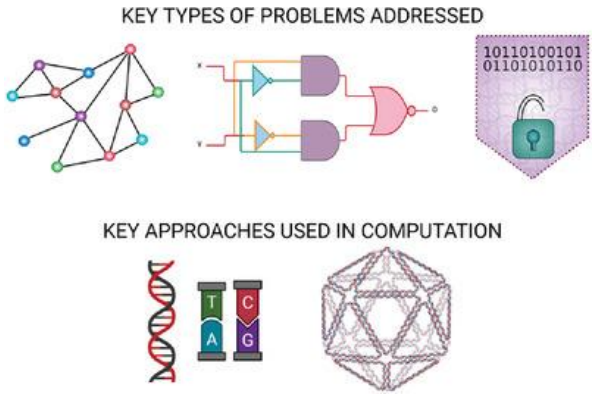
The art of molecular computing: whence and whither

Sahana Gangadharan[1,2] and Karthik Raman[1,2,3]

[1]Bhupat and Jyoti Mehta School of Biosciences, Department of Biotechnology, Indian Institute of Technology Madras, Chennai - 600 036, INDIA
[2]Initiative for Biological Systems Engineering, Indian Institute of Technology Madras
[3]Robert Bosch Centre for Data Science and Artificial Intelligence (RBCDSAI), Indian Institute of Technology Madras; kraman@iitm.ac.in (Corresponding author)

Graphical Abstract



Biomolecules have been exploited as molecular "machines" for a variety of applications ranging from solving mathematical, logical modelling, and network-based problems, to data storage and cryptography. This review outlines the major breakthroughs in this unconventional computing paradigm, and also the future prospects of the field.

ABSTRACT

An astonishingly diverse biomolecular circuitry orchestrates the functioning machinery underlying every living cell. These biomolecules and their circuits have been engineered not only for various industrial applications but also to perform other atypical functions that they were not evolved for—including computation. Various kinds of computational challenges, such as solving NP-complete problems with many variables, logical computation, neural network operations, and cryptography, have all been attempted through this unconventional computing paradigm. In this review, we highlight key experiments across three different 'eras' of molecular computation, beginning with molecular solutions, transitioning to logic circuits and ultimately, more complex molecular networks. We also discuss a variety of applications of molecular computation, from solving NP-hard problems to self-assembled nanostructures for delivering molecules, and provide a glimpse into the exciting potential that molecular computing holds for the future.

Keywords - DNA computing, unconventional computing, logic gates, molecular machines, DNA data storage

INTRODUCTION

Silicon-based computers have been mainstream for the last several decades. They have enabled us to solve many computationally demanding and challenging problems. In a world of conventional paradigms and design architectures, biomolecules offer us an unconventional avenue for solving computational problems. Conceivably, some form of molecular computation underlies cellular decision-making, but can we exploit biomolecular processes to solve computational challenges? Simple manipulations to the existing "operating system" of biomolecules provide us with versatile tools and methods that can be leveraged to solve computational problems. Humans have always strived to exploit living cells for various purposes and make the best use of what's already available in nature. This has led to interesting applications that go well beyond natural biological function.

It is interesting to view the history of molecular computing through the lens of *synthetic biology*. Synthetic biology is a multidisciplinary field that involves assembling biological modules distinctively and predictively, to engineer novel circuits and cells for various applications, simultaneously advancing our understanding of fundamental biology. This, as we can imagine, requires breaking down existing networks to their molecular components and assembling them in a new set-up to yield desirable outcomes. Splicing such molecular components for creating biological modules was first proposed by Salvador Luria *et al.* [1] and later adapted by Jacob and Monod, for their study of the *lac* operon in *E. coli* [2]. This set the stage for other discoveries such as restriction enzymes and strategies to control gene expression, which are the key methodologies used in this field.

The field of synthetic biology has offered fascinating solutions to various problems. It provides a glimpse into the versatility of cellular networks and how they can be adapted for various applications. These applications include formulating "green chemicals'' from agricultural wastes [3], biomimicry [4], rewriting genetic code [5], designing a minimal functioning genome [6], engineering bacteria to replace existing therapeutics [7], and even more interestingly, DNA for data storage [8] and molecular computation [9]. This emerging form of computation involves the use of biological elements to solve computational problems.

Unconventional computing approaches may also require unconventional algorithms—the inherent parallelism of DNA and its error-prone nature of replication may require us to develop radically different algorithmic designs to come up with novel solutions for computationally challenging problems. But what kind of molecular computing systems already exist? Are there other computing paradigms that are amenable for solving such problems? Will they require newer ways of devising algorithms? Can we exploit biological molecules for computation—to recall von Neumann [10, 11] "*is it possible to design reliable systems from unreliable components*"? Do these systems present important advantages beyond their apparently obvious difficulties and disadvantages? In this review, we seek to answer all these questions, presenting a critical account of several important studies over the last few decades.

The following sections provide a brief overview of the history of molecular computation, starting with Leonard Adleman's seminal experiment in 1994 [9]. We delineate the history of molecular computing into three eras: (i) the early history, which pays specific attention to Adleman's classic

work and other 'molecular solutions' to computational problems, followed by (ii) the era of logic circuits, where the emphasis moved to logic circuits and nano-assembly, and finally, (iii) the era of molecular networks, where DNA neural nets, self-assembly-based DNA robots and information processing techniques are emerging as state-of-the-art technologies. Our article will majorly focus on the methodologies and applications developed in DNA computation and briefly touch upon peptide and cellular computation. We cover major breakthroughs in the field of biomolecular computation (see Figure 1) and systematically map out existing research using a generalised morphological analysis (GMA) [12], illuminating important connections between molecular computing approaches, the nature of computationally solved problems, and the molecular techniques used, to ultimately point out potential gaps that future research can address. We discuss the key applications developed in this field, as well as the promise and pitfalls of molecular computation.

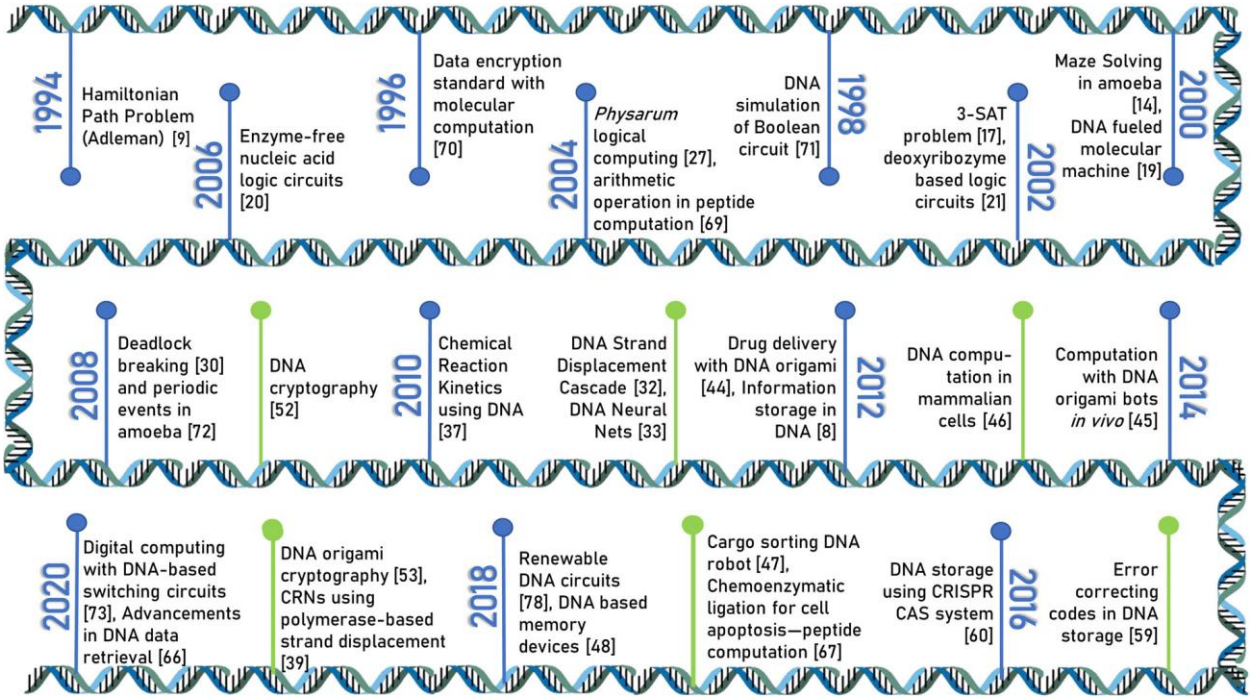


FIGURE 1 Timeline of major breakthroughs in the field of molecular computation. Starting from Adleman's experiment in 1994, the field has come a long way, paving the path for exciting techniques such as DNA Strand Displacement Cascades, DNA self-assembly, nano-robotics with molecules, and neural network computation being the state-of-the-art methodologies today. The numbers in square brackets denote references.

EARLY HISTORY

Although several studies had discussed the potential of biomolecules for computing and self-assembled structures, it was not until Adleman's seminal experiment in 1994 that the field gained due recognition.

Solving the Hamiltonian: Adleman's experiment

In 1994, Leonard Adleman provided a proof-of-concept to solve the Hamiltonian path problem[1] using DNA strands. He used an unusual approach to solve this NP-complete problem[2]. He formulated an algorithm that incorporated DNA strands to represent the input graph and Watson–Crick pairing (see Figure 2) to capture other problem constraints. This landmark experiment laid the foundation for other research and fuelled the growth of this field.

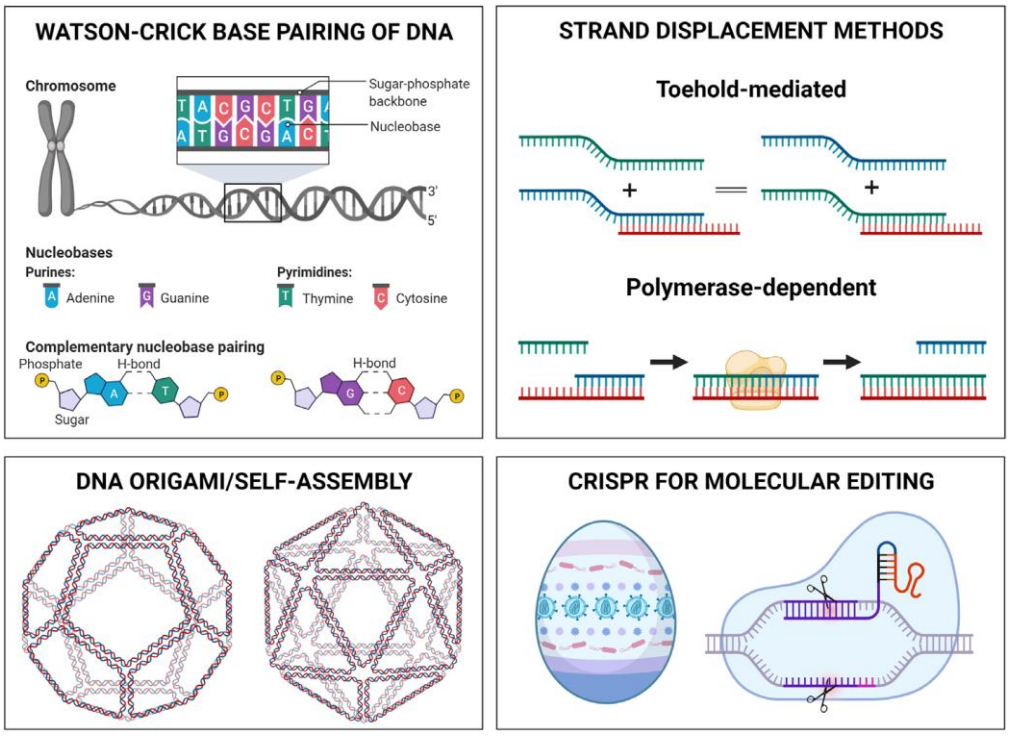


FIGURE 2 Key methodologies employed in the field of DNA computation. This figure demonstrates the key techniques employed in the field of molecular computation. Watson–Crick complementary base pairing is the one rule that binds all the other techniques as well. It states that the nucleotide Adenine (A) always pairs with Thymine (T) via a double hydrogen bond and nucleotide Cytosine (C) always pairs with Guanine (G) via a triple hydrogen bond. The two commonly used strand displacement techniques are Toehold-mediated strand displacement (TMSD) and Polymerase-dependent strand displacement (PSD). The former allows the exchange of one strand of DNA with another by employing the process of branch migration, while the latter employs DNA polymerase to synthesize the strand upon partial binding by another strand. DNA origami/self-assembly is a novel technique in which DNA strands fold themselves to form a 3-dimensional structure that performs specific functions. This technique has been used for several proof-of-concept and therapeutic purposes. CRISPR is the state-of-the-art molecular editing technology in which nucleic acids can be altered in vivo. This methodology has proven very useful for DNA-based in vivo memory devices and for data storage using DNA.

[1]A classic NP-complete computational problem, involving finding a Hamiltonian path in a graph—a path that passes through every vertex in the graph exactly once; it also bears an interesting relation to the travelling salesperson problem.

[2]NP-complete are those problems which do not have a "polynomial time solution", i.e. the time taken to solve the problem typically increases exponentially with problem size. But, if given a solution, it is easy to verify that the solution is correct, in polynomial time.

The algorithm

The algorithm for this problem, as stated by Adleman, is somewhat brute force, yet elegant. The first step is to generate *all* possible paths given a directed network. Next, the paths that begin and end with the specified start and end node are picked. The next step is to identify those paths where all the nodes are visited at least once. Further, it must be ensured that each vertex in the network is visited exactly once. If there remain solutions that satisfy all the above criteria, such paths would be solutions for the Hamiltonian path problem of the given network. This simple and intuitive algorithm can be implemented using a system in which oligonucleotides represent the nodes and the edges of the given network. The experimental strategies are truly novel, making this one of the most exceptional studies.

"DNA implementation" of the algorithm

The implementation of the algorithm was experimentally demonstrated on a 7-node directed graph. The nodes (numbered 0 through 6) were represented by random 20-mer sequences of DNA, and the edges were designed such that they shared sequences with both the vertices that connected them. A ligation reaction, followed by a Polymerase Chain Reaction (PCR) with the 20-mer sequence of node 0 and 20-mer complementary sequence of node 6 as primers, was set up. This generated random paths with the appropriate starting and ending nodes. Once the resulting DNA strands were run on an agarose gel, the portion that corresponded to the desired length of the solution path (140 base pairs), was lysed from the gel. The product from the above step was further affinity-purified to make sure that all the vertices of the network were present in the solution-path exactly once. Finally, a graduated PCR was run on the product to yield a higher concentration of that particular DNA strand—the solution for the Hamiltonian Path Problem of the chosen network.

Of course, merely solving a 7-node Hamiltonian problem was not the major contribution of this work; rather, it heralded a new paradigm for computation, using biomolecules. The massive parallelization involved in generating all paths underlines the power of this approach, given that 10^{14} operations were executed just by the end of the first step. The method was also remarkably energy-efficient, requiring barely 1J of energy to run 2×10^{19} operations.

The dawn of other forms of molecular computation

Beyond DNA, other biomolecules have also been exploited to solve computational problems. The following studies use RNA and a cellular system to solve satisfiability and travelling salesperson problems, respectively.

Molecular computation: RNA solutions to chess problems

Landweber and co-workers [13] proposed an algorithm to solve a modification of the Knights problem, which asks for the configuration of knights on an $n \times n$ chessboard such that no two knights can attack each other, with the help of RNase H Digestion. A 3×3 chessboard, where RNA oligonucleotides represented each position, was employed for the experimental procedure. The key idea was that the RNase would hydrolyse RNA strands that did not fit the constraints of the problem.

The RNA strands were chosen under the basis of certain principles, and the hydrolyzation reaction was done iteratively, according to the logical encoding of the problem. The paper suggests that molecular computation is able to find even the rarest of rare solutions, for instance, even a single strand from over a quadrillion, for a combinatorial problem.

Maze-solving by an amoeboid organism

Physarum polycephalum is a slime mould that has dendritic networks of pseudopodia. It was observed that when this mould was placed at the beginning of a small labyrinth, with nutrient agar at both the start and the end, it always preferred the shortest path to its food source [14]. Instead of spreading out the pseudopodia into different routes, the mould seemed to swirl itself into one thick tube, thereby increasing its foraging efficiency as well as its chance of survival. Thus, it appeared to display the primitive intelligence required to solve an otherwise mathematically hard problem — the Travelling Salesperson Problem (TSP). Here, the cell as a whole indulges in the operation performed thereby introducing cellular computation, rather than biomolecular computation.

THE ERA OF LOGIC CIRCUITS

Following the development of molecular solutions to computational problems, the first decade of the 21st century (2000-2010) saw a number of experiments, mostly rooted in logic gates and arithmetic computations, leading to rapid growth in the field.

Logical Computing with DNA molecules

Adleman's solution to a major combinatorial problem motivated the rest of the scientific community to try out other such problems. One interesting puzzle from that class of problems is the ***3-SAT satisfiability problem***[3] which is NP-complete. Several versions of the problem have been solved through diverse approaches. For instance, Smith and co-workers [15] solved a 3-SAT problem with four variables, analysing $2^4 = 16$ truth assignments, and Sakamoto *et al.* [16] solved a six-variable problem, finding the right solution amidst $2^6 = 64$ possible assignments. Braich *et al.* [17] proposed a method to solve another 3-SAT problem with 20 variables and 24 clauses, scanning through 2^{20} possible truth assignments. Also, to increase the difficulty, the problem was set such that there was only one correct answer in the entire search space.

For each of the 20 variables, a library of DNA strands was synthesized on a solid support, thereby allowing the computation to happen on its surface. These strands were designed to be optimal in that there were no intra- or inter-library strand hybridization. The computation of this work was based on the Stickers Model developed by Roweis *et al.* [18], which uses two basic operations — separation of different types of strands by hybridization, and the application of groups of strands called the memory and sticker strands which together represent a bit string. In this experiment, separation of strands were aided by oligonucleotide probes that were immobilized in an electrophoresis box that

[3] Satisfiability problem is a problem of determining if there exist replacements of variables in a specific Boolean expression such that the formula evaluates to TRUE.

consisted of two chambers — a cold chamber to capture the strands with the correct truth assignments, and a hot chamber that held the remaining. The strands were initially introduced into the hot chamber, and if they did satisfy the set clause, they migrated via electrophoresis into the cold chamber. This step was repeated for all the remaining clauses to identify the one strand representing the factual truth assignments of all the 20 variables. For the most part, the sole "operation" in this experiment was based only on Watson–Crick pairing and melting of DNA.

The implementation of logic gates using DNA was the next breakthrough in this field. The first portrayal of a DNA circuit was proposed by Yurke *et al.* [19]. In this study, they showed two sets of DNA strands that precisely aided one another in nano-scale movements dictated by complementary base-pairing. The first set of strands constituted the circuit, while the second set behaved like a tweezer that opened and closed depending on the complementation of the strands, fuelling the movement of the first set. Nucleic acid-based logic circuits were later introduced by Seelig *et al.* [20] where the designing and modular construction of several logic gates based on miRNA and DNA sequence recognition are discussed, which includes features such as amplification, cascading and feedback loops. Stojanovic *et al.* [21] proposed a ***deoxyribozyme-based modular design*** that was used to generate any Boolean function.

Stojanovic and Stefanovic later improvised on their initial work by devising a molecular automaton [22]. This formulation was designed to play tic-tac-toe with a human opponent, and always guaranteed either a win or a draw. The Molecular Array of YES and ANDANDNOT gates was named MAYA. The human's turn in the play corresponds to the addition of a new DNA strand to which the automaton will initiate a response which could be observed by reading the fluorescence effect from each compartment. MAYA, unlike the methodologies assumed by previously mentioned experiments, works uniquely, by implementing a digital logic circuit using enzymes from metabolic circuits.

Peptide strands for solving mathematically hard problems

Peptide strands are made up of 20 variant building blocks and this confers them advantages over DNA strands for unconventional computation, as the density of information that can be stored in a protein sequence is relatively high. Of course, the kind of tools and techniques required to "operate" on peptide data would differ; the central tool happens to be the specific interaction between epitopes and monoclonal antibodies. There may exist different antibodies for a given epitope and each antibody can possess different binding affinity for the same epitope. This presents an option to detach and reattach different antibodies, which proves useful for computation. It has also been reported that peptide strands composed of unnatural amino acids can also be employed in peptide computation, giving rise to greater variety in peptide strands. Hug and Schuler [23] attempted various combinatorial problems, such as comparing the frequency of a single element in different sets, estimating the number of times an element occurs in a given set, and satisfiability problems. Balan *et al.* [24] provided algorithms for the Hamiltonian path problem and a variant of the set-cover problem. In their work, they also established the algorithm to simulate a Turing machine using a peptide system. Although basic mathematical problems such as addition and subtraction of binary numbers were attempted, other solutions provided by DNA strands were not easily replicable using peptide

computation. This was because there were only a specific set of epitopes–monoclonal antibody pairs that were discovered then, limiting the complexity of problems that could be attempted.

While most of the research in this era focussed on using isolated peptide strands to attempt mathematical problems, Unger and Moult [25] suggested that peptides can be used to construct devices composed of biochemical circuits where inputs can be triggered by biological signals and output might trigger biological processes. As a next step forward, they also discussed the regulations for the implementation of peptide-based logic gates that form the basic unit of digital computers. They described the construction of the NAND gate using peptides and also implemented the model using computer simulations. Ramakrishnan *et al.* [26], described the fundamentals involved in computing with biochemical circuits such as biostability, oscillators and pattern selectivity. These studies also suggested the need for better experimental techniques that would allow us to understand the different states of proteins involved in specific cellular processes.

Logical Computing with cells

Beyond molecules, entire cells have been exploited for computation. Tsuda *et al.* [27] discuss the idea of logical computing in *Physarum* and what happens when there is hardware damage in the system. The logical operations were based on chemotaxis, the aversion phenomenon of *Physarum* against each other and their ability to fuse under extreme situations. AND, OR, and NOT gates were built based on the established guidelines, and expected results were observed. A part of the AND gate was broken to test the emergent behaviour of *Physarum.* In such a scenario, the expected outcome for 1 AND 1 was 0, but on the contrary, the observed results were quite the opposite. One *Physarum* fragment, instead of averting itself from the other fragment, chose to fuse with it and they moved towards the right output path together, leading to 1AND1 = 1. Such inherent behaviour of the *Physarum* leading to self-repair upon damage shows us the possibility of using living organisms to solve logical problems. Jones and Adamatzky [28] computationally designed ***half-adder and full-adder circuits*** using *Physarum,* thereby demonstrating the organism's ability to adapt to any kind of environment and emphasising its robustness.

An interesting application of *Physarum* [29] was a ***bio-robot*** whose only source of instructions came from the physical environment. The external cues were coupled with the intracellular information processing ability of *Physarum* to yield an omni-directional walker that could solve optimisation problems.

Physarum was also observed to display properties that enabled it to flexibly ***break deadlock-like situations*** [30]. There is no higher-level authoritative controller in these systems, thereby making its sub-components behave independently while still maintaining favourable conditions for the whole system. In a sense, *Physarum* implements a recurrent neural network algorithm which enables the amoeba to solve a constrained satisfaction problem. The amoeba is captured in a star-like structure so that any movement in the amoeba makes it traverse in multiple branches. These branches are photosensitive and operate concurrently as processors of the system. An optical feedback control is introduced to create conflicts within the processors, thereby allowing them to function as a recurrent neural network for imposing a particular constraint satisfaction problem.

THE ERA OF MOLECULAR NETWORKS

Digital circuits and Origami with DNA strands

This era began with many striking advances by Erik Winfree and Lulu Qian. They introduce this technique where, metaphorically, ***DNA strands play on a See-Saw***. Two single strands of DNA that have a complementary sequence to a template strand anneal and melt from the template strand in a see-saw fashion, thereby making a wide range of applications possible. This mechanism, called the *'toehold-mediated DNA strand displacement'* (TMSD), was originally proposed by Yurke *et al.* [19], and later popularised by Qian and Winfree. This design was used to construct other circuits such as feedforward digital circuits, relay contact circuits and analogue time-domain circuits [31].

Further, they extended the idea to develop logic gates. A specific arrangement of OR and AND gates also permitted the construction of a 3-bit XOR gate and the computation of square root of a 4-bit number. As the complexity of the problem at hand increased, they set up a debugging tool which was another output 'wire', from an intermediate step, through which the functioning of the entire circuit could be analysed. It is easy to envisage the construction of more complex circuitry from these standard building blocks, which underlie all electronic circuits. This way, they furthered the notion of digital computation using molecular circuits [32].

Next, they built a ***neural network using DNA sequences*** [33], where they assembled a sophisticated circuit, involving 72 DNA species, which resembled a Hopfield network. A Hopfield network [34], is made up of artificial neurons that are fully connected and possess the ability to remember a set of patterns from their training set. The functionality of this molecular network was tested in a cuvette via a game called "read your mind". In this game, a human would select and keep one of four given scientist's names in his/her mind and would answer a set of questions, by giving inputs to the associative memory network. Based on the answers to the predefined questions, the network was able to correctly guess the scientist, sometimes even before the opponent gave out all the available hints by answering the questions. The authors showed that the network can handle robust inputs and any other defect within the system, including sparse connectivity of the network. The 'Winner-take-all' algorithm by Cherry and Qian [35] employed the above algorithm to the MNIST problem[4] using DNA strands that worked on the basis of their ability to remember patterns and classify complex and noisy data. The results of their work have paved the path for biomolecular systems that can learn, memorise and recall.

However, the existing algorithms for implementing DNA logic circuits were time-consuming and involved DNA complexes comprising multiple DNA strands. Song *et al.* [36] proposed a novel architecture of single-stranded DNA logic gates that utilised a strand-displacing DNA polymerase which aided in the computation. They constructed and tested multiple single-stranded logic circuits and scaled them up to form multi-layered linear cascades. They showed that this system is dynamic and also demonstrated shorter response time. Polymerase-based strand displacement (PSD) and TMSD methods are similar in design as shown in Figure 2; yet differ in their primary functioning

[4]MNIST is a large database of handwritten digits, and a gold standard dataset for testing image processing systems.

mechanism. The use of enzyme in PSD method offers an additional source of energy and reduced error caused due to leakage or overlapping signals. Overall, PSD-based logic gates were shown to perform complex computation in a much shorter timespan[5].

Toe-hold mediated strand displacement cascades and DNA polymerase-based displacement cascades can also be employed in several other aspects. The intrinsic analogue nature of biomolecules enables a much broader class of behaviours. In particular, ***Chemical Reaction Networks*** (CRNs), introduced by Soloveichik *et al.* [37] operate as a 'programming language' that facilitates the study of the underlying kinetics in unimolecular and bimolecular reactions, while also discussing limitations and challenges. CRNs can be thought of as a general framework for modelling networks with interacting species. Nano-controllers were implemented by Chen *et al.* [38] using plasmid DNA to test and study fundamental reaction types, eventually leading to *de novo* engineering of interactions between the designed components. Reif and co-workers proposed an alternative approach for implementing arbitrary CRNs solely based on strand-displacing polymerase enzymes [39]. The group designed CRNs for unimolecular and bimolecular reactions and further scaled them to capture large scale network-based reactions such as autocatalytic amplification in molecular biology, molecular-scale consensus network in contemporary economics, and finally a dynamic oscillator to implement the rock–paper–scissors game [40]. The concentration and the length of the primer strands were indicated as the tunable parameters for the stoichiometry and rate of reactions. These principles were used to implement catalytic and non-catalytic reactions. Further, they highlighted and discussed a few challenges such as improperly annealed gates, the requirement of excess biomolecules that might not be suitable for extending into *in vivo* systems, and reaction leaks at higher temperatures, which needed to be addressed before CRNs can be readily exploited for further applications.

In early 1982, Seeman [41] explained the different types of junctions and lattices that can be formed using nucleic acids. Seeman elucidates the basic rules that must be followed to generate covalently bond lattices that are periodic in connectivity and plausibly in space too. DNA-based tile-assembly is a versatile toolbox that provides directions for nanoarchitecture and directed self-assembly of DNA strands [42]. It facilitates structures with increased complexity and behaves as a scaffold that aids in the assembly. Tile-based work also supports robust reprogrammable computations [43] to execute different algorithms, thereby welcoming molecular engineering to the algorithmic era. Jiang *et al.* [44] described an unconventional method for drug delivery, where they used ***DNA origami-based nanostructures*** that could self-assemble (see Figure 2) and more importantly, were biocompatible and spatially addressable. Amidst all the exciting *in vitro* and *in silico* research, Amir *et al.* [45] fabricated a DNA nanobot that could control a specific molecule *in vivo*, inside a living cockroach. Another such circuit that responded to specific miRNAs in the cellular space was constructed, to enable the monitoring and imaging of cell-specific markers [46].

A DNA-based nanobot was designed for sorting two types of biomolecules into separate piles [47]. The bot would perform a random walk across a two-dimensional DNA origami surface, thereby saving energy, picking up and dropping biomolecules as and when it encountered them. DNA is

[5] https://www.microsoft.com/en-us/research/blog/researchers-use-a-strand-displacing-dna-polymerase-to-do-biocomputing/, accessed on 25 April, 2021

biocompatible and can thus be used to record specific intracellular events that might be very significant in diagnostics and treatment [48]. Therefore, DNA can be used as a unique Random Access Memory (RAM)-like memory device to keep track of intracellular events. The general overview of this application is rather straightforward: intracellular molecular events are characterised by specific signals and these signals activate a corresponding genetic circuit for recording, interfaced with the biosensors. Based on the specific event, DNA strands are targeted and modified using a DNA writer. The modification or the writing onto DNA can happen at a single directed location or at multiple locations in either a targeted manner or by accepting stochastic changes in the DNA sequence, mainly by employing the CRISPR/Cas9 system (see Figure 2). The altered DNA is then sequenced and analysed to identify the type and rate of changes that happened in the intracellular environment, to infer specific details regarding the cellular processes.

Similar technologies employing nucleic acids for constructing isothermal assemblies, triggers and switches are explained by Li *et al.* [49]. Mamet *et al.* [50] showcased the solution for the "Monty Hall problem" with DNA strands. The authors used Illumina Next Generation Sequencing to identify the strand that corresponds to the correct answer over multiple simulations. Experimentation and retrieval of results are usually time-consuming; yet, this task was done within 24 hours, including the preparatory work.

DNA Cryptography

Storing information in DNA also inspired the idea of storing and retrieving cryptic messages inside the biomolecule, giving rise to ***DNA Cryptography***. This idea was first proposed in 1995 [51], and since then, many approaches have been tried and tested. A 'cryptographic' experiment was attempted by Ning [52], in which he encoded data in a DNA sequence and allowed it to undergo the steps of the Central Dogma, with introns being spliced in the process. To transfer the information as a secret message, the receiver is given the peptide sequence through a public channel, the additional information about the regions that underwent splicing and a cryptic message regarding the introns is shared through a secure channel. This way, even if the peptide sequence is obtained by an intruder, they will not have the complete information required to decode the original message. Another recent DNA cryptographic experiment involves self-assembling origami structures [53]. The secret message is first encoded as a braille-like pattern in the outer layer. This is followed by the formation of a steganographic intermediate layer. The receiver, upon receiving these scaffolds, uses specific staple sequences to fold the origami, revealing the original patterns under an atomic force microscope. The authors suggested DNA origami cryptography as a biomolecular solution that meets the modern demands of high information security.

Exploring the field through competitions

Various synbio competitions have served to further increase the excitement around molecular computation. The ***International Genetically Engineered Machine*** (*iGEM: http://www.igem.org/)* competition [54], is a worldwide synthetic biology competition which promotes a systematic study of biology and a platform to develop new ideas and test them with support and guidance from the experts. Another initiative is the ***BIOMOD competition*** *(http://biomod.net/)*, led by Shawn Douglas,

where numerous students participate in designing devices with biomolecules for various applications. This competition has brought forth some of the most exquisite works in the field of molecular nanotechnology and computation. The ***Molecular Programming Project*** *(MPP; http://molecular-programming.org/)* is an initiative by 11 scientists from various departments at Caltech, Harvard, University of Washington and University of California, San Francisco. This initiative aims to incorporate computer science and programming principles into molecular systems, thereby creating nanoscale devices that have potential applications in robotics and the biomedical industry. Further information on these competitions and initiatives are provided in Table 1. They have served to accelerate growth in these fields, also creating many opportunities for the future. Most importantly, they have enthused and attracted many young researchers to dabble in this area.

DNA for Data Storage

An unconventional application of DNA is in ***information storage***. DNA arranges itself in a very compact manner, and its miniature size supports extremely dense information storage, with an estimated density of 5.5 Pb/mm^3 [8]. In comparison, a microSDXC card has a density that is approximately a million times lower[6]. Although storing data in DNA and retrieval are tedious, DNA can store large amounts of data for a long time. This technique was first implemented in 1988, by J. Davis [55], when he successfully cloned a picture of *Microvenus,* encoded in DNA *bases*, onto *E. coli*. Since then, many attempts have been made at storing different kinds of data in DNA. In 2012, George Church and his team encoded 70 copies of an HTML version of his book, *Regenesis* [56], as DNA sequences [8]. Following this work, a team led by Nick Goldman encoded all 154 sonnets by William Shakespeare, an audio clip from Martin Luther King's famous speech, *"I have a dream"*, the classic paper on the structure of DNA by James Watson and Francis Crick, a picture of the researcher's institute, followed by the instructions on how the data can be converted into a single DNA sequence [57]. The properties of DNA as a storage device and its efficiency over other contemporary storage media are discussed by Zhirnov *et al.* [58]. Studies have also looked at error-correcting codes [59] and *in vivo* storage using CRISPR Cas systems [60]. An excellent review of DNA data storage has been published elsewhere [61].

Although there are obvious advantages of this technique, it is important to note the limitations as well. The chemical synthesis of DNA is still costly[7] and the mechanism for identification and the retrieval of specific messages is still tricky [62]. However, in the recent past, there has been significant research resulting in the development of appropriate methodologies and tools for accessing and retrieving data from DNA. Yazdi *et al.* [63] proposed PCR-based random access of data while also offering methods for re-writability. Random access over large data was first suggested by Karin Strauss and co-workers [64]. Further, Microsoft demonstrated a fully

[6] The maximum storage of the latest microSDXC cards is 1TB (=8000 Gb); and the dimensions of the card are 32 x 24 x 2.1-1.4 mm^3. Therefore, the information density of this device is - 8000/ (32*24*2.1) - 8000/ (32*24*1.4) = 4.96 - 7.44 Gb/mm^3, which is ~5–7 Gb/mm^3. Source: https://www.theverge.com/2019/5/15/18626729/sandisks-1tb-microsd-card-available-b-h-photo-amazon-price, accessed 8th July 2020

[7] https://www.technologyreview.com/2017/05/22/68387/microsoft-has-a-plan-to-add-dna-data-storage-to-its-cloud/, accessed on 24th September 2020

automated process for encoding and retrieving messages [65]. The reliable retrieval of messages [66] when very few copies of the message are present in the DNA pool, was shown by yet another team from Microsoft. These advances indicate the extensive potential of DNA as "wetware" with its resilience as an added advantage.

Table 1: Recent progress in the field of molecular computation, fuelled by competitions and initiatives.

Competition / Initiative	Team/ Project	URL	Description
iGEM	Team AHUT_China	http://2016.igem.org/Team:AHUT_China	Bio-navigational system that enables a faster solution for the shortest past, given any network
iGEM	Team Thessaloniki	https://2019.igem.org/Team:Thessaloniki/Description	DNA computer to recognise specific DNA-protein interactions
iGEM	Team SEU (Southeast University)	https://2019.igem.org/Team:SEU	Neural network computation using DNA strands that perform ReLu and Sigmoid functions. Open-source software tools that design DNA reactions.
BIOMOD	Team USYD	https://usyd-biomod-2019.webflow.io/project	Cure for Cardiovascular diseases using DNA nanostructures that selectively capture LDL.
BIOMOD	Team TU Berlin	http://biomod.biocat.tu-berlin.de/	Multibrane, a biological membrane with enzyme-rich flagella, that degrades pollutants and heavy metals in water.
MPP	NUPACK	http://www.nupack.org/	Package for analysis and design of nucleic acid structures.
MPP	gro	http://depts.washington.edu/soslab/gro/	Cell programming language that facilitates modelling the behaviour of cells in communities
MPP	caDNAno	https://cadnano.org/	Aids in CAD design of DNA nanostructures
MPP	Single-stranded RNA nanostructure	https://science.sciencemag.org/content/345/6198/799	RNA origami folding where lattices are made by annealing and/or co-transcriptional folding.

Table 1 In this table, we highlight a few studies in the field of biomolecular computation, along with the details of the participating team, and a brief description of their project.

Peptide computation

While DNA was being modified to construct molecular networks, it was only in this era that peptides were employed for designing logic gates and circuits. Unger and Moult [25], as discussed previously, modelled the working of peptide-based logic circuits using computer simulations. Molecular devices with enzymes as processors were adapted to perform logical computing with peptide sequences as inputs into biochemical circuits within cells. In this demonstration by Li *et al.* [67], mammalian cell apoptosis was programmed by a series of peptide-based logic circuits which was interfaced with another regulatory circuit that sensed the expression of secreted biomarkers and triggered apoptosis in cancer cells. Logic operations such as AND (where the presence of both input molecules was essential to initiate the reaction), OR (where the presence of one of the input peptides would suffice to initiate apoptosis) and INHIBIT (where the presence of only one input would initiate cell death) were performed using suitable peptide sequences. Following this, a 3-bit peptide keypad based on concatenated logic gates was designed. This device only works when appropriate sequences are input, especially in the correct order. When this was tested with normal and HeLa cells, apoptosis was observed only in the latter even when the right set of inputs was given to the former. This experiment seems to be extremely beneficial for applications such as targeted cell apoptosis and cell differentiation. Ho *et al.* [68] engineered proteins to construct Boolean circuits with AND and NAND logic gates. The inputs to these gates were the presence or absence of certain molecules; according to the gate's functionality, genes were either activated or suppressed, thereby controlling the cell's functions. However, breaking of coding genes at random places could damage the cell's functions. Hence, transposons (movable genes) were incorporated along with split inteins, which acted as protein glues to maintain the cell's functions.

KEY APPLICATIONS DEVELOPED

The range and scope of molecular computation have expanded extensively, and have come a long way since Adleman's first experiment. We here present a GMA of the literature, capturing important connections between molecular computing approaches across the eras, the nature of computational problems that have been solved, as well as the molecular techniques used [12]. In Table 2, we have listed the functionalities and the approaches taken in biomolecular computing, along the rows, and the different types of problems encountered, along the columns. A cross-consistency matrix is built by placing all the papers referred to in this article, in the appropriate boxes. This matrix systematically maps out literature enabling the identification of unique connections that are otherwise imperceivable to this extent, through conventional methods. The gaps in the matrix suggest either a lack of research in the area, or an error in the literature survey. However, to the best of our knowledge, we have not left out any major contributions in the field. For instance, we speculate that there have not been works employing self-assembly to solve RNA-mediated chemical reactions kinetics. The GMA is thus particularly useful to identify less-researched areas that offer potential research questions to work upon.

Starting from NP-Complete and Combinatorial problems such as the 3-SAT satisfiability problem and TSP, DNA computation has aided in solving diverse sets of problems, including that of logical and neural network computation. By the beginning of the molecular networks era, a DNA computing

device could perform the basic functions of a contemporary silicon-based system. Complex designs such as the implementation of neural networks using DNA sequences, nanobots that could sort different kinds of molecules, DNA architecture that could track and image molecules, both *in vitro* and *ex vivo,* were demonstrated. Renewable circuits that allow the use of the same device for multiple computations, Illumina sequencing that facilitate parallel processing and of course the use of DNA as a medium for storing data and cryptic messages contributed to further advances in the field. The implementations of these problems were predominantly based on Watson–Crick complementary base-pairing, annealing features, deoxyribozyme-based methods, and DNA strand displacement cascades. Advancement in the field of cellular computing paved the pathway for intriguing features such as emergent behaviour, modular design and also the adaptation of chemotaxis for information processing. The growth of peptide computing was also remarkable for its aspects of providing solutions for arithmetic problems, logical computation and finally, the use of peptides as processors [69].

Advantages and Disadvantages

Molecular computing has several advantages. Firstly, the ability to perform a massive number of reactions concurrently makes molecular systems efficient, allowing the device to perform an exhaustive search within hours of running a PCR reaction. Secondly, their miniature architecture also facilitates their application in tracking molecules inside certain living organisms, benefiting the biomedical community. Finally, the energy efficiency of these systems demonstrates their resourcefulness, outweighing any other current-day computers. Despite the continued validity of Moore's law, and the concomitantly increasing computational power, the last decade has seen an intense focus on power-efficient computing, focusing less on mere teraflops and petaflops, but on per-Watt performance. Molecular computing, despite its inherent difficulties, outshines traditional computing in energy efficiency.

However, it is necessary to analyse the various drawbacks of these systems as well. First, the functioning of these systems involves many steps, starting from encoding the data carefully, performing the prerequisites, to finally sequencing the biomolecules to read out the solution, resulting in a very tedious process. What can be presently solved on a desktop computer in microseconds might require a few hours, or even days, with molecular computing devices. Second, the inherent noise in biological systems makes them unpredictable in certain scenarios. The field is still at its inception, and issues like reproducibility and reliability remain to be addressed.

Table 2: An overview of the variety of molecular computation approaches that have been used to tackle various problems.

		Types of Problem						
	Approach/ Functionality	**Combinatorial**	**Decision making**	**Mathematical**	**Chemical Kinetics**	**Logical Modelling**	**Memory based**	**Information Processing and Cryptography**
DNA Computation	Recursive	9, 15, 16, 17	52					
	High-throughput DNA Sequencing		52					54, 69, 70
	RNA mediated		13			48		54
	Toehold mediated strand displacement			33, 36	39, 40	32, 36, 47, 77-79	34	
	Polymerase-based strand displacement				41, 42	37, 38		
	Self-assembly/ Origami					44-49		55
	Logical Operations		13, 22	33, 36		19, 21, 22, 32, 36, 74, 78, 79	34	74
	Cellular Recording						50	
	Biosensing		13	33	40, 80	80	50	55, 63
	Renewable devices					81, 82		67
	Storage device		62					8, 53-55, 60-63,67-70
Peptide Computation	Differential-affinity based	23	23	73	73	26		
	Recursive	24		24				
	Biosensing	23	23	73	73	25, 26		
	Logic Operations		71			72		71
	Cell Regulation		71			72		71
Cellular Computation	Spatiotemporal oscillations/ shuttle streaming	14	14, 30			29	76	29, 76
	Optical Feedback control		30					
	Logic Operations		27, 30	28		27, 28, 30		
	Light-sensing					29		29

We have conducted a GMA and defined various categories for the methodologies employed and the functionalities achieved in the field of molecular computation. The rows denote various approaches or functionalities employed by the molecular "machines" which are further divided into three categories: DNA computation, Peptide computation and Cellular computation. The columns represent the different kinds of problems that have been attempted using biomolecules. Each cell contains references to studies exploring the corresponding intersectional field and the blanks reveal areas that have been hitherto under-explored, to the best of our knowledge, and are potential key "gap" areas for future exploration and research.

WHITHER?

Having discussed the various facets of molecular computation, it is vital to analyse where the field is headed. Applications that employ DNA self-assembly in therapeutics and in the molecular manufacturing sector will certainly benefit the most out of this field. In fact, tackling medical problems with molecular computation has already been taken up by competitions such as iGEM and BIOMOD, (see Table 1). We might also expect applications such as CAD designing for DNA, which the MPP is currently attempting, and automated systems that perform operations based on certain rules. These enhancements will reduce human intervention in the procedures, thereby facilitating easier design and mitigating errors, if any. Automated systems are also immensely helpful in regulating design principles and reducing the noise in the system. With substantial investment from technology giants such as Microsoft, DNA data storage seems to be on an accelerated growth trajectory, with automated systems performing the assigned methodologies.

CONCLUSION

The potential of molecular computation is immense. Employing biomolecules for computation has given rise to various interesting applications from simple arithmetic to logical operations and, finally, to neural networks in which the biomolecular device has the ability to memorise and recall events. Although biomolecular devices are not as effective as silicon-based systems for basic operations, they have proven to be extremely advantageous for specific use cases and applications. Adleman and others, as we have seen, have beautifully adapted biomolecules and their interactions, advancing the art and theory of biomolecular computation.

To summarize, this field isn't just another molecular biology experiment. Many potential applications await, making the future exciting and inspiring for the community to witness and learn. Additionally, it is important to appreciate that a remarkable variety of complex tasks are carried out by the "cells" in every living being, including sophisticated information processing, and complex decision-making to sustain life and evolve. Any work that taps into these naturally available systems would ultimately enhance our ability to understand, exploit and manipulate these building blocks, providing us with a versatile toolbox for solving a variety of challenges confronting humanity.

ACKNOWLEDGEMENTS

The authors thank Hariharan Srikrishnan, Sankalpa Venkatraghavan and Sathvik Ananthakrishnan for critical reading of the manuscript and constructive suggestions. We apologise to colleagues whose work we could not cite due to space constraints.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

REFERENCES

1. Luria, SE., Human, ML. (1952). A nonhereditary, host-induced variation of bacterial viruses. J. Bacteriol., 64, 557.
2. Jacob, F., Perrin, D., Sánchez, C., Monod, J. (2005). L'opéron : groupe de gènes à expression coordonnée par un opérateur [C. R. Acad. Sci. Paris 250 (1960) 1727–1729]. C. R. Biol., 328, 514–20.
3. Tai, Y-S., Xiong, M., Jambunathan, P., Wang, J., Wang, J., Stapleton, C., Zhang, K. (2016). Engineering nonphosphorylative metabolism to generate lignocellulose-derived products. Nat. Chem. Biol., 12, 247–53.
4. Raman, R., Bashir, R. (2017). Biomimicry, Biofabrication, and Biohybrid Systems: The Emergence and Evolution of Biological Design. Adv. Healthc. Mater., 6, 1700496.
5. Mukai, T., Lajoie, MJ., Englert, M., Söll, D. (2017). Rewriting the Genetic Code. Annu. Rev. Microbiol., 71, 557–77.
6. Forster, AC., Church, GM. (2006). Towards synthesis of a minimal cell. Mol. Syst. Biol., 2, 45.
7. Pedrolli, DB., Ribeiro, NV., Squizato, PN., de Jesus, VN., Cozetto, DA., Tuma, RB., … Cerri, MO. (2019). Engineering Microbial Living Therapeutics: The Synthetic Biology Toolbox. Trends Biotechnol., 37, 100–15.
8. Church, GM., Gao, Y., Kosuri, S. (2012). Next-Generation Digital Information Storage in DNA. Science, 337, 1628–1628.
9. Adleman, LM. (1994). Molecular computation of solutions to combinatorial problems. Science, 266, 1021–4.
10. von Neumann, J. . (1956). Probabilistic Logics and the Synthesis of Reliable Organisms From Unreliable Components. In Shannon CE,McCarthy J. ed; Automata Studies. (AM-34). Princeton University Press. 43–98.
11. McCulloch, WS. . (1960). The reliability of biological systems. In Self Organizing Systems. Pergamon. 264–79.
12. Ritchey, T. (2018). General morphological analysis as a basic scientific modelling method. Technol. Forecast. Soc. Change, 126, 81–91.
13. Faulhammer, D., Cukras, AR., Lipton, RJ., Landweber, LR. (2000). Molecular computation: RNA solutions to chess problems. Proc. Natl. Acad. Sci. U. S. A., 97, 1385–9.
14. Nakagaki, T., Yamada, H., Tóth, Á. (2000). Maze-solving by an amoeboid organism. Nature, 407, 470–470.
15. Liu, Q., Wang, L., Frutos, AG., Condon, AE., Corn, RM., Smith, LM. (2000). DNA computing on

surfaces. Nature, 403, 175–9.
16. Sakamoto, K. (2000). Molecular Computation by DNA Hairpin Formation. Science, 288, 1223–6.
17. Braich, RS., Chelyapov, N., Johnson, C., Rothemund, PWK., Adleman, L. (2002). Solution of a 20-Variable 3-SAT Problem on a DNA Computer. Science, 296, 499–502.
18. Roweis, S., Winfree, E., Burgoyne, R., Chelyapov, NV., Goodman, MF., Rothemund, PW., Adleman, LM. (1998). A sticker-based model for DNA computation. J. Comput. Biol. J. Comput. Mol. Cell Biol., 5, 615–29.
19. Yurke, B., Turberfield, AJ., Mills, AP., Simmel, FC., Neumann, JL. (2000). A DNA-fuelled molecular machine made of DNA. Nature, 406, 605–8.
20. Seelig, G., Soloveichik, D., Zhang, DY., Winfree, E. (2006). Enzyme-Free Nucleic Acid Logic Circuits. Science, 314, 1585–8.
21. Stojanovic, MN., Mitchell, TE., Stefanovic, D. (2002). Deoxyribozyme-Based Logic Gates. J. Am. Chem. Soc., 124, 3555–61.
22. Stojanovic, MN., Stefanovic, D. (2003). A deoxyribozyme-based molecular automaton. Nat. Biotechnol., 21, 1069–74.
23. Hug, H., Schuler, R. (2001). Strategies for the development of a peptide computer | Bioinformatics | Oxford Academic. Bioinformatics, 17, 364–8.
24. Balan, MS., Krithivasan, K., Sivasubramanyam, Y. . (2002). Peptide Computing - Universality and Complexity. In Jonoska N,Seeman NC. ed; DNA Computing. Springer Berlin Heidelberg. 290–9.
25. Unger, R., Moult, J. (2006). Towards computing with proteins. Proteins, 63, 53–64.
26. Ramakrishnan, N., Bhalla, US., Tyson, JJ. (2009). Computing with Proteins. Computer, 42, 47–56.
27. Tsuda, S., Aono, M., Gunji, Y-P. (2004). Robust and emergent Physarum logical-computing. Biosystems, 73, 45–55.
28. Jones, J., Adamatzky, A. (2010). Towards Physarum binary adders. Biosystems, 101, 51–8.
29. Tsuda, S., Zauner, K-P., Gunji, Y-P. (2007). Robot control with biological cells. Biosystems, 87, 215–23.
30. Aono, M., Hara, M. (2008). Spontaneous deadlock breaking on amoeba-based neurocomputer. Biosystems, 91, 83–93.
31. Qian, L., Winfree, E. (2011). A simple DNA gate motif for synthesizing large-scale circuits. J. R. Soc. Interface, 8, 1281–97.
32. Qian, L., Winfree, E. (2011). Scaling Up Digital Circuit Computation with DNA Strand Displacement Cascades. Science, 332, 1196–201.
33. Qian, L., Winfree, E., Bruck, J. (2011). Neural network computation with DNA strand displacement cascades. Nature, 475, 368–72.
34. Little, WA. (1974). The existence of persistent states in the brain. Math. Biosci., 19, 101–20.
35. Cherry, KM., Qian, L. (2018). Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. Nature, 559, 370–6.
36. Song, T., Eshra, A., Shah, S., Bui, H., Fu, D., Yang, M., … Reif, J. (2019). Fast and compact DNA logic circuits based on single-stranded gates using strand-displacing polymerase. Nat. Nanotechnol., 14, 1075–81.
37. Soloveichik, D., Seelig, G., Winfree, E. (2010). DNA as a universal substrate for chemical kinetics. Proc. Natl. Acad. Sci., 107, 5393–8.

38. Chen, Y-J., Dalchau, N., Srinivas, N., Phillips, A., Cardelli, L., Soloveichik, D., Seelig, G. (2013). Programmable chemical controllers made from DNA. Nat. Nanotechnol., 8, 755–62.
39. Shah, S., Song, T., Song, X., Yang, M., Reif, J. . (2019). Implementing Arbitrary CRNs Using Strand Displacing Polymerase. In Thachuk C,Liu Y. ed; DNA Computing and Molecular Programming. Springer International Publishing. 21–36.
40. Shah, S., Wee, J., Song, T., Ceze, L., Strauss, K., Chen, Y-J., Reif, J. (2020). Using Strand Displacing Polymerase To Program Chemical Reaction Networks. J. Am. Chem. Soc., 142, 9587–93.
41. Seeman, NC. (1982). Nucleic acid junctions and lattices. J. Theor. Biol., 99, 237–47.
42. Lin, C., Liu, Y., Rinker, S., Yan, H. (2006). DNA Tile Based Self-Assembly: Building Complex Nanoarchitectures. ChemPhysChem, 7, 1641–7.
43. Woods, D., Doty, D., Myhrvold, C., Hui, J., Zhou, F., Yin, P., Winfree, E. (2019). Diverse and robust molecular algorithms using reprogrammable DNA self-assembly. Nature, 567, 366–72.
44. Jiang, Q., Song, C., Nangreave, J., Liu, X., Lin, L., Qiu, D., … Ding, B. (2012). DNA Origami as a Carrier for Circumvention of Drug Resistance. J. Am. Chem. Soc., 134, 13396–403.
45. Amir, Y., Ben-Ishay, E., Levner, D., Ittah, S., Abu-Horowitz, A., Bachelet, I. (2014). Universal computing by DNA origami robots in a living animal. Nat. Nanotechnol., 9, 353–7.
46. Hemphill, J., Deiters, A. (2013). DNA Computation in Mammalian Cells: MicroRNA Logic Operations. J. Am. Chem. Soc., 135, 10512–8.
47. Thubagere, AJ., Li, W., Johnson, RF., Chen, Z., Doroudi, S., Lee, YL., … Qian, L. (2017). A cargo-sorting DNA robot | Science. Science,
48. Sheth, RU., Wang, HH. (2018). DNA-based memory devices for recording cellular events. Nat. Rev. Genet., 19, 718–32.
49. Li, J., Green, AA., Yan, H., Fan, C. (2017). Engineering nucleic acid structures for programmable molecular circuitry and intracellular biocomputation. Nat. Chem., 9, 1056–67.
50. Mamet, N., Harari, G., Zamir, A., Bachelet, I. (2019). Simulating the Monty Hall problem in a DNA sequencing machine. Comput. Biol. Chem., 83, 107122.
51. Beaver, D. (1995). Computing with DNA. J. Comput. Biol., 2, 1–7.
52. Ning, K. (2012). A Pseudo DNA Cryptography Method. Comput. Electr. Eng., 38, 1240–8.
53. Zhang, Y., Wang, F., Chao, J., Xie, M., Liu, H., Pan, M., … Fan, C. (2019). DNA origami cryptography for secure communication. Nat. Commun., 10, 5469.
54. Kelwick, R., Bowater, L., Yeoman, KH., Bowater, RP. (2015). Promoting microbiology education through the iGEM synthetic biology competition. FEMS Microbiol. Lett., 362
55. Davis, J. (1996). Microvenus. Art J., 55, 70–4.
56. Church, GM., Regis, E. . 2014. *Regenesis: How Synthetic Biology Will Reinvent Nature and Ourselves*. Hachette UK.
57. Goldman, N., Bertone, P., Chen, S., Dessimoz, C., LeProust, EM., Sipos, B., Birney, E. (2013). Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. Nature, 494, 77–80.
58. Zhirnov, V., Zadegan, RM., Sandhu, GS., Church, GM., Hughes, WL. (2016). Nucleic acid memory. Nat. Mater., 15, 366–70.
59. Grass, RN., Heckel, R., Puddu, M., Paunescu, D., Stark, WJ. (2015). Robust Chemical Preservation of Digital Information on DNA in Silica with Error-Correcting Codes. Angew. Chem. Int. Ed., 54, 2552–5.

60. Shipman, SL., Nivala, J., Macklis, JD., Church, GM. (2016). Molecular recordings by directed CRISPR spacer acquisition. Science, 353
61. Ceze, L., Nivala, J., Strauss, K. (2019). Molecular digital data storage using DNA. Nat. Rev. Genet., 20, 456–66.
62. De Silva, PY., Ganegoda, GU. (2016). New Trends of Digital Data Storage in DNA. BioMed Res. Int., 2016, 1–14.
63. Tabatabaei Yazdi, SMH., Yuan, Y., Ma, J., Zhao, H., Milenkovic, O. (2015). A Rewritable, Random-Access DNA-Based Storage System. Sci. Rep., 5, 14138.
64. Organick, L., Ang, SD., Chen, Y-J., Lopez, R., Yekhanin, S., Makarychev, K., … Strauss, K. (2018). Random access in large-scale DNA data storage. Nat. Biotechnol., 36, 242–8.
65. Takahashi, CN., Nguyen, BH., Strauss, K., Ceze, L. (2019). Demonstration of End-to-End Automation of DNA Data Storage. Sci. Rep., 9, 4998.
66. Organick, L., Chen, Y-J., Dumas Ang, S., Lopez, R., Liu, X., Strauss, K., Ceze, L. (2020). Probing the physical limits of reliable DNA data retrieval. Nat. Commun., 11, 616.
67. Li, Y., Sun, S., Fan, L., Hu, S., Huang, Y., Zhang, K., … Yao, S. (2017). Peptide Logic Circuits Based on Chemoenzymatic Ligation for Programmable Cell Apoptosis. Angew. Chem. Int. Ed., 56, 14888–92.
68. Ho, TYH., Shao, A., Lu, Z., Savilahti, H., Menolascina, F., Wang, L., … Wang, B. (2021). A systematic approach to inserting split inteins for Boolean logic gate engineering and basal activity reduction. Nat. Commun., 12, 2200.
69. Balan, MS., Krithivasan, K. (2004). Parallel computation of simple arithmetic using peptide–antibody interactions. Biosystems, 76, 303–7.
70. Adleman, LM., Rothemund, PWK., Roweis, S., Winfree, E. (1999). On Applying Molecular Computation to the Data Encryption Standard. J. Comput. Biol., 6, 53–63.
71. Amos, M., Dunne, P. (1998). DNA Simulation of Boolean Circuits. Proc. 3rd Annu. Genet. Program. Conf.,
72. Saigusa, T., Tero, A., Nakagaki, T., Kuramoto, Y. (2008). Amoebae Anticipate Periodic Events. Phys. Rev. Lett., 100, 018101.
73. Wang, F., Lv, H., Li, Q., Li, J., Zhang, X., Shi, J., … Fan, C. (2020). Implementing digital computing with DNA-based switching circuits. Nat. Commun., 11, 121.
74. Song, T., Garg, S., Mokhtar, R., Bui, H., Reif, J. (2016). Analog Computation by DNA Strand Displacement Circuits. ACS Synth. Biol., 5, 898–912.
75. Tang, W., Zhong, W., Tan, Y., Wang, GA., Li, F., Liu, Y. (2020). DNA Strand Displacement Reaction: A Powerful Tool for Discriminating Single Nucleotide Variants. Top. Curr. Chem., 378, 10.
76. Song, T., Shah, S., Bui, H., Garg, S., Eshra, A., Fu, D., … Reif, J. (2019). Programming DNA-Based Biomolecular Reaction Networks on Cancer Cell Membranes. J. Am. Chem. Soc., 141, 16539–43.
77. Eshra, A., Shah, S., Song, T., Reif, J. (2019). Renewable DNA Hairpin-Based Logic Circuits. IEEE Trans. Nanotechnol., 18, 252–9.
78. Garg, S., Shah, S., Bui, H., Song, T., Mokhtar, R., Reif, J. (2018). Renewable Time-Responsive DNA Circuits. Small, 14, 1801470.