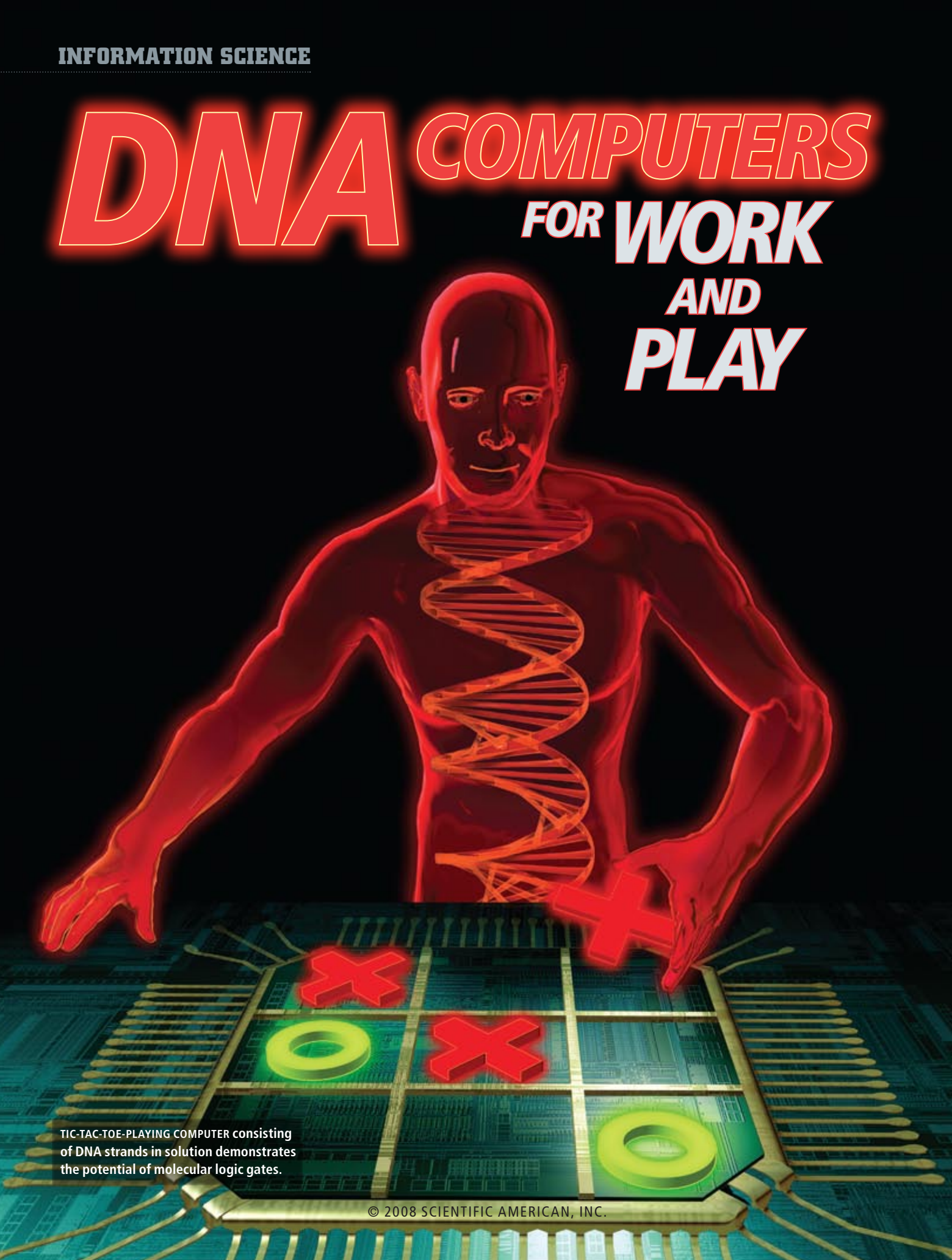
DNA COMPUTERS FOR WORK AND PLAY



TIC-TAC-TOE-PLAYING COMPUTER consisting of DNA strands in solution demonstrates the potential of molecular logic gates.

Logic gates made of DNA could one day operate in your bloodstream, collectively making medical decisions and taking action. For now, they play a mean game of in vitro tic-tac-toe

By Joanne Macdonald, Darko Stefanovic and Milan N. Stojanovic

From a modern chemist's perspective, the structure of DNA in our genes is rather mundane. The molecule has a well-known importance for life, but chemists often see only a uniform double helix with almost no functional behavior on its own. It may come as a surprise, then, to learn that this molecule is the basis of a truly rich and strange research area that bridges synthetic chemistry, enzymology, structural nanotechnology and computer science.

Using this new science, we have constructed molecular versions of logic gates that can operate in water solution. Our goal in building these DNA-based computing modules is to develop nanoscopic machines that could exist in living organisms, sensing conditions and making decisions based on what they sense, then responding with actions such as releasing medicine or killing specific cells.

We have demonstrated some of the abilities of our DNA gates by building automata that play perfect games of tic-tac-toe. The human player adds solutions of DNA strands to signal his or her moves, and the DNA computer responds by lighting up the square it has chosen to take next. Any mistake by the human player will be punished with defeat. Although game playing is a long way from our ultimate goals, it is a good test of how readily the elementary molecular computing modules can be combined in plug-and-play fashion to perform complicated functions, just as the silicon-based gates in modern computers can be wired up to form the complex logic circuits that carry out everything that computers do for us today.

Dissolved Doctors

Near the end of 1997 two of us (Stojanovic and Stefanovic) decided to combine our individual skills in chemistry and computer science and work on a project together. As friends from elementary school in Belgrade, Serbia, we happened to be having dinner, and, encouraged by some wine, we considered several topics, including bioinformatics and various existing ways of using DNA to perform computations. We decided to develop a new method to employ molecules to compute and make decisions on their own.

We planned to borrow an approach from electrical engineering and create a set of molecular modules, or primitives, that would perform elementary computing operations. In electrical engineering the computing primitives are called logic gates, with intuitive names such as AND, OR and NOT. These gates receive incoming electrical signals that represent the 0s and 1s of binary code and perform logic operations to produce outgoing electrical signals. For instance, an AND gate produces an output 1 only if its two incoming inputs are both 1. Modern-day computers have hundreds of millions of such logic gates connected into very complex circuits, like elaborate structures built out of just a few kinds of Lego blocks. Similarly, we hoped that our molecular modules could be mixed together into increasingly complex computing devices.

We did not aim, however, to compete with silicon-based computers. Instead, because Stojanovic had just finished a brief stint with a pharmaceutical company, we settled on developing a system that could be useful for making "smart" therapeutic agents, such as drugs that could sense and analyze conditions in a patient and respond appropriately with no human intervention after being injected. For example, one such smart agent might monitor glucose levels in the blood and decide when to release insulin. Thus, our molecular logic gates had to be biocompatible.

Such molecular modules could have innumerable functions. For instance, in diseases such as leukemia, numerous subpopulations of white blood cells in the immune system display char-

KEY CONCEPTS

- DNA molecules can act as elementary logic gates analogous to the silicon-based gates of ordinary computers. Short strands of DNA serve as the gates' inputs and outputs.
- Ultimately, such gates could serve as dissolved "doctors"—sensing molecules such as markers on cells and jointly choosing how to respond.
- Automata built from these DNA gates demonstrate the system's computational abilities by playing an unbeatable game of tic-tac-toe.

—The Editors

JEAN-FRANCOIS PODEVIN

acteristic markers on their cell surfaces, depending on the cells' lineage and their stage of development. Present-day therapies using antibodies eliminate large numbers of these subpopulations at once, because they target only one of the surface markers. Such indiscriminate attacks can suppress the patient's immune system by wiping out too many healthy cells, leading to serious complications and even death. Molecular modules capable of working together to sense and analyze multiple markers—including performing logical operations such as "markers A and either B or C are present, but D is absent"—might be able to select the specific subpopulations of cells that are diseased and growing out of control and then eliminate only those cells.

Another application of our modules could be in the analysis of DNA, looking for a large array of possible genetic mutations or identifying one of a wide variety of microbiological pathogens. Our most advanced tic-tac-toe-playing automaton combines 32 different short DNA sequences (oligonucleotides). That many logic gate inputs could analyze four billion possible combinations of oligonucleotides and partition them into thousands of patterns, each pattern being characteristic of certain pathogens or genotypes.

Molecular Logic

Researchers reported logic gates based on synthetic molecules as long ago as the early 1990s. In 1993, for instance, A. Prasanna de Silva and his collaborators at Queen's University Belfast made AND gates out of small organic molecules that would fluoresce only if both hydrogen ions (from acid) and sodium ions were bound to them. In 1997 J. Fraser Stoddart, now at Northwestern University, and his co-workers made "exclusive OR" (XOR) gates, in which the molecules fluoresced in the presence of either, but not both, of the inputs (in this case, hydrogen ions and molecules called amines). These examples, however, were not biocompatible, because they required concentrations of acid and other compounds that would harm living cells.

In the mid-1990s other researchers exploited DNA's ability to store information in its sequence of bases—the molecules conventionally abbreviated as A, T, G and C, which pair up to form the rungs connecting the two strands of the famous double-helix structure. Their techniques, however, were very different from the kind of system we envisaged, namely, one in which molecular logic gates floating in solution would process inputs and outputs in a fashion very analogous to the workings of silicon logic gates. Nevertheless, DNA clearly had a lot of potential for biocompatible computation, and a couple of other advances gave us the tools to invent our own brand of DNA logic gates.

First, in 1995 Gerald F. Joyce of the Scripps Research Institute in La Jolla, Calif., developed a method for producing enzymes made out of single strands of DNA that cut other pieces of single-stranded DNA into two segments. These so-called deoxyribozymes have two short arms that will bind only to another stretch of DNA that has the correct complementary sequence of bases, so they are very specific about which substrate DNA strands they will cleave [*see box on page 88*].

Special dye molecules attached to each end of the substrate strands enable laboratory workers to monitor the cleaving process. At one end of the substrate, the dye molecule is a "quencher," which prevents the fluorescent marker dye at the other end from fluorescing as long as the strand remains intact, keeping the quencher close enough to be effective. After the strand is cut, its two pieces move apart and the marker dye molecule can fluoresce unhindered. As the work of the DNA enzymes progresses, cutting more and more strands, the solution gradually lights up with the marker dye's fluorescent color.

The other key advance came soon after our initial planning, when Ronald R. Breaker of Yale University reported a way to integrate a deoxyribozyme with molecular groups acting as recognition modules. These modules work like sensors that either activate or inhibit their attached DNA enzyme when the correct input molecule is bound to them. Breaker even combined two such modules in a construct that could serve as an AND gate with two small input molecules. Very intriguingly, his group has found that such two-sensor constructs have been used by natural riboswitches—molecules made of RNA used by bacteria to control which of their genes actively produce proteins [see "The Power of Riboswitches," by Jeffrey E. Barrick and Ronald R. Breaker; Scientific American, January 2007].

We saw that we could build our logic gates out of DNA enzymes integrated with controlling sensor modules designed to recognize short DNA strands having specific base sequences. The DNA strands would thus act as inputs to the logic gates (an input of 1 if the strand is present; 0 if it is absent), and the gates' enzymes would output "1" by cleaving other DNA strands in the solution. With DNA serving as both inputs and outputs, our gates could in principle be chained together

OTHER DNA COMPUTERS

Researchers over the years have devised several ways to perform computations by exploiting DNA's ability to store information in its sequence of bases.

1994: Leonard M. Adleman of the University of Southern California solved a puzzle known as the Hamiltonian path problem by encoding all the possible solutions (both correct and incorrect) on a large number of DNA molecules and carrying out a series of steps to isolate the molecules with the correct solution [see "Computing with DNA," by Leonard M. Adleman; Scientific American, August 1998].

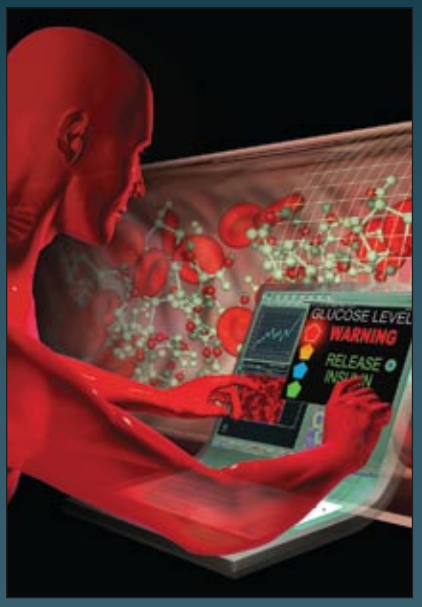
1995: Erik Winfree, now at the California Institute of Technology, proposed that tiles made of DNA could be designed to perform computations by self-assembling into two-dimensional structures [see "Nanotechnology and the Double Helix," by Nadrian C. Seeman; Scientific American, June 2004].

2004: Ehud Shapiro of the Weizmann Institute of Science in Rehovot, Israel, and Yaakov Benenson of Harvard University, building on a proposal by Paul W. K. Rothemund of Caltech, developed a "doctor in a cell." Enzymes operating on DNA analyzed whether a combination of RNA molecules indicative of a disease was present in the solution and responded by releasing another molecule as a model for a drug [see "Bringing DNA Computers to Life," by Ehud Shapiro and Yaakov Benenson; Scientific American, May 2006].

[APPLICATIONS]

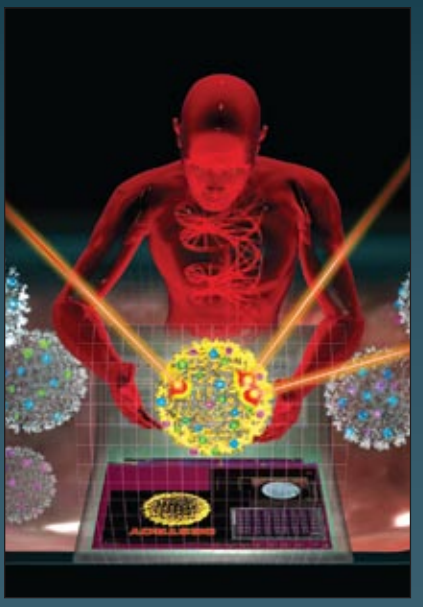
JOBS FOR INTELLIGENT DNA

DNA logic gates could have many applications, ranging from medical treatments to counterterrorism.



INJECTABLE PANCREAS

Logic gates operating in the bloodstream of a diabetic patient could monitor glucose levels and release insulin when appropriate.



TARGETED TREATMENT

Gates that sense different markers on white blood cells and combine their data could target leukemia cells for destruction while sparing healthy cells that may have some but not all the same markers.

COUNTERTERRORISM

DNA-based chemical sensors, along with DNA logic gates, could sniff out previously unknown nerve agents such as Soviet-made "novichok" chemicals as well as more familiar ones such as sarin.

to form complex circuits. Like wires in electrical circuits, the base sequences of the sensors and the enzymes would control which gates' outputs "connected" to which inputs, even as all the gates sloshed around independently in a test tube.

After some less than successful attempts using other designs, we settled on DNA structures known as stem-loops for our recognition modules. Sanjay Tyagi and Fred Kramer, both at the Public Health Research Institute in Newark, N.J., had reported that stem-loops switch between two shapes, or conformations. In the closed conformation the DNA strand making up the stem-loop folds onto itself, and the two ends zip together, forming a stem along with a loop of unzipped DNA, like the outline of a lollipop. An input DNA strand consisting of the sequence of bases complementary to the loop will bind to it, but in forming a stretch of the familiar double helix it pries the stem apart—the double-helical DNA cannot form a tight enough curve to maintain the closed loop.

Depending on how we attach a stem-loop to a DNA enzyme, opening the loop may either activate or inhibit the enzyme's activity. If one of the enzyme's two substrate-matching arms serves as one side of the stem, then the closed stem will block the enzyme's activity. We call this structure a sensor or a YES gate because adding the input strand (say, "input X") for the stem-loop controller opens the stem, exposing the enzyme's substrate-matching region and allowing it to function. The enzyme's output (specific cleaved strands of DNA) in essence says, "YES, input X is present."

Adding a second stem-loop with a different loop sequence (Y) on the other of the enzyme's two arms yields an AND gate. Only if input X AND input Y bind to it can the enzyme function and cleave DNA [*see box on page 89*].

We make an inhibitory controller—one that will deactivate the enzyme when the correct input binds to the loop—by plugging a stem-loop sequence into the "back" of the enzyme. Now when the stem is closed, the enzyme is intact and produces output. The relevant input strand will open the stem-loop and deform the enzyme enough to inactivate it. Of course, this inactivation will not remove output strands already produced by the gate, so in isolation this NOT gate does not function as conveniently as an electronic NOT gate. But the NOT unit comes into its own when combined with the AND gate structure. The resulting gate, which we call AND-

[THE AUTHORS]

Joanne Macdonald, Darko Stefanovic and Milan N. Stojanovic bring very different backgrounds to the task of programming DNA to compute. Macdonald is an associate research scientist at Columbia University. She conducts biology-related research within the division of clinical pharmacology and experimental therapeutics, and pursues practical applications of DNA computing for viral detection. Stefanovic is an associate professor of computer science at the University of New Mexico working on algorithms for memory management in computers. He is the recipient of a U.S. National Science Foundation (NSF) CAREER award. Stojanovic is associate director of the division of clinical pharmacology and experimental therapeutics at Columbia and director of the NSF Center for Molecular Cybernetics. He is a Leukemia & Lymphoma Society Fellow. MAYA is named after his daughter.

JEAN-FRANCOIS PODEVIN

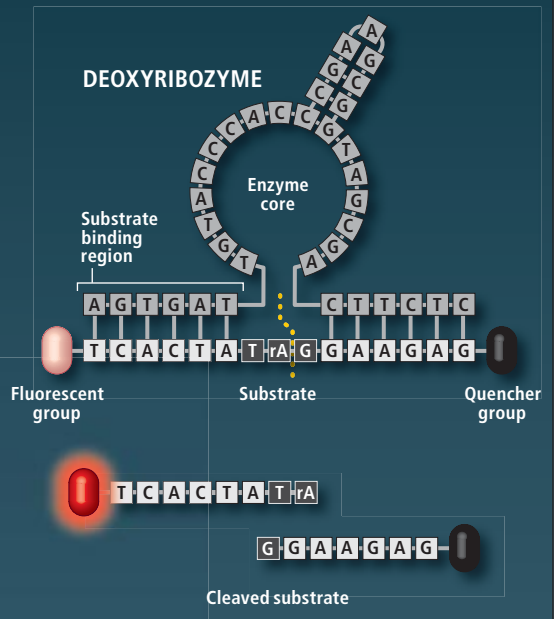
[WORKING PARTS]

MOLECULAR MODULES

To perform as logic gates analogous to silicon ones, a technology must produce specific outputs in response to a variety of inputs. DNA enzymes and recognition modules provide these output and input functions for a system based on DNA in solution.

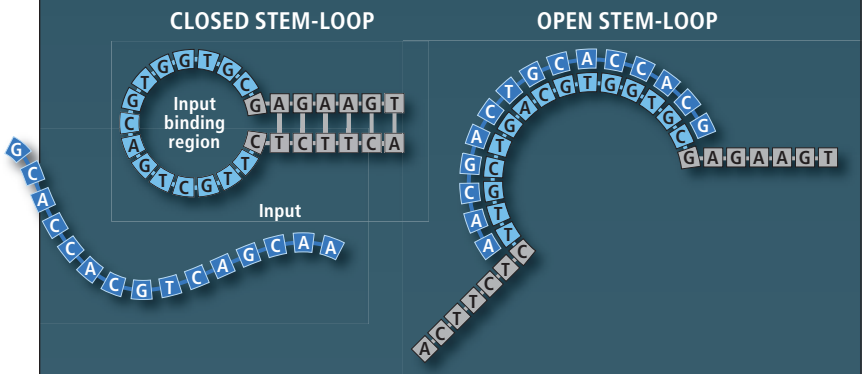
OUTPUT ENZYME

A DNA enzyme called a deoxyribozyme (*top*) consists of single-stranded DNA folded into a "core" structure with arms at each end that can bind to a substrate DNA strand that has the complementary sequences of bases separated by a specific sequence of three other bases (*dark gray*). The enzyme cleaves the strand into two pieces (*bottom*). The process can be monitored by attaching a fluorescent molecule at one end of the substrate strands and a quencher molecule at the other end. The quencher molecule blocks fluorescence until the cleaving of the strand takes it out of range.



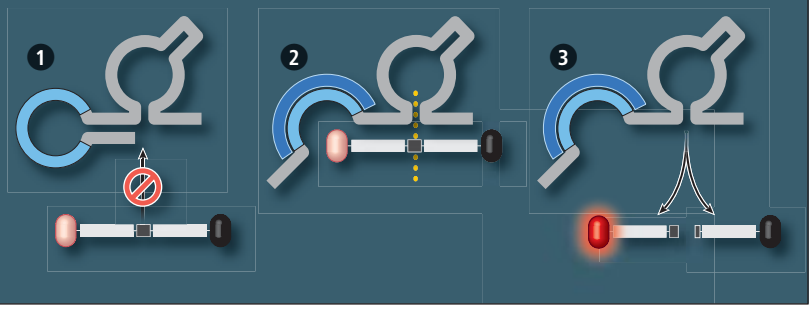
INPUT SENSOR

In a DNA structure called a stem-loop, the DNA folds onto itself and zips together to form a double-stranded stem with a single-stranded loop (*left*). When a matching input strand binds to the loop, it pries the stem apart (*right*).



SENSOR GATE

A stem-loop attached to the arm of an enzyme blocks the enzyme's function (*1*) until an input DNA strand opens the controller and exposes the arm (*2*), enabling the enzyme to bind and cleave substrates (*3*). This structure is also called a YES gate because it signals, "Yes, the input is present."



AND-NOT, produces output only if inputs X AND Y AND NOT Z are present. That function, also known as an INHIBIT gate, turned out to be very useful for our tic-tac-toe automata.

The most important aspect of our system is that it is highly modular. We can use hundreds and theoretically millions of different base sequences for the inputs, and we can also change the sequences of the output strands. We could even switch the underlying enzyme to be a ligase, one that joins together short strands to produce longer ones. Indeed, Andrew D. Ellington's group at the University of Texas at Austin has studied ligase-based switches extensively.

The functioning of the gates is also autonomous. That is, once we trigger a computation by adding the input to the solution, no more human intervention is required. In essence, DNA molecules make the decisions on their own, based on whatever inputs they receive.

Our gates do have some significant differences, however, from the silicon-based logic in electrical circuits. First, we cannot reset our gates. Once an input strand is bound to a stem-loop controller, it tends to remain there for the rest of the computation. Nor can the cleaved oligonucleotide output strands be reassembled. Our ultimate biomedical goals do not require a gate-reset function, but it would be useful for potential molecular robotics applications (involving moving parts). We are exploring the use of ligase enzymes to reassemble output strands.

Second, electronic gates have a threshold voltage at which their switching happens, and their outputs are tied to specific voltages so that they cannot linger at an intermediate voltage. Thus, the 0s and 1s are well defined, and the logic is truly digital. Solutions of our gates, in contrast, change in continuous fashion between the inactive and the fully active forms depending on how many inputs we add to the fluid. This behavior would be important if we were attempting to build the molecular equivalent of a personal computer, but it does not matter for many biomedical applications.

DNA Plays Tic-Tac-Toe

With a general approach to constructing molecular logic gates in our hands, we looked for an objective test of their ability to compute. We wanted to apply our logic gates in a situation in which everyone would immediately see that the molecules were making decisions. A traditional test for a new computer system is to make it play a game of strategy. The rules of a game provide

TOMMY MOORMAN

a challenge with a straightforward measure of success: the system will either be able to play the game or not. Game-playing ability is intimately connected with general computational ability.

We chose the classic children's game of tic-tac-toe for our demonstration. In this game, played on a 3 × 3 grid, two players try to put three marks in a row while blocking the opponent from doing the same. Tic-tac-toe is one of the simplest two-player games of perfect information, meaning that a player knows everything that there is to know about the state of the game at each move (unlike, for instance, most card games, in which rivals' cards are unknown). Tic-tac-toe will always end in a draw if both parties play well, but our device will exploit any mistake the opponent makes.

The game is simple enough that we can encode all decision making into logic operations that examine only the opponent's moves. That is, when you are using a fixed strategy, even if you remember only what your opponent's moves have been, you can work out what your own past moves must have been and therefore what the current board position is and what your strategy dictates as your next move. We condensed that chain of reasoning down to a network of logic gates that takes the opponent's moves as inputs and produces your next move as the output. In 2002 we set out to build just such a network out of DNA logic gates, a tic-tac-toe-playing automaton that we christened MAYA (*m*olecular *a*rray of YES and *A*ND-AND-NOT gates).

MAYA consists of nine wells corresponding to the squares of the tic-tac-toe grid. Each well contains its own precisely defined set of DNA logic gates in solution [*see box on next page*]. The enzymes of these gates are all designed to cleave the same substrate DNA strand, which is also in all the wells, but they require magnesium ions to function. Thus, adding magnesium ions stirs MAYA into action. Because the enzymes in the central well have no stem-loop controllers on them, they start cleaving the substrate immediately. The fluorescence from the central well increases, signaling that MAYA has taken the central square as the opening move.

The human (let's call him Harry) has eight input strands (one for each of the eight remaining squares) for inputting his moves. The base sequences of these strands are complementary to the sequences on the stem-loops that control MAYA's DNA gates. To move in square 4, for instance, Harry adds input 4 to all nine of MAYA's wells. MAYA signals its move in re-

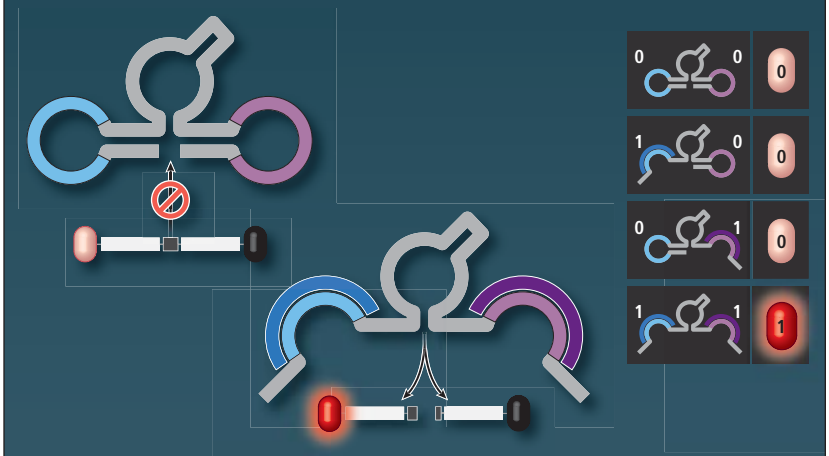
[LOGIC GATES]

HOW DNA COMPUTES

Combining DNA enzymes with stem-loop controllers yields a variety of fundamental logic gates that use short strands of DNA as both inputs and outputs. The cleaving action of the enzyme produces the strands that serve as the gate's output of 1. No cleaving is an output of 0.

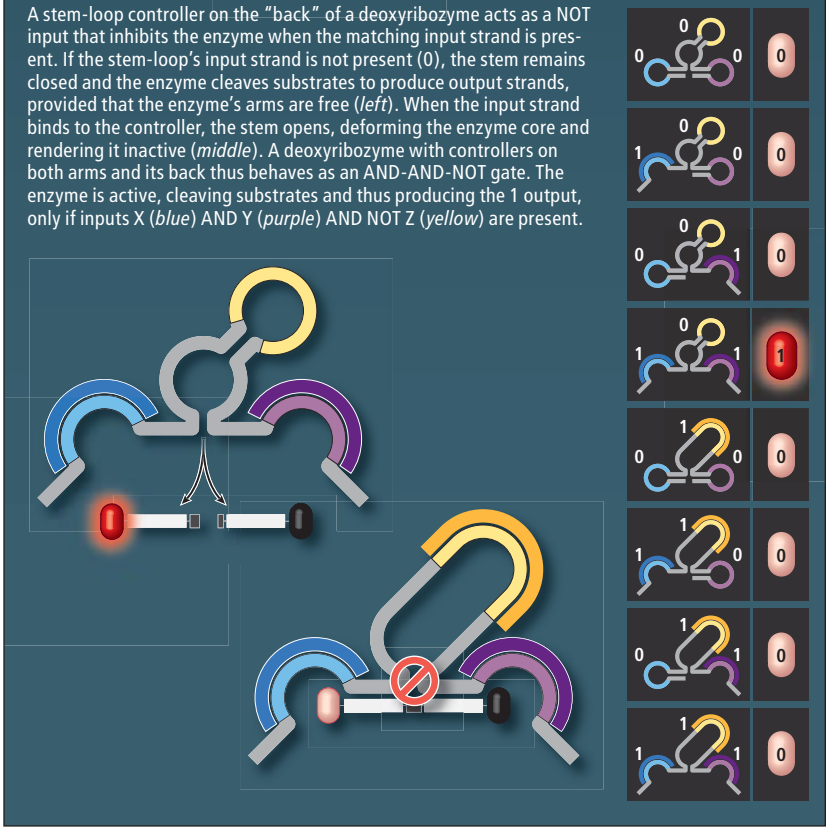
AND GATE

A logical AND gate has two inputs and produces an output of 1 only if both inputs are 1. A deoxyribozyme with a stem-loop on each of its arms acts as an AND gate. The closed stems disable the enzyme (*left*), and only when both loops' matching input strands are added can the enzyme cleave substrates (*middle*). Truth table (*right*) summarizes the gate's function.



AND-AND-NOT GATE

A stem-loop controller on the "back" of a deoxyribozyme acts as a NOT input that inhibits the enzyme when the matching input strand is present. If the stem-loop's input strand is not present (0), the stem remains closed and the enzyme cleaves substrates to produce output strands, provided that the enzyme's arms are free (*left*). When the input strand binds to the controller, the stem opens, deforming the enzyme core and rendering it inactive (*middle*). A deoxyribozyme with controllers on both arms and its back thus behaves as an AND-AND-NOT gate. The enzyme is active, cleaving substrates and thus producing the 1 output, only if inputs X (*blue*) AND Y (*purple*) AND NOT Z (*yellow*) are present.



TOMMY MOORMAN

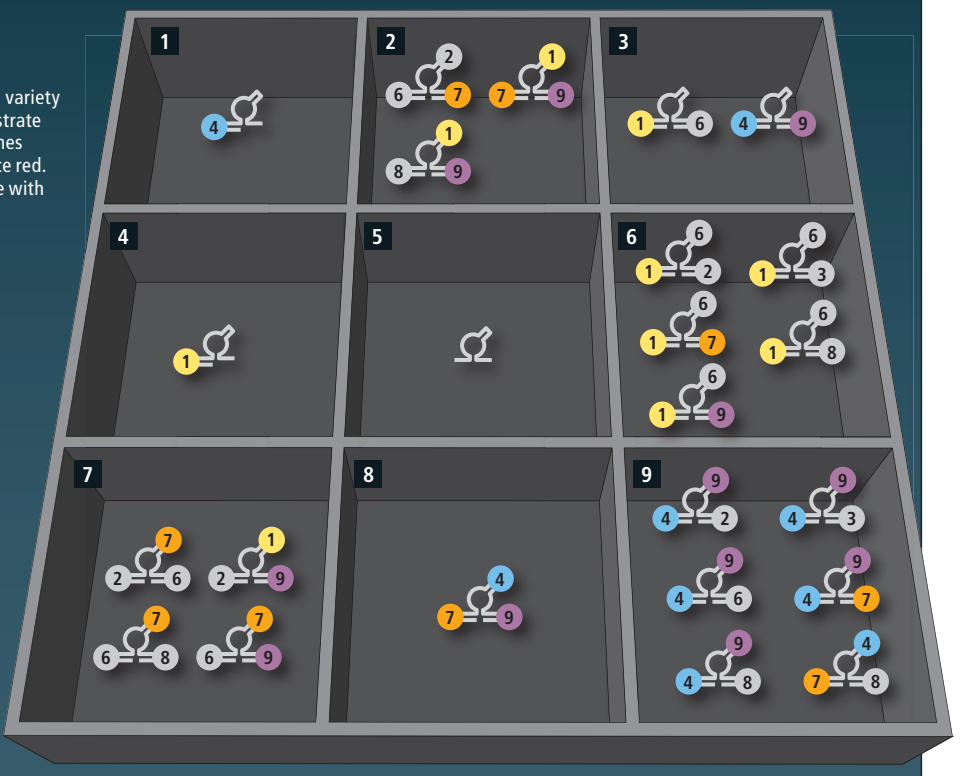
[PROOF OF PRINCIPLE]

PLAYING TIC-TAC-TOE WITH DNA

The first-generation automaton, MAYA-I, proves the potential of DNA logic gates by playing a perfect game of tic-tac-toe, albeit with some restrictions to simplify its programming. MAYA plays first, selecting the central square (*5*), and the human player's first move must be in either the upper left corner (square 1) or the left side (square 4).

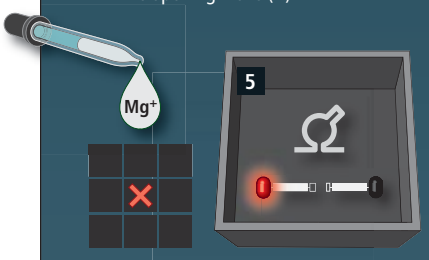
MAYA-I'S STRUCTURE

The computer's 3 × 3 array of wells contains a variety of molecular gates in solution, along with substrate strands (*not shown*). In wells where any enzymes become active, the cleaved substrates fluoresce red. The "gate" in the central well is a DNA enzyme with no stem-loop controllers.

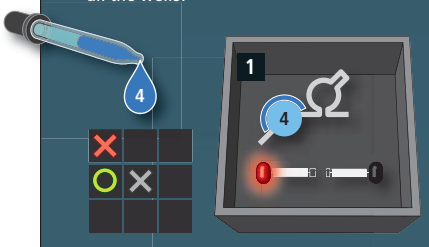


EXAMPLE GAME

The human, "Harry," adds magnesium ions to all nine wells to switch MAYA on. The enzymes in well 5 cleave substrate strands, and the well lights up, signaling MAYA's opening move (X).

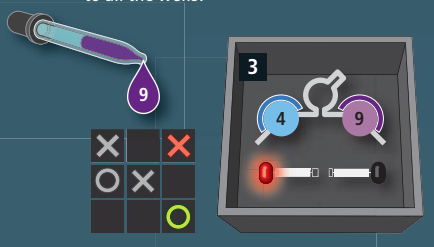


For his first move, Harry takes square 4. He tells MAYA by adding input strand 4 to all the wells.



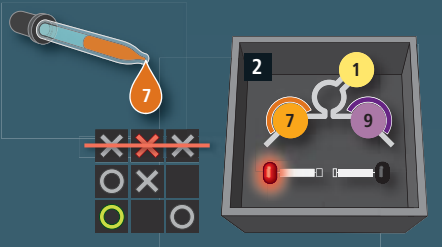
Input strand 4 activates the YES-4 gates in well 1, which lights up; MAYA has taken square 1 for its second move.

To block MAYA from taking the diagonal, Harry takes square 9 by inputting strand 9 to all the wells.



His two inputs activate the 4-AND-9 gates in well 3; MAYA takes that square.

Harry desperately tries blocking MAYA by taking square 7.



Unfortunately for Harry, his inputs now activate the 7-AND-9-AND-NOT-1 gate in well 2 (he has not added strand 1), and MAYA takes that square to win.

sponse by turning on the fluorescence in another of the wells.

As the game progresses, each well contains input strands representing all Harry's moves, and the combination of gates in each well processes those inputs. After every move, one of the wells contains a gate that the last input triggers in combination with the previous inputs. That well lights up to indicate a move by MAYA.

To simplify MAYA's programming, we restricted Harry's first move to be either the upper left corner (square 1) or the left side (square 4). Those two moves are representative of all the moves that Harry might make in response to MAYA's opening move in the center because the board is symmetric. If he moved somewhere else, the board could be rotated to make it a move in either square 1 or 4. With that restriction, the strategy we chose for MAYA allows 19 different possible games to be played. In one of the games,

TOMMY MOORMAN

Harry plays perfectly and the game ends in a draw. In the remaining 18 games, MAYA exploits his mistakes and wins.

To work out all the required gates for the automaton, we considered every move in all 19 games and determined which gates would produce the desired move. The hardest part was matching the strategy requirements with our logic-gate technology. Although our gates are designed to output DNA strands that could in principle serve as inputs to other gates, for MAYA we chose to avoid relying on that feature and the extra complications it might engender. Altogether we took less than three months to design and develop MAYA and fully test all 19 games in the laboratory.

MAYA-II

Not content with MAYA's limitations, we built an unrestricted version, MAYA-II. We also made MAYA-II more user-friendly, displaying both players' moves in two different fluorescent colors. The automaton still goes first and claims the middle square, but Harry the human can then take any of the remaining eight squares. MAYA-II plays four times as many possible games as MAYA, winning 72 of them and drawing four.

We wrote a computer program (for a standard silicon-based computer) to determine an appropriate arrangement of logic gates. The resulting design calls for 128 different logic gates, 96 for deciding and signaling the automaton's moves using red fluorescence and 32 to highlight Harry's moves in green fluorescence.

The sheer size of this automaton made building and testing MAYA-II an enormous challenge. One of us (Macdonald) led the project and trained several high school students to test automata, mostly during summers and on Saturdays. The students checked all 76 games multiple times. They had to make changes in MAYA-II's design to deal with several problems (and then recheck all the games after each tweak).

Our chief concern going into the project was that some sequences might bind in unintended places. Our computer-modeling tools were not advanced enough to be able to predict such difficulties. In fact, spurious binding was relatively rare. Instead the more serious problem turned out to be individual gates cleaving their substrates at different rates. We (or, rather, our students) had to adjust concentrations and structures to correct for this variability. We also quickly discovered that some gates acted differently within a mixture than they did on their own, necessitating other redesigns. Finally, after three consecutive summers and many Saturdays, through some changes of inputs and many small adjustments of gate sequences and concentrations, our team had a system in which we could clearly distinguish active and inactive gates in all wells, for all the games, reproducibly.

Implications

Integrating more than 100 molecular logic components in a single system represented a substantial milestone. In the jargon of electronics, MAYA-II is the first "medium-scale integrated molecular circuit." Our work on a device of such complexity let us refine our deoxyribozyme logic gates as plug-and-play computing primitives. New efforts in our laboratories now proceed more smoothly with existing components, and we can design gates that usually work immediately without needing any fine-tuning.

We could integrate our method with other molecular computing approaches developed recently. For example, Erik Winfree's group at the California Institute of Technology came up with impressive "strand displacement cascades," which could be used to analyze mixtures of oligonucleotides in a similar fashion. In this scheme, strands of DNA combine, joining and displacing one another mostly without the need for any catalysts analogous to the DNA enzymes of our gates. Winfree's system has been demonstrated with a cascade of five units. In comparison, our present system suffers from becoming prohibitively slow if three layers of gates are combined. MAYA-II, for all its complexity, functions as a single layer of gates and takes around 15 minutes to carry out a move.

For our decision-making molecules, we are now very confident about putting many gates together, and tasks representing fresh challenges beckon. We hope one day to report a mixture of molecules that can be taught a strategy by playing example games with them or by introducing some selection to eliminate the gates that encode losing strategies. We might then develop automata that we can train to recognize cancer cells.

But perhaps the most important next step of our program is to incorporate new primitives to carry out more functions, such as sensing and moving (or "actuating"). These are automata that would take action based on the presence of a given input. Our plug-and-play system would then be moving well beyond "play" and would be ready for some real work. ■

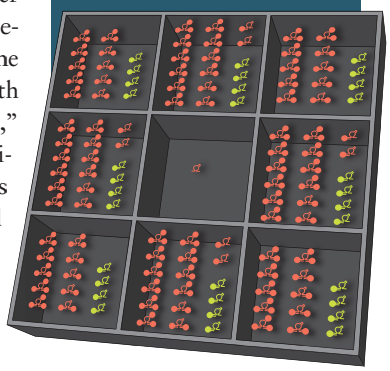
MAYA-II

The second generation of the authors' tic-tac-toe-playing DNA computer, MAYA-II, goes beyond MAYA-I in several respects.

The human player may make any legal move in response to MAYA-II's opening move, increasing the number of possible games to 76. MAYA-II wins 72 of them and draws 4.

32 logic gates cleave green-fluorescing substrates to highlight the human's squares.

96 logic gates compute MAYA-II's moves and indicate them with red fluorescence. A computer program designed the arrangement of gates.



TOMMY MOORMAN

MORE TO EXPLORE

A Deoxyribozyme-Based Molecular Automaton. Milan N. Stojanovic and Darko Stefanovic in *Nature Biotechnology,* Vol. 21, No. 9, pages 1069–1075; September 2003.

Medium Scale Integration of Molecular Logic Gates in an Automaton. Joanne Macdonald et al. in *Nano Letters,* Vol. 6, No. 11, pages 2598–2603; November 2006.

MAYA II, a Second-Generation Tic-Tac-Toe Playing Automaton. Online at **http://tinyurl.com/4mvbnm**

Eric Winfree's home page: **www.dna.caltech.edu/~winfree**