Molecular Computation of Complex Markov Chains with Self-Loop State Transitions

Sayed Ahmad Salehi[1], *Marc D. Riedel*[2], *Keshab K. Parhi*[2]

[1]Department of Computer Science and Computer Engineering, Utah Valley University, Orem, UT
[2]Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN
SSalehi@uvu.edu,{mriedel, parhi}@umn.edu

***Abstract*—This paper describes a systematic method for molecular implementation of complex Markov chain processes with self-loop transitions. Generally speaking, Markov chains consist of two parts: a set of states, and state transition probabilities. Each state is modeled by a unique molecular type, referred to as a *data* molecule. Each state transition is modeled by a unique molecular type, referred to as a *control* molecule, and a unique molecular reaction. Each reaction consumes data molecules of one state and produces data molecules of another state. As we show in this paper, the produced data molecules are the same as the reactant data molecules for self-loop transitions. Although the reactions corresponding to self-loop transitions do not change the molecular concentrations of the data molecules, they are required in order for the system to compute probabilities correctly. The concentrations of control molecules are initialized according to the probabilities of corresponding state transitions in the chain. The steady-state probability of Markov chain is computed by equilibrium concentration of *data* molecules. We demonstrate our method for a molecular design of a seven-state Markov chain as an instance of a complex Markov chain process with self-loop state transitions. The molecular reactions are then mapped to DNA strand displacement reactions. Using the designed DNA system we compute the steady-state probability matrix such that its element (i,j) corresponds to the long-term probability of staying in state j, given it starts from state i. For example, the error in the computed probabilities is shown to be less than 2% for DNA strand-displacement reactions.**

Keywords—Molecular computation, Markov chain, self-loop state transition, molecular reaction, DNA strand-displacement.

I. INTRODUCTION

Due to the advantage of having well-defined theory and extensive simulation software tools, a set of molecular reactions, more often called chemical reaction networks (CRNs), has been used as programming language or models and abstractions in different applications. For example, there has been a groundswell of interest in molecular computations in recent years [1-6]. Since 1994, several approaches have been investigated for molecular computation; these include: solving NP-computational and combinatorial problems such as the Hamiltonian path problem [1] and finding maximal clique problem [7], computing of deterministic functions and algorithms [8],[9], implementation of logical functions [10]-[14], and signal processing operations [15]-[17].

This paper presents a systematic methodology for modeling complex Markov chains by a set of chemical reactions in order to compute the steady-state probabilities of its states. The produced set of molecular reactions is implemented and validated by DNA strand displacement reactions. Markov chain has been frequently used for modeling and analyzing systems of chemical reactions [4],[18],[19]; however, this paper addresses the reverse problem, i.e., modeling Markov chain and computing its steady-state probabilities by a system of chemical reactions. Since Markov processes are commonly used in numerous processing and statistical modeling applications [20]-[22], design of a systematic method for synthesizing Markov chains with DNA strand displacement reactions leads to a systematic method for implementing these applications using DNA.

The implementation of simple Markov Chain processes has been discussed in [23]. In this paper, however, we present a systematic method for implementing complex Markov chain processes using molecular reactions. In particular, for the first time, we describe the molecular implementation of complex Markov Chain processes with self-loop transition states. Our method can be used to implement any Markov chain that includes both transient and recurrent states, or even super states. We use a method similar to the method presented in [23]. However, in our method we have self-loop state transitions as well as transitions with different source and destination states. In this method, each state in the Markov chain is modeled by a unique *data* molecular type and each state transition is modeled by a molecular reaction and a unique *control* molecule. *Data* molecule for each state or *control* molecule for each state transition is distinguishable from molecules corresponding to other states or state transitions. All the reactions have the form of $C_{ij} + D_i \rightarrow C_{ij} + D_j$, where C_{ij} is the *control* molecule that facilitates transition from state i to j and D_i and D_j are *data* molecules for states i and j, respectively. The final concentration of *data* molecules related to each state determines the probability of that state. Since all of the reactions have the same form, that is to say, they have two reactants and two products, we design a template of DNA reactions to implement them. The DNA template consists of a sequence of three DNA strand-displacement reactions for each molecular reaction in our design.

In Section II we present the proposed methodology for modeling Markov chain by molecular reactions. In Section III we explain mapping of the molecular reactions of the proposed model to DNA strand displacement reactions and also present simulation results. Finally Section IV concludes the paper.

This work is supported by the National Science Foundation under grant number CCF-1423407.

II. MODELING BY MOLECULAR REACTIONS

This section describes the methodology of constructing a model for a Markov chain process using molecular reactions. This model can be used to compute the steady-state probability of each state in the Markov chain diagram. The methodology has two parts: *initialization* and *transition reactions*.

Initialization: This stage consists of initializing two groups of molecules: *data* molecules and *control* molecules.

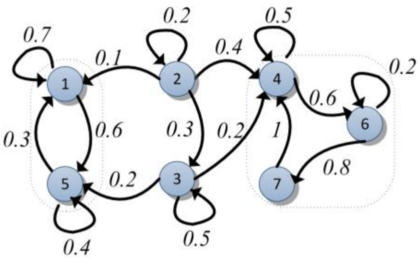

Fig. 1. A seven-state Markov chain with self-loop state transitions.

Data molecule for each state of Markov chain is a unique type of molecule assigned to that state. The initial quantity for each data molecule, except the start state, is zero. For the start state the initial value can be any large nonzero number; however, larger the initial value, more accurate the probability estimates are.

Control molecules are used to control transformation of data molecules of one state to data molecules of other states according to the transition probabilities in the Markov chain diagram. A unique type of molecule is devoted for each state transition in the chain. The quantities of control molecules are time invariant and can be determined according to the probabilities corresponding to their transition in the chain; the ratio of quantity of a control molecule over total quantities of all control molecules in a state equals the probability of the corresponding transition.

In general, the number of unique molecular types in our model is the sum of the number of states and the number of transitions in the Markov chain.

Transition Reactions: The transition reactions determine how data molecules transfer in order to implement the desired Markov chain. There is a transition reaction for each transition in the chain. This reaction transfers data molecules in the source state of transition to the data molecules in the destination state. Each transition reaction uses a control molecule for transferring data molecules. However, transition reactions should not change the concentration of control molecules. Therefore, if a control molecule is used as a reactant in a reaction, it should also be a product of the reaction.

We illustrate our methodology by explaining the design of molecular model for the seven-state Markov Chain shown in Fig. 1. The transition probability matrix for this Markov chain is given by (1).

$$\begin{array}{c|ccccccc} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} \\ \hline \mathbf{1} & 0.7 & 0 & 0 & 0 & 0.3 & 0 & 0 \\ \mathbf{2} & 0.1 & 0.2 & 0.3 & 0.4 & 0 & 0 & 0 \\ \mathbf{3} & 0 & 0 & 0.5 & 0.3 & 0.2 & 0 & 0 \\ \mathbf{4} & 0 & 0 & 0 & 0.5 & 0 & 0.5 & 0 \\ \mathbf{5} & 0.6 & 0 & 0 & 0 & 0.4 & 0 & 0 \\ \mathbf{6} & 0 & 0 & 0 & 0 & 0 & 0.2 & 0.8 \\ \mathbf{7} & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \tag{1}$$

As it is discussed in [22], states 2 and 3 are transient while, other than states 2 and 3, all states in this chain are recurrent. Furthermore, states 1 and 5 form a super state and states 4, 6, and 7 form another super state.

The steady-state probabilities of the above mentioned Markov chain can be mathematically computed as shown in (2).

$$\begin{array}{c|ccccccc} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} \\ \hline \mathbf{1} & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 \\ \mathbf{2} & 0.1833 & 0 & 0 & 0.3412 & 0.0917 & 0.2132 & 0.1706 \\ \mathbf{3} & 0.2667 & 0 & 0 & 0.2824 & 0.1333 & 0.1765 & 0.1412 \\ \mathbf{4} & 0 & 0 & 0 & 0.4706 & 0 & 0.2941 & 0.2353 \\ \mathbf{5} & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 \\ \mathbf{6} & 0 & 0 & 0 & 0.4706 & 0 & 0.2941 & 0.2353 \\ \mathbf{7} & 0 & 0 & 0 & 0.4706 & 0 & 0.2941 & 0.2353 \end{array} \tag{2}$$

The element (*i,j*) in the steady-state matrix shows the probability of being in state j, after a long time, if we start from state i. In other words, the row i of the matrix in (2) shows all long-term probabilities if starting from state i. For example, if we start from state 1, after a long time, with the probability of $\frac{2}{3}$ we will stay in state 1 and with the probability of $\frac{1}{3}$ we will be in state 5. The probability of being in other states is zero. The matrix also shows that no matter which state we start, the probability of staying in states 2 and 3, after a long time, is zero. Furthermore, the matrix shows that if we start from states 1 or 5, we end up staying in states 1 or 5 with the probabilities of $\frac{2}{3}$ or $\frac{1}{3}$, respectively. Similarly, the matrix shows that if we start from states 4, 6, or 7, we end up staying in states 4, 6, or 7.

In order to design its molecular reactions first we assign a *data* molecular type to each state: molecular type 1 for state 1, molecular type 2 for state 2, and all the way to molecular type 7 for state 7.

For this example we have seven data molecular types since there are seven states in the Markov chain. Suppose we want to compute the steady-state probability of each state if the chain starts from state i. For this purpose we set the initial value of *data* molecular type assigned to state i, i.e., D_i, to a nonzero concentration, while the other states have *data* molecules with zero initial values. For simplicity we consider 100 as the initial concentration of data molecules of state i.

Because there are sixteen state transition arcs, ten between states and six self-loops, we have sixteen *Control* molecular types. *Control* molecules in this example are named as C_{ij}, for $1 \leq i,j \leq 7$ where i is the source state and j is the destination state of the transition. For each state transition, we choose initial values of the related *Control* molecule such that the ratio of that *Control* molecule to the summation of the concentration of all *Control* molecules corresponding to the outgoing state transitions of that state is the same as the probability of that state transition. For example for state 3 there are three outgoing state transitions, i.e., to states 5, 4 and 3. Accordingly, we have three *Control* molecules, C_{35}, C_{34}, and C_{33}. If we choose $[C_{35}] = 20, [C_{34}] = 30$, and $[C_{33}] = 50$ then for the probabilities of the outgoing state transitions from state 3 we have:

$$p_{35} = \frac{[C_{35}]}{[C_{35}] + [C_{34}] + [C_{33}]} = 0.2$$

$$p_{34} = \frac{[C_{34}]}{[C_{35}] + [C_{34}] + [C_{33}]} = 0.3$$

$$p_{33} = \frac{[C_{33}]}{[C_{35}] + [C_{34}] + [C_{33}]} = 0.5$$

Notice that the exact values for the concentrations of control molecular types is not important. Only the ratios should be equal to the probabilities of the state transitions.

Similar to the state 3, we can initialize all of the control molecular types as follows:

$$\begin{array}{llll} [C_{11}] = 70 & [C_{23}] = 30 & [C_{35}] = 20 & [C_{55}] = 40 \\ [C_{15}] = 30 & [C_{24}] = 40 & [C_{44}] = 50 & [C_{66}] = 20 \\ [C_{21}] = 10 & [C_{33}] = 50 & [C_{46}] = 50 & [C_{67}] = 80 \\ [C_{22}] = 20 & [C_{34}] = 30 & [C_{51}] = 60 & [C_{74}] = 100 \end{array} \tag{3}$$

Note that any unit could have been used in this paper for the molecular concentrations. In other words, the molecular concentrations are in arbitrary unit.

The final step is to write the molecular reactions related to each state transition. For each transition in the chain a molecular reaction, called transition reaction, is required. In the transition reaction, data molecules of the source state of transition are combined with the control reactions related to that transition to produce the data molecules of the destination state of the transition. The general form of the transition reactions is $C_{ij} + D_i \rightarrow C_{ij} + D_j$, where C_{ij} is the *control* molecule that catalyzes transition reaction for state transition from i to j. D_i and D_j are *data* molecules for states i and j, respectively. As we notice C_{ij} molecules act as catalysts for this reaction. For example, for state 3 there are three outgoing state transitions, i.e., to states 5, 4 and 3. Accordingly, we have three transition molecular reactions as $C_{35} + D_3 \rightarrow C_{35} + D_5$ for the transition to state 5, $C_{34} + D_3 \rightarrow C_{34} + D_4$ for the transition to state 4, and $C_{33} + D_3 \rightarrow C_{33} + D_3$ for the self-loop transition. We can write all molecular reactions for the chain shown in Fig. 1 as listed in Table 1.

Table 1. Molecular reactions for Markov chain shown in Fig. 1.

$C_{15} + D_1 \rightarrow C_{15} + D_5$	$C_{11} + D_1 \rightarrow C_{11} + D_1$
$C_{21} + D_2 \rightarrow C_{21} + D_1$	$C_{22} + D_2 \rightarrow C_{22} + D_2$
$C_{23} + D_2 \rightarrow C_{23} + D_3$	$C_{24} + D_2 \rightarrow C_{24} + D_4$
$C_{33} + D_3 \rightarrow C_{33} + D_3$	$C_{34} + D_3 \rightarrow C_{34} + D_4$
$C_{35} + D_3 \rightarrow C_{35} + D_5$	$C_{44} + D_4 \rightarrow C_{44} + D_4$
$C_{46} + D_4 \rightarrow C_{46} + D_6$	$C_{51} + D_5 \rightarrow C_{51} + D_1$
$C_{55} + D_5 \rightarrow C_{55} + D_5$	$C_{66} + D_6 \rightarrow C_{66} + D_6$
$C_{67} + D_6 \rightarrow C_{67} + D_7$	$C_{74} + D_7 \rightarrow C_{74} + D_4$

Sixteen reactions in Table 1 and twenty three molecular types, i.e., seven data molecular types and sixteen control molecular types, with the initial concentrations, listed in Equation (3), are the proposed molecular model for the Markov chain problem in Fig. 1.

Suppose we want to compute the steady state probabilities for this chain, when we start from a particular state. We initialize all the data molecular types of states to zero but the one that corresponds to our start state. For example, if want to calculate the probabilities of ending up to different states if we start from state 3, we initialize D_3 with 100 and initialize all other states to zero. Then, after we run the simulation of the proposed molecular system the steady-state concentrations of each data molecular type is used to calculate the probability of ending up in that state. For example, if we start with the initial concentration of one of the data molecules is equal to 100 and the others are zero and the final concentrations of data molecules D_3 and D_7 are 40 and 60, respectively, then the steady-state probabilities of states 3 and 7 are $\frac{40}{40+60} = 0.4$ and $\frac{60}{40+60} = 0.6$, respectively.

III. DNA IMPLEMENTATION

A. *Mapping to DNA*

To validate our proposed model with a real molecular system we use DNA molecules. The DNA system that we use is based on DNA strand-displacement reactions. DNA strand-displacement reactions work based on toehold-mediated strand displacement that was introduced by Yurke et al [6] for construction of DNA tweezers. In the toehold-mediated mechanism, a single strand of a double-strand DNA can be replaced by another single strand, provided a toehold binding is feasible. Fig. 2 shows an example of the mechanism. In this representation continuous and dotted lines are used for domain and toehold parts, respectively. Toehold $t1$ in the single strand $t1D1$ binds to its complementary in double strand $D1t1D1$ and causes its domain $D1$ (in blue) to be replaced by the single domain $D1$ (in green). Since this reaction starts by binding toeholds, it is called toehold-mediated strand-displacement reactions.

By properly designing the toeholds in DNA molecules, an arbitrary rate of binding can be achieved. Indeed, Soleveichik et al. [5] demonstrated that DNA strand-displacement reactions can emulate the kinetics of any CRN. Therefore, the molecular reactions designed by our methodology can be implemented by DNA strand displacements reactions using their method. They also presented a software tool that maps chemical CRNs to DNA reactions.

In order to map our molecular reactions to DNA reactions using the method presented in [5], each molecular type is required to be identified by two toeholds and two domains as depicted in Fig. 3 for molecule A.

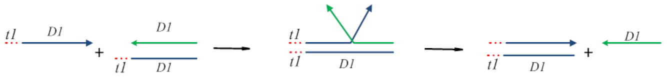

Fig. 2. Toehold-mediated DNA strand-displacement reaction.

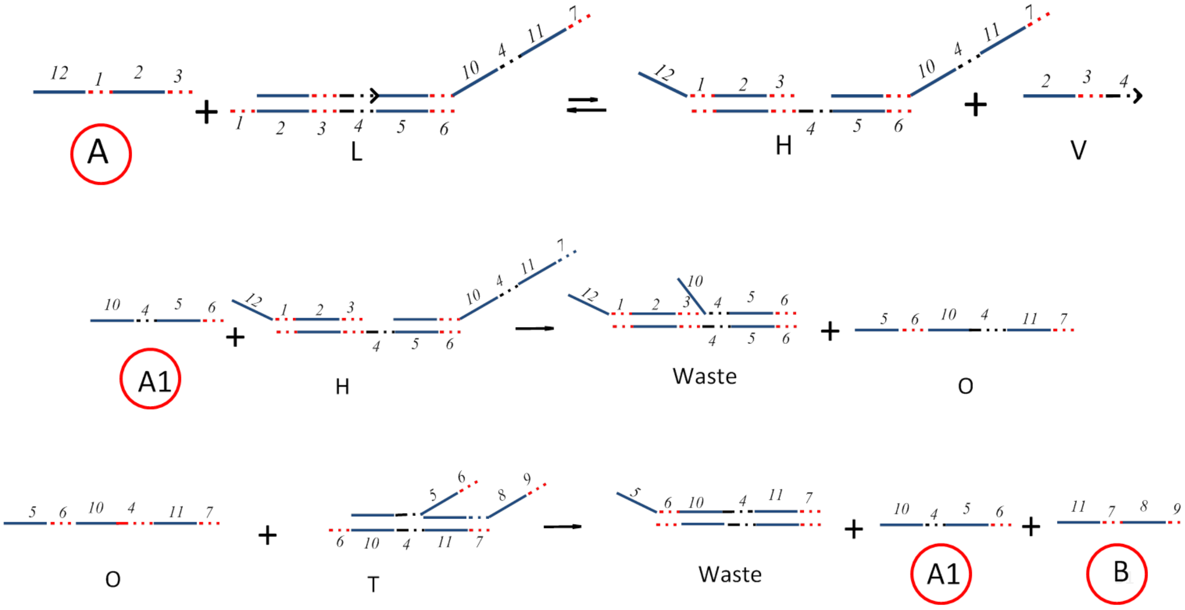

Fig. 4- Three DNA reactions that emulate the molecular reaction $A1 + A \rightarrow A1 + B$

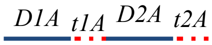

Fig. 3- A DNA strand composed of two domains and two toeholds is used to represent each molecule.

According the method presented in [5] each chemical reaction with m reactants and nonzero products can be emulated by $m + 1$ DNA strand-displacement reactions. Since all of reactions in our designed molecular system are bimolecular reactions with two reactants and two products, they are mapped to 3 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 two products. Three DNA reactions, shown in Fig. 4, implement the molecular reaction $A1 + A \rightarrow A1 + B$.

The reaction starts when toehold 1 of strand A binds to its complementary part of gate molecule L and produces intermediate double strand gate molecule H and single strand molecule V. In the next reaction molecule $A1$ and gate H combine with each other and produce a double strand Waste and single strand O. In the third reaction, strand O and gate T combine to produce A1 and B and another Waste molecule. For more details about the mapping of molecular reactions to DNA strand-displacement reactions, the reader is referred to [5].

B. Simulation Results

To evaluate the DNA implementation of the proposed model, we implement the model for the example shown in Fig. 1. All the molecules are mapped to the DNA strands as described earlier. We use the *Mathematica* tool of Soloveichik et al [5] to simulate the designed DNA system. We use the initial parameters based on the examples in [5]. For all DNA simulations for the presented designs we used the following parameters: The initial concentration of auxiliary complexes, $C_{max} = 10^{-5}M$, the maximum strand displacement rate constant, $q_{max} = 10^{6}M^{-1}s^{-1}$, all reaction rates are the same, i.e., $k = 5.56 \times 10^{4}M^{-1}s^{-1}$.

In order to obtain the simulation values for each row of the steady-state probability matrix, shown in (2), we initialize one of the data molecular types, i.e., D_is, each time and run the simulation until equilibrium. In other words, to obtain the simulation values for the first row of the matrix we initialize $D_1 = 100\,nM$ and other D_is to zero. Then equilibrium concentrations of D_is show us the steady state values to be in the related state if we start from state 1. Similarly, to obtain the simulation values for the second row of the matrix we initialize $D_2 = 100\,nM$ and the equilibrium concentrations of D_is show us the steady state values to be in the corresponding state if we start from state 2. We repeat the simulation for all the seven states in the chain. Fig. 5 shows the simulation results for each case. We run each simulation for 12 minutes (0.2 hours) and then calculate the steady-state probabilities according to the final molecular concentrations. As we described earlier in this paper, the probability value, p_i, for state i, is computed using the Equation (5).

$$p_i = \frac{D_i}{\sum_{j=1}^{7} D_j} \qquad (5)$$

Since only one D_i is initialized to 100 at the beginning of each simulation and other data molecular types initialized to zero Equation (5) can be simplified to

$$p_i = \frac{D_i}{100} \qquad (6).$$

Note that the summation of the concentration of data molecules does not change during the simulation time because the proposed molecular reactions do not consume or produce data molecules; they just transfer one molecular type to the other type. Matrix in Equation (7) shows the computed steady-state probability matrix using DNA reactions.

$$\begin{array}{cccccccc} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} \\ \mathbf{1} & 0.6663 & 0 & 0 & 0 & 0.3327 & 0 & 0 \\ \mathbf{2} & 0.183 & 0 & 0 & 0.3424 & 0.0911 & 0.2167 & 0.1732 \\ \mathbf{3} & 0.265 & 0 & 0 & 0.2844 & 0.1384 & 0.178 & 0.1442 \\ \mathbf{4} & 0 & 0 & 0 & 0.468 & 0 & 0.294 & 0.2312 \\ \mathbf{5} & 0.6663 & 0 & 0 & 0 & 0.3327 & 0 & 0 \\ \mathbf{6} & 0 & 0 & 0 & 0.4729 & 0 & 0.2895 & 0.2361 \\ \mathbf{7} & 0 & 0 & 0 & 0.4729 & 0 & 0.2941 & 0.2361 \end{array} \quad (7).$$

If the concentration of auxiliary species, C_{max}, is much larger than the maximum concentration of other species, (i.e., in the proposed CRNs $C_{max} \gg 100\,nM$) then, as described in [5] 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 [5]. Although these requirements have been met in our simulations, errors exist. The error stems from the fact that each molecular reaction is implemented by a sequence of DNA strand displacement reactions as we described earlier; the concentrations of auxiliary molecules, C_{max}, is bounded. In fact, if $C_{max} \rightarrow \infty$, the DNA simulation results converge to the expected molecular reaction 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 Supplementary Information of [5] and [24].

IV. Conclusion

This paper has presented a method for modeling complex Markov chain processes using molecular reactions. The model is composed of data molecules corresponding to states and control molecules corresponding to the state transition probabilities, where data molecules transfer between states by molecular reactions corresponding to state transitions. One interpretation of the model is that each data molecule represents an agent that participates in the Markov chain process. Whether there is a state transition or not is defined by the molecular reactions and the probabilities of transitions are defined by the concentrations of control molecules.

We ran the DNA simulation of a seven-state Markov chain for 12 minutes. For more complex Markov chain the simulation time can be longer. In fact, the simulation time should be large enough for the data molecular types to get close to their equilibrium concentrations.

References

[1] L. Adleman, “Molecular computation of solutions to combinatorial problems,” *Science*, vol. 266, no. 11, pp. 1021–1024, 1994.

[2] L. Qian, D. Soloveichik, and E. Winfree, “Efficient turing-universal computation with DNA polymers,” in *International Conference on DNA Computing and Molecular Programming*, 2010.

[3] S. A. Salehi, K. K. Parhi and M. D. Riedel, “Chemical Reaction Networks for Computing Polynomials,” *ACS Synth. Biol.,* 6(1), pp. 76-83, 2017.

[4] D. Soloveichik, M. Cook, E. Winfree, and J. Bruck, “Computation with finite stochastic chemical reaction networks,” *Natural Computing*, vol. 7, no. 4, 2008.

[5] D. Soloveichik, G. Seelig, and E. Winfree, “DNA as a universal substrate for chemical kinetics,” *Proceedings of the National Academy of Sciences*, vol. 107, no. 12, pp. 5393–5398, 2010.

[6] B. Yurke, A. J. Turberfield, A. P. Mills, Jr, F. C. Simmel, and J. Neumann, “A DNA-fuelled molecular machine made of DNA,” *Nature*, vol. 406, pp. 605–608, 2000.

[7] Q. Ouyang, P. D. Kaplan, S. Liu and A. Libchaber, “DNA solution of the maximal clique problem,” *Science,* vol. 278, 5337, 446-449, 1997.

[8] F. Guarnieri,M. Fliss, and C. Bancroft, “Making DNA add,” *Science,* vol. 273, 220–223, 1996.

[9] H. L. Chen, D. Doty, and D. Soloveichik, “Deterministic function computation with chemical reaction networks,” *DNA 2012, Lecture Notes in Computer Science*, vol. 7433, Springer-Verlag, 2012, pp. 25-42.

[10] J. C. Anderson, C. A. Voigt, and A. P. Arkin, “A genetic AND gate based on translation control,” *Molecular Systems Biology*, 3(133), 2007.

[11] K. Ramalingam, J. R. Tomshine, J. A. Maynard, and Y. N. Kaznessis, “Forward engineering of synthetic biological AND gates,” *Biochem. Eng. J.* 47, pp.38−47, 2009.

[12] L. Qian, E. Winfree, “Scaling Up Digital Circuit Computation with DNA Strand Displacement Cascades,” *Science*, pp. 1196-1201, 2011.

[13] A. Tamsir, J. J. Tabor, and C. A. Voigt, “Robust multicellular computing using genetically encoded NOR gates and chemical ‘wires’,” *Nature,* vol. 469, pp. 212−215, 2011.

[14] J. Bonnet, P. Yin, M. E. Ortiz, P. Subsoontorn, and D. Endy, “Amplifying Genetic Logic Gates,” *Science*, pp. 599-603, 2013.

[15] H. Jiang, S. Salehi, M. Riedel, and K. Parhi, “Discrete-time signal processing with DNA,” *ACS Synthetic Biology*, vol. 2, no. 5, pp. 245–254, 2013.

[16] S.A. Salehi, H. Jiang, M.D. Riedel, and K.K. Parhi, "Molecular Sensing and Computing Systems (Invited Paper)," *IEEE Transactions on Molecular, Biological, and Multi-Scale Communications*, 1(3), pp. 249-264, Sept. 2015.

[17] S. A. Salehi, M.D. Riedel, and K.K. Parhi, “Asynchronous Discrete-time Signal Processing with Molecular Reactions,” *Proc. of 2014 Asilomar Conference on Signals, Systems, and Computers*, pp. 1767-1772, Nov. 2014.

[18] M. Gibson and J. Bruck, “Efficient exact stochastic simulation of chemical systems with many species and many channels,” *Journal of Physical Chemistry A*, 104:1876–1889, 2000.

[19] D. T. Gillespie, “A General Method for Numerically Simulating the stochastic time evolution of coupled chemical reactions,” *The Journal of Computational Physics*, 22 :403–434, 1976.

[20] S. Resnick, *Adventures in stochastic processes*, Birkhauser, 4th printing, 2005.

[21] P. von Hilgers and A. N. Langville, “The five greatest applications of Markov Chains,” in *Proceedings of the Markov Anniversary Meeting*, Boston Press, Boston, MA, 2006.

[22] R. Durrett, *Essentials of Stochastic Processes*. Springer Science & Business Media. 2012.

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

[24] Y.J. Chen, N. Dalchau, N. Srinivas, A. Phillips, L. Cardelli, D. Soloveichik and G. Seelig, “Programmable chemical controllers made from DNA,” *Nature Nanotechnology* 8: pp.755-762, 2013.

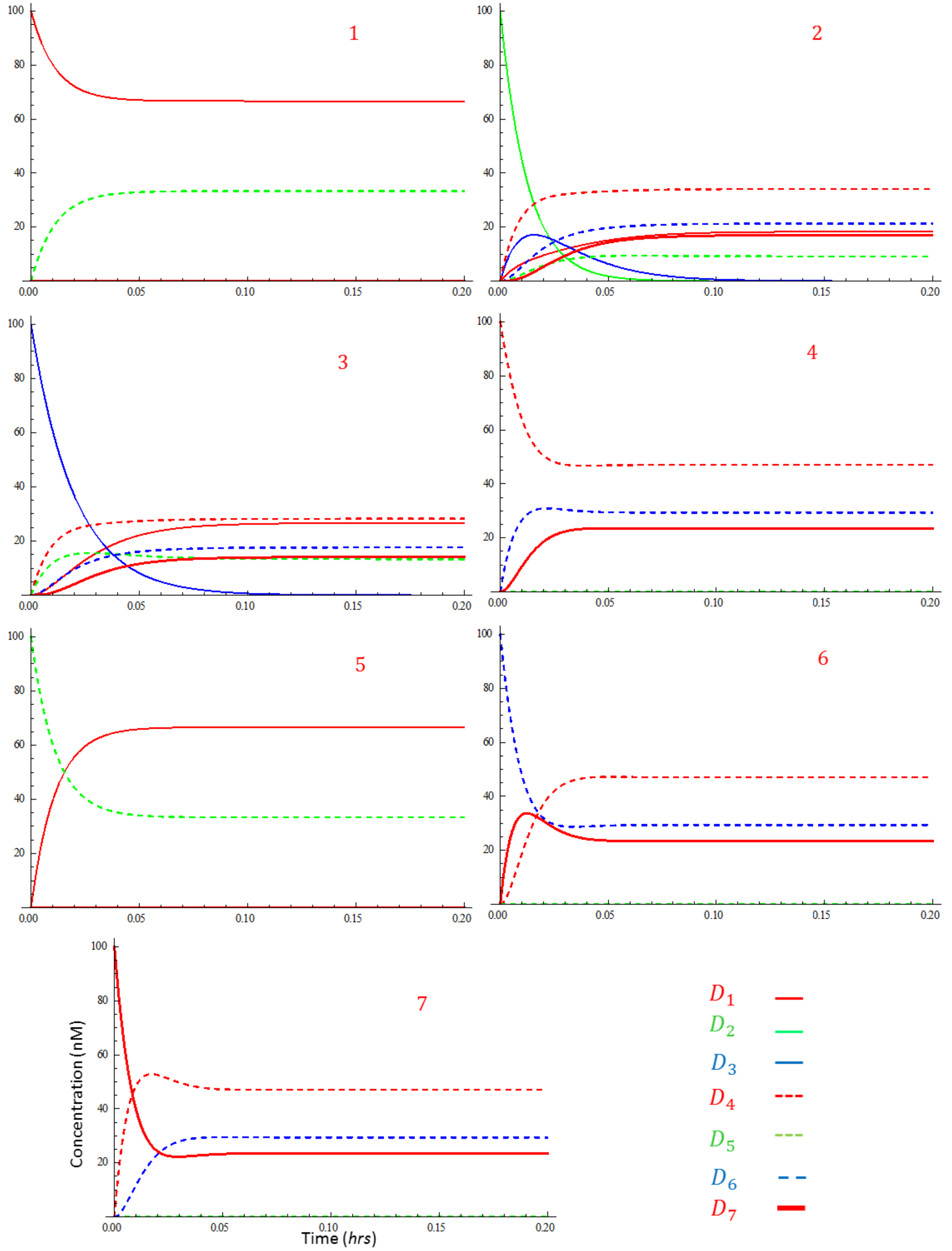

Fig. 5. Simulation results of DNA implementation for the proposed molecular model for Fig. 1. Big red numbers at the top-right corner of each figure represent the start state for that simulation where the related data molecular type is initialized to a nonzero value.