Electronic Preprint for Journal of Information Processing Vol.23 No.6

Regular Paper

Matrix Network: A New Data Structure for Efficient Enumeration of Microstates of a Genetic Regulatory Network

Xiao Cong[1] Tatsuya Akutsu[1,a)]

Received: February 21, 2015, Accepted: August 12, 2015

Abstract: Stochastic processes play an important role in gene regulatory networks. For many years, methods and algorithms have been developed to solve the problems regarding stochastic mechanisms in the cellular reaction system. Discrete Chemical Master Equation (dCME) is a method developed to analyze biological networks by computing the exact probability distribution of the microstates. With this method, because all computations and analyses of probability distribution can be processed based on the enumerated microstates, network microstates enumeration has been considered as a significant and prerequisite step. However, there is no efficient enumeration method. Applications will perform poorly when enumeration must address a complex or large network. To improve these microstate computation and analysis methods, we propose an efficient algorithm to enumerate microstates using Matrix Network, a new data structure we designed. Unlike traditional methods that perform the enumeration using simulation to apply reactions, the proposed approach utilizes the correlation of the microstate values and the geometric structure of the microstate map to accelerate the enumeration computation. In this paper, the theoretical basis, features and algorithms of Matrix Network are discussed. Moreover, sample applications demonstrating computation and analysis using Matrix Network are provided.

Keywords: Matrix Network, gene regulatory network, microstates, microstate enumeration, stochasticity

1. Introduction

Gene regulation has a significant role in cellular processes. Research has continued to explore the secrets of gene regulation, describing these systems using mathematics and biology. Many different kinds of models and methods have been proposed [1] including methods using Bayesian and Boolean networks [2], [3].

Gene regulation involves a complex reaction network consisting of a set of genes, proteins, small molecules, and their mutual regulatory interactions. In genetic regulatory networks, stochastic processes have occupied an important role [4], [5]. Different regulatory proteins are produced to control the development of functions by selecting different reaction pathways [6]. Based on stochastic processes, pathway selection probabilities are determined. Exploiting the advantage of stochastic expression, cells randomize the regulatory outcome and ensure mechanisms that function smoothly and stably.

The high biological significance of stochastic processes has advanced applicable studies. In 1977, Gillespie provided a stochastic simulation algorithm as the fundamental framework to study stochasticity [7]. Under his framework, a microscopic state (microstate) can be defined as a combination of molecules in a chemical system. Then, different independent reactions can be modeled as transitions between these microstates. Because the probability distribution of microstates and transitions describes the full properties of a stochastic system, the study of stochasticity can be seen as a research to determine the methods to study the probability distribution of microstates. The discrete Chemical Master Equation (dCME), one such method, was developed to calculate the probability distribution of a steady state [8]. The dCME method can calculate the statistics of microstates with significantly higher performance, without loss of accuracy, than methods using the SDE and ODE models [9], [10], [11], [12].

In fact, the whole microstates of a biological network and the transitions of the microstates are built up as a Markov process. The dCME method calculates the microstate probability distribution by solving some equation on Markov state transition probabilities. The first step of dCME is to enumerate all possible microstates based on the specific molecular species and related reactions of the given biological network model. The second step is to obtain the Markov state transition matrix $\mathbf{M}$ by the microstates and related reactions. Finally, the probability distribution $\mathbf{P}$ of the microstates is obtained by solving the equation $\mathbf{P} = \mathbf{MP}$.

According to the dCME procedure, an essential step is the enumeration of the microstates. The subsequent computation for probability distribution can be accomplished using the result of the enumeration. However, there is no efficient algorithm to enumerate microstates. In a traditional enumeration, there is a set $\mathbf{X}$ that is used to collect the enumerated microstates. After generating a microstate $\mathbf{m}$, it must be confirmed that $\mathbf{m}$ does not already exist in set $\mathbf{X}$ (to ensure that $\mathbf{m}$ has not been previously generated,

[1] Bioinformatics Center, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto 611–0011, Japan.
[a)] takutsu@kuicr.kyoto-u.ac.jp

that is, that **m** is a new microstate). Only if it is a new microstate, **m** can be added to set **X**. However, even an extremely small genetic network can result in a vast number of microstates. Traditional computation requires excessive time to verify the uniqueness of each microstate generated by a reaction in a large size set **X**.

In this paper, we provide an efficient method to enumerate microstates. We solved the problem of enumeration using geometric relevance based on the structure of a map of the microstates. For a given genetic network, the microstates can be considered as an enormous graph where the vertices are microstates and edges are the transitions (reactions to change microstates) between microstates. In fact, based on the shape of the graph, many parts of the geometry structure can reveal a similar pattern that can be considered as a special feature. To utilize this feature, we designed a new data structure, Matrix Network. We then use Matrix Network to build an efficient microstate enumeration algorithm.

2. Methods

2.1 Matrix Network

2.1.1 Definitions

Matrix Network is defined on theoretical basis that includes M-Vector, M-Set, M-Matrix, R-Matrix and Plus Computation. To discuss Matrix Network, definitions must be provided in advance.

Definition 2.1: M-Vector is a vector where, **m** is the sum of the elements, **n** is the number of elements, and the elements are natural numbers. We denote the M-Vector of **n** elements by **V(m,n)**.

Example 2.2: Vector (1,2,1) is an M-Vector, V(4,3), because the sum of elements is four, and the vector has three elements. Similarly, Vector (1,2) is an M-Vector V(3,2).

Definition 2.3: M-Set (**S(m,n)**) is a set containing all possible M-Vectors that have the same expressions (**V(m,n)**).

Example 2.4: Using Definition 2.1, it is known that both Vector (1,2) and (3,0) are V(3,2) M-Vectors and both belong to the set S(3,2). The full set S(3,2) is {(1,2), (3,0), (0,3), (2,1)}.

M-Set facilitates the definition of M-Matrix, the most essential object used to build the Matrix Network.

Definition 2.5: M-Matrix (**M(m,n)**) is a matrix composed of all the M-Set' (**S(m,n)**) vectors, which have been ordered by left-heavier; each row of M-Matrix is a vector of M-Set. Because a row of M-Matrix has **n** elements, the number of columns of M-Matrix is also **n**.

Example 2.6: As S(3,2) contains four vectors: (1,2), (3,0), (0,3), (2,1), then ordered by left-heavier and combined, these yield the M-Matrix M(3,2):

$$\begin{bmatrix} 3 & 0 \\ 2 & 1 \\ 1 & 2 \\ 0 & 3 \end{bmatrix}.$$

Example 2.7: Other examples: M-Matrix M(1,2) and M(2,2)

$$\text{M(1,2) is } \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \text{ and M(2,2) is } \begin{bmatrix} 2 & 0 \\ 1 & 1 \\ 0 & 2 \end{bmatrix}.$$

Clearly, a vector has a determinate position in the M-Matrix. That is, a vector has a unique sequence number in an M-Matrix.

Example 2.8: (1,1) is the second vector (or row) of M(2,2); (1,2) is the third vector (or row) of M(3,2).

Definition 2.9: Given two M-Matrices with the same column number, **M(a,n)** and **M(b,n)** (**b > a**), we state that **M(b,n)** is at a higher level than **M(a,n)** and the distance between these two M-Matrices is **b** – **a**. If the distance is one, these two M-Matrices are seen as **neighbor M-Matrices** of each other.

Example 2.10: The distance between M(1,2) and M(3,2) is two and M(1,2) is the neighbor matrix of M(2,2).

Definition 2.11: Plus Computation: For a given **d**, the distance of two M-Matrices M1 and M2 (M2 is at higher level than M1), we define Plus Computation (+) as **M1** + **d** = **M2**.

Example 2.12: For the pair of neighbors M(2,3) and M(3,3), we have M(2,3) + 1 = M(3,3).

"Plus 1" computation can express the relationship between neighboring M-Matrices in a simple manner; however, it cannot describe the inherent relationships of two neighboring M-Matrices as to the level of coefficient value. We build another matrix to represent the inherent value relationship between neighboring M-Matrices called **Relationship Matrix** (**R-Matrix**).

Before providing a definition of R-matrix, we assign a representation for the necessary items. For a given M-Matrix, we represent the coefficient of the **j**-th element of the **i**-th row as **c(i,j)** and the **i**-th row as **r(i)** (where **i** > 0 and **j** > 0).

Example 2.13: For a M(2,3), $\begin{bmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix}$,

we have r(2) = (c(2,1),c(2,2),c(2,3)) = (1,1,0)

Definition 2.14: For a pair of neighbor M-Matrices with **n** columns (**M1**, **M2**, **and M1** + **1** = **M2**), we define **R-Matrix**, where coefficient $\mathbf{a_{i,j}}$ can be set by the following computation:

①. For the **i**-th row of M1, $\mathbf{r_1(i)}$, we apply "Plus **1**" operation on its **j**-th element, $\mathbf{c_1(i,j)}$, and acquire a new vector **v**.

②. We find the **k**-th row of **M2**, $\mathbf{r_2(k)}$ that is the same as vector **v**.

③. We set the value of coefficient $\mathbf{a_{i,j}}$ as the value of **k**.

Example 2.15: A more complex example:

$$\text{for M(2,3), } \begin{bmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix}, \text{ and M(3,3), } \begin{bmatrix} 3 & 0 & 0 \\ 2 & 1 & 0 \\ 2 & 0 & 1 \\ 1 & 2 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 2 \\ 0 & 3 & 0 \\ 0 & 2 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{bmatrix},$$

$$\text{R-Matrix is } \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \\ 4 & 7 & 8 \\ 5 & 8 & 9 \\ 6 & 9 & 10 \end{bmatrix}$$

Definition 2.16: Matrix Network is a set of M-Matrices and R-Matrix defined by the rule: for given **M** and **N**, **Matrix Network (M,N)** is the set of all M-Matrices, **M(m,n)** with $\mathbf{m} \leqq \mathbf{M}$, $\mathbf{n} = \mathbf{N}$, and the R-Matrix of the M-Matrix at the highest level. The data is represented by the following structure **[M-Matrices|R-Matrix]**.

Example 2.17: Matrix Network (4,2):

$$[0\ \ 0]\begin{bmatrix}1&0\\0&1\end{bmatrix}\begin{bmatrix}2&0\\1&1\\0&2\end{bmatrix}\begin{bmatrix}3&0\\2&1\\1&2\\0&3\end{bmatrix}\begin{bmatrix}4&0\\3&1\\2&2\\1&3\\0&4\end{bmatrix}\Bigg|\begin{bmatrix}1&2\\2&3\\3&4\\4&5\\5&6\end{bmatrix}$$

Example 2.18: Matrix Network (2,3):

$$[0\ \ 0\ \ 0]\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}\begin{bmatrix}2&0&0\\1&1&0\\1&0&1\\0&2&0\\0&1&1\\0&0&2\end{bmatrix}\Bigg|\begin{bmatrix}1&2&3\\2&4&5\\3&5&6\\4&7&8\\5&8&9\\6&9&10\end{bmatrix}$$

2.1.2 Features of Matrix Network

(1) **M-Matrices of Matrix Network**

In a Matrix Network, a higher-level M-Matrix will indirectly contain all the information of a lower-level M-Matrix. From a higher-level M-Matrix, a lower-level M-Matrix can be calculated directly by deducting the value of their distance on the first column of the higher-level M-Matrix and removing the rows containing a negative number.

(2) **R-Matrix of Matrix Network**

In a Matrix Network, R-Matrices of higher-level M-Matrices are longer than R-Matrices of lower-level M-Matrices. A longer R-Matrix contains all the information of a lower R-Matrix directly, without additional computation. Therefore, in a Matrix Network, we only keep the longest R-Matrix to represent all the relationships of the neighboring M-Matrices.

(3) **Graph of Matrix Network**

Definition 2.19: A Graph of a Matrix Network is a graphic representation of a Matrix Network. For a given Matrix Network, the graph can be drawn under the rule:

①. We draw each row of each M-matrix as a vertex.

②. For the **i**-th row of each Matrix **M**, $\mathbf{r_m(i)}$, we take the **i**-th row of the R-Matrix, $\mathbf{r_r(i)}$. The values of the elements of $\mathbf{r_r(i)}$ represent the rows of $\mathbf{M+1}$ that should be linked with $\mathbf{r_m(i)}$. Then, we link the related vertices in the graph.

Example 2.20: An example of a Matrix Network (3,3) in **Fig. 1**.

The graph of Matrix Network (3,3) is presented in Fig. 1. The different colored lines represent different columns of the R-Matrix.

It can be seen that, from left to right, the graph of the Matrix Network is extended by adding more nodes following the unique pattern of a regular geometric structure. In Fig. 1, the pattern is a three-direction path containing red, green, and black lines as directions. The values of the vertices are changed regularly following the path and then grouped into M-Matrices to become the main portion of the Matrix Network. Therefore, before using Matrix Network, it is necessary to generate M-Matrix.

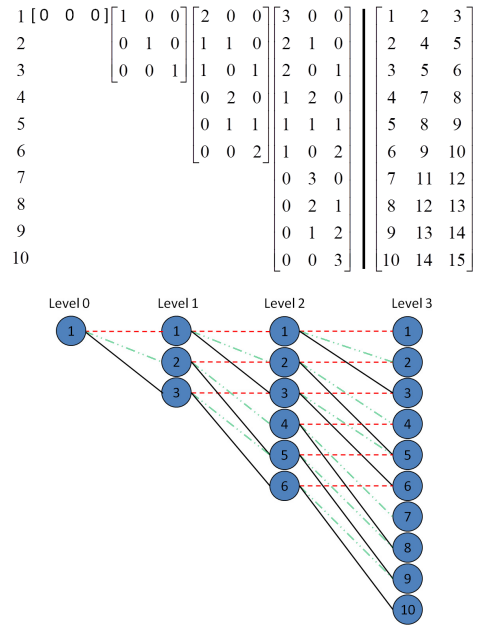


Fig. 1 Matrix Network (3,3) and its graph.

2.1.3 Algorithm to Generate M-Matrix

We developed an efficient algorithm to generate M-Matrix. To build the algorithm with a clear explanation, we must create several functions in advance.

Definition 2.21: Given a matrix **X**, we define a function **F(X)** as increasing the coefficients of the first column of **X** by one.

Example 2.22: For the matrix $\begin{bmatrix}1&0\\0&1\end{bmatrix}$,

$$\text{we have } \mathrm{F}\left(\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)=\begin{bmatrix}2&0\\1&1\end{bmatrix}.$$

Definition 2.23: Given a matrix **X**, we define a function **Z(X)** as inserting a zero column, that is, where all elements are **0**, onto **X** as the first column of **X**.

Example 2.24: For the matrix $\begin{bmatrix}1&0\\0&1\end{bmatrix}$,

$$\text{we have } \mathrm{Z}\left(\begin{bmatrix}1&0\\0&1\end{bmatrix}\right)=\begin{bmatrix}0&1&0\\0&0&1\end{bmatrix}.$$

Definition 2.25: Given two matrices with the same column number, **X**, **Y**, we define a function $\mathbf{C}\left(\frac{\mathbf{X}}{\mathbf{Y}}\right)$ as combining **X** and **Y** vertically into a new matrix.

Example 2.26: For two matrices: $\begin{bmatrix}2&0\\1&1\end{bmatrix}, \begin{bmatrix}1&0\\0&1\end{bmatrix}$,

$$\text{we have } \mathrm{C}\left(\frac{\begin{bmatrix}2&0\\1&1\end{bmatrix}}{\begin{bmatrix}1&0\\0&1\end{bmatrix}}\right)=\begin{bmatrix}2&0\\1&1\\1&0\\0&1\end{bmatrix}.$$

Using the functions above, we have a function to generate M-Matrix, **M(m,n)**.

$$\boldsymbol{M(m,n)} = \boldsymbol{C}\left(\frac{\boldsymbol{F(M(m-1,n))}}{\boldsymbol{Z(M(m,n-1))}}\right), \quad (\mathbf{m},\ \mathbf{n} > 1)$$

Before using the formula above, these facts must be clear:

①. The M-Matrix that can be represented as **M(1,n)** is the identity matrix with **n** columns.

Table 1 Example process of Matrix Network generation for Matrix Network (3,3).

M\N	1	2	3
1	$[\mathbf{1}]$	$\begin{bmatrix}\mathbf{1}&\mathbf{0}\\\mathbf{0}&\mathbf{1}\end{bmatrix}$	$\begin{bmatrix}\mathbf{1}&\mathbf{0}&\mathbf{0}\\\mathbf{0}&\mathbf{1}&\mathbf{0}\\\mathbf{0}&\mathbf{0}&\mathbf{1}\end{bmatrix}$
2	$[\mathbf{2}]$	$\begin{bmatrix}\mathbf{2}&\mathbf{0}\\\mathbf{1}&\mathbf{1}\\\mathbf{0}&\mathbf{2}\end{bmatrix}$	$\begin{bmatrix}\mathbf{2}&\mathbf{0}&\mathbf{0}\\\mathbf{1}&\mathbf{1}&\mathbf{0}\\\mathbf{1}&\mathbf{0}&\mathbf{1}\\\mathbf{0}&\mathbf{2}&\mathbf{0}\\\mathbf{0}&\mathbf{1}&\mathbf{1}\\\mathbf{0}&\mathbf{0}&\mathbf{2}\end{bmatrix}$
3	$[\mathbf{3}]$	$\begin{bmatrix}\mathbf{3}&\mathbf{0}\\\mathbf{2}&\mathbf{1}\\\mathbf{1}&\mathbf{2}\\\mathbf{0}&\mathbf{3}\end{bmatrix}$	$\begin{bmatrix}\mathbf{3}&\mathbf{0}&\mathbf{0}\\\mathbf{2}&\mathbf{1}&\mathbf{0}\\\mathbf{2}&\mathbf{0}&\mathbf{1}\\\mathbf{1}&\mathbf{2}&\mathbf{0}\\\mathbf{1}&\mathbf{1}&\mathbf{1}\\\mathbf{1}&\mathbf{0}&\mathbf{2}\\\mathbf{0}&\mathbf{3}&\mathbf{0}\\\mathbf{0}&\mathbf{2}&\mathbf{1}\\\mathbf{0}&\mathbf{1}&\mathbf{2}\\\mathbf{0}&\mathbf{0}&\mathbf{3}\end{bmatrix}$

②. The M-Matrix that can be represented as **M(m,1)** is the matrix having only one element (a matrix with one column and row) where the value is **m**.

Then, the generation function becomes:

$$M(m,n)=\begin{cases} C\left(\dfrac{F(M(m-1,n))}{Z(M(m,n-1))}\right), & (\mathbf{m},\ \mathbf{n}>1),\\ \text{identity matrix with } \mathbf{n} \text{ columns}, & (\mathbf{m}=1,\ \mathbf{n}>1),\\ \text{one element matrix } [\mathbf{m}], & (\mathbf{m}>0,\ n=1). \end{cases}$$

Example 2.27: Using the facts and generation function above to generate the M-Matrices for Matrix Network (3,3), and the result is presented in the final column of **Table 1**.

2.1.4 Time complexity and the space complexity of Matrix Network

(1) **M-Matrix row number**

We define a function $f(m,n)$ to express the rows number of the M-Matrix **M(m,n)**.

According to the definition of M-Matrix and the mathematical combination method, the rows number of **M(m,n)** is equivalent to the number of ways of selecting **n − 1** items from a collection containing **n + m − 1** total items. Therefore, we have

$$f(m,n)=C_{n-1}^{m+n-1},$$

which means "n + m − 1 choose n − 1".

(2) **Time complexity**

According to our algorithm's procedure, the M-Matrices will be generated from lower levels to higher levels, in another word, when we start to generate **M(m,n)**, we have already generated the lower M-Matrices, which are **M(1,1), M(1,2), M(2,1), …, M(m,n−1), M(m−1,n)**. Thus, the process of generating **Matrix Network (m,n)** will be done once **M(m,n)** has been completely generated. Therefore, we can analyze the computational complexity by analyzing the total work needed to construct **M(m,n)**.

The essence of M-Matrices generation is to generate each single row of the matrices. And for each row, there are 2 kinds of computations:

①. "Plus 1" on the first element of a row,

②. Inserting a zero as the first elements of a row.

According to above discussions, the order of the total computation complexity for **Matrix Network (m,n)** is given by the order of the total computation time of ① and ② of all the row of **M(m,n)**, because at least one of such operations is performed at each recursive step and thus we can ignore the time required for performing recursive calls. It should be noted that we can use the same memory space for **M(m,n)** and for **M(m,n − 1)** and **M(m − 1,n)** by appropriately adjusting indices, and thus we do not need time consuming matrix copy operations.

For ①, the sum of the elements in each row of **M(m,n)** is **m**, and **M(m,n)** have **f(m,n)** rows. Thus, the total number of "plus 1" operations performed in the whole steps will be **mf(m,n)**.

For ②, **M(m,n)** has **n** columns and thus the total number of "insert a zero coefficient" of each row will be **n**. Therefore, the total computation number of "insert a zero coefficient" operations performed in the whole steps will be **nf(m,n)**.

Considering the computation time of ① and ② together, the total time complexity of generating **Matrix Network (m,n)** is

$$T(m,n)=O((m+n)f(m,n))$$

(3) **Space complexity**

For generating **M(m,n)**, we need to combine **M(m − 1,n)** and **M(m,n − 1)** together. And after we get **M(m,n)**, we do not need to keep **M(m − 1,n)** or **M(m,n − 1)** anymore. In another word, in combination process for **M(m,n)**, we can use the same space for **M(m − 1,n)** and **M(m,n − 1)** without allocating any new space. In addition, for generating the whole **Matrix Network (m,n)**, **M(m,n)** is known as the highest and largest M-Matrix. It is able to temporary store other M-Matrices into the space of **M(m,n)**. For example, along with the generation of M-Matrix **M(i,n)**, the related M-Matrices **M(i,j)** where **j** < **n** also have been generated. And all the M-Matrices **M(i,j)** where **j** ≤ **n** can be temporary stored in the space of **M(m,n)**. Then in the process of generating next target M-Matrix **M(i + 1,n)**, the related M-Matrices **M(i + 1,j)** where **j** ≤ **n** can be generated by updating **M(i,j)** directly, and **M(i + 1,j)** can also be stored in the space of **M(m,n)** without allocating any new space. Therefore, it is not necessary to make space for non target M-Matrices, but only for **M(m,n)**. Then the total space for **Matrix Network (m,n)**:

$$nf(m,n)$$

Then the space complexity is

$$S(m,n)=O(nf(m,n))$$

2.2 Microstates Enumeration by Matrix Network

Matrix Network is defined to accelerate the computation of enumeration by utilizing the correlation of the values of the microstates and geometric pattern (or the geometric structure) of the map of microstates. Note that the pattern is dependent on the properties of the reactions and different genetic regulatory networks could produce different geometric patterns.

For a genetic network, to enumerate microstates using Matrix Network, we classify reactions into different types in advance.

Definition 2.28: Type I Reactions are the generation reactions that produce protein directly from DNA (ignore mRNA).

The possible combinations of protein products can be collected by Matrix Network directly. We consider the protein products from a Type I reaction as basic proteins.

Example 2.29: Example of Type I Reactions:
(Gene) -> Protein A;
(Gene) -> Protein B.

Definition 2.30: Type II Reactions are the reactions involving protein synthesis and other interactions among proteins.

Example 2.31: Example of Type II Reactions:
Protein A + Protein B -> Protein C.

Definition 2.32: Type III Reactions are the reactions that change the states of DNA, such as when the regulatory proteins bind to the operator site of a gene.

Example 2.33: Example of Type III Reactions:
Gene + Protein C -> BoundGeneC.

Definition 2.34: A gene state transition network is a finite graph where, each vertex is the state of a gene and the edge is the transition of two states. This graph presents the transitions of all states of a gene. In fact, the transition is a Type III reaction.

Definition 2.35: A protein space or **space of proteins** is a set containing all the possible combinations of the protein **copy numbers**. In this paper, **copy number** means the number of molecules of an element, such as a protein.

For implementation our method, a boundary factor is necessary. In the paper of dCME [8], the boundary condition is defined as **buffer**, and we use the same definition in our method. A buffer is a predefined capacity for containing the generated molecules of the microstates, which is limited by the memory and disk storage.

Choosing the buffer size should be based on the purpose. In this paper, since we would like to compare Matrix Network with a hash table-based method, we used the step increasing buffer sizes of 100, 200 and 300. Though 300 is not the largest size we can choose to compute, it is enough to demonstrate the performance of microstates generation.

Basically, a larger buffer size will generate more microstates, thus improve the precision of the microstates probability distribution by dCME. A large buffer can also cause some resource issues, such as out of memory issue. Therefore, if the purpose is to compute the very precise microstates probability distribution and there is no way to choose the buffer size based on network's biological features, one strategy is to set a large enough buffer size without expending the computation resources. Before implementation, it is able to predict the computation volume by the computation formula of space complexity $\boldsymbol{S(m,n) = O(nf(m,n))}$. It should also be kept in mind that, not only the volume of space complexity, there are also many other factors that can affect memory usage of the program, such as program language and programming style.

According to the above definitions and discussions, the enumeration procedure can be described as:

①. For a given genetic network, we classify all the reactions into three types.

②. According to the Type I reactions, we build a Matrix Network to obtain all the possible combinations of protein copy numbers that can be generated directly from DNA. In general, if the buffer of the system is set as **b**, for **n** Type I reactions, we generate **Matrix Network (b,n)**.

③. The result of ① contains all the possible combinations of protein copy numbers generated directly from DNA. It does not contains, however, the information for other proteins produced by Type II reaction. Our objective is to generate the possible combination of all types of protein. Therefore, we increase the result of ① by one dimension and add the information of each protein that is produced by a Type II reaction. We then calculate the protein space.

④. We generate the gene state transition network based on Type III reactions.

⑤. We join the results of ③ and ④ to obtain all microstates. We remove any nonexistent transitions, if necessary.

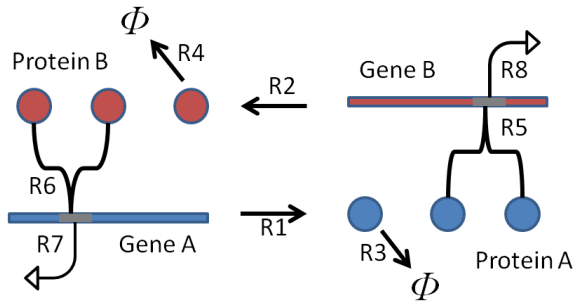
3. Results

3.1 Case I: Toggle Switch Network

Toggle Switch Network is a small network that is used to study stochastic behavior [13], [14], [15], [16]. A toggle switch consists of two genes (see **Fig. 2**). Genes produce two different proteins in this model. The protein product represses the production of the other gene. This model has six species: *GeneA*, *GeneB*, *BoundGeneA*, *BoundGeneB*, *ProteinA*, and *ProteinB*. *GeneA* and *BoundGeneA* are the same gene at different states. The only reason that we consider them as different species is to make the following computation and analysis easier. Similarly, *GeneB* and *BoundGeneB* are the same gene at different states.

There are no reactions concerning the interaction among proteins. Based on the definitions of reaction types, we classify reactions ***R1***, ***R2***, ***R3***, and ***R4*** as Type I reactions, and reactions ***R5***, ***R6***, ***R7***, and ***R8*** as Type III reactions.

For a given buffer B (as boundary condition), on one hand, we generate ***Matrix Network (B, 2)*** to represent the possible pairs of numbers of ***Protein A*** and ***Protein B*** as the protein space. We also simulate applying Type III reactions to obtain a small finite graph as the gene state transition network, which has only four



R1	$(GeneA) \rightarrow ProteinA$
R2	$(GeneB) \rightarrow ProteinB$
R3	$ProteinA \rightarrow \phi$
R4	$ProteinB \rightarrow \phi$
R5	$2 \times ProteinA + GeneB \rightarrow BoundGeneB$
R6	$2 \times ProteinB + GeneA \rightarrow BoundGeneA$
R7	$BoundGeneA \rightarrow 2 \times ProteinB + GeneA$
R8	$BoundGeneB \rightarrow 2 \times ProteinA + GeneB$

Fig. 2 Model of Toggle Switch Network.

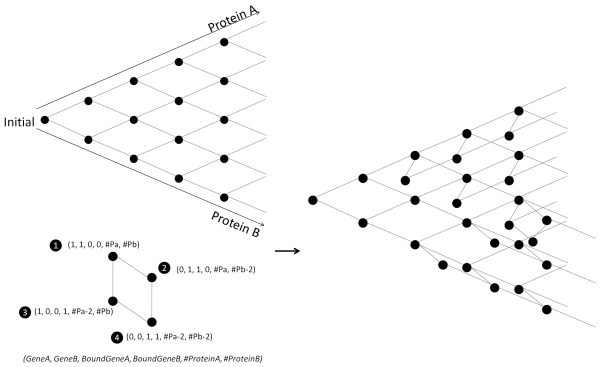


Fig. 3 Joined protein space and gene state transition network.

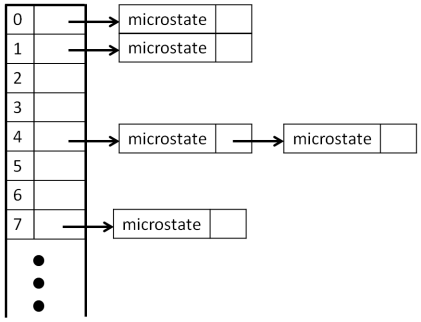


Fig. 4 Separate chaining with linked lists.

Table 2 Results of a Toggle Switch Network using two methods: hash table and Matrix Network.

Buffer / #microstate	100 / 19840	200 / 79604	300 / 179404
Hash Table	**3.196s**	6.602s	21.506s
Matrix Network	3.452s	**4.336s**	**6.095s**

CPU: Core 2 Duo L1700 1.2GHz

vertices. Finally, we combine the results of protein space and the gene state transition network to obtain all the microstates (see **Fig. 3**), details of the combination procedure can be found in A.1.

For comparison with a matrix network, we also use a hash table to enumerate the microstates. We employ a commonly used data structure, hash table, where the structure is separate chaining with linked lists [17]. For each microstate, we first calculate the hash key by hash function; then, using the value of the key, we search for the entry of the list on the chain; finally, we add the microstate to the list. In the list, there are many independent buckets containing microstates sorted by left-heavier on the microstate. The structure of hash table is illustrated in **Fig. 4**.

The hash function is defined by:

$$F(m) = (P_a + 1) * (P_b + 1)\%C,$$

where P_a and P_b are the number of protein A and B of some microstate and C is a constant (that is 10,000 in this experiment).

The results of enumerations using hash table and Matrix Network are presented in **Table 2**.

The results clearly indicate that the proposed enumeration algorithm using Matrix Network is fast and can achieve superior performance compared to the method using hash table.

3.2 Case II: Phage Lambda

Phage lambda is a bacterial virus that consists of a head and a tail and can have fibers [18]. After a Phage lambda infects the bacterial species Escherichia coli, it chooses two pathways to develop, lysis or lysogeny. At normal condition, phage lambda chooses lysogeny to maintain its DNA in the host E.Coli. When the cell suffers damage, the phage will switch pathway to lysis to release many generations and avoid the risk of extinction. The lysis-lysogeny selection allows phage lambda to preserve its species.

The phage lambda network has been seen as the ideal model to study stochasticity because the pathways are switched automatically by the stochastic reactions of the genetic regulatory network of phage lambda. For many years, there has been research based on phage lambda network [19], [20]. Cao et al. have calculated the probability distribution of the microstates of the network and analyzed the features of the phage lambda using dCME [12]. The computational model they used contains 54 biochemical reactions involving 13 molecular species.

We used a similar computational model to Cao et al. We considered the reactions of the production of the same protein with different DNA states as the same reaction, see A.2. We used the same restricted condition as theirs: the maximum copy number of net molecules synthesized in the system was set to 50; they reached approximately 1.7 million microstates. We applied the proposed enumeration algorithm and obtained 1,695,926 microstates (see A.1.2), within 94.145 s (including the time consumed to output the data from memory to a file).

4. Discussion

We designed a new data structure called Matrix Network to represent a type of graph containing the regular geometric pattern obtained from the map of microstates of a genetic regulatory network. Matrix Network is defined under a theoretical basis and includes M-Vector, M-Set, M-Matrix, R-Matrix and Plus Computation. Further, Matrix Network can be efficiently generated using a mathematical method.

Using Matrix Network, we have also provided an efficient algorithm to enumerate microstates. The enumeration algorithm we built using Matrix Network runs at a high performance level. The method can improve microstate-based methods by accelerating the enumeration process.

In the Toggle Switch Network experiment, the results confirm that the proposed enumeration algorithm runs at a high performance level. The phage lambda network is significantly more complex than the Toggle Switch Network. In the experiment, the proposed method enumerated 1,695,926 microstates, which compares with 1.7 million microstates in the Cao et al. paper [12]. This result demonstrates the enumeration algorithm using Matrix Network is also qualified for more complex networks.

Finally, unlike traditional methods that perform enumeration by simulating reactions, Matrix Network utilizes the correlation of both the values of the microstates and geometric structure of the map of microstates. Using Matrix Network, microstate enumeration achieved a high performance level. Moreover, the results of the enumeration potentially provide additional information supporting further computation or analysis.

Acknowledgments We would like to thank assistant profes-

sor Morihiro Hayashida and assistant professor Takeyuki Tamura who always give the valuable advices, continuous encouragement, and generous help for this research. This research was partially supported by JSPS, Japan: Grants-in-Aid 26240034 and 26540125.

References

[1] Jong, H.D.: Modeling and simulation of genetic regulatory systems: A literature review, *Journal of Computational Biology*, Vol.9, No.1, pp.67–103 (2002).

[2] Friedman, N., Linial, M., Nachman, I. et al.: Using Bayesian networks to analyze expression data, *Journal of Computational Biology*, Vol.7, No.3-4, pp.601–620 (2000).

[3] Kauffman, S.A.: *The Origins of Order: Self-Organization and Selection in Evolution*, Oxford Univ. Press (1993).

[4] Elowitz, M.B., Levine, A.J., Siggia, E.D. et al.: Stochastic gene expression in a single cell, *Science*, Vol.297, No.5584, pp.1183–1186 (2002).

[5] McAdamsand, H.H. and Arkin, A.: Stochastic mechanisms in gene expression, *Proc. National Academy of Sciences*, Vol.94, No.3, pp.814–819 (1997).

[6] McAdams, H.H. and Arkin, A.: It's a noisy business! Genetic regulation at the nanomolar scale, *Trends in Genetics*, Vol.15, No.2, pp.65–69 (1999).

[7] Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions, *The Journal of Physical Chemistry*, Vol.81, No.25, pp.2340–2361 (1977).

[8] Cao, Y.F. and Liang, J.: Optimal enumeration of state space of finitely buffered stochastic molecular networks and exact computation of steady state landscape probability, *BMC Systems Biology*, Vol.2, No.1, p.30 (2008).

[9] Santillán, M. and Mackey, M.C.: Why the Lysogenic State of Phage λ Is So Stable: A Mathematical Modeling Approach, *Biophysical Journal*, Vol.86, No.1, pp.75–84 (2004).

[10] Zhu, X.M., Yin, L., Hood, L. et al.: Robustness, stability, and efficiency of phage λ genetic switch: Dynamical structure analysis, *Journal of Bioinformatics and Computational Biology*, Vol.2, No.04, pp.785–817 (2004).

[11] Gillespie, D.T.: The chemical Langevin equation, *The Journal of Chemical Physics*, Vol.113, No.1, pp.297–306 (2000).

[12] Cao, Y., Lu, H.M. and Liang, J.: Probability landscape of heritable and robust epigenetic state of lysogeny in phage lambda, *Proc. National Academy of Sciences*, Vol.107, No.43, pp.18445–18450 (2010).

[13] Ribeiro, A., Zhu, R. and Kauffman, S.A.: A general modeling strategy for gene regulatory networks with stochastic dynamics, *Journal of Computational Biology*, Vol.13, No.9, pp.1630–1639 (2006).

[14] Gardner, T.S., Cantor, C.R. and Collins, J.J.: Construction of a genetic toggle switch in Escherichia coli, *Nature*, Vol.403, No.6767, pp.339–342 (2000).

[15] Kim, K.Y. and Wang, J.: Potential energy landscape and robustness of a gene regulatory network: toggle switch, *PLoS Computational Biology*, Vol.3, No.3, p.e60 (2007).

[16] Tian, T. and Burrage, K.: Stochastic models for regulatory networks of the genetic toggle switch, *Proc. National Academy of Sciences*, Vol.103, No.22, pp.8372–8377 (2006).

[17] Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C.: *Introduction to Algorithms*, MIT Press, Cambridge (2001).

[18] Wikipedia, lambda phage, available from ⟨http://en.wikipedia.org/wiki/Lambda_phage⟩.

[19] Arkin, A., Ross, J. and McAdams, H.H.: Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected Escherichia coli cells, *Genetics*, Vol.149, No.4, pp.1633–1648 (1998).

[20] Gibson, M.A. and Bruck, J.: A probabilistic model of a prokaryotic gene and its regulation, *Computational Methods in Molecular Biology: From Genotype to Phenotype*, MIT Press, Cambridge (2000).

Appendix

A.1 Microstates Enumeration of Toggle Switch Network

In Section 3.1, we classified reactions ***R1***, ***R2***, ***R3***, and ***R4*** as Type I reactions and reactions ***R5***, ***R6***, ***R7***, and ***R8*** as Type III reactions (see Fig. 2). We do not have Type II reactions in this case. For this network, we generate microstates as follow: (a) We generate the protein space (possible combinations of proteins) by Type I reactions. (b) We generate a gene state transition network using Type III reactions. (c) We join the results of (a) and (b) to all microstates.

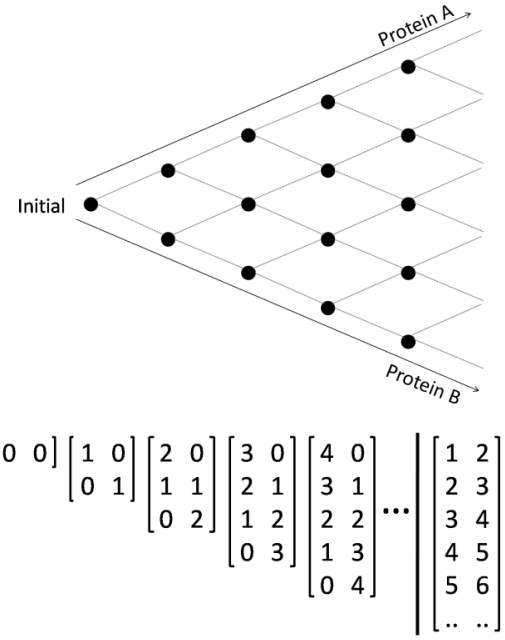


Fig. 5 Protein space.

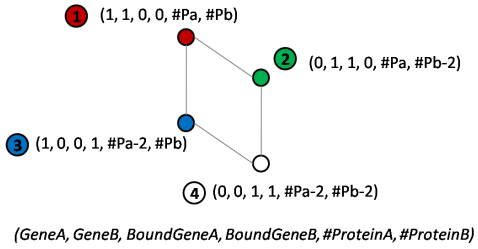


Fig. 6 Gene state transition network.

A.1.1 Generate Protein Space (All Possible Combinations of Proteins)

In this system, according to Type I reactions that produce two kinds of proteins, we generate Matrix Network (b,2) (b is the buffer for the boundary limit) as all the possible combinations of proteins A and B, which are in all rows of all M-Matrices in the Matrix Network.

A.1.2 Gene State Transition by Type III Reactions

According to Type III reactions, we can generate the gene state transition network displayed in **Fig. 6**, which contains the all four kinds of gene state transitions drawn in different colors. In this figure, # represents the copy numbers of a specific protein.

A.1.3 Join Protein Space and Gene State Transition Network

For each state in the gene state transition network, we do not yet have the value of #Pa and #Pb. We input the possible protein combinations from the protein space into each state of the gene state transition network.

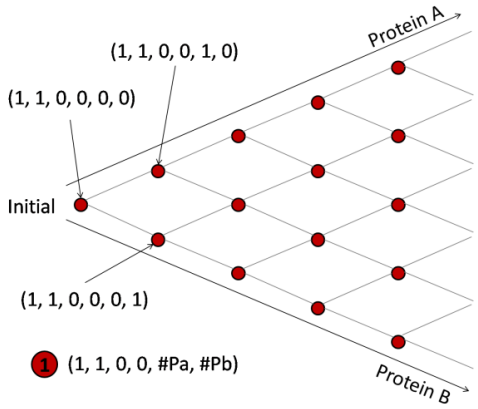


Fig. 7 Result of joined protein space with 1st state in gene state transition network.

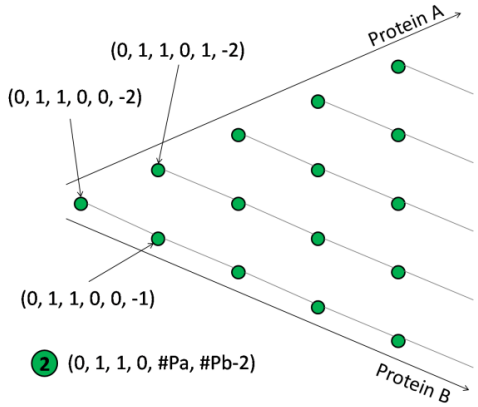


Fig. 8 Result of joined protein space with 2nd state in gene state transition network.

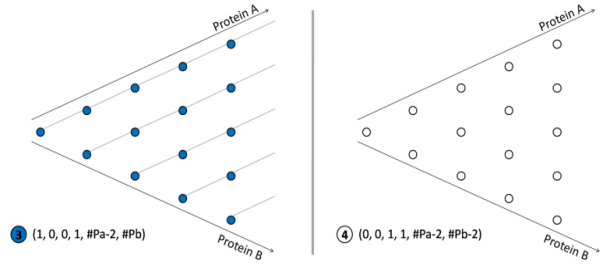


Fig. 9 Result of joined protein space with 3rd and 4th state in gene state transition network.

a) For state 1, (1,1,0,0,#Pa,#Pb), we input the protein space as the value of #Pa and #Pb to obtain **Fig. 7**. Values are displayed in the graph as an example.

b) For state 2, (0,1,1,0,#Pa,#Pb − 2), we do the same computation as above. However, in this case, it should be noticed that in this state, the number of gene A is zero. Therefore the reaction of producing Protein A is not available. Thus, after the same computation as above, we must remove the unavailable links regarding Protein A producing reactions. According to the computation of #Pa − 2, we will have some negative values as illustrated in **Fig. 8**.

c) For state 3 and 4, we can also acquire the values as in **Fig. 9**.

From the results (a, b and c) above, we obtain four new graphs. The points that have the **same position** in these four graphs can be converted to each other following the gene state transition network (Fig. 6). According to the gene state transition network, we combine the results of the four graphs above (see **Fig. 10**).

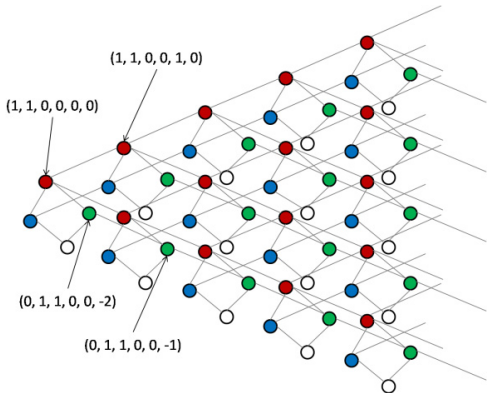


Fig. 10 Result of microstates by joined protein space with gene state transition network.

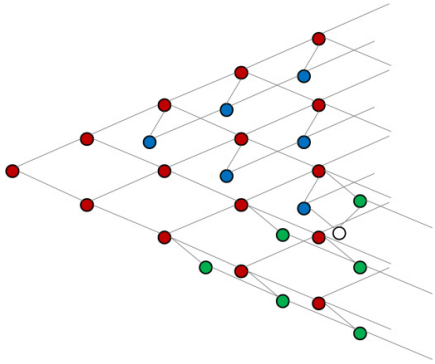


Fig. 11 Result of enumerated microstates.

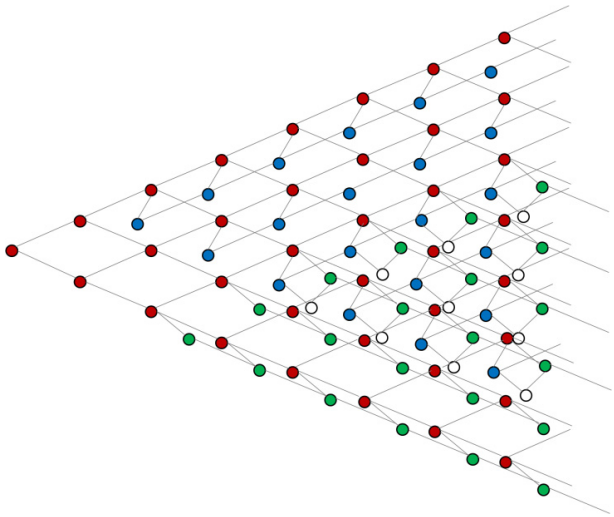


Fig. 12 Additional detailed results of enumerated microstates.

In the process of inputting the possible combination of proteins into the gene state transition, negative values can appear using the computation on *Protein Number A or B less 2*. The negative values can exist in the mathematical model; however, these are not possible in an actual reaction system. Therefore, we remove the unavailable microstates that contain negative value for the number of proteins and then finish the enumeration of all the microstates (see **Fig. 11**). **Figure 12** presents a more detailed result.

Table 3 Portion of results of enumeration of microstates for developmental network of phage lambda by Matrix Network (maximum copy number of net molecules synthesized is 50).

No.	Microstates *1	Transitions of Microstates *2			
1	0 0 0 0 1 1 1 0 0 0 0 0 0 0	2 3 0 0 0 0 0 0 0 0 0 0	27	1 5 0 0 1 1 1 0 0 0 0 0 0 0	34 35 0 12065 0 0 0 0 0 0
2	1 0 0 0 1 1 1 0 0 0 0 0 0 0	4 5 0 0 0 0 0 0 0 0 0 0	28	0 6 0 0 1 1 1 0 0 0 0 0 0 0	35 36 0 12066 0 0 0 0 0 0
3	0 1 0 0 1 1 1 0 0 0 0 0 0 0	5 6 0 0 0 0 0 0 0 0 0 0	29	7 0 0 0 1 1 1 0 0 0 0 0 0 0	37 38 1342 0 0 0 0 0 0 0
4	2 0 0 0 1 1 1 0 0 0 0 0 0 0	7 8 1327 0 0 0 0 0 0 0	30	6 1 0 0 1 1 1 0 0 0 0 0 0 0	38 39 1343 0 0 0 0 0 0 0
5	1 1 0 0 1 1 1 0 0 0 0 0 0 0	8 9 0 0 0 0 0 0 0 0 0 0	...	...	...
6	0 2 0 0 1 1 1 0 0 0 0 0 0 0	9 10 0 12052 0 0 0 0 0 0	1695906	0 3 0 20 0 0 0 0 0 0 1 1 1	1695910 1695911 0 1695921 0 0 0 0 0 0
7	3 0 0 0 1 1 1 0 0 0 0 0 0 0	11 12 1328 0 0 0 0 0 0 0	1695907	4 0 0 20 0 0 0 0 0 0 1 1 1	0 0 1695915 0 0 0 0 0 0 0
8	2 1 0 0 1 1 1 0 0 0 0 0 0 0	12 13 1329 0 0 0 0 0 0 0	1695908	3 1 0 20 0 0 0 0 0 0 1 1 1	0 0 1695916 0 0 0 0 0 0 0
9	1 2 0 0 1 1 1 0 0 0 0 0 0 0	13 14 0 12053 0 0 0 0 0 0	1695909	2 2 0 20 0 0 0 0 0 0 1 1 1	0 0 1695917 1695922 0 0 0 0 0 0
10	0 3 0 0 1 1 1 0 0 0 0 0 0 0	14 15 0 12054 0 0 0 0 0 0	1695910	1 3 0 20 0 0 0 0 0 0 1 1 1	0 0 0 1695923 0 0 0 0 0 0
11	4 0 0 0 1 1 1 0 0 0 0 0 0 0	16 17 1330 0 0 0 0 0 0 0	1695911	0 4 0 20 0 0 0 0 0 0 1 1 1	0 0 0 1695924 0 0 0 0 0 0
12	3 1 0 0 1 1 1 0 0 0 0 0 0 0	17 18 1331 0 0 0 0 0 0 0	1695912	0 0 1 20 0 0 0 0 0 0 1 1 1	1695913 1695914 0 0 0 0 0 0 0
13	2 2 0 0 1 1 1 0 0 0 0 0 0 0	18 19 1332 12055 0 0 0 0 0 0	1695913	1 0 1 20 0 0 0 0 0 0 1 1 1	1695915 1695916 0 0 0 0 0 0 0
14	1 3 0 0 1 1 1 0 0 0 0 0 0 0	19 20 0 12056 0 0 0 0 0 0	1695914	0 1 1 20 0 0 0 0 0 0 1 1 1	1695916 1695917 0 0 0 0 0 0 0
15	0 4 0 0 1 1 1 0 0 0 0 0 0 0	20 21 0 12057 0 0 0 0 0 0	1695915	2 0 1 20 0 0 0 0 0 0 1 1 1	0 0 1695918 0 0 0 0 0 0 0
16	5 0 0 0 1 1 1 0 0 0 0 0 0 0	22 23 1333 0 0 0 0 0 0 0	1695916	1 1 1 20 0 0 0 0 0 0 1 1 1	0 0 0 0 0 0 0 0 0 0
17	4 1 0 0 1 1 1 0 0 0 0 0 0 0	23 24 1334 0 0 0 0 0 0 0	1695917	0 2 1 20 0 0 0 0 0 0 1 1 1	0 0 0 1695925 0 0 0 0 0 0
18	3 2 0 0 1 1 1 0 0 0 0 0 0 0	24 25 1335 12058 0 0 0 0 0 0	1695918	0 0 2 20 0 0 0 0 0 0 1 1 1	0 0 0 0 0 0 0 0 0 0
19	2 3 0 0 1 1 1 0 0 0 0 0 0 0	25 26 1336 12059 0 0 0 0 0 0	1695919	0 0 0 21 0 0 0 0 0 0 1 1 1	1695920 1695921 0 0 0 0 0 0 0
20	1 4 0 0 1 1 1 0 0 0 0 0 0 0	26 27 0 12060 0 0 0 0 0 0	1695920	1 0 0 21 0 0 0 0 0 0 1 1 1	1695922 1695923 0 0 0 0 0 0 0
21	0 5 0 0 1 1 1 0 0 0 0 0 0 0	27 28 0 12061 0 0 0 0 0 0	1695921	0 1 0 21 0 0 0 0 0 0 1 1 1	1695923 1695924 0 0 0 0 0 0 0
22	6 0 0 0 1 1 1 0 0 0 0 0 0 0	29 30 1337 0 0 0 0 0 0 0	1695922	2 0 0 21 0 0 0 0 0 0 1 1 1	0 0 1695925 0 0 0 0 0 0 0
23	5 1 0 0 1 1 1 0 0 0 0 0 0 0	30 31 1338 0 0 0 0 0 0 0	1695923	1 1 0 21 0 0 0 0 0 0 1 1 1	0 0 0 0 0 0 0 0 0 0
24	4 2 0 0 1 1 1 0 0 0 0 0 0 0	31 32 1339 12062 0 0 0 0 0 0	1695924	0 2 0 21 0 0 0 0 0 0 1 1 1	0 0 0 1695926 0 0 0 0 0 0
25	3 3 0 0 1 1 1 0 0 0 0 0 0 0	32 33 1340 12063 0 0 0 0 0 0	1695925	0 0 1 21 0 0 0 0 0 0 1 1 1	0 0 0 0 0 0 0 0 0 0
26	2 4 0 0 1 1 1 0 0 0 0 0 0 0	33 34 1341 12064 0 0 0 0 0 0	1695926	0 0 0 22 0 0 0 0 0 0 1 1 1	0 0 0 0 0 0 0 0 0 0

***1. Microstates**: each row is a microstate as a combination of the copy number of each species in the reaction system. The data structure of the microstate is considered as follows:

(**# of ProteinCI, # of ProteinCro, # of ProteinCI2, # of ProteinCro2, #of Gene1, # of Gene2, # of Gene3, # of BGene1CI, # of BGene2CI, # of BGene3CI, # of BGene1Cro, # of BGene2Cro, # of BGene3Cro**),

where # is the copy number.

***2. Transitions of Microstates**: Columns represent different reactions (R1, R2, …., R10) (see detail of reactions in A.2.1) and the value is the order number of the microstate that it will be changed to if the reaction occurs. For example, for the first microstate, if R1 occurs then it will change to the second microstate and if R2 occurs then it change to the third microstate.

A.2 Microstates Enumeration of Network of Phage Lambda

A.2.1 Computational Model

In this model, the reaction of the production of the same protein with different DNA states has been considered as the same reaction, therefore we have two Type I reactions as follows:

Type I

R1: (Gene with all states) -> Protein CI

R2: (Gene with all states) -> Protein Cro

R1 and R2 represent many reactions (see A.2.2).

Type II

R3: 2 x Protein CI-> Protein CI2

R4: 2 x Protein Cro -> Protein Cro2

Type III

R5: Protein CI2 + Gene1-> BGene1CI

R6: Protein CI2 + Gene2-> BGene2CI

R7: Protein CI2 + Gene3-> BGene3CI

R8: Protein Cro2 + Gene1-> BGene1Cro

R9: Protein Cro2 + Gene2-> BGene2Cro

R10: Protein Cro2 + Gene3-> BGene3Cro

A.2.2 Type I Reactions in the Proposed Model

In the proposed model, Gene1, Gene2, and Gene3 are the three operator sites on the gene. Each operator site can be separately bound by CI2 or Cro2. By a simple computation, it is easily known that the gene in the proposed model has 27 kinds of gene states. According to the 27 gene states, R1 (which means produce protein CI) can represent 27 reactions using mathematics (similarly, R2 also can represent 27 reactions).

1: (Gene1 + Gene2 + Gene3) -> Protein CI

2: (Gene1 + BGene2Cl + Gene3) -> Protein CI

3: (Gene1 + BGene2Cro + Gene3) -> Protein CI

4: (Gene1 + Gene2 + BGene3Cl) -> Protein CI

5: (Gene1 + Gene2 + BGene3Cro) -> Protein CI

6: (Gene1 + BGene2Cl+ BGene3Cl) -> Protein CI

7: (Gene1 + BGene2Cl + BGene3Cro) -> Protein CI

8: (Gene1 + BGene2Cro+ BGene3Cl) -> Protein CI

9: (Gene1 + BGene2Cro + BGene3Cro) -> Protein CI
10: (BGene1Cl + Gene2 + Gene3) -> Protein CI
11: (BGene1Cl + BGene2Cl + Gene3) -> Protein CI
12: (BGene1Cl + BGene2Cro + Gene3) -> Protein CI
13: (BGene1Cl + Gene2 + BGene3Cl) -> Protein CI
14: (BGene1Cl + Gene2 + BGene3Cro) -> Protein CI
15: (BGene1Cl + BGene2Cl+ BGene3Cl) -> Protein CI
16: (BGene1Cl + BGene2Cl + BGene3Cro) -> Protein CI
17: (BGene1Cl + BGene2Cro+ BGene3Cl) -> Protein CI
18: (BGene1Cl + BGene2Cro + BGene3Cro) -> Protein CI
19: (BGene1Cro + Gene2 + Gene3) -> Protein CI
20: (BGene1Cro + BGene2Cl + Gene3) -> Protein CI
21: (BGene1Cro + BGene2Cro + Gene3) -> Protein CI
22: (BGene1Cro + Gene2 + BGene3Cl) -> Protein CI
23: (BGene1Cro + Gene2 + BGene3Cro) -> Protein CI
24: (BGene1Cro + BGene2Cl+ BGene3Cl) -> Protein CI
25: (BGene1Cro + BGene2Cl + BGene3Cro) -> Protein CI
26: (BGene1Cro + BGene2Cro+ BGene3Cl) -> Protein CI
27: (BGene1Cro + BGene2Cro + BGene3Cro) -> Protein CI

It is possible that not all reactions are available in an actual biological reaction system. However, these reactions can assist in building a complete Matrix Network, which contains two sets. One set contains all the possible microstates (M-Matrices) and the other contains the transitions between microstates (R-Matrix). Owing to the data processes, these unavailable reactions will increase the unavailable transitions, which are represented as coefficients in the transitions matrix.

In our case, we provide three solutions to address these unavailable reactions:

①. Do nothing if only the enumeration of the microstates is required.
②. Remove some of the unavailable transitions by replacing the coefficients in the transitions matrix by zero. Using A.1 as a reference, in Fig. 8 we remove the unavailable links by setting pertinent coefficients to zero.
③. Assign these unavailable reactions with a rate of 0.

In this paper, we propose the use of Matrix Network to generate the microstates. Therefore, we choose the first solution and enumerate the microstates in the same manner as computing the Toggle Switch Network.

A.2.3 Example of Results

Under the condition of the maximum copy number of net molecules synthesized in the system set to 50, the proposed method enumerates 1,695,926 microstates and the microstate transitions. A portion of the results can be seen in **Table 3**.

Xiao Cong had engaged in research of bioinformatics algorithms and methods at Akutsu laboratory (laboratory of mathematical bioinformatics), Kyoto University. He received his master's degree from Kyoto University in 2014. In the same year, he joined Rakuten Inc. as an application engineer. He is interested in developing bioinformatics algorithm and data process relating technology and science.

Tatsuya Akutsu received his B.Eng. and M.Eng. in Aeronautics and D.Eng. in Information Engineering from The University of Tokyo, 1984, 1986 and 1989, respectively. From 1989 to 1994, he was with Mechanical Engineering Laboratory. From 1994 to 1996, he was an Associate Professor in the Department of Computer Science at Gunma University. From 1996 to 2001, he was an Associate Professor in Human Genome Center, Institute of Medical Science, The University of Tokyo. Since 2001, he has been a Professor in Bioinformatics Center, Institute for Chemical Research, Kyoto University. His research interests include bioinformatics and the design and analysis of algorithm.