Modeling as the Essential Step in the Construction of Biological Computer Structures

Miha Moškon and Miha Mraz

***Abstract*—We present the properties of simple biological systems which are similar to the properties of computer structures that are used today. Biological computer structures can therefore be synthetized in these manners. In order to design such systems more straightforwardly, *in silico* approaches, i.e. computer modeling can be used. We present the basic steps for the design of biological computer structures among which modeling is described in details. Different models of potential biological computer structures can be established and their analysis can be made before *in vivo* realization. Here we present a sample model of such system and its analysis from the computer structure's viewpoint. In order to perform such analysis different metrics are introduced.**

***Index Terms*—Gene regulatory networks, modeling, transcription based logic, unconventional computing.**

I. Introduction

Basic dynamical properties of modern computer systems are *data processing*, *data memorization* and *data transmission* capability. According to the Moore's law, which predicts the duplication of integrated circuits' complexity every 18 months [1], the continuous minimization of computer systems will bring us to nanometer scale sizes in approximately 10 years. According to the complexity of integrated circuits fabrication techniques which will be unable to follow this trend in such manners, researches on different platforms that promise data processing capabilities are performed. Potential future data processing platforms should be able to overcome the barriers which will prevent the further development of electronic integrated circuits in the meaning of their sizes and in the meaning of their speed. One of the most promising future data processing platforms are also biological systems, which are based on the genetic instructions encoded on the DNA strand. Synthetic biology [2] is a rapidly evolving discipline that combines knowledge from different fields and is aimed toward the realization of novel biological systems with predefined functionalities. Its methods and approaches can also be used in order to construct biological systems with data processing capabilities, which could present potential future data processing platform.

Data processing potential of DNA governed biological systems was shown for the first time two decades ago with an Adleman's experiment which showed that DNA processing can be used in order to solve certain set of NP-complete problems [3]. Large numbers of simple biological systems that can perform as logic gates, oscillators, flip-flops or counters were introduced in the last decade [4-8]. On the other hand methods which were used in their construction were not precise. Success in their construction was a consequence of different experimental procedures with different chemical species. From the computer engineering viewpoint which tries to construct the basic primitives which present complete set of logical functions and consequently connect them into more complex systems the imprecise approaches are not suitable while only simple structures can be built in these manners.

Here we present basic dynamical properties of simple biological systems which have their behavior governed by the instructions encoded on the DNA strand. We analyze their suitabilities from the data processing and data memorizing viewpoints. We also introduce the steps that are necessary in order to synthesize biological systems with data processing and data memorizing capabilities.

Manuscript received May 21, 2012, revised June 25, 2012. The research was supported by the scientific-research programme Ubiquitous Computing (P2-0359) financed by Slovenian Research Agency in years from 2009 to 2012.

The authors are with the University of Ljubljana, Faculty of Computer and Information science, Tržaška cesta 25, SI-1000 Ljubljana, Slovenia (e-mail: miha.moskon@fri.uni-lj.si, miha.mraz@fri.uni-lj.si)

II. Basics of Dynamics In Biological Systems

A. Basics of DNA Expression

Basic term in synthetic biology is a *genome*, which presents the complete hereditary information of a certain organism. It defines the organism's dynamics through different stages of life which are dependent on the genome's environment. Genome is constructed of two *DNA strands*, which interact among each other in the *helix*. Each of the strands is conducted of four different bases, namely *Cytosine (C), Guanine (G), Thymine (T)* and *Adenine (A)*. Hydrogen bonds connect each base to its *complementary pair* on the other DNA strand in the helix. Base *C* (*G*) on the first strand is complementary to the base G (*C*) on the second strand and base *A* (*T*) on the first strand is complementary to the base T (*A*) on the second strand. Genome is therefore constructed of a sequence of so called *base pairs* (*bp*). Its length is dependent on the belonging organism. For example, the length of a human genome is approximately $3 \cdot 10^9$ *bp*. The basic scheme of the genome is presented in Fig. 1.

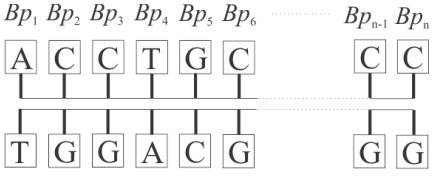

Fig. 1. Genome as a sequence of complementary base pairs.

Each genome can be presented by the sequence of so called *genes*, where each gene describes a certain property of an organism and is a *basic hereditary unit*. Current researches indicate that the number of genes in human organism is somewhere between 20,000 and 25,000. Each gene can be further divided on two parts, i.e. its *regulatory region (promoter)*, which defines the intensity of its *expression* and its *functional part* (*program*), which defines the result of its expression. The basic scheme of the gene is presented in Fig. 2.

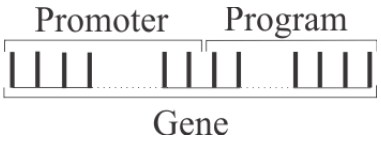

Fig. 2. Genes can be divided in a regulatory region called promoter which defines the conditions for their expression and a program which defines the results of its expression.

Gene expression results the synthesis of so called *output protein*. Intensity of its expression is defined with the presence of *RNA polymerase* and other *transcription factors* which are encoded with the promoter region. RNA polymerase must always be present in order to start the expression. Other transcription factors can be further divided on two groups, i.e. so called *repressors*, which inhibit the expression of the gene with their presence and so called *activators* which induce the expression of the gene with their presence. The role of transcription factors in the system is performed by the *proteins*, which can be a result of an observed gene (feedback loop), a result of some other gene or an input from an outside world. The influence of transcription factors on gene expression is presented in Fig. 3.

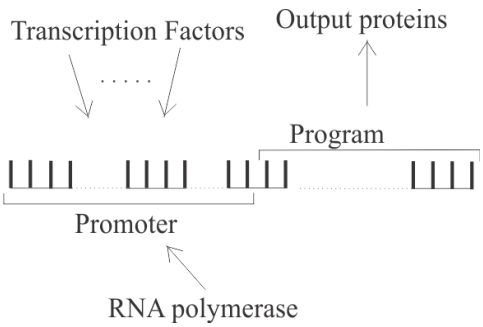

Fig. 3. Regulation of gene expression.

B. Dynamics of Gene Expression

Gene expression can be divided on two phases, i.e. *transcription* and *translation*. The direct result of an expression is an output protein. Its structure is defined with the functional part of the gene (program). Its role in the system is defined with the promoter region of other genes in the system.

Transcription is initiated when RNA polymerase is bounded to promoter with the presence, respectively absence of other transcription factors (see Figure 3). Transcription is not initiated if predefined activators are not present or if predefined repressors are present. After the transcription, translation is initiated for which we can presume that is performed unconditionally. Output protein synthesis is thus only dependent on the absence, respectively presence of transcriptional factors. It is possible to construct artificial genes and therefore define the type and the structure of transcription factors and also the structure of the output proteins that will be expressed.

III. DNA Based Biological Systems as Computer Structures

Here we present the main properties of DNA based biological systems as potential computer structures which present the motivation for our further work:

- *Existence of coding*: each gene can be interpreted as a sequence of codes which are included within the set *{T, G, A, C}*.
- *Density of coding*: with the 0.35 *nm* distance between the bases the density of coding which can be also interpreted as a memory storage space density is 18 megabits per inch which leads us to the 10^6 gigabits/inch^2. When comparing to magnetic hard drives available today this density is 10^3-times higher.
- *Coding redundancy:* each code (*A*, *C*, *T*, *G*) on the first strand has its complementary pair on the second strand in the helix. Ambiguity on the strand can therefore be identified when comparing the contents of each base pair.
- *Linearity of the record on the strand:* linear succession of the record on the strand leads us to the possibility of linear addressing which is inherent to modern computer systems.
- *Inherent parallelism*: dynamics in biological systems is performed consequently on different genes if the conditions for their expression are fulfilled. Parallelism is therefore inherent in such systems.
- *Existence of the program*: functional part of the gene can be interpreted as a program, which defines an output of a certain part of the system (output protein), which is similar to the structure of ordinary computer systems.

Arguments listed above lead us to the conclusion that the nature has designed a system, which calls for the exploitation in the meaning of data processing, data memorization and data transmission. An extension of Fig. 3 is presented in Fig. 4 where power supply, inputs and outputs of biological system performing as a data processing platform are identified.

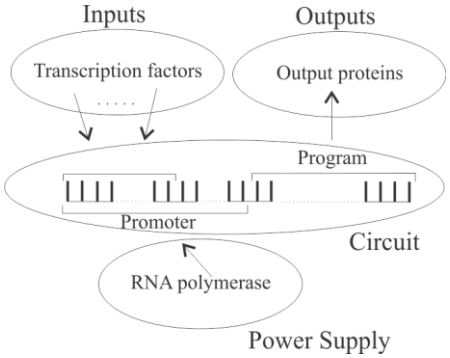

Fig. 4. Gene expression based logic circuit as an data processing platform with denoted input, output and power supply.

Outputs of the circuit (output proteins) are determined

with its inputs (proteins that have a role of transcription factors) and their structure which is defined inside the program of the circuit. Circuit is functional only when power supply (*RNA polymerase*) is present. Strength of the inputs and outputs of the circuit can be measured with the *concentrations* of input and output proteins.

Biological circuit presented in Fig. 4 can present one of the modules that construct our biological system. Product of a certain gene (i.e. output of the circuit) can serve as a transcription factor to some other gene (i.e. input of the circuit) or even as a transcription factor for its own expression (i.e. feedback input). Therefore presumption that output of a certain module serves as an input to some other module can be made. In that way complex systems can be built. Modules are thus connected into complex networks called *gene regulatory networks*.

While the program is written on the DNA strand it is encoded with four different values (*C*, *G*, *T* and *A*). We can therefore make a transition from a binary coding in classical computer structures to *quaternary coding* on the DNA strand.

IV. Basic Steps for the Design of Biological Computer Structures

Basic steps which lead us to the successful design and consequently to the synthesis of biological computer structures should be as follows:

- Identification of an inputs (defined with the regulatory region of the gene – promoter), identification of a program (defined with the functional part of the gene) and identification of an output (proteins that are result of an expression).
- Modeling and analysis of the biological system's dynamics.
- In-vivo realization of the system.

Most of the procedures used today are skipping the second step, i.e. in-vivo realization is performed without the modeling and analysis of the system's dynamics. Comprehension of this step would allow the designers to verify the system's dynamics before the realization, which would drastically reduce the experimental work and therefore reduce the time and expenses consumed. On the other hand systems with higher complexity could be built in these manners.

In the rest of the paper we present the modeling and analysis of an example of a biological computer structure. The analysis is performed with the evaluation of metrics which were introduce in order to objectively estimate the data processing capabilities of biological systems.

V. Modelling the Dynamics of Gene Expression in Biological Systems

Different types of models are used in order to be able to predict the dynamics of gene expression in the designed biological circuits. The behavior of the circuit can therefore be verified *in silico*, i.e. with different computational approaches, before *in vivo* realization is performed. As the number of observed chemical species (i.e. observed proteins, promoters, genes, etc.) and the number of reactions is increased the complexity of biological systems drastically grows. Using the models the prediction of their behavior is still possible. Even more, models can be used in order to decrease the complexity and the amount of experimental work [9], [10]. We can also use *in silico* approach in order to analyze the data processing capabilities within the designed biological system. In that manners different parameters of biological system's *switching dynamics*, i.e. *rise time*, *fall time*, *refresh rate*, etc. can be estimated (see Section VI).

Different types of approaches for modeling the dynamics in biological systems can be used. Models used for modeling the dynamics of gene expression in biological systems can be roughly divided to *deterministic*, *stochastic* and *semiquantitative* [9-11]. Deterministic models are mainly based on *ordinary differential equations* (ODEs). Such models are relatively easy to construct. On the other hand they are unable to predict the influence of noise which can be inherent to observed biological system, i.e. *intrinsic noise* or can be a consequence of external factors, i.e. *extrinsic noise* [12]. If noise is ignored results of such modeling are not in accordance to experimental results in many cases. In order to include the effects of noise, stochastic models can be used. The complexity of such models is much higher. If we are not dealing with trivial biological systems such models cannot be solved precisely. Different approximate approaches are used such as *stochastic simulation algorithm* (SSA) [13] or *τ-leaping method* in order to solve them. If these approaches fail too, *semiquantitative* models can be used which present a compromise between the complexity of stochastic models and simplicity of deterministic ones. Models can be further differentiated within these three groups, but deterministic models are mostly based on so called *Hill equations* [9] and *mass-action kinetics* and stochastic models on *Chemical Master Equation* (CME) [11].

TABLE I: RS Flip-Flop Behaviour, Where R Presents Reset Input, *S* Set Input, *Q* Current Output of the Flip-Flop and D^1Q Output of the Flip-Flop in the Next Time Step. Undefined Output Value is Denoted with "?".

R	***S***	D^1q
0	*0*	*q*
0	*1*	*1*
1	*0*	*0*
1	*1*	*?*

VI. Analyzing the Data Processing Capabilities on the Example of Biological Circuit

A. Toggle Switch Model

We will present the analysis of the data processing capabilities on an example model of *Toggle Switch* circuit [7] which can be interpreted as a biological equivalent of *RS flip-flop*. RS flip-flop is a basic memory element in digital circuits. Its behavior can be described with the look-up table presented in Table I.

The state of the flip-flop (*q*) is preserved when *set* (*S*) and *reset* (*R*) inputs are inactive. If reset input is activated, state of the flip-flop is set to logical 0. If set input is activated, its state is set to logical 1. We can achieve a comparable behavior with a simple gene expression based logic biological circuit

presented in Fig. 5 which is called Toggle Switch circuit.

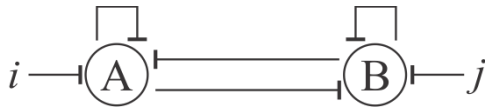

Fig. 5. Biological equivalent of RS flip-flop memory cell, i.e. Toggle Switch. Observed proteins are denoted with A and B, *i* and *j* present the factors which increase a certain protein degradation rate. Arrows present the direction of the repression.

TABLE II: TOGGLE SWITCH BEHAVIOUR, WHERE COLUMNS *I* AND *J* PRESENT THE PRESENCE, RESPECTIVELY ABSENCE OF EXTERNAL FACTORS AND COLUMNS D^1A AND D^1B PRESENT THE CONCENTRATION OF PROTEINS *A* AND *B* IN NEXT TIME STEP.

i	*j*	D^1A	D^1B
absent	*absent*	*A*	*B*
absent	*present*	*high*	*low*
present	*absent*	*low*	*high*
present	*present*	*?*	*?*

Two proteins are observed within the Toggle Switch circuit, i.e. protein A and protein B. As can be seen in Figure 5 protein A represses protein B and vice versa. When the protein A is absent from the system, protein B is present with high concentration and when the protein B is absent, protein A is present with high concentration. In order to limit the high level concentration, each of the proteins also represses itself. Switch from one state to another can be performed with the introduction of one of the external inputs, i.e. *i* or *j*, which drastically decrease the degradation rate of a specific protein, i.e. *i* increases the degradation rate of protein *A* and *j* increases the degradation rate of protein B. Switch from the state, where protein A is present with high and protein B with low concentration is therefore performed with the temporal introduction of external factor *i*. Concentration of protein A drastically decreases and protein B is no longer repressed. Its concentration increases. State where A is present with low and B with high concentration is achieved and sustained even after the external factor is removed from the system. We can present the described behavior with Table II.

We presume that high concentration of observed proteins presents logical value 1 and low concentration logical value 0. We can presume that the concentration of protein A has the same role as the signal *q* in RS flip-flop circuit. We can therefore interpret the concentration of protein B as the signal *q* and *i* and *j* as reset and set inputs of the circuit.

B. Simulation Results of Toggle Switch Model

In order to analyze the data processing capabilities of Toggle Switch circuit series of simulations were performed. Goal of the simulations was to estimate the following parameters:

- logical levels,
- switching times,
- refresh rate.

1) Estimating the Logical Levels

We presume that high concentration of a specific protein presents logical value 1 and low concentration logical value 0. We wanted to estimate specific concentration values that present certain logical level. We measured the stable concentrations of each protein after the switch was performed (see Fig. 6). Regarding the noise, logical value 1 was estimated with concentrations higher than 70 *nM* (i.e. 70 nanomoles per liter) (C_H > 70 *nM*) and logical value 0 with concentrations lower than 20 *nM* (C_L < 20 *nM*). The region between these two values is so called uncertainty region, in which concentrations do not present valid logical levels.

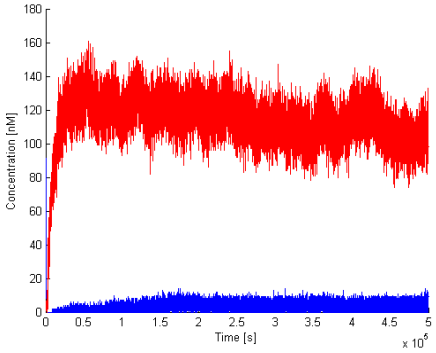

Fig. 6. Estimating the logical levels where red color presents the concentration of protein A and blue color presents the concentration of protein B.

2) Estimating the Switching Times

Two different switching times were estimated, i.e. time needed to perform a switch from logical value 0 to logical value 1 (rise time) and time needed to perform a switch from logical value 1 to logical value 0 (fall time). We can estimate the rise time (t_r) as the time needed to get the average value of the protein concentration from 10% of the concentration presenting logical 1 (*CH*) to 90% of the concentration presenting logical 1 (in our example from 7 *nM* to 63 *nM*) [14] (see Fig. 7). Similarly fall time (t_f) can be estimated as the time needed to get the average value of the protein concentration from 90% of C_H to 10% of C_H (see Fig. 8). It was estimated that rise time equals 4000 s and fall time 150 s. In order to perform a switch both protein concentrations need to be stabilized. Time to perform a switch therefore equals 4000 s.

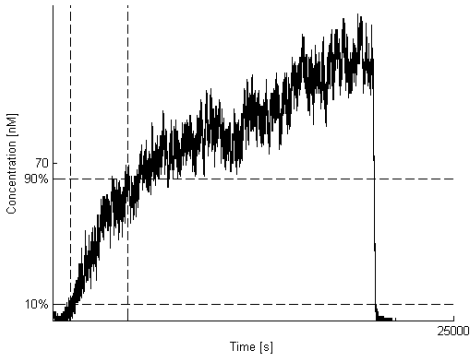

Fig. 7. Estimating the rise time.

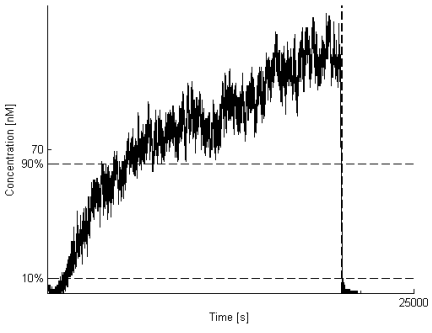

Fig. 8. Estimating the fall time.

3) Estimating the refresh rate

Ideally state of the system would be memorized for unlimited amount of time. As time passes it is possible that transition from stable state to so called metastable state, i.e. both protein concentrations are within the uncertainty region, is reached (as in Dynamic RAM circuits). It is necessary to introduce refreshing of the state in such circuits. Period of refreshing is defined with refresh rate and it has to be defined in such a way, that metastable state is never reached. It is obvious that the state is preserved for very large amount of time in our system (See Fig. 9). According to simulation results refreshing is not necessary for at least 1000 days.

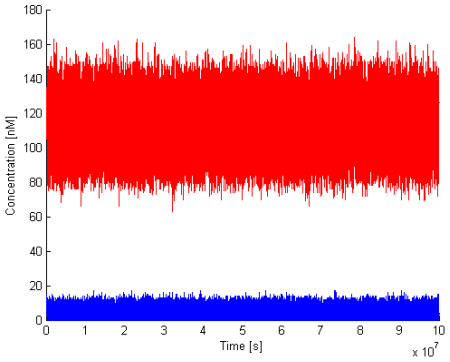

Fig. 9. Memorizing the state of the system for very large amount of time (more than 1000 days). Red color presents the concentration of protein A and blue color presents the concentration of protein B.

VII. Conclusion

Analysis of data processing capabilities on an example of Toggle Switch model was presented in the article. We propose that similar analyses should be made before *in vivo* realization of such circuits is performed. Their suitabilities as data processing building blocks can thus be evaluated objectively. In that way more complex biological computer structures can be made and finally biological computer could be realized.

Acknowledgment

Results presented here are in scope of PhD thesis that is being prepared by Miha Moškon.

References

[1] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, pp. 114–117, 1965.

[2] S. A. Benner and A. M. Sismour, "Synthetic biology," *Nature Reviews*, vol. 6, pp. 533–543, 2005.

[3] L. M. Adleman, "Molecular computation of solutions to combinatorial problems," *Science*, vol. 266, pp. 1021–1024, 1994.

[4] M. B. Elowitz and S. Leibler, "A synthetic oscillatory network of transcriptional regulators," *Nature*, vol. 403, pp. 335–338, 2000.

[5] A. E. Friedland, T. K. Lu, X. Wang, D. Shi, G. Church, and J. J. Collins, "Synthetic gene networks that count," *Science*, vol. 324, pp. 1199–1202, 2009.

[6] G. Fritz, N. E. Buchler, T. Hwa, and U. Gerland, "Designing sequential transcription logic: a simple genetic circuit for conditional memory," *Syst Synth Biol*, vol. 1, pp. 89–98, 2007.

[7] T. S. Gardner, C. R. Cantor, and J. J. Collins, "Construction of a genetic toggle switch in Escherichia coli," *Nature*, vol. 403, pp. 339–342, 2000.

[8] R. Weiss, G. Homsy, and T. Knight, "Toward in vivo digital circuits," in *Proceedings of the Dimacs Workshop on Evolution as Computation*, 1999.

[9] U. Alon, *An Introduction to Systems Biology*, A. M. Etheridge, Ed. Chapman and Hall/CRC, 2007.

[10] M. Kaern, W. J. Blake, and J. Collins, "The engineering of gene regulatory networks," *Annu. Rev. Biomed. Eng.*, vol. 5, pp. 179–206, 2003.

[11] H. El Samad, M. Khammash, L. Petzold, and D. Gillespie, "Stochastic modeling of gene regulatory networks," *International Journal of Robust and Nonlinear Control*, vol. 15, pp. 691–711, 2005.

[12] P. R. Patnaik, "External, extrinsic and intrinsic noise in cellular systems: analogies and implications for protein synthesis," *Biotechnology and Molecular Biology Review*, vol. 1, pp. 121–127, 2006.

[13] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *The Journal of Physical Chemistry*, vol. 81, pp. 2340–2361, 1977.

[14] J. F. Wakerly, *Digital Design: Principles and Practices Package, 4th Edition*. Prentice Hall International, Inc., 2005.

Miha Moškon was born on October 28, 1983 in Ljubljana, Slovenia. He received his BSc degree in Computer Science from the Faculty of Computer and Information Science, University of Ljubljana, Slovenia in 2007. He is currently pursuing a PhD degree of computer science on the Faculty of Computer and Information Science, University of Ljubljana, Slovenia. He is currently employed as a teaching assistant in the Computer Structures and Systems Laboratory at the Faculty of Computer and Information Science, University of Ljubljana, Slovenia, where he is in charge of the following laboratory courses: Introduction to Digital Circuits, Computer Networks Modeling, Mobile and Wireless Networks and Unconventional Processing Platforms.

His research interests are mainly directed towards unconventional computing. He is also interested in fuzzy logic applications and artificial life. He has published his work in several journals and conference proceedings.

Miha Mraz was born on August 20, 1966 in Ljubljana, Slovenia. He received his BSc, MSc and PhD degree in computer science from the Faculty of Computer and Information Science, University of Ljubljana, Slovenia in 1992, 1995 and 2000. He is currently employed as an associate professor in the Computer Structures and Systems Laboratory at the Faculty of Computer and Information Science, University of Ljubljana, Slovenia. He holds this position since 2006.

He is in charge of the following courses: Unconventional Processing Platforms, Computer Networks Modeling and System Reliability and Performance.

His research interests are recently directed towards unconventional processing methods and unconventional processing platforms, such as fuzzy logic, synthetic DNA systems and QCA structures. He has published his work in many distinguished journals such as Nanotechnology, International Journal on Unconventional computing, Japanese journal of applied physics, etc. He is a member of IEEE professional society.