Bacterial Computing

ABSTRACT:

Bacteria are wondrous creatures. They can reproduce in less than half an hour, they can rapidly adapt to changing conditions, and some strains are able to survive in extreme temperatures & conditions. Biologist and computer scientist have designed digital logic gates based on the metabolism of living cells, with the aim of eventually building a computer out of colonies of Escherichia-coli or some other single celled organism. An agar plate coated with microscopic bacteria is not much different from a silicon wafer etched with microscopic transistors. If the components can store and manipulate in a few basic ways, they can compute. Semi conducting particles could then stick to any array of proteins, creating an array of quantum dots and could serve as computer memory, a sensor or a logic device that could calculate. One appeal of bacterial computing is that bacteria are very cheap to manufacture. Scientist can grow trillions of bacteria in a lab for a few dollars So instead of building expensive factories, companies could use cheap programmed bacteria that grow in giant vats to manufacture drugs, fuels and plastic. In fact, bacterial computing is similar in some ways to the way the human brain functions. It can produce a reliable outcome despite unreliable parts. Our brains function reliably even though individual neurons constantly misfire or die. Also, like neurons, the computing power of each bacterium is small compared to silicon transistors, but the sheer number overcomes lack of speed. Analogous to the incredible capacity of the brain, these computers possess an undefined potential to where they can be applied. The biological computer is a wireless device where signals are broadcast to the cell. In a biochemical computer, communication comes for free.

INTRODUCTION TO DIGITAL TECHNOLOGY:

A digital technology usually starts with Boolean logic gates, devices that operate on signals with two possible values, such as true/false, 1/0. An "AND" gate has two or more inputs and one output: the output is true only if all the inputs are true. An "OR" gate is similar except that the output is true if any of the inputs are true. The simplest of all gates is the NOT gate, which takes a single input signal and produces the opposite value as output: true becomes false, false becomes true.

"This is nanotechnology in a wetter form, namely in a living cell," said Dr. James J. Collins, a professor of biomedical engineering at Boston University who is a physicist by training and has become a leader in biocomputing.

BASIC CONCEPTS:

In electronic circuits a NOT gate can be made from a single transistor, wired so that a high voltage at the input produces a low voltage at the output and vice versa.When the gate switches between its two states it does so abruptly like a snap-action light switch. It is this sudden, non-linear response that gives digital devices their resistance to noise & error. There are hundreds of biochemical equivalents to transistor gates. Perhaps the most interesting among them are the mechanisms of genetic control which switch genes on & off.

GENETIC MECHANISM:

The archetypal example of genetic regulation in bacteria is the lac operon of E-coli. The operon is a set of genes and regulatory sequences involved in the metabolism of certain complex sugars, including lactose. The bacterium's preferred nutrient is the simpler sugar glucose but when glucose is scarce , the cell can make do by living on lactose. The enzymes for digesting lactose are manufactured in quantity only when they are needed- specifically when lactose is present and glucose is absent.

As in the expression of any genes, synthesis of the lac enzymes is a two stage process.First the DNA is transcribed into messenger RNA by the enzyme RNA polymerase. Then the m-RNA is translated into protein by ribosomes.The process is controlled in the transcription stage. before the genes can be transcribed, RNA polymerase must bind to the DNA at a special site called the promoter, which is just "upstream" of the genes;

LAC OPERON:

Then the polymerase must travel along one strand of the double helix, reading off the sequence of nucleotide bases and assembling a complementary strand of messenger RNA.One mechanism of control prevents transcription by blocking the progress of RNA polymerase molecules. The blocking is done by the lac repressor protein, which binds to the DNA downstream of the promoter region and stands in the way of the polymerase. When lactose enters the bacterial cell the lac operon is released from this restraint. A metabolite of lactose binds to the lac repressor, changing the proteins shape & there by causing it to loosen its grip on the DNA. As the repressor protein drifts away the polymerase is free to march along the stand & transcribe the operon. The repressor system is only half of the lac control strategy.Even in the presence of lactose the lac enzymes are synthesized only in trace amounts if glucose is also available in the cell. The reason, it turns out is that the lac promoter site is a feeble one, which does a poor job of attracting & holding RNA polymerase. To work effectively, the promoter requires an auxillary molecule called an activator protein, which clamps on to the DNA & makes it more receptive. Glucose causes the activator to fall away from the DNA just as lactose causes the repressor to let go-but the ultimate effect is the opposite. Without the activator, the lac operon lies dormant. All these tangled interactions of activators and repressors can be simplified by viewing the control elements of the operon as a logic gate.

The inputs to the gate are the concentrations of lactose and glucose in the cells environment. The output of the gate is the production rate of the 3 lac enzymes. A factor that tends to steepen the response curve is the cooperative action of multiple subunits in the regulatory proteins. The lac repressor consists of four subunits, and the lac activator has two. Although the first subunit may be slow in binding to the DNA, subsequent units stick to one another as well as to the DNA, and so the binding goes faster. The net effect is to make the threshold for repression or activation sharper. Thus increases the computation speed.

It is also possible to develop design rules and parts catalogue for biological computers, like the comparable tools that facilitate design of electronic integrated circuits. An engineer planning the layout of a silicon chip does not have to define the geometry of each transistor individually; those details are specified in a library of functional units, so that the designer can think in terms of higher-level abstractions such as logic gates and registers. A similar design discipline will be needed before biocomputing can become practical. The elements in the biocomputing design library will be repressor proteins.

DESIGN OF LOGIC GATES:

The logic "family" might be named RRL, for repressor - repressor logic, in analogy with the long established TTL, which stands for transistor - transistor logic. The basic NOT gate in RRL will be a gene encoding some repressor protein (call it Y) with transcription of the Y gene regulated in turn by a different repressor (call it X). Thus whenever X is present in the cell, it binds near the promoter site for Y and blocks the progress of RNA polymerase. When X is absent, transcription of Y proceeds normally. Because the Y protein is itself a repressor, it can serve as the input to some other logic gate, controlling the production of yet another repressor protein, say Z. In this way gates can be linked together in a chain or cascade. Going beyond the NOT gate to other logical operations calls for just a little more complexity. Inserting binding sites for two

repressor proteins (A and B) upstream of a gene for protein C creates a NOR gate, which computes the negation of the logical OR function. With the dual repressor sites in place, the C gene is transcribed only if either A or B is absent from the cell; if either one of them should rise above a threshold level, production of C stops. In other words; C is transcribed only if neither A or B is present. The NOR gate is said to be a universal logical gate and any Boolean function can be generated by linking together a series of NOR gates. Pairs of NAND gates can be coupled together to form a computer memory element known as Flip Flop. Implementing this concept in RRL calls for 2 copies of the genes coding for 2 repressor proteins M and N. One copy of the M gene is controlled by a different repressor R and likewise one copy of the N gene is regulated by repressor R. In the second pair each of these proteins inhibits the other's synthesis. Here's how the Flip Flop works. Through the cross coupling of the second pair, M suppresses the output of N with the collateral result that M's own repressor site remains vacant, so that production of M can continue. But now imagine that the S protein falls below threshold. This event briefly lifts the repression of the N gene in the first pair. The resulting pulse of N protein represses the M gene in the second pair lowering the concentration of protein M, which allows N gene to be manufactured. Thus a momentary change in switches the system from steady production of M to steady production of N. Likewise a brief blip in R would switch it back again.(S and R stand for "set" and "reset")

DISCUSSION:

One conclusion to be drawn from this synopsis of a few RRL devices is that a computer based on genetic circuits will need a sizable repertory of different repressor proteins. Each logic gate inside a cell must have a distinct repressor assigned to it, or else the gate would interfere with one another. In this respect a biomolecular computer is very different from an electronic one, where all signals are carried by the same medium- an electric current. The reason for the difference is that electronic signals are steered by the pattern of conductors on the surface of

the chip, so that they reach only their intended target. The biological computer is a wireless device, where signals are broadcast throughout the cell. The need to find a separate repressor for every signal complicates the designer's task, but there is also a compensating benefit. On electronic chips, communication pathways claim a major share of the real estate. In a biological computer, communication comes for free.A multitude of metabolic pathways have to be kept under control without unwanted cross talk. As a result, cells have evolved thousands of distinct regulatory proteins. Moreover, the biocomputing engineer will be able to mix and match among molecules and binding sites that may never occur together in the natural world. The aim of the RRL design rules is to identify a set of genes and proteins that can be encapsulated as black box components, to be plugged as in needed without any thought about conflicts.

Another important design tool is the simulator, which allows a device to be tested without the substantial effort of building a prototype. The world of electronics has long relied on a simulator called spice, which models the physics of transistors and other electronic components. A biospice simulator can model the dynamics of genetic circuits in a similar way.

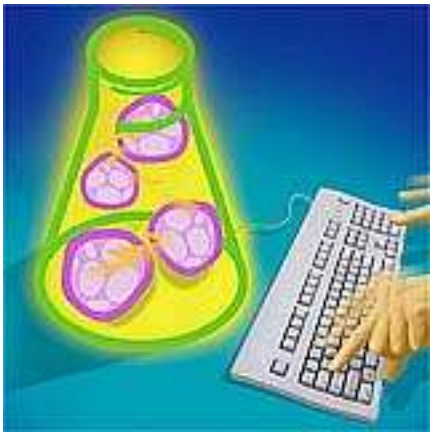
A free running genetic oscillator in E.coli, arranging in three repressor genes so that they act on one another in turn, it was observed periodic fluctuations in gene expression, with a frequency independent of cell's reproductive cycle. In another E.coli experiment, a genetic toggle switch much likes the flip flop with two cross-coupled promoters and repressors. The report is "robust bistability". The eventual aim is the construction of "genetic applets"- self-contained program modules that could be "downloaded" into organisms.

BUILDING COMPUTER CHIPS FROM BACTERIA:

Using a heat tolerant protein from Sulfolobus shibatae, a bacterium that lives in geo thermal hot-springs, they are able to form ring shaped cells just 10 or 20 nanometers across called Chaperonins. Each cell is analogous to cells in arrays produced by lithographic techniques for computer chips, but even the best

modern techniques are limited to around 100nm. These biological based arrays of nano particles could have future applications in computer memories, sensors or logic devices.

These arrays are formed by applying the chaperonins to a substrate of silicon where they self-assemble into a repeating hexagonal pattern. A slurry of gold nano particles and a semi conducting material, cadmium selenide-zinc sulphide, is then added. The materials adhere only to active sites around the hole in each protein ring resulting in a precise, regular array of nano particles.



APPLICATIONS:

- **Computer chips can be built from a bacterium, Sulfolobus shibatae.**
- **Biological based arrays of nano particles could have future applns in computer memories, sensors or logic devices.**
- **A BioSpice simulator can model the dynamics of genetic circuits.**
- **A free running genetic oscillator in E-coli can absorb periodic fluctuations in gene expression.**
- **Flip flops with 2 cross coupled promoters and repressors can be designed in E-coli.**

CONCLUSION:

Persuading a living cell to perform useful computations is quite a trick. The real goal is to get a billion cells working in concert on the same task. The hard part

is organizing a population of cells so that they work toward some specified goal. A more likely prospect is a crop of programmable biological sensors, actuators and messengers. One contemplated application of such organisms is the assembly of nanoscale structures; instead of replacing semiconductor circuits, the cells would fabricate them. Another possibility is the old fantasy of a microscopic robot that could enter the human body to repair diseased tissues or combat infections. If this daydream of an intravenous computer is ever to happen, success seems more likely with the tools of genetic engineering than with a soldering iron. Despite few obstacles Biocomputing in the long run can prove to be a revolution in the field of computers.

REFERENCES:

www.technicalpapers.co.nr