DNA computing for combinational logic

Chuan ZHANG[1,2,3*†], Lulu GE[1,2,3†], Yuchen ZHUANG[1,2,3], Ziyuan SHEN[1,2,3],
Zhiwei ZHONG[1,2,3], Zaichen ZHANG[2,3] & Xiaohu YOU[2]

[1]*Lab of Efficient Architectures for Digital-communication and Signal-processing (LEADS),*
Southeast University, Nanjing 211189, China;
[2]*National Mobile Communications Research Laboratory, Southeast University, Nanjing 211189, China;*
[3]*Quantum Information Center, Southeast University, Nanjing 211189, China*

Abstract With the progressive scale-down of semiconductor's feature size, people are looking forward to More Moore and More than Moore. In order to offer a possible alternative implementation process, researchers are trying to figure out a feasible transfer from silicon to molecular computing. Such transfer lies on bio-based modules programming with computer-like logic, aiming at realizing the Turing machine. To accomplish this, the DNA-based combinational logic is inevitably the first step we have taken care of. This timely overview study introduces combinational logic synthesized in DNA computing from both analog and digital perspectives separately. State-of-the-art research progress is summarized for interested readers to quick understand DNA computing, initiate discussion on existing techniques and inspire innovation solutions. We hope this study can pave the way for the future DNA computing synthesis.

1 Introduction

With the intense pursuit of More Moore and More than Moore [1,2], other computing modes have emerged to meet people's ever-increasing demands on both computing speed and volume. Among all promising candidates, molecular computing, especially DNA computing, attracts extensive research interest for its massively parallel operations and high computational speed. With its final goal to operate as a Turing machine shown in Figure 1, DNA computing requires the synthesis of combinational logic to be the first step [3, 4]. In the blue print, such as the field of synthetic biology [5–7], cells are employed as living devices to perform useful tasks like disease diagnosis and therapeutic drug release inside a body [8]. However, the limited existing techniques leads to a current consensus—to programme bio-based modules with computer-like logic.

In consideration of its undoubted significance, both academia and industry have paid a lot of attention to the researches on combinational logic based on DNA computing. Introducing and discussing this timely special issue, this overview study summarizes main synthesizing techniques for combinational logic in DNA computing. We hope this study can help readers to have a brief view on the cutting-edge

---

* Corresponding author (email: chzhang@seu.edu.cn)
† Chuan ZHANG and Lulu GE have the same contribution to this work.
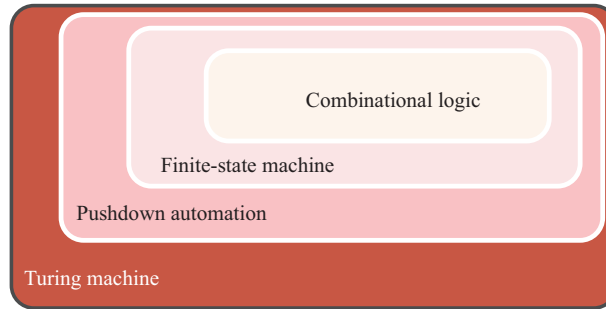
Figure 1   (Color online) Automata theory [4].

technologies and progress from two perspectives—analog and digital. On this basis, more and more innovation solutions are hoped to be inspired to pave the way for DNA computing's realization.

The remainder of this article is organized as follows. Section 2 reviews DNA computing in the aspect of its history, property and application. Several preliminaries, such as the illustration of gene expression, are also given in this section for a good further explanation. Analog circuits design based on DNAs are illustrated in Section 3. Section 4 reviews the combinational logic from the perspective of digital circuits, strongly appealing a trend which combines analog and digital together. In closing, Section 5 concludes the entire study and points out the possible future development of combinational logic in DNA computing.

2 DNA computing

As long ago as the 1950s, Feynman, who was the founder of nanotechnology, envisioned that we could manufacture an object maneuvering what we want on a very small scale [9]. However, it was not until 1994 that this notion of performing computations at a molecular level was realized by Adleman [10]. Due to his seminal work on computing with DNA, the field of DNA computing was initially developed. Since then, a tremendous amount of work has been done in this intriguing and nascent field, theoretically and experimentally.

2.1 DNA vs. silicon

The basic idea in DNA computing is from silicon to carbon [11]. With digital switching primitives replaced by DNA molecules, the computational substrate, which may be defined as "a physical substance acted upon by the implementation of a computational architecture", is changed from silicon to DNA [12]. For the conventional silicon-based computer with the von Neumann architecture [13], there exist two main barriers which prevent it going further on a small scale. (1) The inherent von Neumann bottleneck caused by the machine architecture [14]. Literally, this von Neumann bottleneck refers to a connecting tube transmiting a single word between the central processing unit (CPU) and the memory. A data traffic of a problem is inevitable when pumping vast numbers of words back and forth in this bottleneck, which seriously limits the effective processing speed of CPU. As data size increases, the bottleneck has become more of a problem. It also sets a limit on the speed of the computer. (2) The limitation derived from the nature of the computational substrate, silicon. At one time, 5 nm silicon technology node was once anticipated by some experts to be the end of Moore's Law, because quantum tunnelling will occur in transistors smaller than 7 nm scale [1]. Consequently, the unwanted interference severely affects the overall performance.

To overcome these two barriers of traditional silicon-based computers, some people are seeking for other computing modes. Their efforts are made mainly for two things. (1) Break through our word-at-a-time thinking style given by the von Neumann bottleneck and seek for non-von Neumann principles for new computer architectures. (2) Prove the superiority of new implementations over the conventional one and then investments should be considered to offer an economic basis for further developments. Among

all the promising candidates, include quantum, optical, and DNA-based computers, this study concerns DNA computing for its high-density storage and vast parallelism.

2.2 Merit and demerit

2.2.1 *Advantages of DNA computing*

The following lists the main merits of DNA computing.

- **Vast parallelism.** The most appealing advantage of DNA computing is its ability to address millions of operations in parallel [15]. If forced to behave sequentially, DNA computer will lose its edge [16].
- **Gigantic memory capacity.** One kilogram of DNA could meet the storage requirement from the whole world, under the premise of packaging the information as densely as it is in the genes of Escherichia coli. With over 100 years of data retention, the data density of bacterial DNA is estimated around 10^{19} bits per cm^3 [17].
- **Powerful computational capacity.** For DNA computing, the seminal work done by Adleman uncovers its ability to solve difficult computational problems, which are intractable for conventional computers [18].
- **Easy availability.** DNA, as a non-toxic material, is easy to be fetched from the nature, as long as there are cellular organisms. Such large supply makes DNA an affordable resource [19, 20].

2.2.2 *Limitations for DNA computing*

Current limitations on DNA computing are listed below.

- **One-time "instance" computers.** The current DNA computer is good at one instance of one specific problem, and typically performs computations only one time [16].
- **Long time consumption.** Since massive and messy, reactions themselves are very slow, requiring time measured in hours or days for the final computational result.
- **Limited synthesis technique.** DNA computing requires DNA synthesis and sequencing. If the high-quality synthesis is considered with no defect, the current practical length of synthetic oligonucleotides do not exceed 150–200 bases [21–23]. This hinders the capability of the constructed DNA-based computer to handle problems of larger size.
- **Error prone characteristic.** DNA synthesis is prone to errors [15]. Problems might sometimes be caused by even the simplest mixing operation.
- **Resource-intensive problems.** Employing the computational paradigm developed by Adleman, even for some relatively simple problems, unrestricted solution space and impractical amounts of memory may be required [12].
- **No visual output mechanism.** Since envisioned by [11], the designed DNA-based computer is in the form of tube with liquid. The final result of this liquid computer is difficult to be observed by naked eyes. There should be a mechanism to deeply analyze and visually show the computational results.

2.3 What does a DNA computer look like?

To the naked eye, a clean water solution in a test tube can be a DNA computer. No mechanical device there. A single drop of water surprisingly contains a trillion bio-molecular devices. Results are parsed by a technique that permits scientists to see the length of the final DNA molecule [16], rather than show up on a computer screen.

2.4 Application: what can DNA-based computers do?

Undoubtedly, DNA computer can do almost whatever the silicon-based one can. Consider the fact that silicon-based computer operates in a linear manner, while DNA computer is good at parallel processing. If DNA computer is forced to do as sequentially as silicon one, hours even years might be required to wait for a final result. This cannot be tolerant. Therefore, to develop in their best way, the domain of DNA
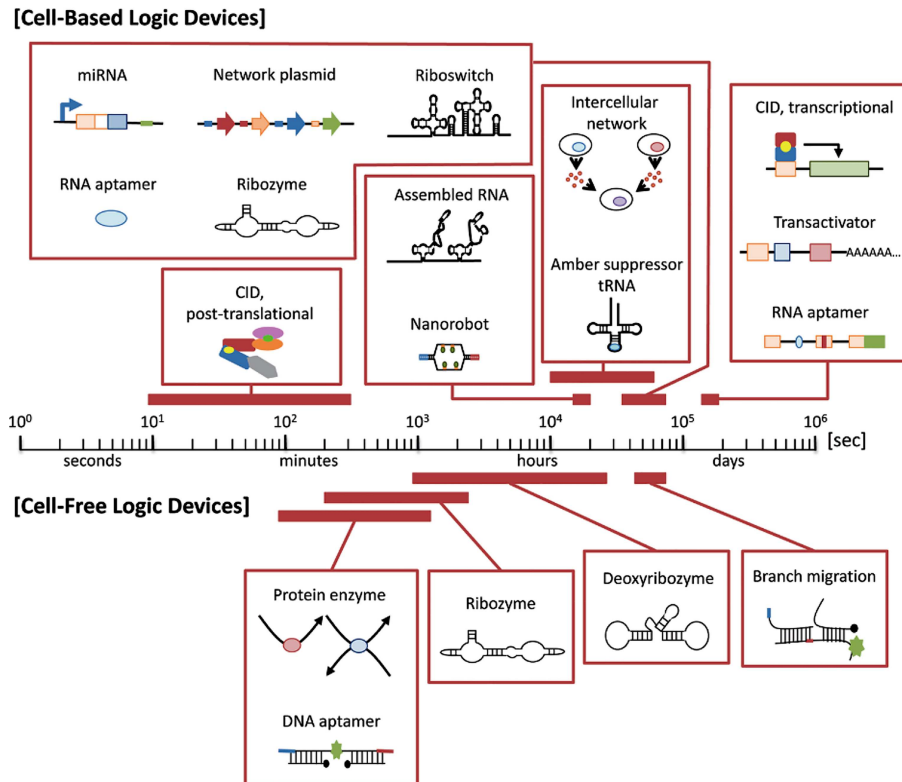
Figure 2 (Color online) Representative time-scale for the activation time of biomolecule-based logic devices in [28].

computers possibly covers the issues whose algorithms can be majorly parallelized, while the speciality of silicon-based computers resolves those problems with sequential algorithms.

Disease treatment. DNA computer can help with disease treatment. Early in 2004, researchers constructed a simple molecular-scale DNA computer with both input and output modules [24]. It is theoretically capable of correct cancerous diagnosis and additionally can release the corresponding anti-cancer drugs.

DNA^{2}DNA applications. This application is short for DNA to DNA computations, where conventional computers cannot outshine DNA computers [25]. Its key idea is to employ DNA computation with certain known DNA strands to operate on unknown pieces of DNA. After the unknown pieces of DNA are re-coded, reasonable computations can be performed on them to solve many interesting problems. Unlike "classic" DNA computations, these DNA^{2}DNA computations do not require the operations to work perfectly. Thus the results are much more error tolerant.

DNA cryptography. Due to the size of one-time-pad, practical applications of cryptographic systems based on one-time-pads are limited in traditional electronic media. Owing to the encryption of natural DNA and artificial DNA encoding binary data, DNA computing can be utilized in the cryptography field [26, 27].

2.5   Several categories

In [28], biomolecule-based Boolean logic gates fall into two categories: within cell and cell-free systems. According to the different substrates, they can also be categorized into two classes: nucleic acid-based computation and protein-based counterpart. Based on whether there is enzyme or not, nucleic acid-based computation includes both enzyme-based one and enzyme-free one. Figure 2 vividly and precisely displays the time-scale of activation time of cell-based and cell-free logic devices.

2.6 Chemical reaction networks

Chemical reaction networks (CRNs) can be used as a programmable language for modelling molecular systems, which comprise a set of reactants, products, and reactions. Moreover, CRNs could be employed to not only analyse the existing chemical systems but also construct new chemical systems in a convenient way, without considering the physical substrate. Using CRNs, researchers have designed novel molecular devices that achieve functions originally in electronic field, e.g., combinational logic, sequential logic, discrete-time signal processing computations, and arithmetic elements [29–47]. Demonstrated by ordinary differential equations (ODEs) [48–51], CRNs can always be modeled in terms of mass-action kinetics [52,53]. Therefore the ODE-based simulations are usually used to validate the CRN-based design. After the CRN-based design has been validated through simulations, engineering materials are required to physically implement the molecule devices. There are mainly three reasons why DNA is an ideal engineering material for CRN-based molecular devices: (1) DNA is a critical part of the central dogma [54], and as well the genetic material of most creatures. Thus, DNA devices could interface with a living body without any rejection reaction that might be caused by silicon-based devices. (2) Diverse reaction mechanisms could be easily realized by simple DNA sequence design because DNA strictly conforms to the Watson-Crick complementarity principle. This shows the facility and flexibility of CRN implementations using DNA. (3) Literature [55] theoretically proves that any CRN could be mapped into DNA strand displacement reactions, as long as it only contain bimolecular and unimolecular reactions. This makes CRN-based designs promising for biomedical applications.
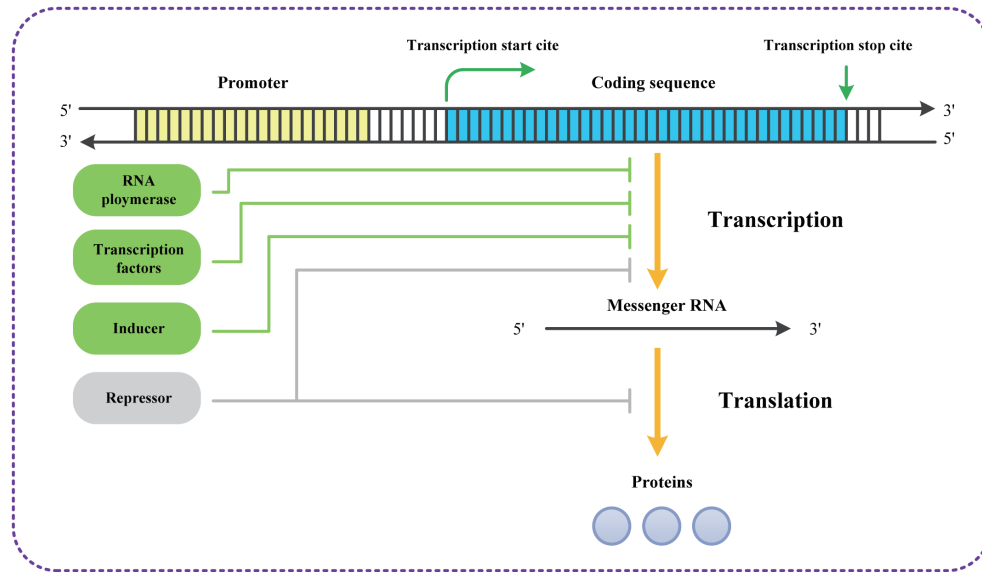
2.7 DNA strand displacement reactions

Short for toehold-mediated branch migration and strand displacement reactions, DNA strand displacement reactions can be viewed as an approach to physically perform complex computations or behaviors. As a kind of enzyme-free system, these reactions exhibit lower catalytic speedup than those catalyzed by enzymes, resulting in a long time consuming process for the final results. Driven by the free energy, DNA strand displacement reactions are limited by the initial amount of reactants. This means that no more information processing can be done as long as the system reaches an equilibrium state [56]. One important property of DNA strand displacement reactions is that they can be cascaded. Thus one reaction's output can be used as an input for its downstream reaction. This feature enables DNA strand displacement reactions to scale up and perform more complex behaviors. Exclusion of limited lifetime and long time consuming, the prediction of nucleic acid hybridization and strand displacement kinetics reveals a control over DNA strand displacement reactions design. This control is also attested by an observation that we can quantitatively control the rate of strand displacement reactions over a factor of 10^6 by adjusting the length and sequence composition of toeholds [56,57]. Therefore, the ever-increasingly important sequence design motivates the development of its automated software, for example, the DNA Strand Displacement (DSD) [58]. Empirically, to balance maximizing sequence space with minimizing crosstalk, it is useful to employ a three letter alphabet of $\{A, T, C\}$ to design the input and output strands, because G is known to be the most promiscuous nucleotide for its strongest base pair formation and its strongest mismatches [56,59]. More importantly, a conspicuous feature that can be further exploited is that DNA strand displacement reactions can be programmed by chemical reaction networks (CRNs) [55,60]. This facilitates their design process of complex behaviors using CRNs as a programmable language.

Figure 3 offers a basic DNA strand displacement overview. Panel (a) vividly shows the structure of a DNA double helix. With the hook indicating the $3'$ end, directional lines are typically used to denote DNA molecules. The numbers 1, 2, 3 actually represent different domains with specific DNA sequences. Panel (b) shows how a DNA strand displacement reaction works. Note that both A and B are single-stranded DNA molecules, while complex X and Y are double-stranded ones. Once the toehold domains 3 and 3^* bind together, the strand displacement reaction is triggered. After the domain 2 undergoes branch migration, the strand displacement reaction is finished. For more details, please refer to [56,57,61].

Figure 3 (Color online) DNA strand displacement overview in [56]. (a) The structure of DNA; (b) an example of a DNA strand displacement reaction.

2.8 Gene expression

Gene expression [54, 62] is the process of transforming genetic information hidden in DNA sequences into functional proteins. It contains two steps: transcription and translation. Figure 4 systematically illustrates a basic molecular gene expression. Both eukaryotes and prokaryotes are covered and merged. More specifically, transcription occurs first. In this step, the specific DNA sequence is copied into messenger RNA by the RNA polymerase (The green line indicates facilitation). Notably, the initiation of transcription is typically facilitated by a promoter (a special region of the same DNA strand) and transcription factors (proteins). After that, translation is triggered, in which ribosomes create polypeptides based on the messenger RNA. Then the polypeptides fold into functional proteins. In gene expression, the final output signals are the generated proteins; the input signals are molecules that can affect the gene expression. For example, a repressor could inhibit transcription or translation (the gray line indicates inhibition); the RNA polymerase and transcription factors initiate the gene expression as mentioned above; and an inducer could facilitate transcription by inhibiting the repressor or activating transcription factors. Different gene expressions could be viewed as different modules.

Actually, we categorise gene circuits into two types: analog and digital. In analog gene circuits, the concentrations of molecules above are used to indicate the values of input and output signals. Attention should be mainly paid to the functional relationships between the inputs and outputs. As for the digital circuits, logic values "0" and "1" are indicated by the presence and lack of the particles. Efforts should be focused on the establishment of the Boolean logic relationships between the inputs and outputs. The two types of gene circuits have one thing in common that they all have the potential to be cascaded due to the fact that the generated proteins of one gene expression might be used to adjust the expression of the other genes, functioning as its transcription factors, repressor or other relevant molecules.

3 Analog logic

One distinctive difference between analog DNA computation and digital DNA computation is that the input and output signals of analog DNA circuits are usually represented by the values of molecular concentrations, i.e., an analog DNA device is able to sense specific molecules' concentrations in their environment and then produce certain concentrations of outputs through proper analog computations [24]. Although digital circuits have taken a dominant place and achieved success in the field of electronics, analog circuits are more suitable and powerful than digital circuits in the field of DNA computation for the following reasons. (1) Up to a given accuracy, much fewer gates are applied in analog circuits

**Figure 4** (Color online) Basic molecular genetic processes.

to perform numerical computations than digital circuits [63]. This resource-saving property of analog circuit is useful in living cells where resources are limited. (2) Analog circuits are also more efficient than digital circuits in that cellular activities are analog computations in nature. Signals in cells are analog instead of digital [64]; the representation "1" and "0" in digital computation is too simple to indicate the effect of signals in cells.

To construct analog DNA circuits, one intuitive method is to translate traditional analog circuits into DNA circuits. Similar to the transformation from conventional digital electronics into digital DNA circuits, in which elementary DNA logic gates are synthesized to implement complex circuits, the transformation from analog electronics into analog DNA circuits should have focused on implementing DNA transistors. However, due to the fact that behaviour and property of a transistor are much more complicated and difficult to predict than those of a logic gate, the construction of a transistor is hardly analogous with behaviours of DNA strands, which only perform binding and unbinding operations in toehold-mediated strand displacement reactions. Therefore so far it is impractical to systematically synthesize DNA transistors to construct arbitrary analog circuits. One adopted alternative is to map analog electronics to analog DNA circuits at a behavioural level, i.e., to design DNA circuits achieving functions that originally implemented by silicon-based analog circuits. The following two subsections discusses: (1) analog circuits only involving DNA strands, (2) analog circuits involving gene expression.

Typically, in analog DNA circuits, the concentrations of single-stranded DNA signals are utilized to indicate the values of input signals and output signals; double-stranded DNA complexes are used to process input signals and generate output signals. The analog DNA circuits are driven by toehold-mediated strand displacement reactions. Diverse functions have been achieved. For example, Ref. [65] provides DNA devices to perform analog computation, including addition, subtraction, and multiplication based on DNA reactions. Using these elementary arithmetic modules, it also describes how to compute polynomial functions and even functions beyond the scope of polynomials. Ref. [66] utilizes DNA reactions to construct the feedback control circuit, which is common in analog electronics. Ref. [67] employs three DNA reactions to synthesize a decision-making controller, which works on two types of DNA strands and is able to convert the one in lower initial concentration into the one in higher initial concentration. Theoretically, these analog DNA circuits could be connected to achieve more complex functions, since they all rely on the same mechanism: toehold-mediated strand displacement reactions. However, consider that the input and output signals in different circuits might be in different structures (the position of the toehold domain and the recognition domain), extra interface modules, composed of DNA reactions, are required to transform the output signals of one circuit into the input signals of another circuit
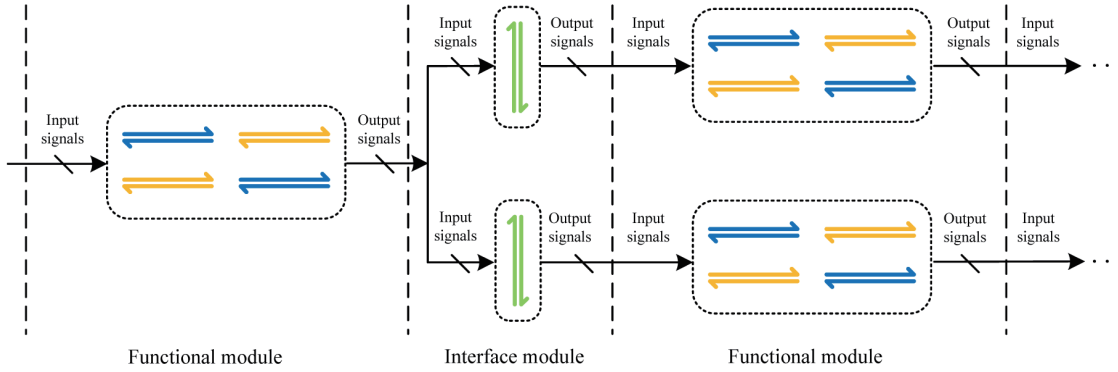
Figure 5 (Color online) Combination of different analog DNA circuits.
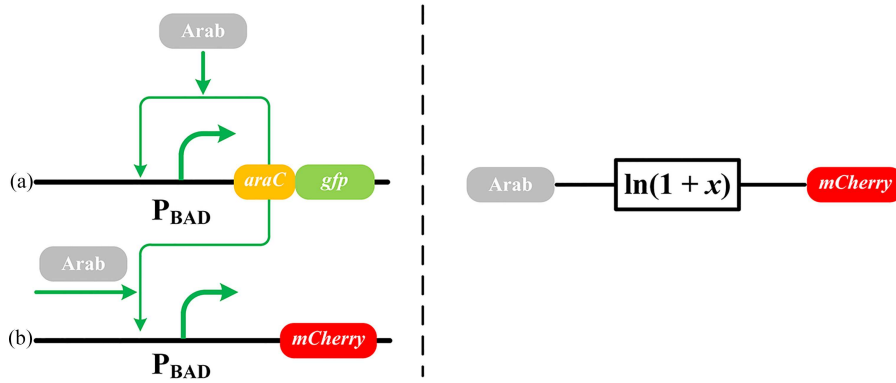
Figure 6 (Color online) Implementation of function $y = \ln(1 + x)$ with gene circuits in [69].

without destructing the distinctness of signals as shown in Figure 5. Consider that toehold-mediated strand displacement reactions do not require any enzyme as catalyst, such circuits are assumed to be well applied in an enzyme-free environment, i.e., in vitro.

Here we introduce analog gene circuits. Such circuits differ from analog DNA circuits in that the input and output signals of gene circuits are not DNA strands. Instead, in gene expression (DNA $\rightarrow$ RNA $\rightarrow$ protein), RNA polymerase, transcription factors, inducer, and repressor could be viewed as input signals; the final product-protein could be viewed as output signals. Gene circuits could be easily and directly synthesised in living cells. For instance, an analog accumulator could be implemented by two inputs generating a common output protein from independent genetic promoters, in which the two molecular fluxes will be automatically added to the common output protein as the final sum of the adding function [68]. In some cases, the generated proteins could also be used as transcription factors or repressors, thus making the gene circuits cascadable. For example, by using three gene parts, Ref. [69] constructs the log-domain analog circuit in living cells, implementing function $y = \ln(1 + x)$ as shown in Figure 6, in which x refers to the concentration of the inducer (Arab) and y refers to the concentration of the protein (mCherry). Specifically, in the first gene expression labelled "a", the inducer (Arab) helps the transcription factor (araC) bind to the promoter ($\mathrm{P_{BAD}}$), so as to generate the output protein-araC, which is also the transcription factor. Apparently, the generated proteins will further facilitate the gene expression. This could be viewed as a positive-feedback loop. In the second gene expression labelled "b", the output proteins (araC) mentioned above are shunted away and also function as the transcription factors, generating the output protein (mCherry). Experiments in [69] illustrate that the concentrations of Arab and mCherry conform to the function $y = \ln(1 + x)$. In addition, Ref. [63] has proven that in the field of electronics, the most efficient computation in use of resources is the mixture of analog computation and digital computation. Thus we might assume that the same conclusion applies to DNA computation as we translate electronics into DNA circuits. Then the molecular analog-to-digital converters and digital-
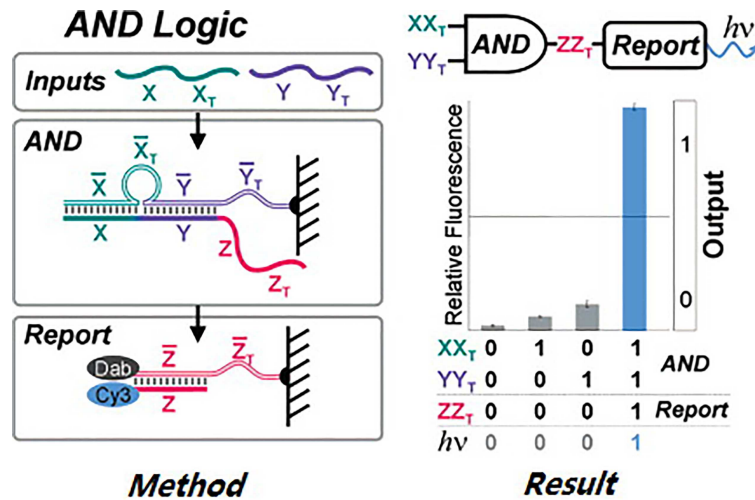
Figure 7 (Color online) Experiment method and result of AND logic in [71].

to-analog converters proposed in [70] will be critical in the mixture of the two forms of computation in the long run.

4 Digital logic

In the design of digital circuits, the Boolean values should be defined in an appropriate way and the bits of the binary system in the calculation should also be taken into account. The following existing studies show different angles of treating this issue.

An experimental study [71] utilizes a similar method to represent Boolean values in logic circuits. This research is also established on the DNA strand displacement reactions. However, it is mainly about those experiments the researchers have done. Digital logic gates, like AND (shown in Figure 7), OR, NAND and XOR gates have been all accomplished in the experiments. To get larger scale logic circuits, the inputs and the outputs of different levels ought to be isolated until they are not involved in the computations. In the results, they monitor the relative concentration of fluorescence in certain DNA strands and define a threshold to classify the Boolean values. If the value is above the threshold, it is seen as "1". If not, the value "0" is achieved. However, the drawbacks of the method applied in the experiment are also obvious. All designs include the inputs and the outputs approximation in analog waves of digital circuits, and there is no perfect strand displacement. Thus, the imperfect strand displacements generate a minor loss in the input and output signals. And as the scale is getting larger, this phenomenon can be taken place at each level of circuits operation, which limits the size of the biological circuits designed by this method.

Single-rail representations like above are studied, while the dual-rails values can also make sense. The binary system is defined differently in [72]. Dual-rails value is utilized to represent the value "0" or "1". Given a certain reactant w, the existence of 0_w represents "0" while the existence of 1_w represents "1". All the designed chemical reactions obey this principle. The research employs this method to design some other parts of the digital circuits, for example the NAND gates, the fan-out gates and the signal restoration gates in chemical reactions. Also, the combination of these parts is studied. For experimental realization, an introduction of the reaction environment and the biology principles behind the reactions are listed as well. In the final parts of the article, the drawback of this kind of method is analysed that dual-rails make the detection of absence more complex. Aside from that, this method should make every effort to deal with some unideal inputs, like 0.99 and 0.01, which is often encountered in many researches in this area. The design cannot be renewable and the threshold is some kind of a too ideal thing in such methods.

Jiang's research [29] mainly talks about realizing digital modules with chemical reactions. The whole
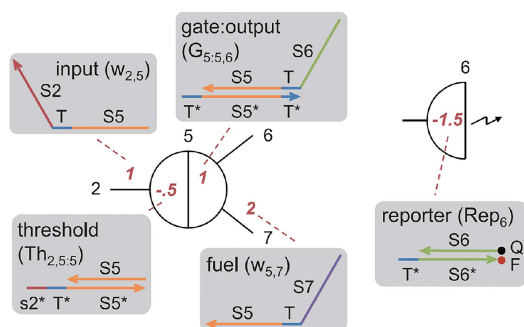
Figure 8 (Color online) A seesaw gate in [73].
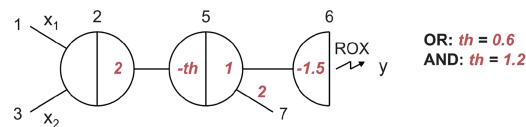
**Figure 9** (Color online) Two cascaded seesaw gates with a reporter in [73].

design is based on DNA strand displacement reactions. The digital values of "0" and "1" are represented by two different reactants, namely "X_0" and "X_1". The presence of "X_0" represents 0 and the presence of "X_1" represents 1. Similar to the method in [72], the two reactants cannot coexist with each other. In this method, the principles of each bit calculating in the logic gates can be easily studied out. According to the principles, the basic digital modules are transplanted to the corresponding chemical reactions. Apart from the basic digital logic gates, both the D latches and D flip-flops are also designed with the same idea. To ensure the validity, the corresponding ODEs towards each reaction network are also listed. With these basic modules prepared, more complicated modules can be combined with each other, like full adders, half adders and linear feedback shift registers. So far, the general method of designing more complicated digital logic circuits is achieved.

Besides the traditional designs, a specialized seesaw gate can be fabricated to combine all the logic gates together [73]. Employing dual-rail logic to represent binary values as shown in Figure 8, this seesaw gate has four basic components, including input, threshold, fuel and gate:output. A pair of seesaw gates, once cascaded, essentially perform AND or OR operations as shown in Figure 9. To perform which operation depends on the devised threshold's initial concentration. Combing with dual-rail logic, NOT operation is introduced. This means that any AND-OR-NOT circuit can be synthesized with seesaw gates using dual-rail logic, and translated into DNA. With this, designers can easily acquire the target DNA circuits following five sequential steps: a given AND-OR-NOT circuit, dual rail logic, seesaw gates, DNA domain and DNA sequence. Moreover, owing to the defined integrating gates and amplifying gates, the built logic circuits can support both multiple inputs and multiple outputs. Though use-once, the systematically synthesized digital circuits can be large-scale.

To program living cells, Ref. [74] offers a new platform Cello[1] to automatically design genetic circuits, according to the logic information hidden in the hardware description language Verilog. The mechanism can be described as follows. To realize the final target, the Verilog text should be parsed first. According to the results of the parse, the circuit diagram can be created. For the reason that Cello designs genetic circuits via drawing upon a library of Boolean logic gates (shown in Figure 10), the next step is to assign those gates to accomplish the circuit design. Then we can achieve the conversion from traditional logic gates to genetic circuit DNA sequence. In Figure 11, raised arrows stand for promoters, circles on stems indicate ribozyme insulators, hemispheres are ribosome binding sites (RBSs), protein-coding sequences are denoted by large arrows, and terminators are represented by "T"s. Part colors are in accordance with physical gates. Combining the gates together and making great efforts to isolate the different gates, the results of the insulated circuits are quite correct. Similar to some other researches, fluorescent protein is applied to present the final results. If the fluorescence is above the threshold, we can define that result as Boolean value "1", and otherwise as "0".

Different corresponding methods for synthesizing digital and analog circuits on biology level are also reviewed in [75]. In this review, the researchers mainly show their interests in digital memory, computations
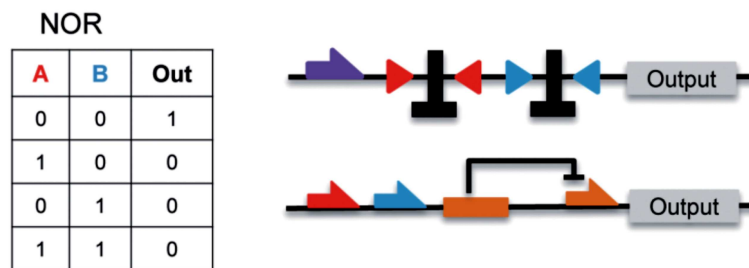
---

1) http://www.cellocad.org/.
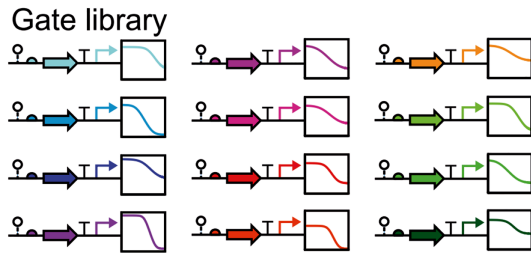
Figure 10 (Color online) The gate library in [74].
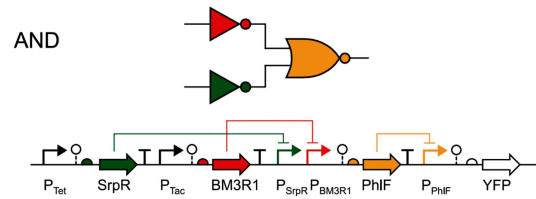
Figure 11 (Color online) Implementation of AND gate generated by Cello in [74].
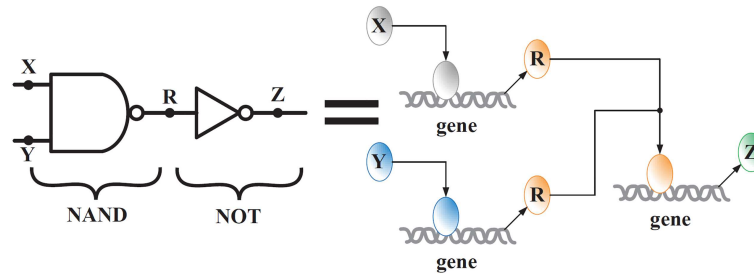

Figure 12 (Color online) An example of NOR digital logic gate synthesized in [75].

and multi-input logic gates. In the mentioned biology-based circuits, the chemical concentrations can be utilized to represent the digital signals. The designed memory based on DNA also has great potential in stability and scalability because of its high-parallelism, nevertheless it does not have an effective mean to maintain the states. To realize the circuits that are able to detect inputs, the biological environment and the endogenous cell can be used. And similar to [71], this study also prefers to use a proper threshold to transfer the original analog signals to digital ones. The multi-input logic gates integrate all input signals combinatorially, which refer to the digital computations that are only related to the current inputs and are without the usage of memory. Figure 12 gives an example of a NOR digital logic gate.

In [76], a more biological approach to establish genetic circuits is put forward. This article aims at creating synthetic gene networks that enable people to engineer cells as easy as they currently control computers and robots. And commonly, the scale is gradually increased. The research describes the genetic circuit building blocks, ranging from existing prototype circuits to signal processing circuits. The latter circuits are combined with the former smaller-scaled circuits. The main cellular blocks include those intracellular computation components and external communication devices. For a general study on this study as shown in Figure 13, the biochemical inverter can be taken as an example. Both the transcription and translation cellular processes are used in biochemical inversion. To be more specific, the input mRNA is translated by ribosomal RNA into a chain and then folds into a 3-dimensional protein structure. Once the protein binds an operator of the gene's promoter, the transcription of the gene is prevented by RNA polymerase (RNAp). If the repressor protein does not exist, RNAp transcribes the gene into the output mRNA. The other parts like NAND gates apply the same method of transcription and translation cellular process. Then the larger-scale circuits can be established based on these basic genetic circuits.

In [30], it proposes a systematic and straightforward approach to synthesize combinational digital logic with bimolecular reactions based on Karnaugh map. Boolean values' representation is the same as that in [29]. However, when implementing combinational logic, this work focuses more on the intimate relationship between the function requirement and the design process. A direct link between the Karnaugh map and the reaction network is established, thus sophisticated circuit structure is saved. Such elaborate design discovers and exploits the natural programmability of biochemical systems in terms of digital logic, providing a much easier mapping procedure. Also, ODE simulation has ensured both accuracy and robustness. Figure 14 lists all 2-input logic gates synthesized by [30]. Its extended version shown in [31]

Figure 13 (Color online) An example of NOR digital logic gate synthesized in [76].

offers more valid proofs from a mathematical perspective for the proposed approaches. Multi-input gates could be high efficiently synthesized with [31]. Figure 15(a) partially shows a 3-input gate implementation with CRNs. The corresponding simulation result is shown in Figure 15(b).

In addition, Ref. [77] proposes a design approach for the operation of all the Boolean logic gates, including AND (shown in Figure 16), NAND, OR and their opposite operations. Beyond that, a fuzzy logic gate system, as a proof-of-concept, is successfully implemented in a DNA origami box. These basic Boolean DNA logic gates are all with a universal structure, namely two types of DNA complexes, and can be readily constructed into larger scale. In this study, the input signals are typically DNA strands, while the output are denoted by fluorophores resonance energy transfer (FRET) signals. Such proposed DNA-based logic gates could be employed to perform those simple yes/no operations in a sensor, and would enable the calculations in vivo as well as cancer diagnostics at the molecular level.

5 Future research

As a non-von Neumann architecture machine, DNA computer tries to rethink the notion of computation. Operating in a parallel manner, it starts all the bio-operations from encoding information over the alphabet $\{A, T, G, C\}$. Just like the silicon-based computer performs complex calculations on a basic suite of arithmetic and logical operations, DNA has cutting, copying and many others. "Born" to solve a seven-point Hamiltonian path problem, DNA computing shows its overwhelming benefit in solving difficult problems, especially those that conventional computers cannot address. Seemingly as the smallest computer, about a kilogram of its computational substrate DNA could meet the world's storage requirements as long as the information could be packaged densely enough. Due to certain limitations, there is a current consensus that DNA computing will not be the rival for silicon-based one. This does not indicate that DNA computing is destroyed in the water. Most likely, the rich potential of DNA computing lies in vivo computing as well as in being combined with existing machinery to form DNA/silicon hybrid architectures. A future is predicted, in which a myriad of DNA computers patrol our bodies to monitor our health condition and make sure that proper drugs could be released to repair unhealthy tissue. Automatization is also potential, freeing most bio-operations from manual handling of tubes. Expected to be good at doing DNA-related things, DNA computing might be used in information storage, encryption, medical diagnosis, drug delivery and so on and so forth. To develop DNA computing in its full motion, there is still a long way off, full of both hopefulness and challenges.

As the core components of digital circuits, combinational logic gates could be synthesized with DNA strands to perform basic programmable logical operations. They as well could be cascaded to form more complex logic circuits—including sequential logic—and even larger systems to realize complex tasks and bio-sensing. Employing combinational logic gates is essentially an approach belonging to digital field, while analog signals are more common in nature. Thus, a new trend calling for more efficient DNA computation is to mix analog and digital computation together. To this end, both the molecular analog-to-digital converters and digital-to-analog converters are heated in the study of mixing these two forms of computation in the long run. In addition, there is a bold assumption or application beyond basic combinational logics, that is to implement fuzzy logic, probabilistic logic systems and even a Turing
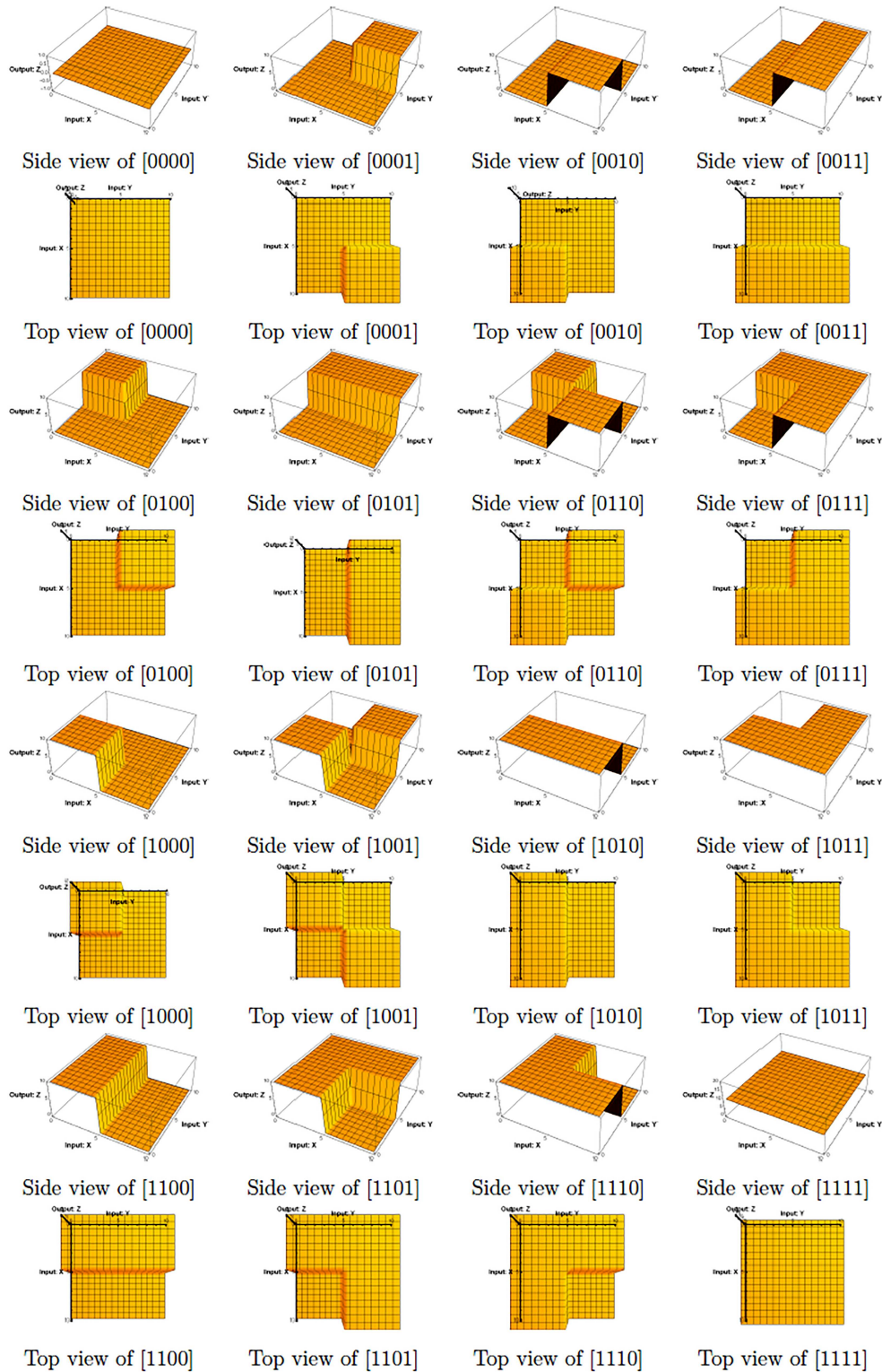
Figure 14 (Color online) Simulation of all 2-input gates [30].

machine. Thus they operate with analog logic whose inputs and outputs are values between "0" and "1" or have partial truth statements [77]. To elaborate a little further, unlike the deterministic interpretation of a truth table we are familiar with, a value is interpreted as the probability that the proposition is true. The significance of fuzzy logic lies in its role in both academia and industry, covering the field of medical diagnosis, image understanding [78], medicine delivery [79,80], artificial neural networks [81–83],
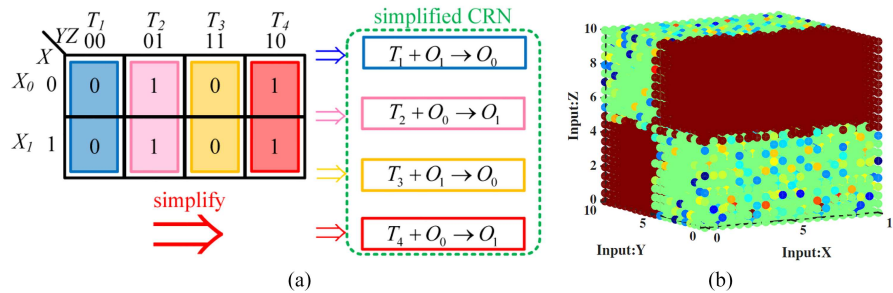
Figure 15 (Color online) (a) An approach for a 3-input gate in [31]. (b) Simulation of this 3-input gate in [31].
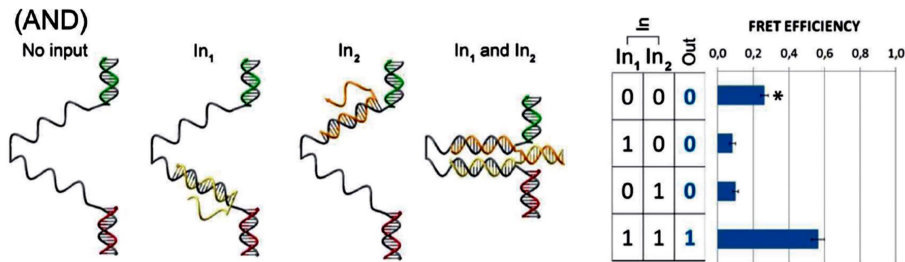
Figure 16 (Color online) Design principles of the AND gate based on DNA strands in [77].

RNA secondary structure prediction [84], processing by artificial intelligence [85], and fuzzy electronics [77, 86, 87]. Therefore, to understand fuzzy logic and additionally apply it to artificial synthetic biology can help us understand natural systems better and then provide a new tool to perform soft computing.

References

1 Kish L B. End of Moore's law: thermal (noise) death of integration in micro and nano electronics. Phys Lett A, 2002, 305: 144–149

2 Desai S B, Madhvapathy S R, Sachid A B, et al. MoS2 transistors with 1-nanometer gate lengths. Science, 2016, 354: 99–102

3 Yahiro W, Hagiya M, Implementation of Turing machine using DNA strand displacement. In: Proceedings of International Conference on Theory and Practice of Natural Computing. Berlin: Springer, 2016. 161–172

4 Wikipedia. Combinational logic. 2018. https://en.wikipedia.org/wiki/Combinational_logic

5 Khalil A S, Collins J J. Synthetic biology: applications come of age. Nat Rev Genet, 2010, 11: 367

6 Siuti P, Yazbek J, Lu T K. Synthetic circuits integrating logic and memory in living cells. Nat Biotechnol, 2013, 31: 448–452

7 Andrianantoandro E, Basu S, Karig D K, et al. Synthetic biology: new engineering rules for an emerging discipline. Molecular Syst Biol, 2006, 2: 28

8 Green A A, Kim J, Ma D, et al. Complex cellular logic computation using ribocomputing devices. Nature, 2017, 548: 117–121

9 Feynman R P. There's plenty of room at the bottom. Eng Sci, 1960, 23: 22–36

10 Trautman J K, Macklin J J, Brus L E, et al. Near-field spectroscopy of single molecules at room temperature. Nature, 1994, 369: 40–42

11 Paun G, Rozenberg G, Salomaa A. DNA Computing: New Computing Paradigms. Berlin: Springer, 2005

12 Amos M. Theoretical and experimental DNA computation. Bull European Assoc Theor Comput Sci, 1999, 67: 125–138

13 von Neumann J. First draft of a report on the EDVAC. IEEE Ann Hist Comput, 1993, 15: 27–75

14 Backus J. Can programming be liberated from the von Neumann style?: a functional style and its algebra of programs. Commun ACM, 1978, 21: 613–641

15 Deaton R, Murphy R C, Rose J A, et al. A DNA based implementation of an evolutionary search for good encodings for DNA computation. In: Proceedings of IEEE International Conference on Evolutionary Computation, Indianapolis,

1997. 267–271

16 Tagore S, Bhattacharya S, Islam M, et al. DNA computation: application and perspectives. J Proteom Bioinform, 2010, 3: 234–343

17 Extance A. How DNA could store all the world's data. Nature, 2016, 537: 22–24

18 Hameed K. DNA computation based approach for enhanced computing power. Int J Emerg Sci, 2011, 1: 23–30

19 Saxena S. Introduction to DNA computing. Int Acadmey Eng Medical Res, 2016, 1: 1–3

20 Kumar S N. A proper approach on DNA based computer. American Nanomater, 2015, 3: 1–14

21 Ma S, Tang N, Tian J. DNA synthesis, assembly and applications in synthetic biology. Curr Opin Chem Biol, 2012, 16: 260–267

22 Bornholt J, Lopez R, Carmean D M, et al. A DNA-based archival storage system. SIGOPS Oper Syst Rev, 2016, 50: 637–649

23 Hughes R A, Ellington A D. Synthetic DNA synthesis and assembly: putting the synthetic in synthetic biology. Cold Spring Harb Perspect Biol, 2017, 9: a023812

24 Benenson Y, Gil B, Ben-Dor U, et al. An autonomous molecular computer for logical control of gene expression. Nature, 2004, 429: 423–429

25 Landweber L F, Lipton R J, Rabin M O. DNA^{2}DNA computations: a potential "killer app"? In: Proceedings of International Colloquium on Automata, Languages, and Programming (ICALP). Berlin: Springer, 1997. 56–64

26 Watada J, binti abu Bakar R. DNA computing and its applications. In: Proceedings of the 8th International Conference on Intelligent Systems Design and Applications, Kaohsiung, 2008. 288–294

27 Gehani A, LaBean T, Reif J. DNA-based cryptography. Asp Mol Comput, 2003, 2950: 167–188

28 Miyamoto T, Razavi S, DeRose R, et al. Synthesizing biomolecule-based Boolean logic gates. ACS Synth Biol, 2012, 2: 72–82

29 Jiang H, Riedel M D, Parhi K K. Digital logic with molecular reactions. In: Proceedings of International Conference on Computer-Aided Design (ICCAD), San Jose, 2013. 721–727

30 Zhang C, Ge L L, Zhong Z W, et al. Karnaugh map-aided combinational logic design approach with bistable molecular reactions. In: Proceedings of IEEE International Conference on Digital Signal Processing (DSP), Singapore, 2015. 1288–1292

31 Ge L, Zhong Z, Wen D, et al. A formal combinational logic synthesis with chemical reaction networks. IEEE Trans Mol Biol Multi-Scale Commun, 2017, 3: 33–47

32 Wen D L, Ge L L, Lu Y X, et al. A DNA strand displacement reaction implementation-friendly clock design. In: Proceedings of IEEE International Conference on Communications (ICC), Paris, 2017

33 Zhang X C, Ge L L, You X H, et al. Synthesizing LDPC belief propagation decoding with molecular reactions. In: Proceedings of IEEE International Conference on Communications (ICC), Kansas City, 2018

34 Zhong Z W, Li Z, Ge L L, et al. Implementation of Mealy machine with molecular reactions. In: Proceedings of IEEE International Conference on Communications (ICC), Kansas City, 2018

35 Lu Y X, Ge L L, You X H, et al. Implementation of sinusoids and pulse width modulation with chemical reactions. In: Proceedings of IEEE International Conference on Communications (ICC), Kansas City, 2018

36 Li M H, Ge L L, You X H, et al. Basic arithmetics based on analog signal with molecular reactions. In: Proceedings of IEEE International Conference on Communications (ICC), Kansas City, 2018

37 Shen Z, Ge L, Wei W, et al. Molecular synthesis for probability theory and stochastic process. J Sign Process Syst, 2018, 90: 1479–1494

38 Fang C, Shen Z, Zhang Z, et al. Synthesizing a neuron using chemical reactions. In: Proceedings of IEEE International Workshop on Signal Processing Systems (SiPS), Cape Town, 2018

39 Zhuang Y, Zhang Z, You X, et al. Arithmetic computations based on chemical reaction networks. In: Proceedings of IEEE International Workshop on Signal Processing Systems (SiPS), Cape Town, 2018

40 Zhong Z, Ge L, Shen Z, et al. CRN-based design methodology for synchronous sequential logic. In: Proceedings of IEEE International Workshop on Signal Processing Systems (SiPS), Lorient, 2017

41 Shen Z Y, Ge L L, Wei W, et al. Synthesizing Markov chain with reversible unimolecular reactions. In: Proceedings of International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, 2017

42 Zhuang Y C, Ge L L, Wei W, et al. A synthesis flow for fast convolution unit based on molecular reactions. In: Proceedings of International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, 2017

43 Shen Z, Zhang C, Ge L, et al. Synthesis of probability theory based on molecular computation. In: Proceedings of IEEE International Workshop on Signal Processing Systems (SiPS), Dallas, 2016

44 Ge L, Zhang C, Zhong Z, et al. A formal design methodology for synthesizing a clock signal with an arbitrary duty cycle of M/N. In: Proceedings of IEEE International Workshop on Signal Processing Systems (SiPS), Hangzhou, 2015

45 Jiang H, Riedel M D, Parhi K K. Synchronous sequential computation with molecular reactions. In: Proceedings of the 48th Design Automation Conference (DAC), San Diego, 2011. 836–841

46 Salehi S A, Riedel M D, Parhi K K. Asynchronous discrete-time signal processing with molecular reactions. In: Proceedings of the 48th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, 2014

47 Senum P, Riedel M D. Rate-independent constructs for chemical computation. PLoS ONE, 2011, 6: e21414

48 Howard P. Analysis of ODE models. 2009. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.441.4759&rep=rep1&type=pdf

49 Strogatz S H. Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering. Boulder: Westview Press, 2014

50 Zauderer E. Partial Differential Equations of Applied Mathematics. Hoboken: John Wiley & Sons, 2011

51 Hale J K, Lunel S M V. Introduction to Functional Differential Equations. Berlin: Springer, 2013

52 Érdi P, Tóth J. Mathematical Models of chemical Reactions: Theory and Applications of Deterministic and Stochastic Models. Manchester: Manchester University Press, 1989

53 Horn F, Jackson R. General mass action kinetics. Arch Rational Mech Anal, 1972, 47: 81–116

54 Crick F. Central dogma of molecular biology. Nature, 1970, 227: 561–563

55 Soloveichik D, Seelig G, Winfree E. DNA as a universal substrate for chemical kinetics. Proc Natl Acad Sci USA, 2010, 107: 5393–5398

56 Zhang D Y, Seelig G. Dynamic DNA nanotechnology using strand-displacement reactions. Nat Chem, 2011, 3: 103–113

57 Zhang D Y, Winfree E. Control of DNA strand displacement kinetics using toehold exchange. J Am Chem Soc, 2009, 131: 303–314

58 Phillips A, Cardelli L. A programming language for composable DNA circuits. J R Soc Interface, 2009, 6: S419–S436

59 SantaLucia Jr J, Hicks D. The thermodynamics of DNA structural motifs. Annu Rev Biophys Biomol Struct, 2004, 33: 415–440

60 Shapiro E, Ran T. DNA computing: molecules reach consensus. Nat Nanotech, 2013, 8: 703–705

61 Zhang D Y. Dynamic DNA strand displacement circuits. Dissertation for Ph.D. Degree. Pasadena: California Institute of Technology, 2010

62 Leavitt S. Deciphering the genetic code: Marshall Nirenberg. Office of NIH History, 2004

63 Sarpeshkar R. Analog versus digital: extrapolating from electronics to neurobiology. Neural Comput, 1998, 10: 1601–1638

64 Sauro H M, Kim K. Synthetic biology: It's an analog world. Nature, 2013, 497: 572–573

65 Song T, Garg S, Mokhtar R, et al. Analog computation by DNA strand displacement circuits. ACS Synth Biol, 2016, 5: 898–912

66 Yordanov B, Kim J, Petersen R L, et al. Computational design of nucleic acid feedback control circuits. ACS Synth Biol, 2014, 3: 600–616

67 Chen Y J, Dalchau N, Srinivas N, et al. Programmable chemical controllers made from DNA. Nat Nanotech, 2013, 8: 755–762

68 Sarpeshkar R. Analog synthetic biology. Philos Trans R Soc A-Math Phys Eng Sci, 2014, 372: 20130110

69 Daniel R, Rubens J R, Sarpeshkar R, et al. Synthetic analog computation in living cells. Nature, 2013, 497: 619–623

70 Salehi S A, Jiang H, Riedel M D, et al. Molecular sensing and computing systems. IEEE Trans Mol Biol Multi-Scale Commun, 2015, 1: 249–264

71 Frezza B M, Cockroft S L, Ghadiri M R. Modular multi-level circuits from immobilized DNA-based logic gates. J Am Chem Soc, 2007, 129: 875–879

72 Chinifooroshan E, Doty D, Kari L, et al. Scalable, time-responsive, digital, energy-efficient molecular circuits using DNA strand displacement. In: Proceedings of the 16th International Conference on DNA Computing and Molecular Programming, Hong Kong, 2010. 25–36

73 Qian L, Winfree E. Scaling up digital circuit computation with DNA strand displacement cascades. Science, 2011, 332: 1196–1201

74 Nielsen A A, Der B S, Shin J, et al. Genetic circuit design automation. Science, 2016, 352: aac7341

75 Roquet N, Lu T K. Digital and analog gene circuits for biotechnology. Biotech J, 2014, 9: 597–608

76 Weiss R, Basu S, Hooshangi S, et al. Genetic circuit building blocks for cellular computation, communications, and signal processing. Nat Comput, 2003, 2: 47–84

77 Zadegan R M, Jepsen M D E, Hildebrandt L L, et al. Construction of a fuzzy and Boolean logic gates based on DNA. Small, 2015, 11: 1811–1817

78 Zhang Y, Wirkert S J, Iszatt J, et al. Tissue classification for laparoscopic image understanding based on multispectral texture analysis. J Med Imag, 2017, 4: 015001

79 Lu C H, Willner B, Willner I. DNA nanotechnology: from sensing and DNA machines to drug-delivery systems. ACS Nano, 2013, 7: 8320–8332

80 Li J, Pei H, Zhu B, et al. Self-assembled multivalent DNA nanostructures for noninvasive intracellular delivery of immunostimulatory CpG oligonucleotides. ACS Nano, 2011, 5: 8783–8789

81 Qian L, Winfree E, Bruck J. Neural network computation with DNA strand displacement cascades. Nature, 2011, 475: 368–372

82 Schneider G, Wrede P. Artificial neural networks for computer-based molecular design. Prog Biophys Mol Biol, 1998, 70: 175–222

83 Noordewier M O, Towell G G, Shavlik J W. Training knowledge-based neural networks to recognize genes in DNA sequences. In: Proceedings of Advances in Neural Information Processing Systems, Denver, 1991. 530–536

84 Zuber J, Sun H, Zhang X, et al. A sensitivity analysis of RNA folding nearest neighbor parameters identifies a subset of free energy parameters with the greatest impact on RNA secondary structure prediction. Nucleic Acids Res, 2017, 45: 6168–6176

85 Brady M. Artificial intelligence and robotics. Artif Intell, 1985, 26: 79–121

86 Ray K S, Mondal M. Similarity-based fuzzy reasoning by DNA computing. Int J Bio-Inspired Comput, 2011, 3: 112–122

87 Jeng D J, Watada J, Wu B, et al. Fuzzy forecasting with DNA computing. In: Proceedings of International Workshop on DNA-Based Computers. Berlin: Springer, 2006. 324–336