ACS Editors' Choice

ACCOUNTS of chemical research

Article

pubs.acs.org/accounts

Terms of Use

Exercises in Molecular Computing

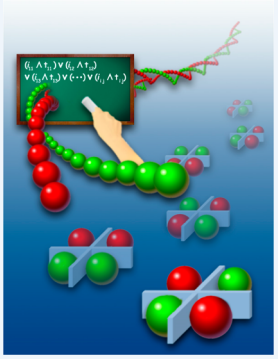
Milan N. Stojanovic,*[,†] Darko Stefanovic,[‡] and Sergei Rudchenko[§]

[†]Departments of Medicine, Biomedical Engineering, and Systems Biology, Columbia University, New York, New York 10032, United States

[‡]Department of Computer Science and Center for Biomedical Engineering, University of New Mexico, Albuquerque, New Mexico 87131, United States

[§]Hospital for Special Surgery, New York, New York 10021, United States

CONSPECTUS: The successes of electronic digital logic have transformed every aspect of human life over the last half-century. The word "computer" now signifies a ubiquitous electronic device, rather than a human occupation. Yet evidently humans, large assemblies of molecules, can compute, and it has been a thrilling challenge to develop smaller, simpler, synthetic assemblies of molecules that can do useful computation. When we say that molecules compute, what we usually mean is that such molecules respond to certain inputs, for example, the presence or absence of other molecules, in a precisely defined but potentially complex fashion. The simplest way for a chemist to think about computing molecules is as sensors that can integrate the presence or absence of multiple analytes into a change in a single reporting property. Here we review several forms of molecular computing developed in our laboratories.



When we began our work, combinatorial approaches to using DNA for computing were used to search for solutions to constraint satisfaction problems. We chose to work instead on logic circuits, building bottom-up from units based on catalytic nucleic acids, focusing on DNA secondary structures in the design of individual circuit elements, and reserving the combinatorial opportunities of DNA for the representation of multiple signals propagating in a large circuit. Such circuit design directly corresponds to the intuition about sensors transforming the detection of analytes into reporting properties. While this approach was unusual at the time, it has been adopted since by other groups working on biomolecular computing with different nucleic acid chemistries.

We created logic gates by modularly combining deoxyribozymes (DNA-based enzymes cleaving or combining other oligonucleotides), in the role of reporting elements, with stem–loops as input detection elements. For instance, a deoxyribozyme that normally exhibits an oligonucleotide substrate recognition region is modified such that a stem–loop closes onto the substrate recognition region, making it unavailable for the substrate and thus rendering the deoxyribozyme inactive. But a conformational change can then be induced by an input oligonucleotide, complementary to the loop, to open the stem, allow the substrate to bind, and allow its cleavage to proceed, which is eventually reported via fluorescence. In this Account, several designs of this form are reviewed, along with their application in the construction of large circuits that exhibited complex logical and temporal relationships between the inputs and the outputs.

Intelligent (in the sense of being capable of nontrivial information processing) theranostic (therapy + diagnostic) applications have always been the ultimate motivation for developing computing (i.e., decision-making) circuits, and we review our experiments with logic-gate elements bound to cell surfaces that evaluate the proximal presence of multiple markers on lymphocytes.

INTRODUCTION

A chemist, a computer scientist, and a biophysicist are writing an Account of molecular computing for a special issue of *Accounts of Chemical Research* focusing on Ned Seeman's founding of the field of DNA nanotechnology. While this sentence sounds like the beginning of a joke, it is probably the most straightforward testimony of the breadth of Ned's intellectual footprint. And, the reader will immediately notice a similarity between this Account and those from this same issue that deal with structural DNA nanotechnology: We all use elementary units that can be combined at various scales into more complex functions, and we use, primarily and repeatedly, basic concepts such as Watson–Crick base pairing to plan and predict the behaviors of these simple units and, by extension, their more complicated mixtures or assemblies. It is this kind of "keep it simple and you will go much further" thinking that Ned brought to traditional synthetic (bio)chemistry, attracting at the same time scientists and engineers from other disciplines to the field and enriching chemistry in the process with a series of new concepts and approaches.

We start this Account by recapitulating our early approach to molecular computing,[1,2] including the first reported complete set of nucleic-acid-based logic gates[3] that could be directly combined into some traditional circuits[4,5] and some less traditional game-playing automata.[6−8] This approach was

Special Issue: Nucleic Acid Nanotechnology

Received: February 16, 2014
Published: May 30, 2014

   dx.doi.org/10.1021/ar5000538 | *Acc. Chem. Res.* 2014, 47, 1845−1852

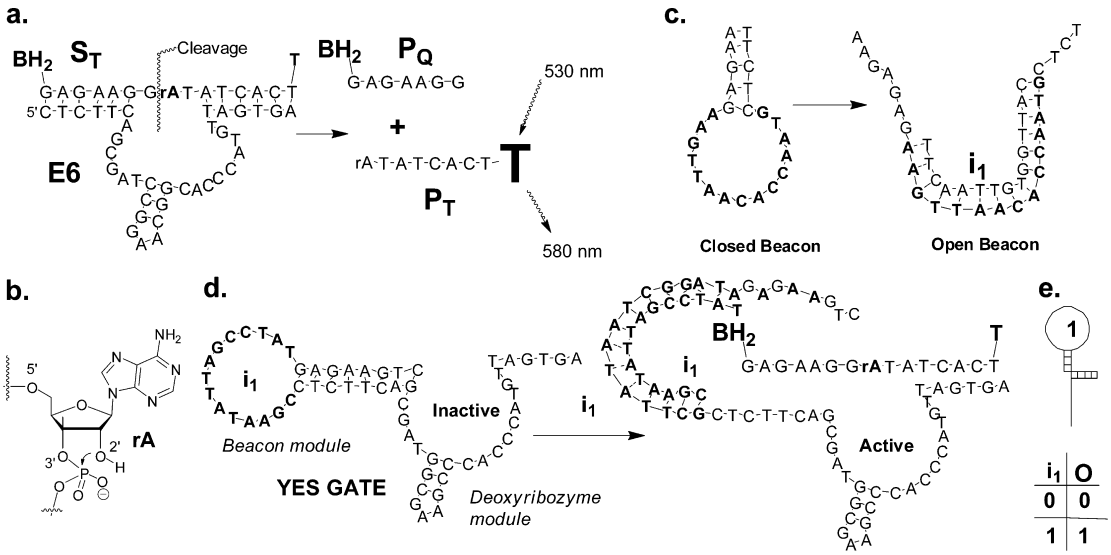


Figure 1. (a) An example of deoxyribozyme (**E6**[18]) shown in complex with its substrate ($\mathbf{S_T}$). The cleavage reaction of $\mathbf{S_T}$ produces two shorter products ($\mathbf{P_Q}$ and $\mathbf{P_T}$) as output (**O**); output production can be monitored by fluorogenic cleavage ($\mathbf{BH_2}$ is black hole quencher, while **T** is fluorescent dye carboxytetramethylrhodamine, TAMRA). (b) Enzyme **E6** is a phosphodiesterase, meaning it cleaves a phosphodiester bond in the substrate; in this case, it cleaves at the position at which a single ribonucleotide is inserted in an oligonucleotide (**rA** as in part a), probably by activating a 2′OH group through a general base catalysis by a metal ion. (c) The molecular beacon stem–loop as a recognition module: a closed beacon has a stem–loop conformation, but adding an input oligonucleotide ($\mathbf{i_1}$) complementary to the loop opens the stem. (d) The catalytic molecular beacon, or $\mathbf{YESi_1}$, or sensor gate, is constructed by attaching a beacon module to one of the substrate recognition regions of the deoxyribozyme module. Upon addition of an input ($\mathbf{i_1}$), the gate switches to its active form. The reaction can be monitored fluorogenically. (e) Schematic representation of beacon sensitive to input $\mathbf{i_1}$ and corresponding input–output correlation table.

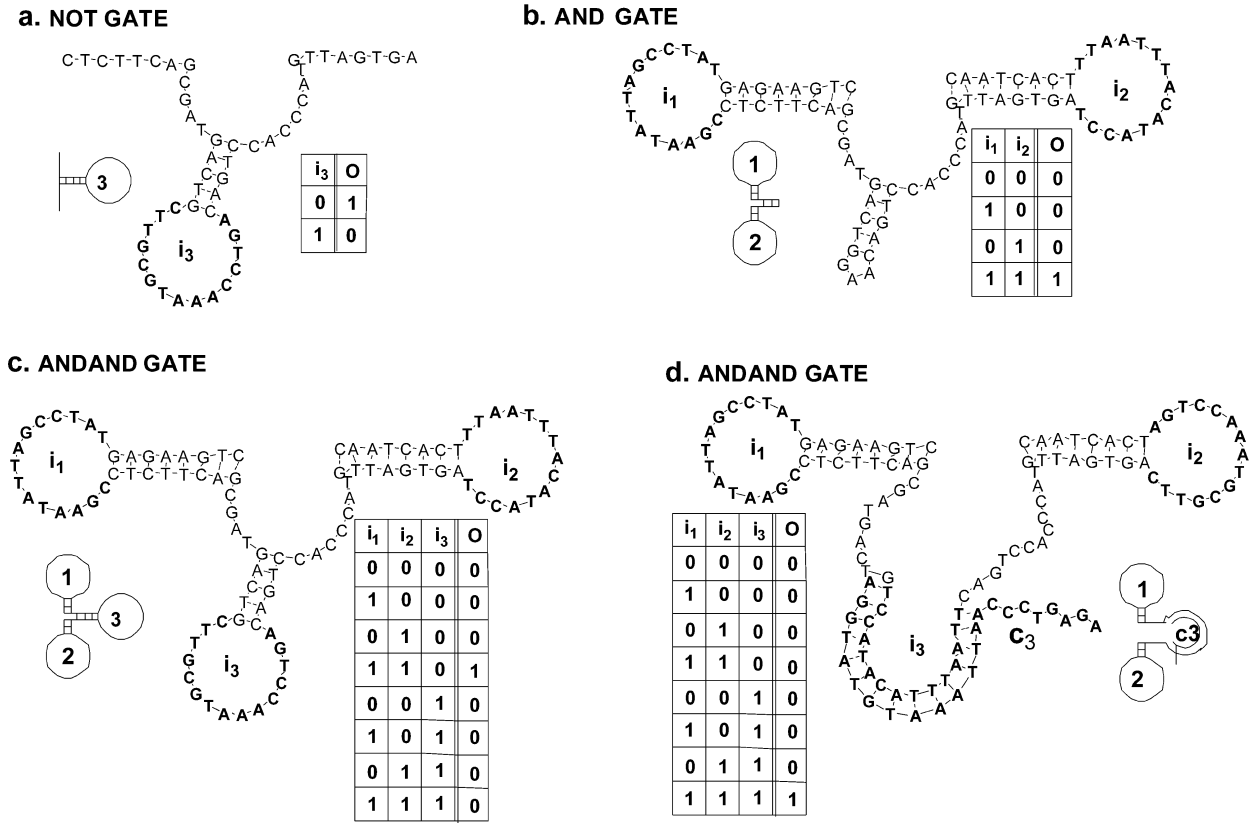
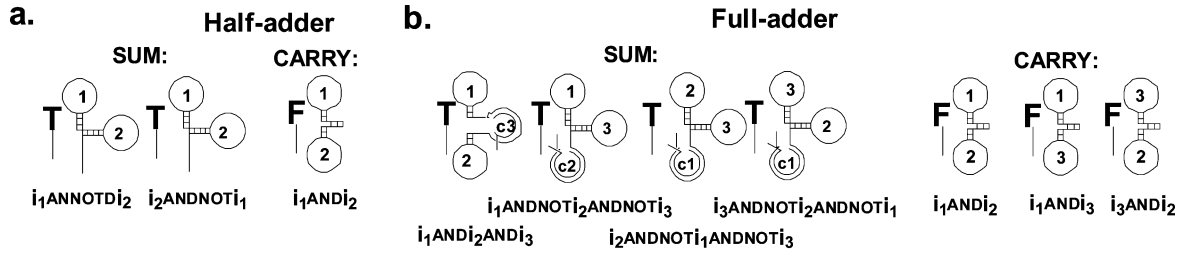


Figure 2. (a) $\mathbf{NOTi_3}$ gate, that is, a single input gate inhibited by the presence of input, with corresponding input–output correlation (or truth) table. (b) $\mathbf{i_1ANDi_2}$ gate with two inputs, both needed to generate an output. (c) $\mathbf{i_1ANDi_2ANDi_3}$ gate with two inputs promoting and one inhibiting generation of output. (d) $\mathbf{i_1ANDi_2ANDi_3}$ gate using precomplexation with a complement of an input ($\mathbf{c_3}$) to achieve output production only when all three inputs are present.

dx.doi.org/10.1021/ar5000538 | Acc. Chem. Res. 2014, 47, 1845–1852



Input		Result		Output	
i_1	i_2	dec	bin	S	C
0	0	0	00	0	0
0	1	1	01	1	0
1	0	1	01	1	0
1	1	2	10	0	1

Input			Result		Output	
i_1	i_2	i_3	dec	bin	S	C
0	0	0	0	00	0	0
0	0	1	1	01	1	0
0	1	0	1	01	1	0
1	0	0	1	01	1	0

Input			Result		Output	
i_1	i_2	i_3	dec	bin	S	C
1	1	0	2	10	0	1
1	0	1	2	10	0	1
0	1	1	2	10	0	1
1	1	1	3	11	1	1

Figure 3. (a) A mixture of three gates that behaves as a half-adder, that is, an element that can add up to 1 + 1 binary. The $\mathbf{i_1ANDNOTi_2}$ and $\mathbf{i_2ANDNOTi_1}$ gates cleave one output substrate (for the sum output **S**), while the $\mathbf{i_1ANDi_2}$ gate cleaves another (for the carry output **C**). (b) Full adder consists of a total of seven gates and three possible inputs, one of which could be a 'carry' from previous layer of gates.

extensively reviewed before, including in the popular literature.[9] We then discuss two projects that evolved from our initial work in molecular computing, computing with beads and on cell surfaces. For lack of space, we will not discuss here a related approach to molecular robotics, called molecular spiders.[10,11]

DEVELOPING DEOXYRIBOZYME-BASED LOGIC GATES AND THEIR CIRCUITS

Two of us (D.S. and M.N.S.), being both at one of those troubling career-forks in life, decided to do something together; we settled on DNA computing, inspired by Adleman's seminal paper[12] and discussions how molecules could add two numbers and cure diseases at the same time. Things started to take a turn toward reality following Breaker and Ellington's back-to-back reviews[13,14] on nucleic acid catalysts and aptamers. Read in a single breath, these reviews inspired thinking about nucleic acid catalysts controllable by external inputs through coupled recognition elements, an idea that was ripe for implementation.[15,16] Some of our early attempts looked like Woody Allen's Earl of Sandwich experiments, "His first completed work—a slice of bread, a slice of bread on top of that, and a slice of turkey on top of both—fails miserably",[17] and will not be discussed further.

Our first result that was reported in peer-reviewed literature was a modular design,[15] called either a catalytic molecular beacon[18] when sensitive to one oligonucleotide or a *gate* (Figure 1) when sensitive to one or more oligonucleotides.[3,18] In these, the catalytic activity of a deoxyribozyme[19] module is controlled by up to three oligonucleotide *inputs* through recognition modules based on molecular beacons.[20] When active, the deoxyribozyme has phosphodiesterase activity and cleaves another oligonucleotide,[18] the substrate, and this cleavage is the *output* of the gate. By labeling the substrate fluorogenically, we can monitor the progress of the cleavage via an increase of fluorescence. Thus, just like electronic logic gates, a gate is switched by one or more *inputs*, namely, specific oligonucleotides, producing output. A recognition module (a stem–loop oligonucleotide) can be placed to block the substrate from accessing the deoxyribozyme (Figure 1); introduction of that loop's complementary sequence (an input) then removes the block and *activates* the gate. This placement is possible at either binding arm of the deoxyribozyme. Another possible placement of a recognition module is within the catalytic core of the deoxyribozyme (provided its structure allows it); in this case introduction of the input *inhibits* the gate by distorting this core (Figure 2a).

The three allosteric binding sites are for all practical purposes modulated independently, which means that we can easily predict input–output relationships, that is, describe whether the presence or absence of each of the several inputs will result in cleavage, and we can express these relationships as truth tables or as conjunctive formulas of Boolean algebra (Figure 2).

Furthermore, we are free to vary the input sequences, as long as no interfering secondary structure arises in the loop and as long as the inputs have no other preferred interaction with deoxyribozymes. As a result, each gate design is really a template for a very large number of possible enzymes. This lets us construct systems comprising large numbers of such enzymes operating in parallel by predicting their behavior compositionally, initially ignoring second-order effects of mutual interference ("cross-talk") between gates. The composability of gates, together with the logical abstraction, permits the application of design principles from classical digital logic design and shows a way to organize biological molecules into systems that carry out potentially complex computations.

We use a single stem–loop region to block access of the substrate to the deoxyribozyme's substrate recognition region (Figure 1c, $\mathbf{YESi_1}$), which results in a **YES** gate[18] (i.e., a signal detector or repeater or basic catalytic molecular beacon). When the oligonucleotide complementary to the loop is present, it binds to the loop, opening the stem and thus allowing the substrate to bind to the deoxyribozyme, whereupon it is cleaved. The binding of an oligonucleotide to the loop of a stem–loop structure inserted into the catalytic core of a deoxyribozyme (Figure 2a) turns its enzymatic activity *off*, resulting in a **NOT** gate.[3] Using two stem–loop regions to block both substrate recognition regions of the deoxyribozyme results in an **AND** gate (Figure 2b), in which the presence of two inputs is needed for cleavage. Together, **AND** and **NOT** gates, with unlimited connectivity, theoretically suffice for logic circuit design, but it is advantageous to obtain more complex functionality directly within a single gate. Thus, for instance,

dx.doi.org/10.1021/ar5000538 | Acc. Chem. Res. 2014, 47, 1845–1852

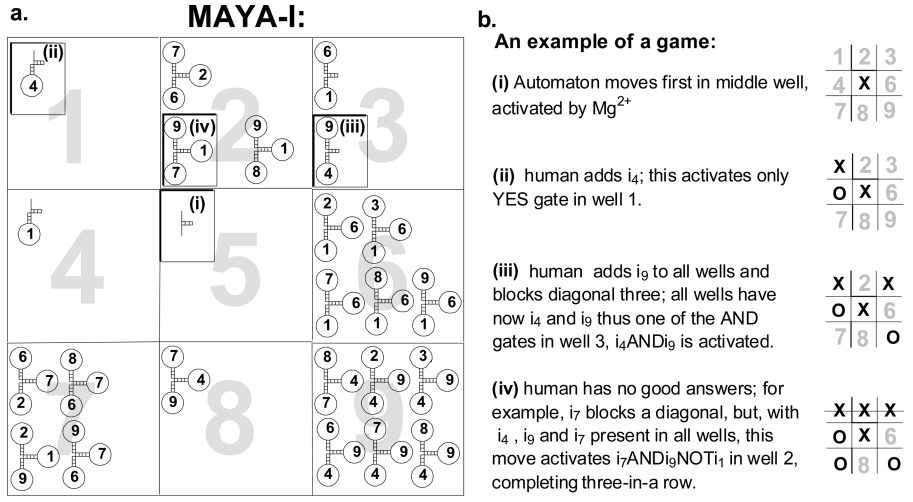


Figure 4. MAYA-I, an automaton that plays a symmetry-pruned tic-tac-toe game: (a) Distribution of gates in wells. The center well (5) contains a constitutively active deoxyribozyme, while the other wells contain logic gates. Gates used in our example game are boxed. (b) An example of a game in which the human does not play perfectly and therefore loses. There are a total of 19 games encoded in this distribution of logic gates.

using three stem−loop elements, we can create an **ANDANDNOT** gate (Figure 2c) that analyzes three inputs.[3,5] This gate computes the Boolean expression $\mathbf{i_1}$ **AND** $\mathbf{i_2}$ **AND NOT** $\mathbf{i_3}$; if more general gates are desired, the sense of the action of the input oligonucleotides can be reversed by precomplexing with their complements, which allows us to turn **ANDANDNOT** gates into **ANDAND** and **ANDNOT-NOT** gates (Figure 2d).[5] In this approach, there is no need for explicit **OR** gates because any two gates that cleave the same substrate are implicitly in an **OR** connection.

Having constructed elementary logic gates using deoxyribozymes, we set to arrange simple computational systems to take advantage of them. Multiple gates can operate in parallel by querying some of a set of inputs and then cleaving either the same shared substrate (in an implicit **OR** connection), or different substrates for multiple outputs (which can be monitored fluorogenically as different colors).

An early circuit that we constructed[4] was a half-adder (Figure 3a): in digital electronics, a half-adder receives two bits (binary digits) from two numbers being added together, and yields one bit of the sum and one bit of carry-forward. Our molecular representation of numbers is straightforward: the presence of the first oligonucleotide $\mathbf{i_1}$ stands for the value 1 of the first input bit and its absence for the value 0; similarly the presence or absence of $\mathbf{i_2}$ encodes the second bit. We take as outputs the cleavage of two separate substrates, which we monitor as fluorescence on two channels, having labeled the two substrates with different fluorophores. An unusual feature of this circuit, from the point of view of biochemistry, is that one and the same effector, say $\mathbf{i_1}$, functions either as an inhibitor or a promotor, depending on the presence of another effector, $\mathbf{i_2}$. Subsequently, we developed a larger circuit, the full adder,[5] which analyzes three inputs (two bits and a carry-in), to produce the sum and carry-out bits (Figure 3b). Multiple such circuits could be cascaded to yield multidigit adders; in electronic computers, such adders are common building blocks for their arithmetic-logical units.

Inspired by reading about Donald Michie's MENACE project,[21] we decided to develop automata for playing a game of strategy against human opponents, as a demonstration of information processing with molecules. The human can engage with these circuits in a dialogue of game moves, that is, the circuits coherently respond to a series of molecular stimuli by changing their state. To the extent that strategies for games can be rendered into Boolean logic at all,[22] the requisite formulas tend to be large and complex, and thus implementing a chosen Boolean formula for a particular game strategy represents an objective test of the engineering readiness of a computational medium.

We have built three generations of game-playing automata, MAYA I−III (originally, molecular array of YES and AND gates), in this vein. Common to all, the human conveys his game moves to the automaton using input oligonucleotides one at a time, and the automaton returns a sequence of fluorogenic responses for its game. **MAYA-I**[6] played a symmetry-pruned game of tic-tac-toe (Figure 4). **MAYA-II**[7] played the unrestricted game using a richer encoding of inputs (not shown). **MAYA-III**[8] could be trained to play specific strategies in a specially designed simple game (Figure 5).

In **MAYA-I**, the tic-tac-toe board is mapped to a 3 × 3 section of a well plate, numbered 1−9 (Figure 4). The automaton immediately claims well 5, and the human's first move is restricted to well 1 among corners and well 4 among sides. Thanks to this symmetry pruning, the automaton's strategy includes just 19 legal games, making the task of fully testing the circuit manageable. The implemented strategy is favorable: the automaton never loses, and it wins in 18 cases, drawing only if the human plays perfectly.

To initiate a game, we add Mg^{2+} to all wells. A plain deoxyribozyme in well 5 then becomes active and claims the well for **MAYA**. Human moves are keyed as eight input oligonucleotides and added to all wells, for example, to signal a move into well 1 the human adds input $\mathbf{i_1}$ everywhere. The automaton contains appropriate Boolean logic gates in each well, 23 in total, such that at each turn in the game, it correctly calculates a single-well response to the opponent's latest move.

The follow-on automaton **MAYA-II** contained 128 deoxyribozyme logic gates in 9 wells, and the human player was free to choose any of the peripheral wells, rather than a set corner or side, on the first move. There were 76 games within its designed strategy, which was implemented using 96 logic gates to respond to human moves. We chose to increase the number

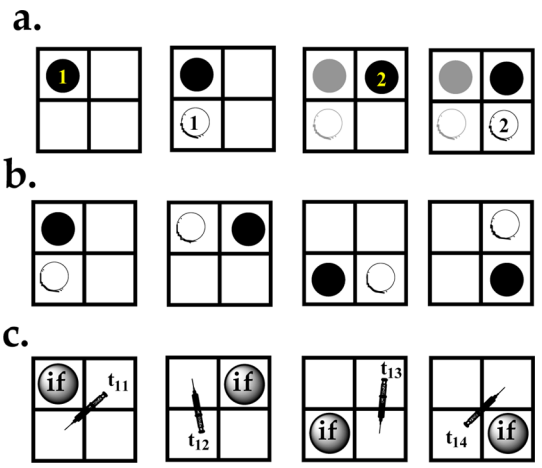


Figure 5. (a) An example of tit-for-tat game-play, with human moves shown as filled circles and automaton moves as hollow circles, with first and second moves labeled. Past moves are shown in gray and are not labeled any more; current moves are in black and are labeled as made. (b) An example of the strategy (a set of possible responses to all human first moves; in legal strategies, the automaton does have any choices in the second moves). (c) A part of the training session (first move) that teaches the automaton to play the strategy shown in part b. Training consists of injecting training inputs (t_{mn}) in an intuitive way, mimicking the actual game-play. The complete training protocol also covers second moves.

of inputs from 8 to 32 to encode both the well identity and the order of the move within the game. We also made the automaton easier to use by having it echo the human's moves in another color, using 32 **YES** gates with a different substrate. Thus, **MAYA-II** represented an engineering feat of scaling molecular circuitry up. One of the lessons of this scale-up was that while individual deoxyribozymes can work perfectly, there is a good chance that in a large mixture deoxyribozymes will interfere with each other in ways that cannot be predicted readily using simple Watson−Crick-based interaction models.

The foregoing two automata were molecular circuits hardwired to play versions of the game of tic-tac-toe. In contrast, **MAYA-III**[8] was designed with the idea that we can start with a blank slate, that is, an automaton without any recognizable function, and then train it to carry out various functions. **MAYA-III** can be trained to play any strategy possible for the retributive game *tit-for-tat* (an example of which is shown in Figure 5a), invented explicitly for the purpose of demonstrating the ability to train molecular automata. This two-player two-move game is played on a 2 × 2 board. While all such games are trivial, they have the advantage that after some restrictions in the way we observe moves (focusing only on the remaining fields for the second move), their complete action space (and more, because of the redundancies, that is, gates not used) can be represented with an array consisting of four **YES** gates, responding to the inputs keyed to the first moves, and 12 **AND** gates, responding to all legal combinations of inputs keyed to the first and second moves. Thus, we can select any function within this game space, all with only 16 gates using field programmable (reconfigurable) molecular logic arrays and "teaching by example" as a way of reconfiguring this logic. To achieve that, we placed these 16 gates under the control of additional instructional inputs that can be used in a very intuitive way, that is, without any knowledge of molecular programing. This turns each single-input **YES** gate into a two-input **AND** gate and each **AND** gate into a three-input **ANDAND** gate (Figure 2d).

The automaton's goal is to match each human move into a field with a move to any free field. The game has 81 winning strategies, defined as complete sets of responses to all possible moves by the human (cf. Figure 5b) leading to the automaton's fulfilling its goal (each strategy has 4 × 2 = 8 possible game plays). Besides oligonucleotides that represent human moves (total of eight, four for the first moves, and four for the second moves), we use training oligonucleotides, which let us activate the required gates in individual wells of the automaton in training sessions. Individual training sessions resemble actual game play and result in the automaton's learning how to play all possible games within one strategy (Figure 5c shows a training session for all possible responses to first moves). Each training session turns a fully symmetric distribution of gates in four wells into **MAYA-I**-like situations with the automaton playing only one strategy, chosen by training. The human trainer need not understand any molecular logic in order to select a strategy; the procedure requires one only to have a key for using the training inputs.

We are often asked whether there will be more **MAYA**s. Yes, we hope so, but with the advent of other molecular computing methods beyond deoxyribozymes, as well as broadened interest in deoxyribozyme computing,[23] it takes more time to select a truly novel concept to be demonstrated with automata.

Since our early "let's do something together" discussions we have wanted to apply molecular computing to solving medical problems. In retrospective, some of the early ideas and attempts to formulate some feasible approach to a practical application seem very naïve now. For example, we can look into our paper on connecting logical elements with aptamers,[24] which, at the time and still now, we considered an interesting proof-of-concept of a cascaded reaction in which a molecular computing element controls potentially therapeutic outcomes based on evaluation of biomarkers[24] (Figure 6). One problem that we

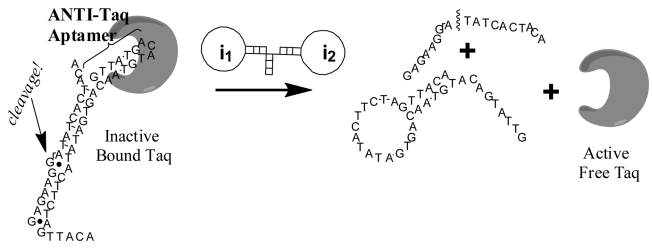


Figure 6. An **AND** gate controlling an anti-Taq aptamer. This is an example of a downstream event being controlled by molecular logic.

encountered when we started seriously considering how to address an actual disease in an actual living organism using molecular computing was that many of these initial ideas, such as this one, grew rapidly in complexity (e.g., measured by the number of subprojects that had to be finished before key proof-of-concept experiments are even attempted). Thus, to move forward more tangibly, we decided to go back to basics. We will now describe two approaches that we have demonstrated in vitro and are currently testing in animal models. They were demonstrated in the laboratory of our third coauthor Sergei Rudchenko, who provided expertise in flow cytometry and cell biology.

dx.doi.org/10.1021/ar5000538 | *Acc. Chem. Res.* 2014, 47, 1845−1852

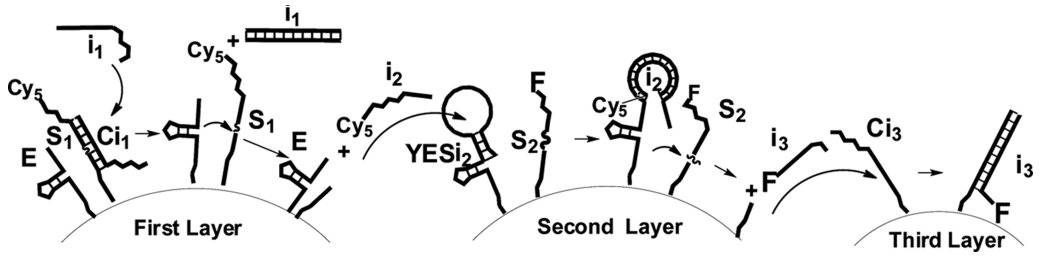


Figure 7. Three-layer cascade: The cascade starts with a first-layer bead that senses oligonucleotide $\mathbf{i}_1$. This bead is coated with a nucleic acid enzyme **E** and its substrate $\mathbf{S}_1$. The substrate is blocked by a complement to the input ($\mathbf{Ci}_1$), thus only upon the addition of the input can the enzyme cleave its substrate. The product $\mathbf{i}_2$ is released, behaving as an input for the $\mathbf{YESi}_2$ gate on the second-layer bead. The $\mathbf{YESi}_2$ gate, when activated, cleaves the $\mathbf{S}_2$ substrate, releasing product $\mathbf{i}_3$, which is captured by a complementary oligonucleotide ($\mathbf{Ci}_3$) at the third-layer beads. Product $\mathbf{i}_2$ is labeled with Cy5, while $\mathbf{i}_3$ is labeled with fluorescein, allowing us to observe information transfer down the cascade using flow cytometry.

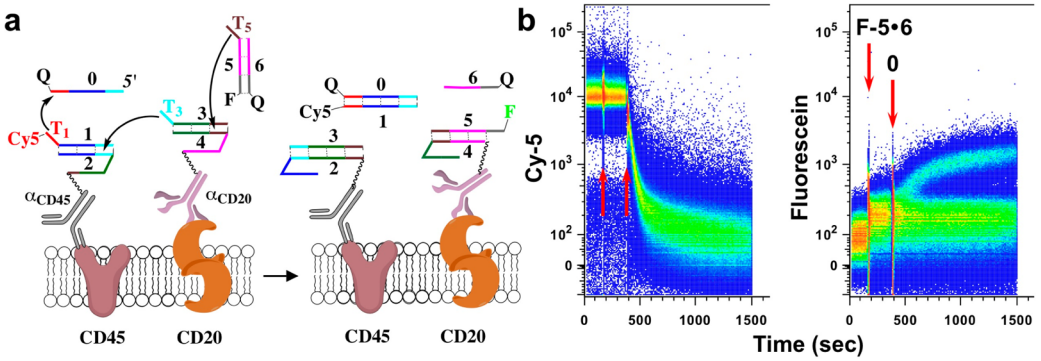


Figure 8. Example of an automaton assessing the presence of two cell surface markers. (a) Schematic representation of a **YESCD45YESCD20** automaton with the reaction $\mathbf{0} + \mathbf{1}^*\mathbf{2}_{\alpha CD45} + \mathbf{3}^*\mathbf{4}_{\alpha CD20} + \mathbf{5}^*\mathbf{6} \rightarrow \mathbf{0}^*\mathbf{1} + {}_{\alpha CD45}\mathbf{2}^*\mathbf{3} + {}_{\alpha CD20}\mathbf{4}^*\mathbf{5} + \mathbf{6}$ occurring on the cell surface: **1** is labeled with Cy5 and **0** labeled with a quencher for Cy5, and **5** is labeled with fluorescein and **6** with a quencher for fluorescein. (b) Flow cytometry monitoring of state transitions in automata in the instance of **YESCD45YESCD20**: Kinetics of the cascade reaction on $\mathbf{CD20}^+$ B-cells; (left panel) removal of Cy5-**1** after the triggering reaction with **0** monitors the transition occurring on all $\mathbf{CD45}^+$ cells; (right panel) fluorescein-labeled **5** is taken up from solution by $\mathbf{CD20}^+$ B-; this is used for monitoring the transition occurring on CD45 and CD20 positive cells. The events are indicated by arrows: (i) addition of **5***6, followed by a small immediate fluorescence increase on all cells due to the incomplete quenching; (ii) Addition of **0** triggers the cascade and the separation of subpopulations of cells.

INFORMATION EXCHANGE BETWEEN BEADS

At the time we developed bead-to-bead communication as a concept, we became interested in expanding the possibilities of nanotechnology. In then-current approaches to nanomedicine, more complex functions of particles were achieved by loading a single particle with more functionalities. Instead, we proposed to achieve an increase in complexity of functions by forming networks of simpler particles, which can be individually targeted to the same cell or different cells. To explain this concept, we use the example of a cascade formed by three sets of beads coated with elements similar to those used in deoxyribozyme-logic gates.

The elementary unit of a network is a single particle covered with a DNA computing or sensing element (Figure 7). An individual bead senses the presence of an input stimulus (or multiple stimuli) in solution, and according to a set of rules encoded on this bead by computing elements, it releases an oligonucleotide-based signal as an output. This can occur through any one of the logic gates that are deposited on the bead together with substrate, and the released oligonucleotide is one of the products. This signal can diffuse and interact with another DNA element on another bead (downstream element of a cascade), leading to information transfer between two particles and a cascade. The communication between elements requires no physical contact, and it occurs over the long-range through diffusion of signaling molecules. We can monitor the network activity with polychromatic flow cytometry, if we label individual oligonucleotides with different fluorophores.

In a three-layer cascade that we demonstrated, the first layer consists of a bead coated homogeneously with deoxyribozyme gate and its substrate $\mathbf{S}_1$. The substrate is blocked with its complement, which can be removed via a more complementary input $\mathbf{i}_1$. Thus, adding the input initiates the cleavage of substrate and release of a product, $\mathbf{i}_2$. Product $\mathbf{i}_2$ is also an input for the second-layer bead, which is homogeneously coated with $\mathbf{YESi}_2$ and $\mathbf{S}_2$. Activation of $\mathbf{YESi}_2$ results in the second-layer beads releasing the product $\mathbf{i}_3$, which is captured by the third and final layer beads, resulting in fluorescence at those beads. Of note, the omission of either first or second layers results in no increase in fluorescence of the third layer. A more complex network, an **AND** hub, can be found in our initial publication.[25] These results represent a step toward a network that could detect the proximity of two cell types, which we hope to demonstrate will have practical applications in imaging.

COMPUTING ON CELL SURFACES

To expand molecular computing to actual medical problems,[26] we started by selecting a set of problems that can be solved using drugs that would "know" Boolean algebra. Most traditional drugs (currently popular polypharmacology aside) work because they have a single target. However, we quickly focused on targeting lymphocytes, because these cells are

dx.doi.org/10.1021/ar5000538 | Acc. Chem. Res. 2014, 47, 1845−1852

characterized in a way that is very much Boolean in nature (credit for firt proposing targeting these cells goes to Dr. Vincent Butler, Emeritus Professor of Medicine, Columbia University, and Immunologist, who helped M.N.S. write his first grants on this topic). Numerous subpopulations of lymphocytes are defined by the presence or absence of multiple cell surface markers, that is, their lineages and stages of differentiation are uniquely characterized through the different levels of expression of multiple cell surface markers known as clusters of differentiation or CDs; we use CD45 (and its isoforms, CD45RA and CD45RO), CD20, CD3, and CD8 as examples. Blood cells, of which lymphocytes are one type, are commonly characterized by flow cytometry based on binding of fluorescently labeled antibodies to these markers (e.g., anti-CD*x* antibody will provide characteristics such as **CD*x*$^+$**, **CD*x*$^-$**, or **CD*x*^{high}**, **CD*x*^{low}**, or **CD*x*^{dim}** for intermediate expression levels of marker ***x***). For example, in our first demonstrations we used **CD45$^+$CD20$^+$** and **CD45$^+$CD3$^+$** to characterize B- and T-cells, respectively, which are examples of Boolean **AND** logic over two markers, respectively, **CD45ANDCD20** (Figure 8) and **CD45ANDCD3**. While in standard Boolean logic the order of inputs does not matter, in the way we implemented logic on lymphocytes the order does matter; to emphasize this, we write **YESCD45YESCD20** rather than **CD45ANDCD20**.

So, how would we implement an automaton that evaluates the cell for the presence of two markers, CD45 (marker of all hematopoietic cell) and CD20 (B-cell marker)? We can have two oligonucleotides, one attached to an anti-CD45 (e.g., **2** in Figure 8) and one attached to an anti-CD20 antibody (e.g., **4** in Figure 8), signal their presence to each other when they are close by.[25] While signaling can be done through enzyme−substrate cleavage, as we did it on beads, for various practical reasons, it was easier to have an oligonucleotide (e.g., **3**) move from its less-complementary oligonucleotide attached to anti-CD20 to the more complementary oligonucleotide attached to anti-CD45 antibody. This move of an oligonucleotide to form a longer double helix and dismantle a shorter double helix is then nothing but a strand displacement reaction,[27] in which **2** displaced **3** from its complex with **4**, that is, **2** + **3***4 → **2***3 + **4**. In order to trigger the reaction (i.e., to block it from occurring before we want it to occur), we introduce a solution-phase oligonucleotide **0** to displace **1** from the less complementary **2** (triggering information exchange between proximal cell-surface markers). In order to monitor the appearance of **4**, we can have it react with solution phase **5***6, in which **5** is labeled with fluorescein (**F**). Thus, in the end, we have a coupled series of strand-displacement reactions **0** + **1***2 + **3***4 + **5-F***6 → **2***3 + **4***5-F + **6**, driven by higher complementarity of oligonucleotides, which results in labeling only those cells that have both CD45 and CD20 on their surfaces with fluorescently labeled oligonucleotide **5-F**, thus executing a Boolean **AND** function.

Simple expansion of the cascade, that is, attaching **5***6 to the third antibody, expands analysis to the third marker. Modification of the cascade can also lead to a **NOT** function on automata, in which acquisition of an oligonucleotide such as **5-F** from the solution is blocked by the faster acquisition of the similar (blocking) oligonucleotide from the proximal surface marker. For example, **CD45$^+$** cells that do not have the CD20 marker on the surface would be labeled if we add **3-F***4 in solution, while **CD45$^+$CD20$^+$** cells, such as in Figure 8, would not.

What we have shown up to now is that we can take multiple antibodies, typically used in polychromatic flow cytometry, and condense information about their presence and absence on cell surface in a color. Perhaps this approach has applications in expanding the number of antibodies we can use in standard flow cytometry protocols, while keeping the number of fluorochromes minimal. Also, the ability to contract recognition by several antibodies may have application in rapid elimination of cell subpopulations in vitro as part of preparations for autologous transplantation. However, our next goal is to implement cascades like this in vivo and to couple them with cell elimination, for example, by substituting a toxin for fluorescein.

■ CONCLUSIONS

For our younger colleagues—Accounts readers who are considering starting an academic career and pondering what kind of risk to take in selecting their first projects (i.e., those at "life-forks")—it may be useful to glean some historical perspective on doing a long-term project with very little immediate practical consequence and very little precedent.

So, to recapitulate, more than 15 years have passed since we first considered a project on DNA computing and about 13 years since we first submitted a grant proposal on this topic and constructed the first real logic gate with oligonucleotides. It may be useful to consider these years in contrast with typical three-to-five year grant horizons. At the time of our first discussion, DNA computing looked much different; the only implemented approach had been human-experimentator guided (i.e., nonautonomous): Adleman's solution of a small example of a small traveling salesman problem.[12] Moreover, even within our own work, the systems that we pursue now for in vivo development look nothing like what we thought or proposed they would after our first experiments. Importantly, despite all claims about bias against projects that are not typical, we were sufficiently funded at all times by different agencies and organizations. There were two communities that were particularly supportive of our research. First, in our experience the medical community (not chemists, but practicing physicians and physician-scientists) has been very receptive, likely because they realize that there ought to be long-term investment into completely new approaches to treat, for example, cancers and diabetes. Second, we were fortunate to be able to apply for our first grants at the right time, that is, a community of open-minded computer scientists (including those program officers serving at that time and now at NSF) has been primed to accept that concepts from computer science are ready to be applied and expanded to the world of molecules. We readily admit that, aside from Len Adleman's initial experiment,[12] we had largely Ned to thank for this, as it was he who, through a joint project with Erik Winfree, broke the ice.[28]

■ AUTHOR INFORMATION

Corresponding Author

*E-mail address: mns18@columbia.edu.

Funding

The authors have been generously supported by NSF and NIH. The first project directly proposing specific medical applications of molecular computing was funded by NASA. M.N.S. was a Lymphoma and Leukemia Society Fellow.

Notes

The authors declare no competing financial interest.

Biographies

Milan N. Stojanovic is an organic chemist interested in molecular computing, robotics, sensing, and traditional drug discovery. His Ph.D. and postdoctoral advisers were Yoshito Kishi and Donald Landry, respectively.

Darko Stefanovic is a computer scientist. He trained in programming languages under Eliot Moss, Kathryn M^{c}Kinley, and Margaret Martonosi. Over the past decade, molecular computing became for him first a pastime, then an occupation.

Sergei Rudchenko is a biophysicist (Ph.D. adviser was Enno Ruuge) with additional postdoctoral training in protein chemistry (under Joan Sobel), cellular biology (under Vladimir Smirnov), and immunology and molecular biology (under Kathrin Calame). Over the last dozen years, his interest has been focused on application of molecular computing in cell biology and immunology.

REFERENCES

(1) Credi, A. Molecules That Make Decisions. *Angew. Chem., Int. Ed.* **2007**, *46* (29), 5472−5475.
(2) Katz, E.; Privman, V. Enzyme-Based Logic Systems for Information Processing. *Chem. Soc. Rev.* **2010**, *39* (5), 1835−1837.
(3) Stojanovic, M. N.; Mitchell, T. E.; Stefanovic, D. Deoxyribozyme-Based Logic Gates. *J. Am. Chem. Soc.* **2002**, *124* (14), 3555−3561.
(4) Stojanović, M. N.; Stefanovic, D. Deoxyribozyme-Based Half-Adder. *J. Am. Chem. Soc.* **2003**, *125* (22), 6673−6676.
(5) Lederman, H.; Macdonald, J.; Stefanovic, D.; Stojanovic, M. Deoxyribozyme-Based Three-Input Logic Gates and Construction of a Molecular Full Adder. *Biochemistry* **2006**, *45* (4), 1194−1199.
(6) Stojanovic, M. N.; Stefanovic, D. A Deoxyribozyme-Based Molecular Automaton. *Nat. Biotechnol.* **2003**, *21* (9), 1069−1073.
(7) Macdonald, J.; Li, Y.; Sutovic, M.; Lederman, H.; Pendri, K.; Lu, W.; Andrews, B.; Stefanovic, D.; Stojanovic, M. N. Medium Scale Integration of Molecular Logic Gates in an Automaton. *Nano Lett.* **2006**, *6* (11), 2598−2603.
(8) Pei, R.; Matamoros, E.; Stefanovic, D.; Stojanovic, M. N. Training a Molecular Automaton to Play a Game. *Nat. Nanotechnol.* **2010**, *5* (11), 773−777.
(9) Macdonald, J.; Stefanovic, D.; Stojanovic, M. N. Smart DNA: Programming the Molecule of Life for Work and Play. *Sci. Am.* **2008**, *299*, 84−91.
(10) Pei, R.; Taylor, S.; Rudchenko, S.; Stefanovic, D.; Mitchell, T. E.; Stojanovic, M. N. Behavior of Polycatalytic Assemblies in a Substrate-Displaying Matrix. *J. Am. Chem. Soc.* **2006**, *128* (39), 12693−12699.
(11) Lund, K.; Manzo, A. J.; Dabby, N.; Michelotti, N.; Johnson-Buck, A.; Nangreave, J.; Taylor, S.; Pei, R.; Stojanovic, M. N.; Walter, N. G.; Winfree, E.; Yan, H. Molecular Robots Guided by Prescriptive Landscapes. *Nature* **2010**, *465* (7295), 206−210.
(12) Adleman, L. M. Molecular Computation of Solutions to Combinatorial Problems. *Science* **1994**, *266* (5187), 1021−1024.
(13) Osborne, S. E.; Ellington, A. D. Nucleic Acid Selection and the Challenge of Combinatorial Chemistry. *Chem. Rev.* **1997**, *97* (2), 349−370.
(14) Breaker, R. R. *In Vitro* Selection of Catalytic Polynucleotides. *Chem. Rev.* **1997**, *97* (2), 371−390.
(15) Tang, J.; Breaker, R. R. *Chem. Biol.* **1997**, *4* (6), 453−459.
(16) Robertson, M. P.; Ellington, A. D. In Vitro Selection of an Allosteric Ribozyme That Transduces Analytes to Amplicons. *Nat. Biotechnol.* **1999**, *17* (1), 62−66.
(17) Allen, W. Yes, But Can the Steam Engine Do This. *The Insanity Defense: The Complete Prose*; Random House: New York, 2007; p 32.
(18) Stojanovic, M. N.; de Prada, P.; Landry, D. W. Catalytic Molecular Beacons. *ChemBioChem* **2001**, *2* (6), 411−415.
(19) Breaker, R. R.; Joyce, G. F. A DNA Enzyme with Mg^{2+}-Dependent RNA Phosphoesterase Activity. *Chem. Biol.* **1995**, *2* (10), 655−660.
(20) Tyagi, S.; Krammer, F. R. Molecular Beacons: Probes That Fluoresce upon Hybridization. *Nat. Biotechnol.* **1996**, *14* (3), 303−309.
(21) Gardner, M. *The Colossal Book of Mathematics: Classic Puzzles, Paradoxes, and Problems* W.W. Norton & Company: New York, 2001; pp 472−473.
(22) Stefanovic, D.; Stojanovic, M. N. Computing Game Strategies. *The Nature of Computation, 9th Conference on Computability in Europe, Milan, Italy July 1-July 5, 2013, Proceedings* Bonizzoni, P., Brattka, V., Löwe, B., Eds.; Springer-Verlag: Berlin, 2013; pp 383−392.
(23) Elbaz, J.; Lioubashevski, O.; Wang, F.; Remacle, F.; Levine, R. D.; Willner, I. DNA Computing Circuits Using Libraries of DNAzyme Subunits. *Nat. Nanotechnol.* **2010**, *5* (6), 417−422.
(24) Kolpashchikov, D.; Stojanovic, M. N. Boolean Control of Aptamer Binding States. *J. Am. Chem. Soc.* **2005**, *127* (32), 11348−11352.
(25) Yashin, R.; Rudchenko, S.; Stojanovic, M. N. Networking Particles over Distance Using Oligonucleotide-Based Devices. *J. Am. Chem. Soc.* **2007**, *129* (50), 15581−15584.
(26) Rudchenko, M.; Taylor, S.; Pallavi, P.; Deschovskaia, A.; Khan, S.; Butler, V. P., Jr; Rudchenko, S.; Stojanovic, M. N. Autonomous Molecular Cascades for Evaluation of Cell Surfaces. *Nat. Nanotechnol.* **2013**, *8* (8), 580−586.
(27) Seelig, G.; Soloveichik, D.; Zhang, D. Y.; Winfree, E. Enzyme-Free Nucleic Acid Logic Circuits. *Science* **2006**, *314* (5805), 1585−1588.
(28) Winfree, E.; Liu, F.; Wenzler, L. A.; Seeman, N. C. Design and Self-Assembly of Two-Dimensional DNA Crystals. *Nature* **1998**, *394* (6693), 539−544.