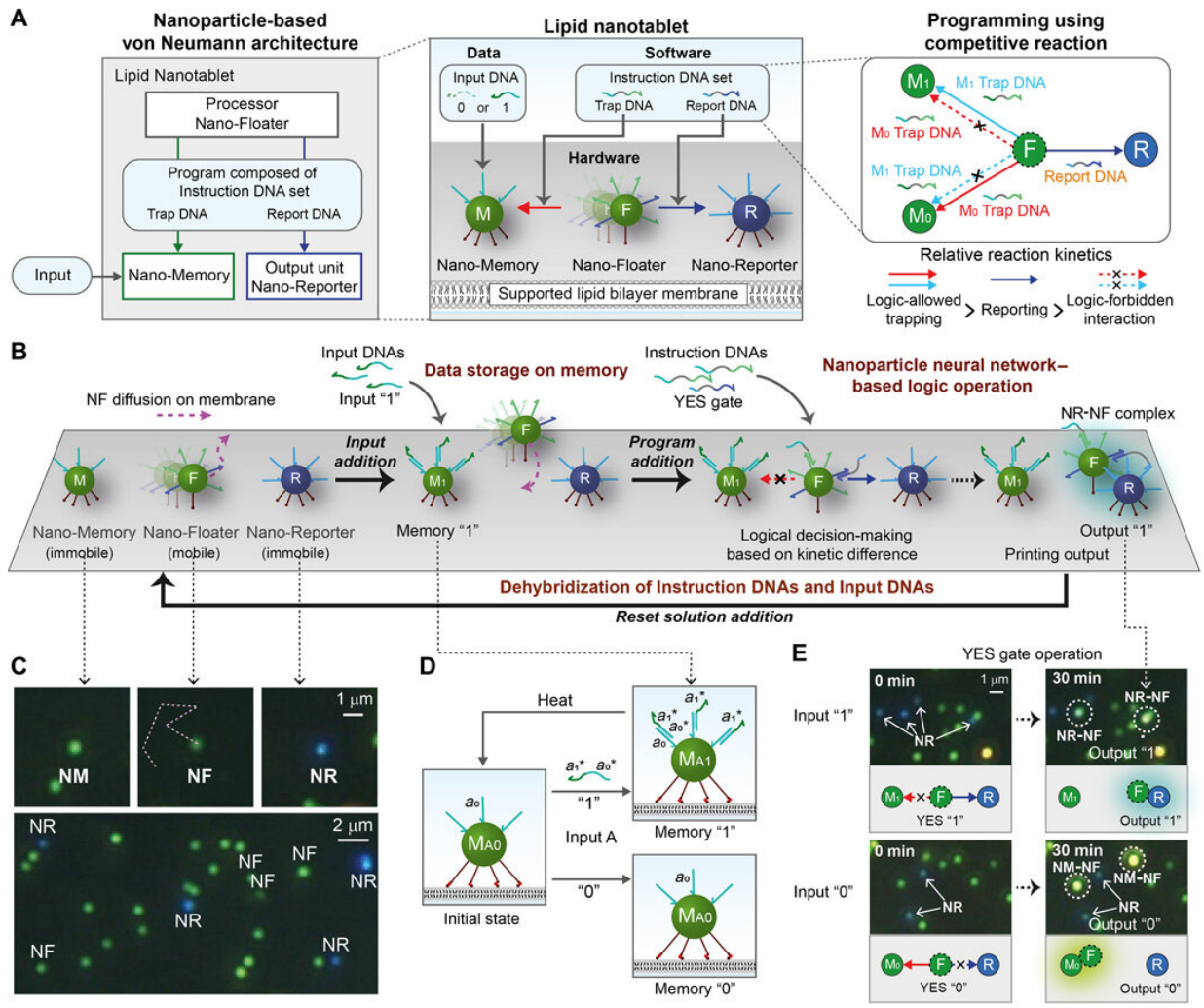
Nanoparticle-based computing architecture for nanoparticle neural networks

September 2 2020, by Thamarasee Jeewandara



The nanoparticle-based von Neumann architecture (NVNA) on a lipid nanotablet (LNT) chip. (A) Schematic of NVNA-LNT. The LNT is operated with software composed of Instruction DNAs in solution and hardware composed of nanoparticles on a lipid bilayer. The hardware consists of a data storage unit,

NM; an output unit, NR; and a processing unit, NF. A set of Instruction DNAs programs logic operation using a kinetic difference between nanoparticle reactions with memory storage state. (B) LNT protocol: (i) data storage on NM, (ii) neural network (NNN) operation by Instruction DNA set addition, and (iii) reset by dehybridizing DNAs for the next executions. (C) Time-lapse dark-field microscopic imaging can differentiate each nanoparticle on LNT via scattering color and mobility. The non-labeled nanoparticles are NM. (D) Molecular information storage on the NM changes the exposed single-stranded domain. (E) YES, gate operation results. Input "1" results in output "1," printing the NF-NR. Otherwise, all NFs are trapped to NM and exhibit no reaction on NR, which is output "0." Credit: Science Advances, doi: 10.1126/sciadv.abb3348

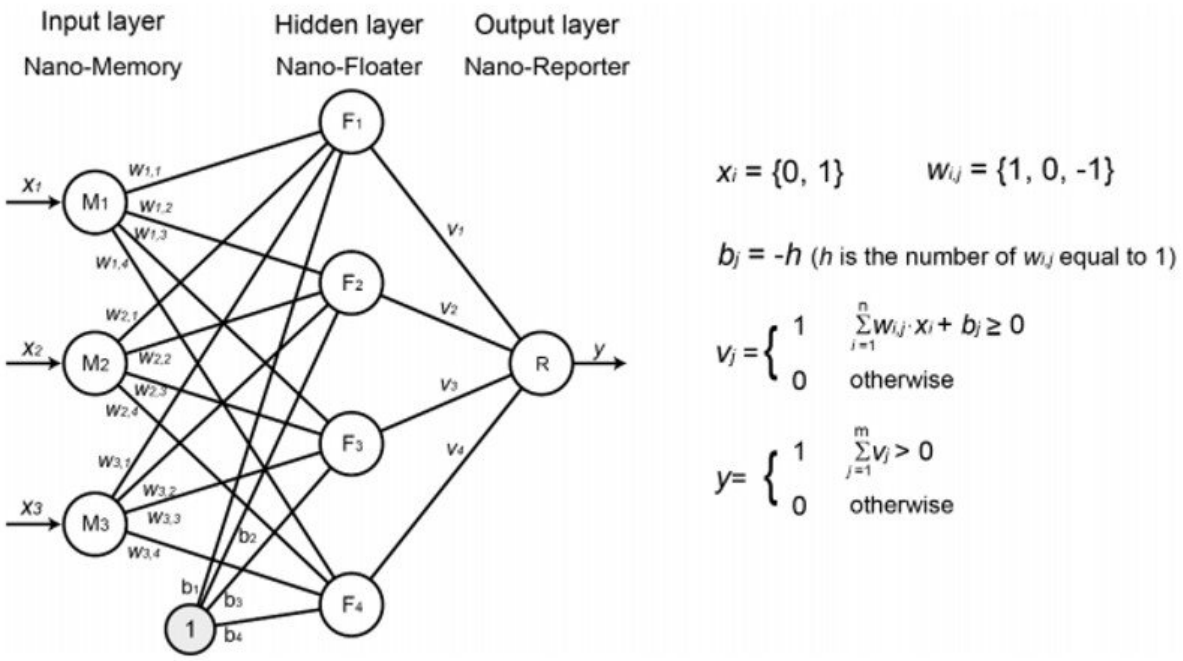
Scalable nanoparticle-based computing architectures have several limitations that can severely compromise the use of nanoparticles to manipulate and process information through molecular computing schemes. The von Neumann architecture (VNA) underlies the operations of multiple arbitrary molecular logic operations in a single chip without rewiring the device. In a new report, Sungi Kim and a team of scientists at the Seoul National University in South Korea developed the nanoparticle-based VNA (NVNA) on a lipid chip. The nanoparticles on the lipid chip functioned as the hardware—featuring memories, processors and output units. The team used DNA strands as the software to provide molecular instructions to program the logic circuits. The nanoparticle-based von Neuman architecture (NVNA) allowed a group of nanoparticles to form a feed-forward neural network known as a perceptron (a type of artificial neural network). The system can implement functionally complete Boolean logical operations to provide a programmable, resettable and scalable computing architecture and circuit board to form nanoparticle neural networks and make logical decisions. The work is now published on *Science Advances*.

The von Neumann architecture in modern computing

and molecular computing

Electronic computers of the past could only run a fixed program and researchers had to physically rewire and restructure processes to reprogram such machines. The von Neumann architecture (VNA) developed by John von Neumann in 1945 and later cited by Alan Turing in his proposal for the automatic computing engine, details a stored-program computer to execute a set of instructions. The system processed information by sequentially fetching the stored data and instructions from the memory to generate outputs. The powerful programmability of the VNA is applicable for modern computers and in quantum computing .

Molecular computing with nanostructures can allow a variety of technologies such as nanoparticle logic gates, single-molecule biosensors and logic sensing, although such systems are limited to a single program much like early electronic computers. The limits arose since researchers incorporated the software (function) and nanostructural hardware as a single unit. To overcome this challenge, they can include lipid bilayers to compartmentalize molecules and nanoparticles. Kim et al. had previously developed a computing platform with nanoparticles on a lipid bilayer to form a nano-bio-computing lipid nanotablet (LNT). In this work, they designed and realized a nanoparticle-based von Neuman architecture (NVNA) platform for molecular computing on a lipid nanotablet (LNT).

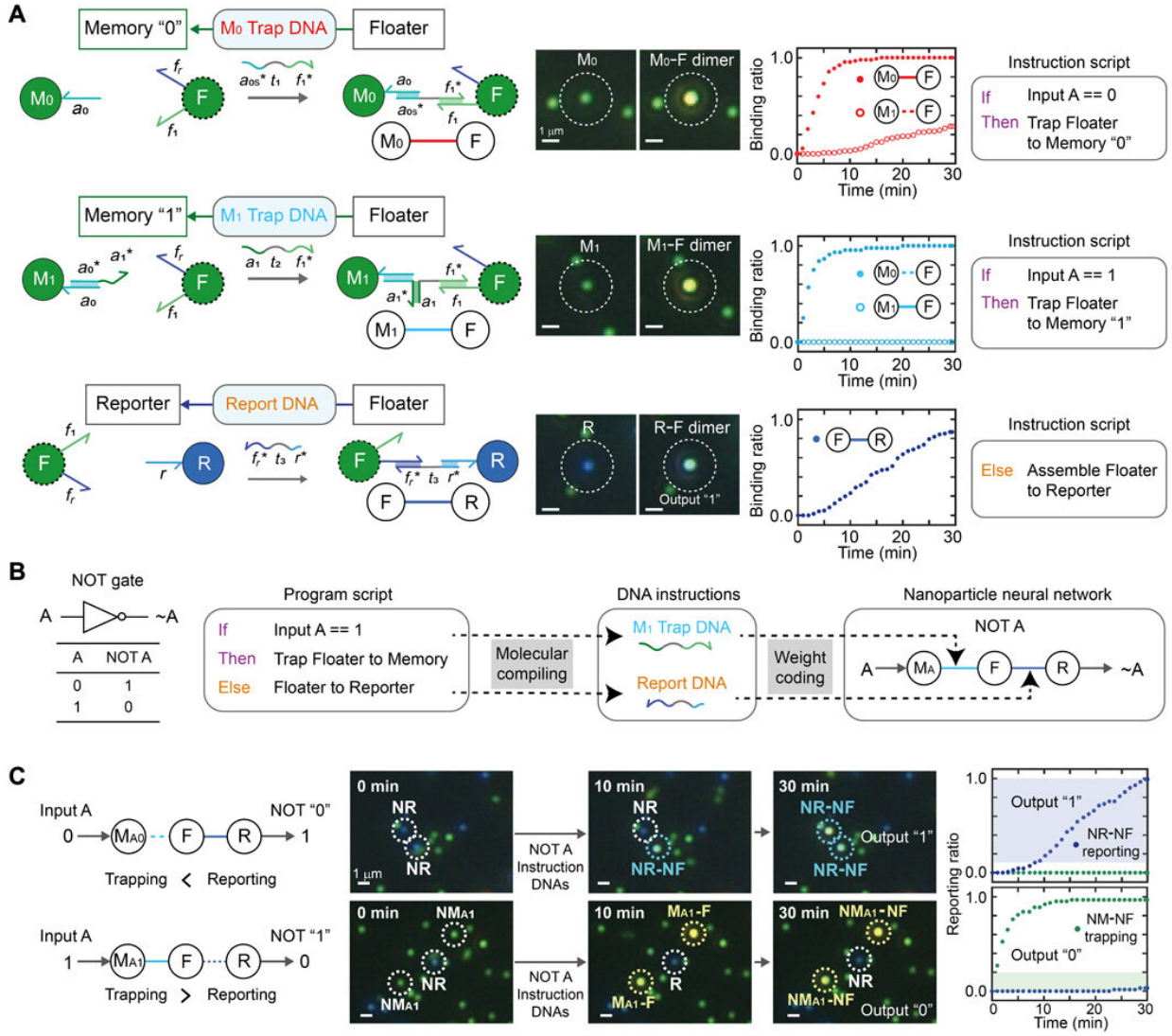


Nanoparticle neural network (NNN) for a functionally complete 3-input system. The system can be represented with a multi-layer perceptron diagram with three layers (input, hidden and output layers), where xi is an input, wi,j and vj are weights, and y is an output. Each layer has three input nodes, four hidden nodes and one output layer, respectively. NF calculates a weighted sum of inputs and a bias and can be activated with an activation function of Heaviside step function. The NM0 and the NM1 Trap DNAs can be represented by discrete weights of 1 and -1, respectively, as the NM0 Trap DNA deactivates the NF at input 0 and the NM1 Trap DNA deactivates the NF at input 1. As they set the threshold for activation function as 0, the bias is required to balance the positive and negative values of the weighted sum of inputs. The bias is defined as the number of NM0 Trap DNA. Activated NFs can bind to NR as output “1”. Credit: Science Advances, doi: 10.1126/sciadv.abb3348

Hardware and software of the nanoparticle-based von Neuman architecture (NVNA)

The team created a stored-program device to implement molecular computing via the von Neumann architecture with nanoparticles, while including the concept of memory to store molecular information. They separated the software and hardware for scalability of information processing in the lipid nanotablet (LNT) to perform multiple computational tasks without developing a new device each time. To compose the LNT hardware chip, they used three types of DNA-modified nanoparticles, including the nano-memory (NM), nano-floater (NF), and nano-reporter (NR). The nano-memory and nano-reporter were immobile nanoparticles that functioned as a molecular information storage device and output unit, respectively. They referred to the mobile nanoparticles as nano-floaters that freely diffused and collided with immobile particles. The scientists functionalized the plasmonic nanoparticles by modifying them with thiolated DNA oligonucleotides. Then for data storage, they loaded different concentrations of NF, NM and NR nanoparticles on to the lipid nano-tablet (LNT). To develop the software, Kim et al. used a set of instruction DNAs in solution, and the logic operation followed three steps.

The team first stored the molecular information on the nano-memory (NM) unit via DNA hybridization. For example, a single NM particle could form a one-bit memory device in which zero or one represented the bistable state. In the second step, they performed the logic operation as a combination of instruction DNAs, to initiate competitive nanoparticle-nanoparticle assembly with different kinetics based on the nano-memory state. To reset the computer chip to its initial state, Kim et al. added a reset solution (low salt buffer and high temperature), which detached the input and instructional DNA base pairings on the chip.

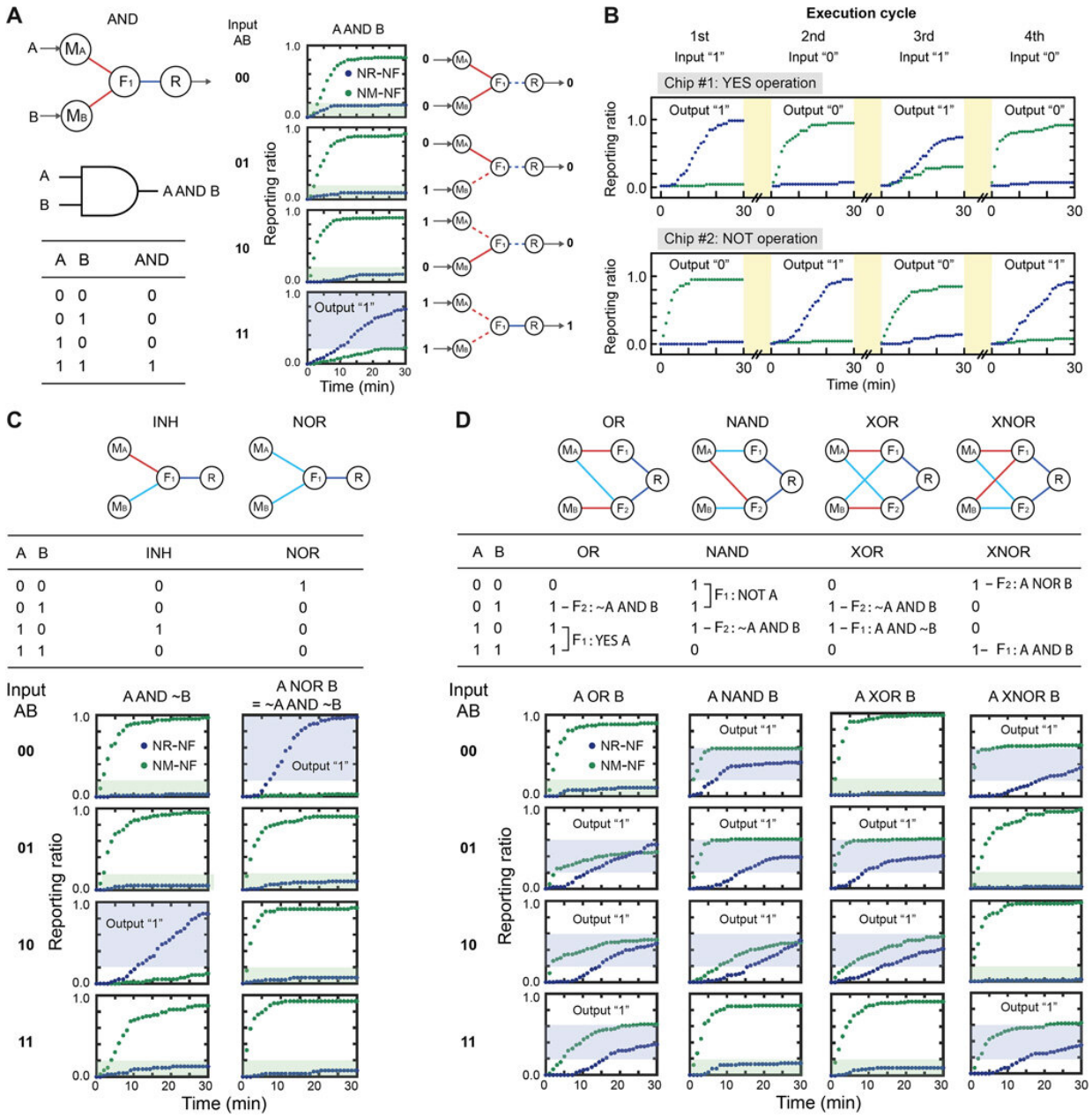


Software programming strategy using Instruction DNAs. (A) Reaction kinetics of three types of Instruction DNAs. The addition of 8 nM NM0 and NM1 Trap DNAs allows fast logic-allowed trapping (solid lines) of NFs to NM with the "0" and "1" states, respectively, and no or slow logic-forbidden binding (dotted lines). The 1 nM Report DNA addition shows binding of NFs to NRs with a lag time. (B) Programming of NOT gate from an If-Then-Else statement to a combination of Instruction DNAs coding the NNN. (C) NOT gate operation in the LNT. For input "0," the NF has no specific interaction with M0 and generates NF—NR assemblies (cyan dotted circle) as the output "1" (reporting ratio > 0.2, green box). For DNA input "1" stored in the NM, the NFs are trapped to the NM1 (yellow dotted circle), resulting in the output "0" (reporting

ratio =

Programming strategy

Kim et al. used two types of instruction DNAs named Trap and Report DNAs to provide instructions for the nano-floaters. They specifically designed Trap DNA to bind the nano-floaters to form logical decision making nanoparticles. The team optimized the concentration of instruction DNAs and the density of each nanoparticle to induce fast trapping kinetics compared with reporting. The competitive trapping and reporting behaviors resulted in binding kinetics expressed as an if-then-else statement, allowing them to first search whether the If condition satisfied TRUE or FALSE operations and then operate the "then" or "else" statement. The scientists implemented the logical operation by mixing trap DNA and Report DNA in the NVNA-LNT chip. During the process, they noted the assembly of a few logically forbidden states, which they further optimized.

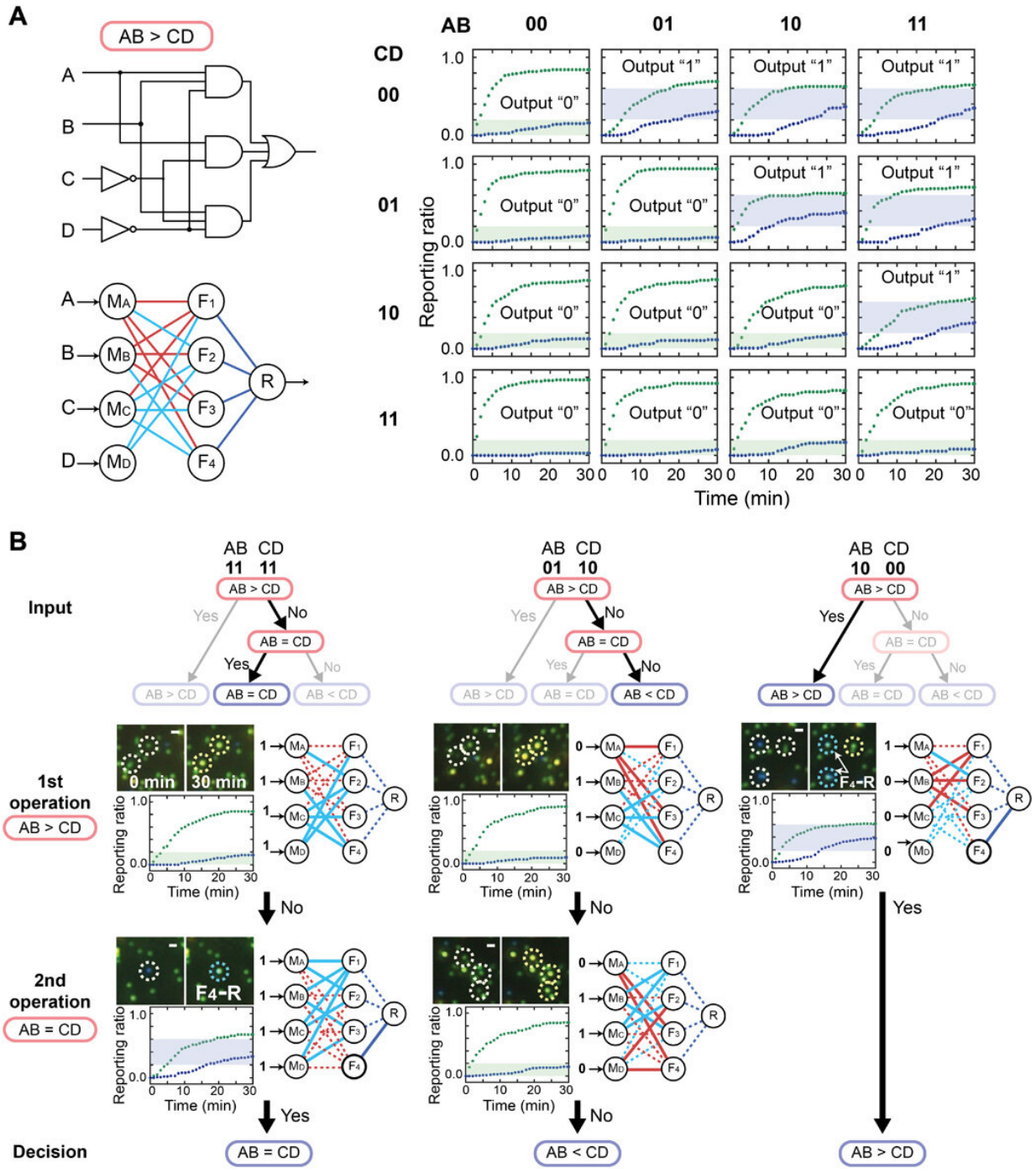


Programming a two-input Boolean logic gate with NNN and demonstration of a reset function. (A) Single-layer perceptron for an AND logic gate. The nanoparticle network at four input combinations is represented with the solid lines indicating the nanoparticle assembly reaction and the dotted lines indicating no or a suppressed reaction. The output "1" (blue box) is represented by NF—NR reporting (blue dots) to NF—NM trapping (green dots) over 0.2 (green box). (B) Multiple executions of logic gates in a single chip by resetting after each execution (yellow box). (C) Execution of INH and NOR logic gates using weight coding. (D) Execution of OR, NAND, XOR, and XNOR logic gates using multilayer perceptron with two types of NF. The output "1" is represented by a

reporting ratio between 0.2 and 0.6 because a single NF between two NFs generates the output "1."Credit: Science Advances, doi: 10.1126/sciadv.abb3348

Nanoparticle neural network with reset and reusability

The team represented the reaction network between multiple nanoparticles connected via instruction DNAs, using a perceptron—a type of artificial neural network for a binary classifier. They expanded the programming strategy to construct the nanoparticle neural network (NNN) on the LNT platform and implemented arbitrary Boolean logic circuits for two-bit inputs. Then they calculated the number of nanoparticle nodes needed to functionally complete Boolean logic operators on the neural network. The hardware relied on covalently modified nanostructures on a lipid chip for multiple executions. They tested the reset function of the system for reusability by dehybridizing all DNA assemblies after exchanging the buffer solution in the setup. The reset allowed thiolated DNAs alone to remain on the nanoparticles, thereby returning to the initial state for the next function.



Execution of a 2-bit comparator with decision tree on a single chip. (A) Digital logic circuit and NNN diagram for AB > CD, and operation result of 16 combinations of two 2-bit input AB and CD. (B) Decision trees for the magnitude comparator. The two-layered tree structure generates three results, indicating the relative magnitude of two 2-bit binary inputs. Four-bit inputs of 1111, 0110, and 1000 result in AB = CD, AB CD respectively. Scale bars, 1 μm. Credit: Science Advances, doi: 10.1126/sciadv.abb3348

The decision-making process and the fan-out logic gate

Kim et al. then explored the system with a sequential decision tree. The decision tree resembled a flowchart to produce a final decision of YES or NO in the nanoparticle neural network. Due to their nanoscale geometric features and optical properties, the plasmonic nanoparticle core of the lipid nanotablet was critical for computing. As the number of nanoparticle nodes and the accompanying complexity of the logic circuit increased, the reaction kinetics remained identical due to parallel reactions of the multilayer perceptron. The team used powerful programmability and the reset function of the setup to sequentially operate the two-bit comparator.

In this way, Sungi Kim and colleagues developed a nanoparticle perceptron with the nanoparticle-based von Neuman architecture (NVNA) on a lipid nanotablet (LNT) chip and explored the system with a sequential decision-making tree. The setup included a reset function for reusability. The nanoparticle-based computing architecture and the nanoparticle neural network (NNN) provided a platform for molecular computing alongside instruction DNAs. The process allowed scalability and paves the way to use nanoparticles in deep learning, neural interfaces and neuromorphic computing to manage and analyze complex biomolecular information. This computing architecture can be embedded in microfluidics to mimic and interrogate complex living systems to develop smart drug screening systems.

More information: Sungi Kim et al. Nanoparticle-based computing architecture for nanoparticle neural networks, *Science Advances* (2020). DOI: 10.1126/sciadv.abb3348

Maxim P. Nikitin et al. Biocomputing based on particle disassembly, *Nature Nanotechnology* (2014). DOI: 10.1038/nnano.2014.156

Kevin M. Cherry et al. Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks, *Nature* (2018). DOI: 10.1038/s41586-018-0289-6