principle as the delayed dec_2 reaction of the RM simulation. We adjust $\#A$ so that all reactions in a microstep have a chance to finish before the system transitions to the next microstep (see Section A.3).

Since as a function of s_{ct}, the reactions constituting a microstep the CTM simulation finish in expected time $O(\frac{vs_{ct}}{k3^{s_{ct}}})$, by increasing s_{ct} via padding of the CTM tape with extra bits we can decrease exponentially the amount of time we need to allocate for each microstep. This exponential speedup is only slightly dampened by the increase in the number of CTM steps corresponding to a single step of the TM (making the worst case assumption that the padded bits must be traversed on every step of the TM, Lemma A.9).

In total we obtain the following result (see Section A.3). It shows that we can simulate a TM with only a polynomial slowdown, and that computation can be sped up by increasing the molecular count of some species through a "padding parameter" Δ.

THEOREM 3.2 (Bounded computation: TM simulation). *For any TM, there is an SCRN such that for any non-zero error probability δ, any amount of padding Δ, any input, any bound on the number of TM steps t_{tm}, and any bound on TM space usage s_{tm}, there is an initial amount of the accuracy species A that allows simulation of the TM with cumulative error probability at most δ in expected time $O(\frac{v(s_{tm}+\Delta)^{7/2}t_{tm}^{5/2}}{k\,3^{(2s_{tm}+\Delta)}\delta^{3/2}})$, where v is the volume, and k is the rate constant.*

Under realistic conditions relating v, s_{tm}, and t_{tm}, this theorem implies that the SCRN simulates the TM in polynomial time, specifically $O(t_{tm}^6)$. The finite density constraint introduced earlier requires that the solution volume be proportional to the maximum molecular count attained in the course of the computation. This constraint limits the speed of the simulation: there is a minimum volume to implement a particular computation, and if the volume is larger than necessary, the finite density constraint bounds Δ. In most cases, the total molecular count will be dominated by $3^{2s_{tm}+\Delta}$ (see Sec. A.3). Thus the maximum allowed padding satisfies $3^{2s_{tm}+\Delta} = \Theta(v)$, yielding total expected computation time $O(\frac{(\log v)^{7/2}t_{tm}^{5/2}}{k\,\delta^{3/2}})$. This implies that although Δ cannot be used to speed up computation arbitrarily, it can be used to minimize the effect of having a volume much larger than necessary since increasing the volume decreases the speed of computation poly-logarithmically only. Alternatively, if we can decrease the volume as long as the maximum density is bounded by some constant, then the best speed

is obtained with zero padding and the smallest v possible: $v = \Theta(3^{2s_{tm}})$. Then the total computation time is $O(\frac{s_{tm}^{7/2} t_{tm}^{5/2}}{k\delta^{3/2}})$. Since we can always ensure $s_{tm} \leq t_{tm}$, we experience at most a polynomial (6^{th} order) slowdown overall compared with a regular error-free TM.

4. Unbounded Algorithms. The above simulations are not Turing-universal because they incur a fixed probability of error per step of the computation. Since the probability of correct computation decreases exponentially with the number of steps, only finite computation may be performed reliably. Additionally the TM simulation has the property that the tape size must be specified a priori. We now prove that a fixed SCRN can be Turing-universal with an arbitrarily small probability of error over an unlimited number of steps. In the course of a computation that is not a priori bounded, in addition to stirring faster and injecting energy and raw materials, the volume needs to grow at least linearly with the total molecular count to maintain finite density. Therefore, in this section our model is that the volume dynamically changes linearly with the total molecular count as the system evolves over time. We desire that the total error probability over arbitrarily long computation does not exceed δ and can be set by increasing the initial molecular counts of the accuracy species A.

We now sketch how to modify our constructions to allow Turing-universal computation. Consider the RM simulation first. We can achieve a bounded total probability of error over an unbounded number of steps by sufficiently decreasing the probability of error in each subsequent error-prone step. Only *dec* steps when the register is non-zero are error-prone. Further, if dec_2 occurs then either the register value was zero and no error was possible, or an error has just occurred and there is no need decreasing the error further. Therefore it is sufficient to decrease the probability of error after each dec_1 step by producing A as a product of dec_1. If the clock Markov chain length is $l = 3$, then adding a single molecule of A as a product of every dec_1 reaction is enough: the total probability of error obtained via Lemma A.4 is $O(\sum_{\#A=i_0}^{\infty} 1/\#A^2)$; since this sum converges, the error probability over all time can be bounded by any desired $\delta > 0$ by making the initial number of A's, i_0, sufficiently large. Using $l = 3$ is best because $l > 3$ unnecessarily slows down the simulation. The total expected computation time is then $O(t(1/\delta+t)^2(1/\delta+t+s_0)/k)$, where s_0 is the sum of the initial register counts (see Section A.4).

A similar approach can be taken with respect to the TM simulation. The added difficulty is that the tape size must no longer be fixed, but must grow as needed. This can be achieved if the SCRN triples the molecular count of the state species, M, T, D, and P whenever the tape needs to increase by an extra bit. However, simply increasing $\#A$ by 1 per microstep without changing $\#A^*$ as in the RM construction does not work since the volume may triple in a CTM step. Then the clock would experience an exponentially increasing expected time. To solve this problem, in Sec. A.5 we show that if the SCRN triples the amount of A and A^* whenever extending the tape and increases $\#A$ by an appropriate amount, $\Theta(3^{s_{ct}})$, on every step then it achieves a bounded error probability over all time and yields the running time claimed in Theorem 4.1 below. The clock Markov chain of length $l = 5$ is used. All the extra operations can be implemented by reactions similar to the types of reactions already implementing the CTM simulation (Fig. 3.2). For example, tripling A can be done by reactions akin to $A \rightarrow A^{\dagger}$

and $A^\dagger \to 3A$ catalyzed by different state species in two non-consequitive microsteps.[5]

THEOREM 4.1 (Turing-universal computation). *For any TM, there is an SCRN such that for any non-zero error probability δ, and any bound s_{tm0} on the size of the input, there is an initial amount of the accuracy species A that allows simulation of the TM on inputs of size at most s_{tm0} with cumulative error probability at most δ over an unbounded number of steps and allowing unbounded space usage. Moreover, in the model where the volume grows dynamically in proportion with the total molecular count, t_{tm} steps of the TM complete in expected time (conditional on the computation being correct) of $O((1/\delta + s_{tm0} + t_{tm}s_{tm})^5 t_{tm}s_{tm}/k)$ where s_{tm} is the space used by the TM, and k is the rate constant.*

For $s_{tm} \approx t_{tm}$ this gives a polynomial time (12th order) simulation of TMs. This slowdown relative to Theorem 3.2 is due to our method of slowing down the clock to reduce errors.

Can SCRNs achieve Turing-universal computation without error? Can we ask for a guarantee that the system will eventually output a correct answer with probability 1?[6] Some simple computations are indeed possible with this strong guarantee, but it turns out that for general computations this is impossible. Intuitively, when storing information in molecular counts, the system can never be sure it has detected all the molecules present, and thus must decide to produce an output at some point without being certain. Formally, a theorem due to Karp and Miller [35] when adapted to the SCRN context (see Section A.6) rules out the possibility of error-free Turing universal computation altogether if the state of the TM head can be determined by the presence or absence (or threshold quantities) of certain species (i.e. state species in our constructions). Here recall that in computer science a question is called decidable if there is an algorithm (equivalently TM) that solves it in all cases. (Recall a state of a SCRN is a vector of molecular counts of each of the species. Below operator $\geq$ indicates element-wise comparison.)

THEOREM 4.2. *For any SCRN, given two states $\mathbf{x}$ and $\mathbf{y}$, the question of whether any state $\mathbf{y}' \geq \mathbf{y}$ is reachable from $\mathbf{x}$ is decidable.*

How does this theorem imply that error-free Turing universal computation is impossible? Since all the constructions in this paper rely on probabilities we need to rule out more clever constructions. First recall that a question is undecidable if one can prove that there can be no algorithm that solves it correctly in all cases; the classic undecidable problem is the halting problem: determine whether or not a given TM will eventually halt [36]. Now suppose by way of contradiction that someone claims to have an errorless way of simulating any TM in a SCRN. Say it is claimed that if the TM halts then the state species corresponding to the halt state is produced with non-zero probability (this is weaker than requiring probability 1), while if the TM never halts then the halt state species cannot be produced. Now note that by asking whether a state with a molecule of the halting species is reachable from the initial

[5] A slight modification of the clock module is necessary to maintain the desired behavior. Because of the need of intermediate species (e.g. $A^\dagger$) for tripling $\#A$ and $\#A^*$, the clock reactions need to be catalyzed by the appropriate intermediate species in addition to A and A^*.

[6] Since in a reaction might simply not be chosen for an arbitrarily long time (although the odds of this happening decrease exponentially), we can't insist on a zero probability of error at any fixed time.

state, we can determine whether the TM halts: if such a state is reachable then there must be a finite sequence of reactions leading to it, implying that the probability of producing a halt state species is greater than 0; otherwise, if such a state is not reachable, the halt state species can never be produced. This is equivalent to asking whether we can reach any $\mathbf{y}' \geq \mathbf{y}$ from the initial state of the SCRN, where $\mathbf{y}$ is the all zero vector with a one in the position of the halting species – a question that we know is always computable, thanks to Karp and Miller. Thus if an errorless way of simulating TMs existed, we would violate the undecidability of the halting problem.

Finally note that our Turing-universality results imply that the their long-term behavior of SCRNs is unknowable in a probabilistic sense. Specifically, our results imply that the question of whether a given SCRN, starting with a given initial state $\mathbf{x}$, produces a molecule of a given species with high or low probability is in general undecidable. This can be shown using a similar argument: if the question were decidable the halting problem could be solved by encoding a TM using our construction, and asking whether the SCRN eventually produces a molecule of the halting state species.

5. Discussion. We show that computation on molecular counts in the SCRN model of stochastic chemical kinetics can be fast, in the sense of being only polynomially slower than a TM, and accurate, in the sense that the cumulative error probability can be made arbitrarily small. Since the simulated TM can be universal [36], a single set of species and chemical reactions can perform any computation that can be performed on any computer. The error probability can be manipulated by changing the molecular count of an accuracy species, rather than changing the underlying chemistry. Further, we show that computation that is not a priori bounded in terms of time and space usage can be performed assuming that the volume of the solution expands to accommodate the increase in the total molecular count. In other words SCRNs are Turing universal.

The Turing-universality of SCRNs implies that the question of whether given a start state the system is *likely* to produce a molecule of a given species is in general undecidable. This is contrasted with questions of possibility rather than probability: whether a certain molecule *could* be produced is always decidable.

Our results may imply certain bounds on the speed of stochastic simulation algorithms (such as variants of τ-leaping [32]), suggesting an area of further study. The intuition is as follows: it is well known by the time hierarchy theorem [36] that certain TMs cannot be effectively sped up (it is impossible to build a TM that has the same input/output relationship but computes much faster). This is believed to be true even allowing some probability of error [37]. Since a TM can be encoded in an SCRN, if the behavior of the SCRN could be simulated very quickly, then the behavior of the TM would also be determined quickly, which would raise a contradiction.

Our results were optimized for clarity rather than performance. In certain cases our running time bounds can probably be significantly improved (e.g. in a number of places we bound additive terms $O(x + y)$, where $x \geq 1$ and $y \geq 1$, by multiplicative terms $O(xy)$). Also the roles of a number of species can be performed by a single species (e.g. A^* and C in the RM

simulation).

A number of previous works have attempted to achieve Turing universality with chemical kinetics. However, most proposed schemes require increasing the variety of molecular species (rather than only increasing molecular counts) to perform larger computation (e.g. [38] which shows finite circuit computation and not Turing universal computation despite its title). Liekens and Fernando [10] have considered computation in stochastic chemistry in which computation is performed on molecular counts. Specifically, they discuss how SCRNs can simulate RMs. However, they rely on the manipulation of rate constants to attain the desired error probability per step. Further, they do not achieve Turing-universal computation, as the prior knowledge of the length of the computation is required to set the rate constants appropriately to obtained a desired total error probability. While writing this manuscript, the work of Angluin et al [11] in distributed computing and multi-agent systems came to our attention. Based on the formal relation between their field and our field, one concludes that their results imply that stochastic chemical reaction networks can simulate a TM with a polynomial slowdown (a result akin to our Theorem 3.2). Compared to our result, their method allows to attain a better polynomial (lower degree), and much better dependence on the allowed error probability (e.g. to decrease the error by a factor of 10, we have to slow down the system by a factor of $10^{3/2}$, while an implementation based on their results only has to slow down by a factor polynomial in $\log 10$). However, because we focus on molecular interactions rather than the theory of distributed computing, and measure physical time for reaction kinetics rather than just the number of interactions, our results take into account the solution volume and the consequences of the finite density constraint (Section 3). Further, while they consider only finite algorithms, we demonstrate Turing universality by discussing a way of simulating algorithms unbounded in time and space use (Section 4). Finally, our construction is simpler in the sense that it requires far fewer reactions. The relative simplicity of our system makes implementing Turing-universal chemical reactions a plausible and important goal for synthetic biology.

Appendix A. Proof Details.

A.1. Clock Analysis. The following three lemmas refer to the Markov chain in Fig. A.1. We use $p_i(t)$ to indicate the probability of being in state i at time t. CDF stands for cumulative distribution function.

LEMMA A.1. *Suppose the process starts in state l. Then $\forall t$, $p_1(t) \leq (1 - p_0(t))\mu$ where $\mu = 1/(1 + \frac{r}{f} + (\frac{r}{f})^2 + \cdots + (\frac{r}{f})^{l-1})$.*

Proof. Consider the Markov chain restricted to states $1, \ldots, l$. We can prove that the invariance $p_{i+1}(t)/p_i(t) \geq r/f$ (for $i = 1, \ldots, l-1$) is maintained at all times through the following argument. Letting $\phi_i(t) = p_{i+1}(t)/p_i(t)$, we can show $d\phi_i(t)/dt \geq 0$ when $\phi_i(t) = r/f$ and $\forall i', \phi_{i'}(t) \geq r/f$, which implies that for no i can $\phi_i(t)$ fall below r/f if it starts above. This is done by showing that $dp_i(t)/dt = p_{i+1}(t)f + p_{i-1}(t)r - (r+f)p_i(t) \leq 0$ since $\phi_i(t) = r/f$ and $\phi_{i-1}(t) \geq r/f$, and $dp_{i+1}(t)/dt = p_{i+2}(t)f + p_i(t)r - (r+f)p_{i+1}(t) \geq 0$ since $\phi_i(t) = r/f$ and $\phi_{i+1}(t) \geq r/f$ (the p_{i-1} or the p_{i+2} terms are zero for the boundary cases).

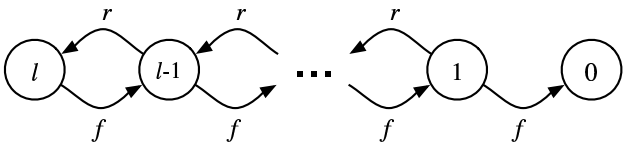

FIG. A.1. *Continous-time Markov chain for Lemmas A.1–A.3. States $i = 1, \ldots, l$ indicate the identity of the currently present clock species $C_1, \ldots, C_l$. Transition to state 0 represents reaction dec_2 for the RM simulation or the state transition initiation reaction of the CTM simulation.*

Now $p_i(t) = \phi_{i-1}(t)\phi_{i-2}(t)\cdots\phi_1(t)p_1(t)$. Thus $\sum_i p_i(t) = 1$ implies $p_1(t) = 1/(1 + \phi_1 + \phi_2\phi_1 + \cdots + \phi_{l-1}\phi_{l-2}\cdots\phi_1) \leq 1/(1 + \frac{r}{f} + (\frac{r}{f})^2 + \cdots + (\frac{r}{f})^{l-1})$. This is a bound on the probability of being in state 1 given that we haven't reached state 0 in the full chain of Fig. A.1. Thus multiplying by $1 - p_0(t)$ gives us the desired result. □

LEMMA A.2. *Suppose for some μ we have $\forall t$, $p_1(t) \leq (1 - p_0(t))\mu$. Let T be a random variable describing the time until absorption at state 0. Then $\Pr[T < t] \leq 1 - e^{-\lambda t}$ for $\lambda = f\mu$ (i.e. our CDF is bounded by the CDF for an exponential random variable with rate $\lambda = f\mu$).*

Proof. The result follows from the fact that $dp_0(t)/dt = p_1(t)f \leq (1 - p_0(t))\mu f$. □

LEMMA A.3. *Starting at state l, the expected time to absorb at state 0 is $O((\frac{r}{f})^{l-1}/f)$ assuming sufficiently large r/f.*

Proof. The expected number of transitions to reach state 0 starting in state i is $d_i = \frac{2pq\left((q/p)^l - (q/p)^{l-i}\right)}{(1-2p)^2} - \frac{i}{q-p}$, where $p = \frac{f}{f+r}$ is the probability of transitioning to a state to the left and $q = 1 - p$ is the probability of transitioning to the state to the right. This expression is obtained by solving the recurrence relation $d_i = pd_{i-1} + qd_{i+1} + 1$ $(0 > i > l)$ with boundary conditions $d_0 = 0, d_l = d_{l-1} + 1$. Thus $d_l < \frac{2pq(q/p)^l}{(1-2p)^2} = \frac{2(r/f)^{l+1}}{(r/f-1)^2}$. This implies that for r/f larger than some constant, $d_l = O((\frac{r}{f})^{l-1})$. Since the expected duration of any transition is no more than $1/f$, the desired bound is obtained. □

By the above lemmas, the time for the clock to "tick" can be effectively thought of as an exponential random variable with rate $\lambda = f/(1 + \frac{r}{f} + (\frac{r}{f})^2 + \cdots + (\frac{r}{f})^{l-1}) = \Theta(\frac{f}{(r/f)^{l-1}})$. Lemma A.2 shows that the CDF of the tick is bounded by the CDF of this exponential random variable. Further, lemma A.3 shows that the expected time for the tick is bounded by (the order of) expected time of this exponential random variable. Note that Lemma A.2 is true no matter how long the clock has already been running (a "memoryless" property). For our clock construction (Fig. 3.1(b)), we set λ by changing $\#A$ and $\#A^*$ which define the forward and reverse rates f and r. Specifically, we have $\lambda = \Theta(\frac{k\#A^{*l}}{v\#A^{l-1}})$.

A.2. Time/space-bounded RM simulation. LEMMA A.4. *For the finite RM simulation, the probability of error per step is $O((1/\#A)^{l-1})$. Further, the expected time per step is bounded by $O((\#A)^{l-1}v/k)$.*

Proof. Consider the point in time when the RM simulation enters a state in which it should decrement a non-empty register. If the time until dec_2 occurs were an exponential random variable with rate λ then the probability of error per step would be bounded by $\lambda/(k/v+\lambda)$. (We are making the worst case assumption that there is exactly one register molecule; otherwise, the error is even smaller.) The time until dec_2 is not exponentially distributed, but by Section A.1,

it can be bounded by an exponential random variable with rate $\lambda = O(\frac{k}{v\#A^{l-1}})$ ($\#A^* = 1$ for the RM construction). Note that the clock may have been running for a while since the last *dec* operation (while the RM performs *inc* operations for example); however, this amount of time is irrelevant by the memoryless property established in Section A.1. Thus the probability of error per step is bounded by $\lambda/(k/v+\lambda) = O((1/\#A)^{l-1})$. The expected time per RM step is bounded by the expected time for dec_2 which is $O((\#A)^{l-1}v/k)$ by Section A.1. □

The above lemma implies that we can use $\#A = \Theta((t/\delta)^{1/(l-1)})$ resulting in the expected time for the whole computation of $O(\frac{vt^2}{k\delta})$ and the total probability of error being bounded by δ.

A.3. Time/space-bounded CTM simulation. In the following lemmas, we say a reaction *completely finishes* when it happens enough times that one of the reactants is used up.

LEMMA A.5. *Starting with $\Theta(m)$ molecules of X and $\Theta(m)$ molecules of Y, the expected time for the reaction $X + Y \to Y$ to completely finish is $O(\frac{v}{km}\log m)$. The variance of the completion time is $O((\frac{v}{km})^2)$.*

Proof. When there are q molecules of X remaining, the waiting time until next reaction is an exponential random variable with rate $\Theta(kqm/v)$ and therefore mean $O(\frac{v}{kqm})$. Each waiting time is independent. Thus the total expected time is $\sum_{q=1}^{\Theta(m)} O(\frac{v}{kqm}) = O(\frac{v}{km}\log m)$.[7] The variance of each waiting time is $O((\frac{v}{kqm})^2)$. Thus the total variance is $\sum_{q=1}^{\Theta(m)} O((\frac{v}{kqm})^2) = O((\frac{v}{km})^2)$. □

LEMMA A.6. *Starting with $\Theta(m)$ molecules of X and $\Theta(m)$ molecules of Y such that $\Delta = \#Y - \#X = \Omega(m)$ the expected time for the reaction $X + Y \to \emptyset$ to completely finish is $O(\frac{v}{km}\log m)$. The variance of the completion time is $O((\frac{v}{km})^2)$.*

Proof. This case can be proven by reducing to Lemma A.5 with initial amounts $\#Y' = \Delta$ and $\#X' = \#X$. □

LEMMA A.7. *Starting with $\Theta(m)$ molecules of X and 1 molecule of Y, the expected time for the reaction $X + Y \to 2Y$ to completely finish is $O(\frac{v}{km}\log m)$. The variance of the completion time is $O((\frac{v}{km})^2)$.*

Proof. Consider splitting the process into two halves, with the first part bringing the amount of X to half its initial value and the second part using up the remainder. The time-reverse of the first part, as well as the second part, can both be bounded by processes covered by Lemma A.5. (Assume that $\#X$ is fixed at its minimal value for part one, and assume $\#Y$ is fixed at its minimal value for part two. The variance can only decrease.) □

LEMMA A.8. *Some $\lambda = \Theta(\frac{k\varepsilon^{3/2}3^{s_{ct}}}{vs_{ct}})$ attains error at most ε per microstep of the CTM simulation.*

Proof. Using the above lemmas with $m = 3^{s_{ct}-1}$, by Chebyshev's inequality,[8] with probability at least $1-\varepsilon/2$ all reactions finish before some time $t_f = \Theta(\frac{v}{km}(\log(m)+1/\sqrt{\varepsilon})) = O(\frac{v\log m}{km\varepsilon^{1/2}})$.

[7] As $m \to \infty$, the difference between $\sum_{q=1}^{m}(1/q)$ and $\log m$ approaches the Euler-Mascheroni constant.

[8] Chebyshev's inequality states that for a random variable X with expected value μ and finite variance σ^2, for any $d > 0$, $\Pr[|X - \mu| \geq d\sigma] \leq 1/d^2$.

Now we set λ such that the probability that the clock ticks before time t_f is smaller than $\varepsilon/2$ (for a total probability of error ε). Since the time until the clock ticks is bounded by the CDF of an exponential random variable with rate λ (Sec A.1), it is enough that $\lambda < \frac{\varepsilon}{2t_f}$ and so we can choose some $\lambda = \Theta(\frac{\varepsilon^{3/2}km}{v \log m})$. □

LEMMA A.9. *Any TM with a two-way infinite tape using at most s_{tm} space and t_{tm} time can be converted to a CTM using $s_{ct} = 2s_{tm}$ space and $t_{ct} = O(t_{tm}s_{tm})$ time. If Δ extra bits of padding on the CTM tape is used, then $t_{ct} = O(t_{tm}(s_{tm} + \Delta))$ time is required.*

Proof. (sketch, see [34]) Two bits of the CTM are used to represent a bit of the TM tape. The extra bit is used to store a TM head position marker. To move in the direction corresponding to moving the CTM head clockwise (the easy direction) is trivial. To move in the opposite direction, we use the temporary marker to record the current head position and then move each tape symbol clockwise by one position. Thus, a single TM operation in the worst case corresponds to $O(s)$ CTM operations. □

In order to simulate t_{tm} steps of a TM that uses s_{tm} bits of space on a CTM using Δ bits of padding requires $t_{ct} = O(t_{tm}(s_{tm}+\Delta))$ CTM steps and a circular tape of size $s_{ct} = 2s_{tm}+\Delta$ (Lemma A.9). Recall that in our CTM simulation, there are four microsteps corresponding to a single CTM operation, which is asymptotically still $O(t_{ct})$. Thus, in order for the total error to be at most δ, we need the error per CTM microstep to be $\varepsilon = O(\frac{\delta}{t_{tm}(s_{tm}+\Delta)})$. Setting the parameters of the clock module $(\#A, \#A^*)$ to attain the largest λ satisfying Lemma A.8, the expected time per microstep is $O(\frac{vs_{ct}}{k3^{s_{ct}}\varepsilon^{3/2}}) = O(\frac{v(s_{tm}+\Delta)^{5/2}t_{tm}^{3/2}}{k3^{2s_{tm}+\Delta}\delta^{3/2}})$. This can be done, for example, by setting $\#A^{*l} = \Theta(\frac{3^{s_{ct}}}{s_{ct}})$ and $\#A^{l-1} = \Theta(\frac{1}{\varepsilon^{3/2}})$. Since there are total $O(t_{tm}(s_{tm} + \Delta))$ CTM microsteps, the total expected time is $O(\frac{v(s_{tm}+\Delta)^{7/2}t_{tm}^{5/2}}{k\,3^{(2s_{tm}+\Delta)}\delta^{3/2}})$.

How large is the total molecular count? If we keep δ constant while increasing the complexity of the computation being performed, and setting $\#A^*$ and $\#A$ as suggested above, we have that the total molecular count is $\Theta(m + \#A)$ where $m = 3^{2s_{tm}+\Delta}$. Now m increases at least exponentially with $s_{tm} + \Delta$, while $\#A$ increases at most polynomially. Further, m increases at least quadratically with t_{tm} (for any reasonable algorithm $2^{s_{tm}} \geq t_{tm}$) while $\#A$ increases at most as a polynomial of degree $(3/2)\frac{1}{l-1} < 2$. Thus m will dominate.

A.4. Unbounded RM simulation. After i dec_2 steps, we have $\#A = i_0 + i$ where i_0 is the initial number of A's. The error probability for the next step is $O(1/\#A^2) = O(1/(i_0+i)^2)$ by Lemma A.4 when $l = 3$. The total probability of error over an unbounded number of steps is $O(\sum_{i=0}^{\infty} 1/(i_0 + i)^2)$. To make sure this is smaller than δ we start out with $i_0 = \Theta(1/\delta)$ molecules of A.[9]

Now what is the total expected time for t steps? By Lemma A.4 the expected time for the next step after i dec_2 steps is $O(\#A^2v/k) = O((i_0 + i)^2v/k)$. Since each step at most increases the total molecular count by 1, after t total steps v is not larger than $O(i_0 + t + s_0)$, where s_0 is the sum of the initial values of all the registers. Thus the expected time for the tth step is bounded by $O((i_0 + i)^2(i_0 + t + s_0)/k) = O((1/\delta + t)^2(1/\delta + t + s_0)/k)$ and so the expected

[9] If $i_0 > 1/\delta + 1$, then $\delta > \int_{i_0-1}^{\infty} \frac{1}{x^2}dx > \sum_{x=i_0}^{\infty} \frac{1}{x^2}$.

total time for t steps is $O(t(1/\delta + t)^2(1/\delta + t + s_0)/k)$.

A.5. Unbounded CTM simulation. We want to follow a similar strategy as in the RM simulation (Section A.4) and want the error probability on the ith CTM step to be bounded by $\varepsilon = 1/(\Theta(1/\delta) + i)^2$ such that the total error probability after arbitrarily many steps is bounded by δ. By Lemma A.8, we can attain per step error probability (taking the union bound over the 4 microsteps in a step) bounded by this ε when we choose a small enough $\lambda = \Theta(\frac{k\varepsilon^{3/2}3^{s_{ct}}}{vs_{ct}}) = \Theta(\frac{k3^{s_{ct}}}{v(1/\delta+i)^3 s_{ct}})$, where s_{ct} is the current CTM tape size. Recall that λ is set by $\#A$ and $\#A^*$ such that $\lambda = \Theta(\frac{k\#A^{*l}}{v\#A^{l-1}})$ (Sec. A.1). It is not hard to see that we can achieve the desired λ using clock Markov chain length $l = 5$, and appropriate $\#A = \Theta((i_0 + i)3^{s_{ct}})$ and $\#A^* = \Theta(3^{s_{ct}})$, for appropriate $i_0 = \Theta(1/\delta + s_{ct0})$, where s_{ct0} is the initial size of the tape. These values of $\#A$ and $\#A^*$ can be attained if the SCRN triples the amount of A and A^* whenever extending the tape and increases $\#A$ by an appropriate amount $\Theta(3^{s_{ct}})$ on every step.

How fast is the simulation with these parameters? From Sec. A.1 we know that the expected time per microstep is $O(1/\lambda) = O(\frac{v(1/\delta+s_{ct0}+i)^4}{k3^{s_{ct}}})$. Since the total molecular count is asymptotically $O(\#A) = O((1/\delta + s_{ct0} + i)3^{s_{ct}})$, this expected time is $O((1/\delta + s_{ct0} + i)^5/k)$. However, unlike in the bounded time/space simulations and the unbounded RM simulation, this expected time is conditional on all the previous microsteps being correct because if a microstep is incorrect, A and A^* may increase by an incorrect amount (for example reactions tripling $\#A$ akin to $A \to A^\dagger$ and $A^\dagger \to 3A$ can drive A arbitrarily high if the catalyst state species for both reactions are erroneously present simultaneously). Nonetheless, the expected duration of a microstep conditional on the entire simulation being correct is at most a factor of $1/(1-\delta)$ larger than this.[10] Since we can assume δ will always be bounded above by a constant less than one, the expected duration of a microstep conditional on the entire simulation being correct is still $O((1/\delta + s_{ct0} + i)^5/k)$. By Lemma A.9, this yields total expected time time to simulate t_{tm} steps of a TM using at most s_{tm} space and with initial input of size s_{tm0} is $O((1/\delta + s_{tm0} + t_{tm}s_{tm})^5 t_{tm}s_{tm}/k)$ assuming the entire simulation is correct.

A.6. Decidability of Reachability. We reduce the reachability question in SCRNs to the reachability question in Vector Addition Systems (VAS), a model of asynchronous parallel processes developed by Karp and Miller [35]. In the VAS model, we consider walks through a p dimensional integer lattice, where each step must be one of a finite set of vectors in $\mathbb{N}^p$, and each point in the walk must have no negative coordinates. It is known that the following reachability question is decidable: given points $\mathbf{x}$ and $\mathbf{y}$, is there a walk that reaches some point $\mathbf{y}' \geq \mathbf{y}$ from $\mathbf{x}$ [35]. The correspondence between VASs and SCRNs is straightforward [39]. First consider chemical reactions in which no species occurs both as a reactant and as a product (i.e. reactions that have no catalysts). When such a reaction $\alpha = \langle \mathbf{l}, \mathbf{r}, k\rangle$ occurs, the state of the SCRN changes by addition of the vector $-\mathbf{l} + \mathbf{r}$. Thus the trajectory of states is a walk

[10] This follows from the fact that $\mathbb{E}[X|A] \leq (1/\Pr[A])\,\mathbb{E}[X]$ for random variable X and event A, and that the expected microstep duration conditional on the previous and current microsteps being correct is the same as the expected microstep duraction conditional on the entire simulation being correct.

through $\mathbb{N}^p$ wherein each step is any of a finite number of reactions, subject to the constraint requiring that the number of molecules of each species remain non-negative. Karp and Miller's decidability results for VASs then directly imply that our reachability question of whether we ever enter a state greater than or equal to some target state is decidable for catalyst-free SCRNs. The restriction to catalyst-free reactions is easily lifted: each catalytic reaction can be replaced by two new reactions involving a new molecular species after which all reachability questions (not involving the new species) are identical for the catalyst-free and the catalyst-containing networks.

Acknowledgments: We thank G. Zavattaro for pointing out an error in an earlier version of this manuscript. This work is supported in part by the "Alpha Project" at the Center for Genomic Experimentation and Computation, an NIH Center of Excellence (grant no. P50 HG02370), as well as NSF Grant No. 0523761 and NIMH Training Grant MH19138-15.

REFERENCES

[1] Milan N. Stojanovic, Tiffany Elizabeth Mitchell, and Darko Stefanovic. Deoxyribozyme-based logic gates. *Journal of the American Chemical Society*, 124:3555–3561, 2002.

[2] Nathan D. McClenaghan A. Prasanna de Silva. Molecular-scale logic gates. *Chemistry - A European Journal*, 10(3):574–586, 2004.

[3] David Sprinzak and Michael B. Elowitz. Reconstruction of genetic circuits. *Nature*, 438:443–448, 2005.

[4] Geord Seelig, David Soloveichik, David Y. Zhang, and Erik Winfree. Enzyme-free nucleic acid logic circuits. *Science*, 314:1585–1588, 2006.

[5] Charles H. Bennett. The thermodynamics of computation – a review. *International Journal of Theoretical Physics*, 21(12):905–939, 1982.

[6] Paul W.K. Rothemund. A DNA and restriction enzyme implementation of Turing machines. In *Proceedings DNA Computers*, pages 75–120, 1996.

[7] Paul W.K. Rothemund, Nick Papadakis, and Erik Winfree. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biology*, 2:e424, 2004.

[8] Gerard Berry and Gerard Boudol. The chemical abstract machine. In *Proceedings of the 17th ACM SIGPLAN-SIGACT annual symposium on principles of programming languages*, pages 81–94, 1990.

[9] Gheorghe Paun and Grzegorz Rozenberg. A guide to membrane computing. *Theoretical Computer Science*, 287:73–100, 2002.

[10] Anthony M. L. Liekens and Chrisantha T. Fernando. Turing complete catalytic particle computers. In *Proceedings of Unconventional Computing Conference, York*, 2006.

[11] Dana Angluin, James Aspnes, and David Eisenstat. Fast computation by population protocols with a leader. Technical Report YALEU/DCS/TR-1358, Yale University Department of Computer Science, 2006. Extended abstract to appear, DISC 2006.

[12] Michael B. Elowitz and Stanislas Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403:335–338, 2000.

[13] Joanne Macdonald, Yang Li, Marko Sutovic, Harvey Lederman, Kiran Pendri, Wanhong Lu, Benjamin L. Andrews, Darko Stefanovic, and Milan N. Stojanovic. Medium scale integration of molecular logic gates in an automaton. *Nano Letters*, 6:2598–2603, 2006.

[14] Donald A. McQuarrie. Stochastic approach to chemical kinetics. *Journal of Applied Probability*, 4:413–478, 1967.

[15] N.G. van Kampen. *Stochastic Processes in Physics and Chemistry*. Elsevier, revised edition edition, 1997.

[16] Péter Érdi and János Tóth. *Mathematical Models of Chemical Reactions : Theory and Applications of Deterministic and Stochastic Models*. Manchester University Press, 1989.

[17] Daniel T. Gillespie. A rigorous derivation of the chemical master equation. *Physica A*, 188:404–425, 1992.

[18] Thomas G. Kurtz. The relationship between stochastic and deterministic models for chemical reactions. *The Journal of Chemical Physics*, 57:2976–2978, 1972.

[19] Stewart N. Ethier and Thomas G. Kurtz. *Markov Processes: Characterization and Convergence*. John Wiley & Sons, 1986.

[20] Purnananda Guptasarma. Does replication-induced transcription regulate synthesis of the myriad low copy number proteins of Escherichia coli? *Bioessays*, 17:987–997, 1995.

[21] Benjamin Levin. *Genes VII*. Oxford University Press, 1999.

[22] Harley H. McAdams and Adam P. Arkin. Stochastic mechanisms in gene expression. *Proceedings of the National Academy of Sciences*, 94:814–819, 1997.

[23] Michael B. Elowitz, Arnold J. Levine, Eric D. Siggia, and Peter S. Swain. Stochastic gene expression in a single cell. *Science*, 297:1183–1185, 2002.

[24] Gurol M. Suel, Jordi Garcia-Ojalvo, Louisa M. Liberman, and Michael B. Elowitz. An excitable gene regulatory circuit induces transient cellular differentiation. *Nature*, 440:545–550, 2006.

[25] Adam P. Arkin, John Ross, and Harley H. McAdams. Stochastic kinetic analysis of a developmental pathway bifurcation in phage-l Escherichia coli. *Genetics*, 149:1633–1648, 1998.

[26] Stochastic simulation implementations. Systems Biology Workbench: `http://sbw.sourceforge.net`; BioSpice: `http://biospice.lbl.gov`; Stochastirator: `http://opnsrcbio.molsci.org`; STOCKS: `http://www.sysbio.pl/stocks`; BioNetS: `http://x.amath.unc.edu:16080/BioNetS`; SimBiology package for MATLAB: `http://www.mathworks.com/products/simbiology/index.html`.

[27] Karan Vasudeva and Upinder S. Bhalla. Adaptive stochastic-deterministic chemical kinetic simulations. *Bioinformatics*, 20:78–84, 2004.

[28] Andrzej M. Kierzek. STOCKS: STOChastic kinetic simulations of biochemical systems with Gillespie algorithm. *Bioinformatics*, 18:470–481, 2002.

[29] David Adalsteinsson, David McMillen, and Timothy C. Elston. Biochemical network stochastic simulator (BioNetS): software for stochastic modeling of biochemical networks. *BMC Bioinformatics*, 5, 2004.

[30] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81:2340–2361, 1977.

[31] Michael Gibson and Jehoshua Bruck. Efficient exact stochastic simulation of chemical systems with many species and many channels. *Journal of Physical Chemistry A*, 104:1876–1889, 2000.

[32] Daniel T. Gillespie. Stochastic simulation of chemical kinetics. *Annual Review of Physical Chemistry*, 58:35–55, 2007.

[33] Marvin L. Minsky. Recursive unsolvability of Post's Problem of 'tag' and other topics in theory of Turing machines. *Annals of Math*, 74:437–455, 1961.

[34] Turlough Neary and Damien Woods. A small fast universal Turing machine. Technical Report NUIM-CS-2005-TR-12, Dept. of Computer Science, NUI Maynooth, 2005.

[35] Richard M. Karp and Raymond E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3(4):147–195, 1969.

[36] Michael Sipser. *Introduction to the Theory of Computation*. PWS Publishing, 1997.

[37] Boaz Barak. A probabilistic-time hierarchy theorem for 'slightly non-uniform' algorithms. In *Proceedings of RANDOM*, 2002.

[38] Marcelo O. Magnasco. Chemical kinetics is Turing universal. *Physical Review Letters*, 78:1190–1193, 1997.

[39] Matthew Cook. *Networks of Relations*. PhD thesis, California Institute of Technology, 2005.