Living computers powered by biochemistry

F. Veronica Greco, Matthew J. Tarnowski and Thomas E. Gorochowski
(University of Bristol, UK)

Electronic computers have revolutionized virtually all aspects of our lives. However, long before these existed, cells have relied on computations implemented using biochemistry to make decisions about how to improve their chance of survival. The emerging field of synthetic biology offers a new perspective on life, attempting to apply engineering principles to modify and repurpose biological systems or even create new ones from scratch. This is opening up exciting opportunities to reprogram cellular functions, enabling us to better understand how biological computations are implemented, as well as providing a window into the inner workings of the living computers that surround us.

Cells compute

The world is a dynamic and ever-changing place. For cells to survive, they must continually sense the environment, process this information and then make decisions about how to respond. These steps are controlled by biological computations (or 'biocomputations' as we will refer to them) implemented using biochemistry whose outputs carefully coordinate changes in cell physiology. Due to the huge diversity of biochemical components and processes available during evolution, biocomputations in nature often work differently from those implemented using the fast and reliable electronic circuits we are more used to. If we are to appreciate, better understand and potentially engineer these living computers, it is crucial for us to scrutinize how information is encoded within cells and the ways that it is sensed, converted and transformed by biological circuitry.

Figure 1. Harnessing the computational power of living cells offers new ways to tackle challenges in many areas such as manufacturing, medicine and agriculture.

Why reprogram living cells?

An ability to create new biocomputations that reliably control the behaviour of living cells forms the basis of many exciting biotechnological applications (see Figure 1). In biomanufacturing, the introduction of non-native metabolic pathways to sustainably produce new chemicals from renewable feedstocks often imparts a significant burden on a cell, leading to stress responses that reduce overall yield. By incorporating cellular computations that implement feedback control mechanisms, stress can be continually monitored and the expression of components in the synthetic pathway reduced if a sufficiently high level is reached. This not only helps improve productivity but creates a more robust system able to dynamically tune its behaviour in response to additional unknown demands that might arise in the future, e.g. the need to produce multiple chemicals simultaneously.

The ability to reprogram cells also offers innovative solutions to challenging problems in healthcare. For example, it was recently shown that cancer cells express a unique pattern of small RNAs (sRNAs) that differ from those found in healthy cells. Using this knowledge, bioengineers have been able to create synthetic regulatory circuits implemented using RNA interference (RNAi) that can sense key sRNAs, compute if the expression pattern matches that of a cancerous cell, and if so, trigger the production of a deadly apoptosis-promoting protein to kill it. By selectively targeting cancerous cells this biological circuit holds promise as a highly specific cancer therapy with low off-target effects. In addition, it can be easily reprogrammed to combat other forms of cancer with other unique sRNA signatures.

Unlike with electronic computers, reprogramming living cells also offers access to the vast capabilities of biological systems. For example, cells can synthesize a diversity of chemicals, self-replicate and even exploit

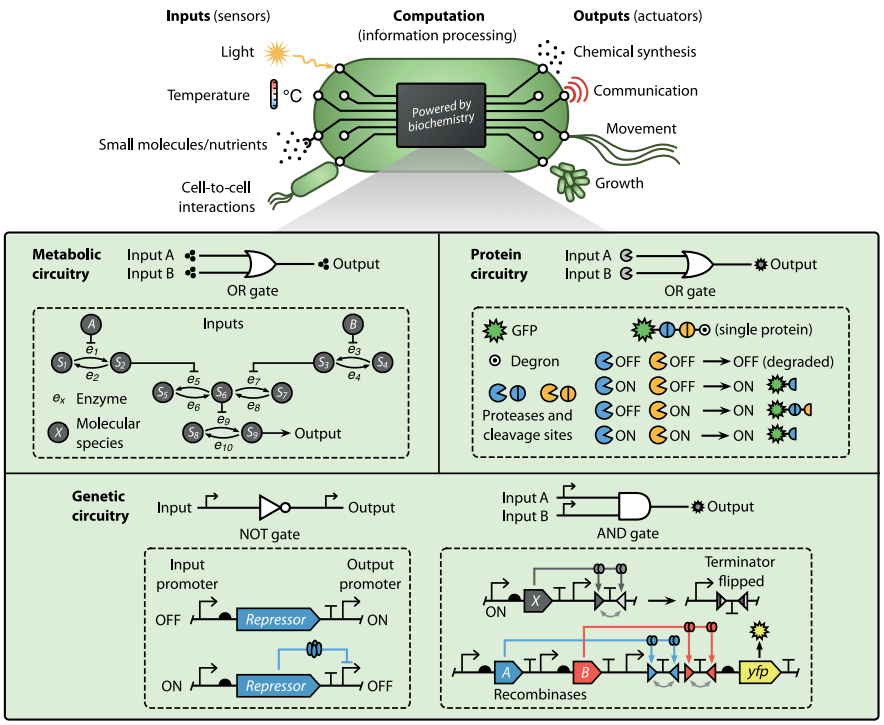


Figure 2. Overview of some typical inputs and outputs to biological computations and the wide array of biochemical machinery that can be used to process this information in living cells. The metabolic circuitry example shows the implementation of an OR logic function using molecular concentrations as inputs and output (see Arkin and Ross (1994) Biophy. J. **67**, 560–578). The protein circuitry example shows an OR logic function implemented using the CHOMP system (see Gao et al. (2018) Science **361**, 1252–1258). Degrons are short amino acid sequences that if included in a protein signal its rapid degradation. The genetic circuitry example shows two logic functions (NOT and AND) using repressor DNA-binding proteins and recombinase enzymes (see Brophy and Voigt (2014) Nat. Methods **11**, 508–520). All logic gates show the different input and output signals and genetic designs are drawn using Synthetic Biology Open Language Visual notation (see Cox et al. (2018) J. Integr. Bioinform. 20170074).

nanoscale self-assembly; features not generally accessible to an electronic computer. Thus, being able to reprogram cells offers far more than just an alternative way to perform mathematical calculations, it enables diverse aspects of biology to be closely integrated with the computational processes themselves. However, in contrast to electronic systems where we have clear rules for how components can be connected together to produce new functionalities, biological systems are generally less well understood and more difficult to control, resulting in a significant amount of time and effort to develop a desired biocomputation.

Computing with biochemistry

To be able to effectively reprogram living cells, it is important to understand that all computations have a common structure: information gathered from inputs is fed in, transformed and processed to produce a single or multiple outputs (see Figure 2). Crucially, information in these systems must be encoded as signals that are compatible with the components carrying out the computational operations. In electronics, signals are normally implemented as a flow of electrons (an electrical current) with wires used to guide signals between the various components of a circuit. Even though sensors might capture information about other types of signal (e.g. temperature, pH, the key pressed on a keyboard, etc.) these must first be converted into an electrical current before computations using this information can occur.

Common signals in biocomputations include the transcription and translation rates of genes, the concentrations of molecules, protein phosphorylation states and the structural orientation of DNA segments, to name but a few. Sensors also exist in cells to convert external stimuli into common cellular signals that can then be processed. Two-component systems found throughout biology are used to sense a wide range of external conditions. They consist of a membrane-bound sensor (typically a histidine kinase), which when active initiates the phosphorylation of a response regulator to control changes in gene expression. This allows a shared transcriptional regulatory network to be used to process information from many different sensors which then controls a single response or pathway in a coherent way. For example, in the predatory Gram-positive bacterium *Myxococcus xanthus*, it is thought that more than 200 proteins related to two-component systems help sense and coordinate complex behaviours like the swarming.

Biochemistry offers a wealth of molecular machinery and regulatory mechanisms for synthetic biologists to perform biocomputations (see Figure 2 for some examples). For signals based on transcription, the most common approach is through the use of DNA-binding proteins that recruit (activate) or block (repress) the initiation of RNA polymerase at transcriptional promoters (see Figure 3). These repressors and activators are hindered by the fact that the DNA-binding sequence and hence specificity is fixed, limiting the promoters that can be controlled using these components. To tackle this problem, clustered regularly interspaced short palindromic repeats (CRISPR)/Cas9 systems have recently been repurposed to create programmable transcription factors. These work by employing a catalytically inactive 'dead' version

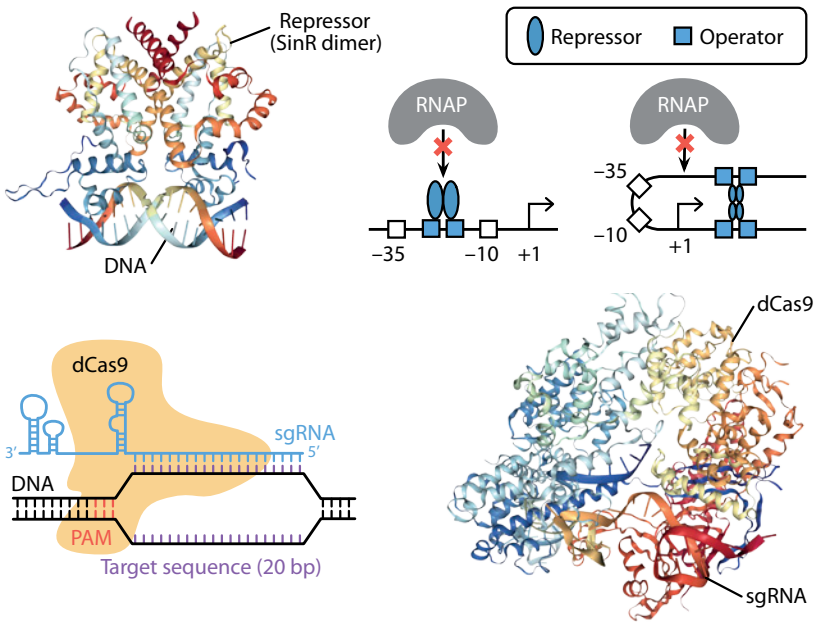


Figure 3. Controlling transcription initiation by using DNA-binding repressor proteins (top) and dCas9 (bottom). In both cases, physical blocking of RNA polymerase (RNAP) binding to the promoter region either through steric hindrance with the repressor, dCas9, or through looping of the DNA. Molecular visualizations show (top) SinR from *Bacillus subtilis* which regulates the process of sporulation, and (bottom) a nuclease-inactive 'dead' *Streptococcus pyogenes* Cas9 (D10A/H840A) in complex with single-guide RNA. In the dCas9 DNA binding schematic (bottom left), the 3 bp CCN Protospacer Adjacent Motif (PAM) is shown in red.

of the RNA-guided DNA endonuclease Cas9 protein (dCas9) with a single guide RNA (sgRNA) that targets a promoter that the bioengineer aims to regulate (see Figure 3). Binding of the dCas9-sgRNA complex at the promoter physically blocks initiation of RNA polymerase and represses expression of any associated genes. Furthermore, because an alternative target can be easily chosen by creating a different sgRNA with a complementary targeting sequence, large and complex regulatory networks using sets of different sgRNAs can be easily constructed.

For signals based on the concentrations of small molecules (e.g. metabolites), chemical reaction networks employing enzymes to catalyse certain reactions can be used to encode regulatory relationships approximating basic logic functions (see Box 1 for a description of Boolean logic). For example, an allosterically regulated enzyme E that catalyses the reaction $A \rightarrow B$ but is negatively regulated by metabolite X will cause the concentration of B to capture a B = NOT X Boolean function as long as A is always in excess and B is swiftly degraded. Specifically, if X is low (off) the enzyme is active causing B to be high (on), and if X is high (on) the enzyme will be inhibited causing B to be low (off). Unlike transcriptional systems where the propagation of a signal requires the time-consuming synthesis of large RNA molecules, chemical reaction networks rely on simpler chemical conversions and so are able to respond much quicker.

Using protein-based components offers another way to implement fast biocomputations either through the use of chemical modifications (e.g. phosphorylation) or by modifying a protein's structure (e.g. cleavage by proteases). Recently, a new system called CHOMP (circuits of hacked orthogonal modular proteases) was created that employed viral proteases able to interact and cleave each other in a programmable way (see Figure 4). This allowed a wide array of logic gates and dynamic signal processing algorithms to be implemented. In addition, the circuits developed are: (1) able to respond quickly because transcription and translation are not required, (2) can be localized to work in specific cell compartments or organelles because they do not rely on interactions with DNA and (3) their viral origin ensures few unwanted interactions with endogenous cellular processes.

Structural modifications to DNA sequences can also be used to carry out biocomputations and create DNA-based memory, allowing cells to effectively remember events or stimuli. By using recombinase enzymes that catalyse structural DNA rearrangements between specific recognition sites it is possible to precisely flip the orientation, excise or insert DNA at a desired location (see Figure 5). A permanent memory can be created by having the orientation of a piece of DNA encode a single 'bit' of information (i.e. whether something is true/false or a has the value of 1 or 0). Logic can be implemented by having the DNA regions flipped contain elements to control gene expression. For example, an expression construct can be designed with multiple transcriptional terminators inserted between a promoter and protein-coding region to initially stop transcription reaching the protein-coding region. If each terminator is also flanked by specific recombinase sites, expression of an associated recombinase enzyme will flip the orientation of a single terminator, allowing transcription to pass this element. Because all terminators need to be flipped in order for transcription to reach the protein-coding region, the construct implements a basic AND Boolean function (see Box 1) where recombinase 1 AND recombinase 2… need to be expressed in order for the protein-coding region to also be expressed (see Figure 2). A major benefit of these types of biocomputation is that each segment of DNA can have one of only two orientations. These fixed orientations provide clear on and off states that are more robust than the high and low concentration levels or rates used to encode on and off states with other types of signal where cell-to-cell variability can make distinguishing each difficult.

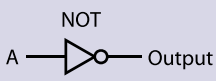
Challenges facing the field

Advances in DNA synthesis and assembly mean that it is now possible to create genetic constructs encoding large and complex biocomputations. Unfortunately, due to our limited understanding of how biological components behave in new contexts and how they are best pieced together, often these systems don't work as expected. Unlike in electronic systems where many tools exist to probe and debug such failures, few tools are available to monitor the many types of signal that might be used during a biocomputation. This is, however, starting to change with the recent application of RNA-sequencing

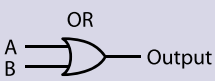
Box 1

Combinatorial logic is a powerful way to describe many types of computational function without worrying about how they might be biologically implemented. In this framework all inputs and output take Boolean values, i.e. can either be true (1) or false (0). Logic gates perform basic functions and can be connected together by wiring the output of one gate to the input of another. This enables the implementation of more complex functions. The function of a logic gate can be described by a truth table, consisting of columns for each input/output and rows for each unique combination of inputs. In the context of biological systems, sensors are connected to inputs and outputs are used to drive other cellular processes (see Figure 2 for some examples).

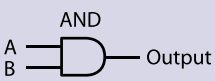
Common logic gates with truth tables



A	Output
0	1
1	0

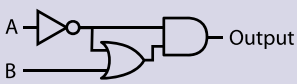


A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1



A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

Logic circuit



A	B	Output
0	0	1
0	1	1
1	0	0
1	1	0

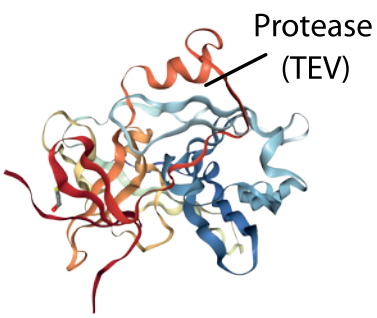
techniques to allow transcriptional signals across an entire circuit and the host cell to be monitored. Deviations from expected behaviours can then be used to ensure the correct parts are replaced to fix the observed faults. Being able to ensure biocomputations are functioning as we expect is likely to grow in importance, especially for complex layered circuits and applications where failures could have devastating effects (e.g. medicines).

Future directions

So far, the design of new biocomputations has mostly used concepts from digital circuit design. An awareness that this perspective is not always best suited when using biochemistry could open new opportunities to learn from the forms of computation biology excels at. For example, unlike electronic circuits, biological systems evolve. Exploring ways to exploit this feature to create adaptive computations able to refine their function over time would harness the natural capabilities of biology rather than fight against them. Furthermore, the running of evolutionary algorithms on electronic computers has proved a valuable tool for optimization tasks. An ability to implement these using biological substrates that are naturally suited to the underlying operations of mutation, recombination and selection could offer significant advantages for future evolutionary algorithms.

The vast number of cells present in many biological systems also offers exciting opportunities to implement many different computations concurrently at a low cost. For example, a small tube of bacterial culture contains billions of cells and a handful of soil contains thousands of species of microorganism. By moving from biocomputations constrained to work within a single cell to the coordination of many independent computational units (cells), calculations could be performed at scales that dwarf what is possible with electronic computers.

Biology relies on the ability of cells to compute. By using synthetic biology to create our own biocomputations, we not only have a powerful means of tackling challenges spanning from climate change to unmet healthcare needs, but also gain a deeper understanding of the computational processes supporting all life. ■



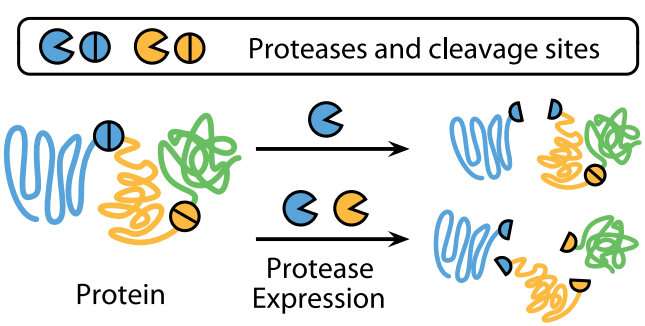


Figure 4. Proteases can be used to cleave proteins at specific points enabling the creation of many physically separated protein products. This process can be controlled to generate different sets of protein products from the same starting protein depending on the proteases expressed. Molecular visualization shows the *tobacco etch virus* (TEV) nuclear-inclusion-a endopeptidase commonly used to control the cleavage of fusion proteins due to its high amino acid sequence specificity.

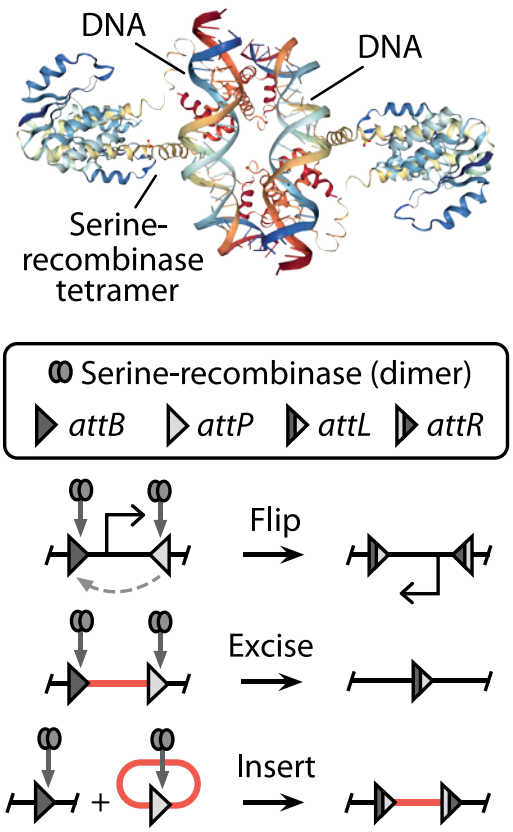


Figure 5. Serine-recombinase enzymes can be used to precisely manipulate the structure of DNA sequences. Depending on the location and orientation of the *attB* and *attP* recognition sites in a DNA sequence, an associated recombinase enzyme can flip, excise or insert regions, generating *attL* and *attR* sites that are then not processed further. Molecular visualization shows a serine recombinase tetramer in complex with DNA.

Further reading

- Bray, D. (2011) Wetware: A Computer in Every Living Cell. Yale University Press, USA
- Morton, O. (2019) The engineering of living organisms could soon start changing everything. The Economist. Accessed: https://www.economist.com/technology-quarterly/2019/04/04/the-engineering-of-living-organisms-could-soon-start-changing-everything
- Brophy, J.A.N. and Voigt, C.A. (2014) Principles of genetic circuit design. Nat. Methods **11**, 508–520
- Xie, Z., Wroblewska L., Prochazka L., Weiss R. and Benenson Y. (2011) Multi-input RNAi-based logic circuit for identification of specific cancer cells. Science **333**, 1307–1311
- Gander, M.W., Vrana, J.D., Voje, W.E., Carothers, J.M. and Klavins, E. (2017) Digital logic circuits in yeast with CRISPR-dCas9 NOR gates. Nat. Comms. **8**, 15459
- Green, A.A., Kim, J., Ma, D., Silver, P.A., Collins, J.J. and Yin, P. (2017) Complex cellular logic computation using ribocomputing devices. Nature **548**, 117–121
- Arkin, A. and Ross, J. (1994) Computational functions in biochemical reaction networks. Biophys. J. **67**, 560–578
- Gao, X.J., Chong, L.S., Kim, M.S. and Elowitz, M.B. (2018) Programmable protein circuits in living cells. Science **361**, 1252–1258
- Sheth, R.U. and Wang, H.H. (2018) DNA-based memory devices for recording cellular events. Nat. Rev. Genet. **19**, 718–732
- Gorochowski, T.E., Borujeni, A.E., Park, Y., et al. (2017) Genetic circuit characterization and debugging using RNA-seq. Mol. Syst. Biol. **13**, 952

Veronica Greco is a Royal Society-funded PhD student working in the Biocompute Lab at the School of Biological Sciences, University of Bristol, UK. Her research involves the systematic study of DNA-based memory devices in living cells using high-throughput assembly and sequencing techniques to better understand their design principles.
Email: veronica.greco@bristol.ac.uk

Matthew Tarnowski is a PhD student in the Biocompute Lab at the School of Biological Sciences, University of Bristol, UK. His research is at the interface of experimental and computational biology, using DNA sequencing to understand and engineer molecular bio-systems.
Email: matt.tarnowski@bristol.ac.uk

Thomas Gorochowski is a Royal Society University Research Fellow at the University of Bristol, UK and heads up the Biocompute Lab (www.biocomputelab.org) within the School of Biological Sciences. His research is focused on better understanding the computational architecture of biological systems from the molecular to ecosystem level and developing tools to enable the rational engineering of new functionalities across these scales.
Email: thomas.gorochowski@bristol.ac.uk

