BIOLOGICAL CELL AND ITS SYSTEM SOFTWARE

N. Bozinovska[1], Gj. Jovancevski[1], S. Bozinovski[2]

Institute of Informatics, Faculty of Natural Sciences and Mathematics,
Sts. Cyril and Methodius University,
Arhimedova bb, PO BOX 162, Skopje, Macedonia
{nevena, gjorgji}@ii.edu.mk
[2]Electrical Engineering Faculty
Sts "Cyril and Methodius" University,
1000 Skopje, Macedonia
bozinovs@rea.etf.ukim.edu.mk

Abstract: Looking for a role that DNA has in the cell, we are elaborating that DNA can be viewed as a Cell Operating System. In the paper we consider several aspects of that approach and we propose some possible explanations for some of the not yet understood phenomena in molecular genetics in terms of systems software.

Keywords: computer science, biological cell, protein biosynthesis, cell control system, cell system software, cell operating system

1. Introduction

In computer science there is a growing interest in molecular genetics. Most of the interest of the researchers in computer science so far has been directed toward the DNA computing. Both the fundamental processes in molecular genetics: *replication* and *protein biosynthesis* are governed by the DNA molecule.

From the computer science viewpoint, we are looking for the best metaphor that will describe the DNA, taking into account both the fundamental processes. Recently, a new view toward DNA was proposed [4]: DNA is actually a cell Operating System which contains and takes care of a huge database, but also governs all the complex control processes. It is believed that this view has a potential to offer some answers to relevant questions in molecular cell biology. This paper elaborates further on that idea.

2. The Cell as an Agent

The framework of our study bases on the observation that the biological cell is an agent that exists in an environment in which in which it behaves and interacts, as shown in Fig. 1.

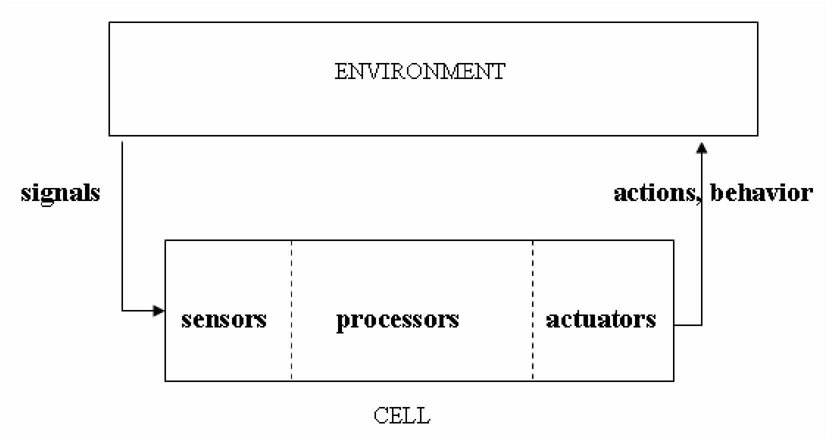

Figure 1: The cell as an agent

The cell receives signals from the environment and responds to those signals. The cell has special sensors for various signals, and some of the cells have motors to make movable actions in the environment. There are three types of response of a cell: behavioral response (for example movement), a product manufacturing response, and cell division and multiplication (generation change).

3. Central Dogma

The central dogma of molecular genetics states that the information processing in genetic systems goes in the direction DNA → RNA → protein.

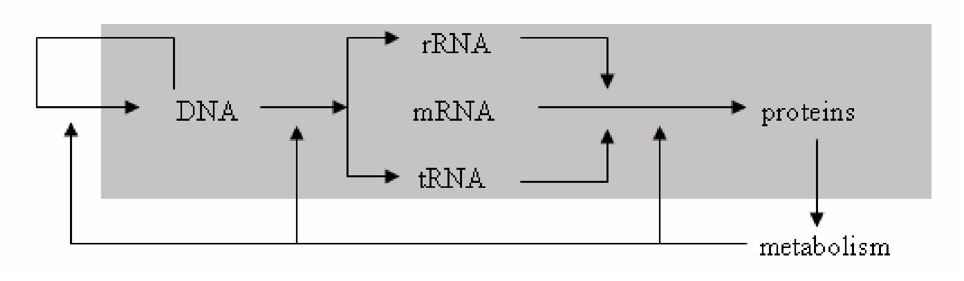

Figure 2: Extended central dogma with the regulatory feedback

Fig. 2 represents what we call extended central dogma of molecular genetics where *protein feedback control mechanisms* are included.

The process of replication is carried out by many enzymes, the most prominent being the *DNA polymerase*, which reads DNA and produces new DNA. The process of transcription is carried out by many enzymes, the most prominent being the *RNA polymerase*, which reads DNA and produces new RNA. The process of translation is carried out by many enzymes, the most prominent being the *ribosome*, which reads RNA and produces new protein. All of the mentioned enzymes act like Turing machines. They read an information tape and produce another information tape.

4. The Cell Control System

The protein synthesis process can be represented as a system of two pipelined Turing machines, the RNA polymerase, and the ribosome (Fig. 3.). The ribosome has two distinct subunits, which from the Turing machines viewpoint can be recognized as reading and a writing head respectively. The system shown in Fig. 3, which given a *gene produces a protein(s)*, is denoted as *genozome* [5]. Gene is a segment of the DNA that contains program or data for producing a protein or RNA. The DNA is the cell database [3, 1].

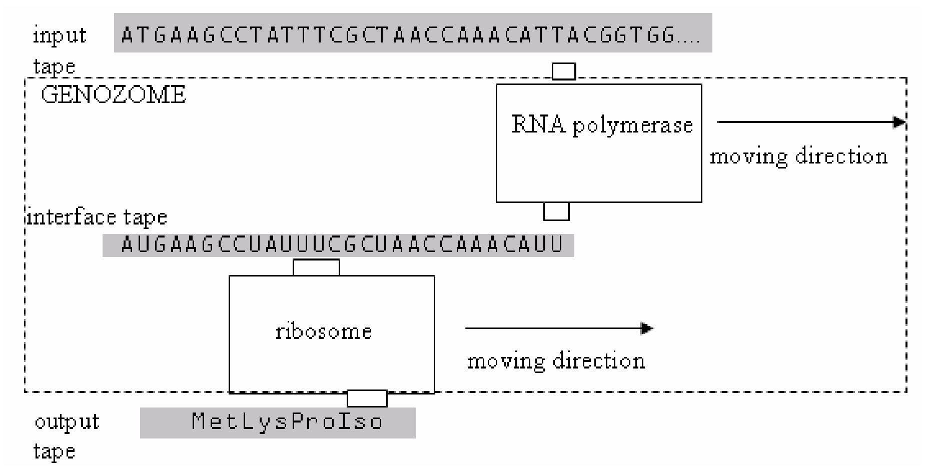

Figure 3: Genozome as system of Turing machines

The genozome is a component, a module of global metabolic control system. Fig. 4 uses such a view and describes the basic control circuit in prokaryote genetics, the operon system.

Fig. 4 shows that a genozome is activated when the associated gene is activated. The activation of a gene is controlled by the corresponding operon. An operon is a control structure that consists of regulated genes and at least two regulatory strings, promoter and operator. The promoter sends an activation signal to the regulated genes to start their expression, but the operator is a control gate that enables or disables the activation signal. Operator is sensitive to the environment changes by means of signal proteins. The signal proteins represent events, some change of the state of the environment, on which the cell reacts producing possibly several event-related proteins. In a feedback setup, the protein will deal with the event that was a reason for its production.

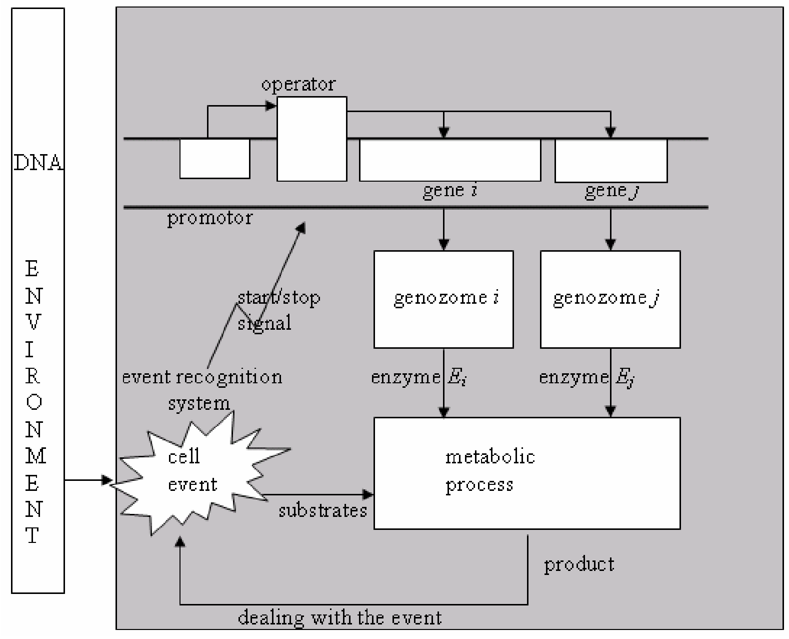

Figure 4: Dealing with an event by cell control circuit

There are several levels of hierarchical control systems over the operon level. Several operons can be controlled by a regulatory system which is denoted as regulon, and a set of several operons and regulons can be controlled by a control system denoted as modulon. The number of hierarchical levels of the cell control system is unknown.

Cell regulatory systems have been found at the level of transcription, post-transcription editing, exporting RNAs to the cytoplasm, translation control (mRNA degradation) and the protein activity control.

5. The Cell System Software

The cell is governed by an event recognition and control systems, at several levels. However our observation is that there should be a general control structure that deals with all the priorities. For example, when a cell is going to replicate, all other processes are oriented toward the support of the replication.

To get the feeling of the potential control complexity, the three main levels of activities should be considered: the *resources*, the *processors* and the *processes*. Let us look at the potential complexity of *Escherichia coli*. It has a genome (set of all genes) of 4800 genes, a proteome (set of all proteins) of about 2500 proteins and a metabolome (set of all metabolic reactions) of an unknown number. Not all the proteins are processors, but we can estimate it is a parallel distributed processing system of about thousand processors executing their programs written in a genome, and carrying out an unknown number of processes.

The observation that the cells have to manage processes, resources (including memory management and processor management) by assigning priorities, leads to the conclusion that this is a job of an Operating System. We observe that a Cell Operating System (COS) exists, and it resides in the DNA. Furthermore, it can be observed as a database operating system which has a large database, DNA, to take care for. Database operating systems are of interest in systems software.

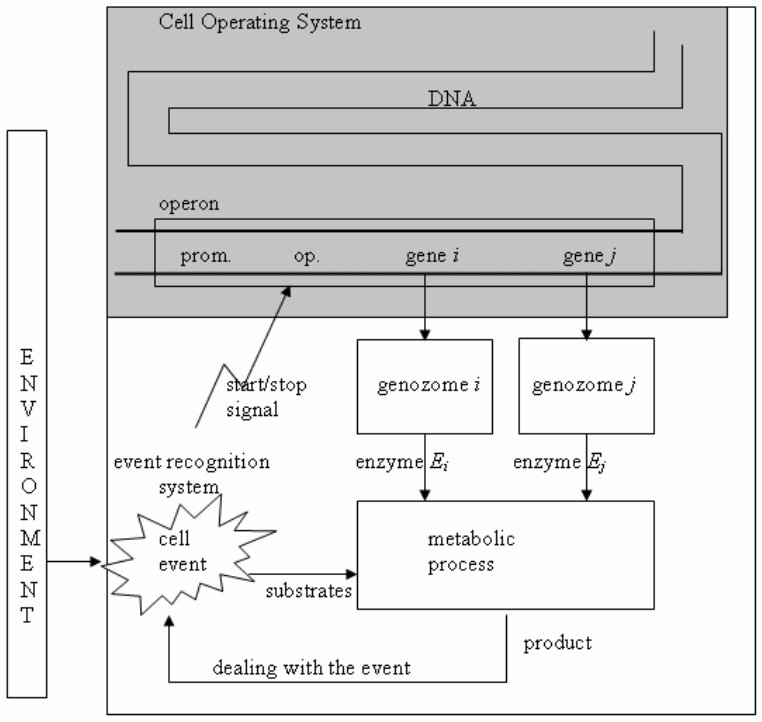

Figure 5: Dealing with an event by a cell operating system

Fig. 5 considers the Cell Operating System (COS) as a crucial concept of the cell control system. As shown in Fig. 5, the lower level of control system consists of various feedforward and feedback loops controlled by operons or other control structures. For example, the activation of a genozome is in the feedforward loop of the event, while the acting proteins are in the feedback loop to compensate the event. The Cell Operating System (COS) as a *real-time operating system*, comprises the hierarchy of control levels of the cell control system including the information system control.

Viewing the DNA control segments as elements of the Cell Operating System, we have tried to find elements of homomorphism with the human made, computer operating systems.

One of the basic concepts of an operating system is a concept of file. Looking for such a concept in DNA we can say that the scriptons are cell files. Scripton is a transcription unit, a part of DNA that is transcribed to RNA. An operating system can not copy less than a file with its standard operating system operations. The cell also cannot transcribe less than a scripton.

Further, we recognize the operon as a monitor a piece of software that contains the data, but also it controls the access over its data. It is an object in a sense that it contains data structures (genes) and control structures (operators) for accessing the data. This suggests that the prokaryote systems software is object oriented. In the cell hierarchical control system, the operon is the smallest object.

From the operating system point of view, it is natural that some files are fragmented. After years of evolution the eukaryotic file system has developed possibility for fragmenting its files. The space between the fragmented files, the intron, can be just a space determining distance between exons, or a segment of another file or program. Example of an intron-program is the *Tetrahymena* ribozyme, which, being an intron, cleaves itself from the precursor mRNA. The intron splicing process can be viewed as the well known "garbage collection" process performed by the operating systems.

The exons can be viewed as subprograms, programs, or data of modular software. In some cases they are pieces of reusable software, since combination of exons can produce different programs, as in the case with program preparation for antibodies. So, the RNA processing can be viewed as a linkage editing process, linkage editor being a system software concept from the old OS/360 systems. Since the exons of a cell program can be distributed among several chromosomes, the cell operating system is a typical distributed file system.

Genetic files can contain data and executable programs. For example, mRNA is a program which is used in its primary, text structure. In contrast, tRNA and rRNA are executable pieces of software and are used as a tertiary structure. Their folding into secondary and tertiary structure is actually the compilation process of the Cell Operating System. By folding, tRNA and rRNA become executable space programs, actually executable machines. As executable programs they still use some of their data structures, such as anticodon sequence used by tRNA. The executable software is encoded in the space representation of the cell processors, enzymes and ribozymes.

The Cell Operating System has mechanism for preserving the database and the programs against entropy (random mutations) and against its own mistakes. For example, the RNA polymerase has mechanisms of self-correction of some kind of errors. Due to the evolution, significant changes have been made to the database and the whole operating system. However, some of the sequences, for example the part of the rRNA program that enables moving of the amino acid to the evolving protein, seems to be preserved over the evolution tree of the species.

One of the problems in operating systems, and the concurrent engineering in general, is management of the access to resources. A mutual exclusion mechanism is usually applied in standard operating systems, when several processes attempt to use the same resource. Part of that problem in DNA is solved by replication of the frequently used data. For example, the same sequences are found on several places of DNA. So, the access to those sequences can be done in parallel, and the problem is solved by a *memory management* mechanism rather than only by a *process management* mechanism. The redundancy enables parallelism and also possible recovery of mutations. When several copies are kept, a database operating system can keep the consistency against mutations using a “voting” mechanism: if mutation appears in one copy, it can be recovered regarding the content of other copies.

An important issue is the self migration of the Cell Operating System. In some prokaryotes the system is transmitted as a whole. For example, a bacteriophage operating system transports itself as a whole. In eukaryotes, in order to self-migrate, the Cell Operating System has a problem with the huge amount of data that should be transferred as a file over an insecure environment. A solution is found in making packages, chromosomes, which can be transmitted regardless of the sequence. Upon arriving at the destination the correct sequence will be restored and make the operating systems operational. Chromosomes contain compressed (“.zip”) file structures. In the so called cell interphase, when cell OS is stationary (non-migrating), the files are uncompressed, at least those which are transcribed. In the interphase, eukaryotic chromosomes can be viewed as resident disks of distributed database operating system. Those disks have multiple reading heads, and actually each RNA polymerase is such a head.

6. Discussion and Conclusion

Various metaphors have been used in order to understand the complexity of protein biosynthesis system in the cell. Examples of those metaphors are the linguistic metaphor, the factory metaphor and the flexible manufacturing systems (FMS) metaphor.

This paper elaborates the idea that some kind of operating system exists in the cell. This leads to the proposal of the concept of the Cell Operating System, a kind of real time, database operating system that manages the activities over cell database and real-time activities carried by the cell processors. This opens a new viewpoint, not used before in cell research, the *systems software metaphor*. We believe that this metaphor offers some candidate explanations for relevant and non-answered questions in molecular biology, as what are exons, what are cell data files, what are executable files, what are chromosomes etc.

We believe that the systems software metaphor is a promising one. However, we are aware that this new metaphor is just one of the many metaphors that will be used before humankind really recognizes, explains and systemizes the knowledge about all the phenomena in a biological cell.

7. References

1. S. Bolsover, J. Hyams, S. Jones, E. Shephard, H. White, From Genes to Cells, Wiley – Liss
2. S. Bozinovski, Operating Systems and System Software, Gocmar, 1998
3. S. Bozinovski, L. Bozinovska, "Flexible production lines in genetics: A model of the protein biosynthesis process", Proceedings of the International Symposium on robotics, p. 1-4, Dubrovnik, 1987,
4. S. Bozinovski, G. Jovancevski, N. Bozinovska, "DNA as a Real Time, Database Operating System", Proceedings of the World Multiconference on Systems, Cybernetics and Informatics, p. 65-70, Orlando, 2001
5. S. Bozinovski, B. Muller, F. Primio, Biometric Autonomous Factories: Autonomous Manufacturing Systems and System Software, GMD Report 115, German National Center for Information Technology, Sankt Augustin, 2000
6. C. Guyton, J.E. Hall, Textbook of Medical Physiology, W. B. Saunders Company, 1996