

The screenshot shows a web browser window titled "Untitled Document" with the address bar displaying "http://www.guinnessworldrecords.com/content_pages/dna.htm". The browser interface includes navigation buttons (Back, Forward, Reload, Stop), a search bar with "Google" and "Mac News Tabs" tabs, and a sidebar. The main content area has a dark blue background with white text. The text is an interview transcript between Guinness World Record science editor David Hawksett and Ehud Shapiro and Yaakob Benenson, discussing the world's smallest biological computing device. The transcript includes a question about how the device works, followed by an explanation of the stored program computer model, the finite automaton model, and the molecular finite automaton.

The following is a transcript of an interview between Guinness World Record science editor David Hawksett and Ehud Shapiro and Yaakob Benenson, two of the creators of the world's smallest biological computing device.

DH: How does the smallest computing device work?

ES: All electronic computers work according to a mathematical model – developed by John von Neumann in the 1940s – called the "stored program computer". In it, both the program and data are stored as words in the computer memory. Each memory word can be accessed by its address. The electronic computer fetches program instructions one by one from its memory and executes them. Typically, the instructions are to load data from a certain memory location into a register, or to store the content of a register into a memory location, or to perform an operation on registers, such as adding the content of two registers and store the result in a third register.

The stored program computer model is ideal for realization using electronic circuits, and hence it has been the basis of all electronic computers for more than 50 years.

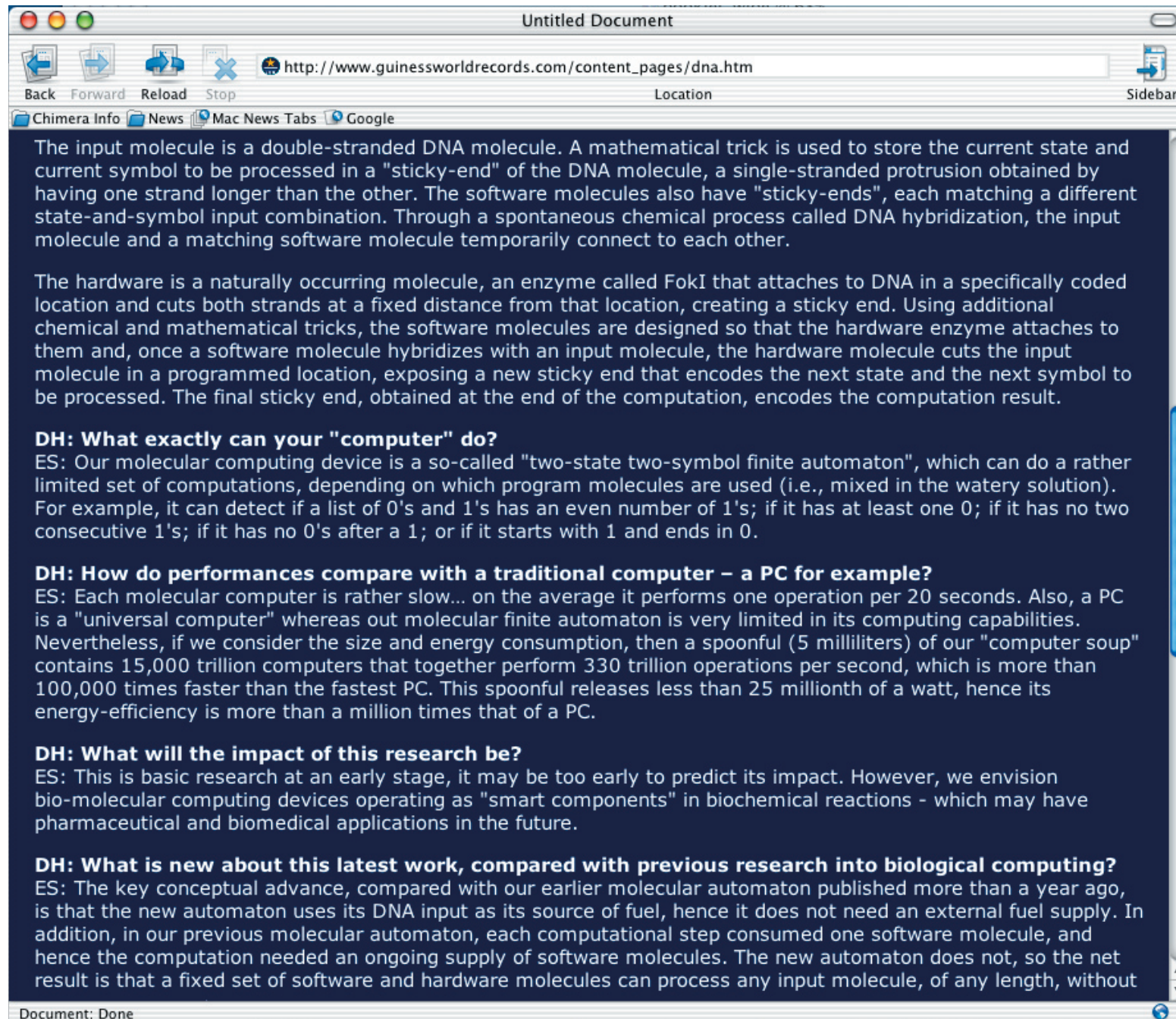
Our molecular computer is based on a very different model, called a "finite automaton", which is a simplified version of the Turing machine, a mathematical model of computation developed by the British mathematician Alan Turing in the 1930s.

The Turing machine has data stored as a string of symbols. (Turing envisioned a paper tape divided into squares, each holding one symbol.) A control unit, which holds the program rules, is in a particular location on the string, and can be in one of a finite number of states. The Turing machine starts the computation in an initiate state and on the leftmost symbol in the string. In each cycle, a Turing machine reads a symbol and executes a program rule that specifies what action to take according to the symbol read and the internal state. The actions can be to replace the symbol with a new symbol, move one symbol to the left or to the right, and/or change internal state.

A "finite automaton" is a Turing machine that cannot change the string of symbols, and in each step moves one symbol to the right. Hence the result of the computation of the finite automaton is the final state reached upon reaching the last input symbol.

Both the stored program computer and the Turing machine are so-called "universal computers" and as such have equivalent computing power. A finite automaton is much weaker.

Our "molecular finite automaton" has three components. An *input molecule*, which stores the data to be processed (as well as the fuel needed for the computation), *software molecules*, which encode program rules, and a *hardware molecule*, which performs the computation, as directed by the software and the input. The molecules float in a watery solution containing some salts needed for the operation of the hardware molecule.



The screenshot shows a web browser window titled "Untitled Document". The address bar contains the URL http://www.guinnessworldrecords.com/content_pages/dna.htm. The browser interface includes navigation buttons (Back, Forward, Reload, Stop), a search bar with "Google" entered, and a sidebar icon. The main content area has a dark blue background with white text. The text discusses DNA computing, including the use of FokI enzymes and the capabilities of molecular computers. The article is structured with questions (DH) and answers (ES).

The input molecule is a double-stranded DNA molecule. A mathematical trick is used to store the current state and current symbol to be processed in a "sticky-end" of the DNA molecule, a single-stranded protrusion obtained by having one strand longer than the other. The software molecules also have "sticky-ends", each matching a different state-and-symbol input combination. Through a spontaneous chemical process called DNA hybridization, the input molecule and a matching software molecule temporarily connect to each other.

The hardware is a naturally occurring molecule, an enzyme called FokI that attaches to DNA in a specifically coded location and cuts both strands at a fixed distance from that location, creating a sticky end. Using additional chemical and mathematical tricks, the software molecules are designed so that the hardware enzyme attaches to them and, once a software molecule hybridizes with an input molecule, the hardware molecule cuts the input molecule in a programmed location, exposing a new sticky end that encodes the next state and the next symbol to be processed. The final sticky end, obtained at the end of the computation, encodes the computation result.

DH: What exactly can your "computer" do?
ES: Our molecular computing device is a so-called "two-state two-symbol finite automaton", which can do a rather limited set of computations, depending on which program molecules are used (i.e., mixed in the watery solution). For example, it can detect if a list of 0's and 1's has an even number of 1's; if it has at least one 0; if it has no two consecutive 1's; if it has no 0's after a 1; or if it starts with 1 and ends in 0.

DH: How do performances compare with a traditional computer – a PC for example?
ES: Each molecular computer is rather slow... on the average it performs one operation per 20 seconds. Also, a PC is a "universal computer" whereas our molecular finite automaton is very limited in its computing capabilities. Nevertheless, if we consider the size and energy consumption, then a spoonful (5 milliliters) of our "computer soup" contains 15,000 trillion computers that together perform 330 trillion operations per second, which is more than 100,000 times faster than the fastest PC. This spoonful releases less than 25 millionth of a watt, hence its energy-efficiency is more than a million times that of a PC.

DH: What will the impact of this research be?
ES: This is basic research at an early stage, it may be too early to predict its impact. However, we envision bio-molecular computing devices operating as "smart components" in biochemical reactions - which may have pharmaceutical and biomedical applications in the future.

DH: What is new about this latest work, compared with previous research into biological computing?
ES: The key conceptual advance, compared with our earlier molecular automaton published more than a year ago, is that the new automaton uses its DNA input as its source of fuel, hence it does not need an external fuel supply. In addition, in our previous molecular automaton, each computational step consumed one software molecule, and hence the computation needed an ongoing supply of software molecules. The new automaton does not, so the net result is that a fixed set of software and hardware molecules can process any input molecule, of any length, without

Untitled Document

http://www.guinnessworldrecords.com/content_pages/dna.htm

Back Forward Reload Stop Location Sidebar

Chimera Info News Mac News Tabs Google

an external supply of energy.

YB: As we mentioned, DNA would break into pieces unless high energetic barriers prevent it from doing it. Nuclease enzymes lower these barriers and allow DNA to decompose. Some of the nucleases cleave very precisely at certain locations, determined by the exact sequence of the nucleotides at or near their cleavage sites. Yet, even such specific cleavage would only generate useless heat. In our molecular computer, we use this energy to drive a useful process, namely computation.

It should be understood that on the physico-chemical level we have a sequence of chemical reactions, resulting in a certain chemical product. This chemical process would not happen if energy were not released at each step. The energy is indeed released because we break DNA into smaller pieces with the help of FokI restriction nuclease.

At a conceptual level, the sequence of chemical reactions is the steps of computation, and the final product is a computational output. The whole computation process and output formation depend on the energy stored in the starting material, a molecule encoding an input data. Therefore, the computation (an abstract notion) is made possible thanks to the (physical) energy stored in the (physical implementation of its) input relative to the energy stored in its output. There are no other sources of energy; the computation is fueled solely by its input.

So, there are five main differences between what we've done now and what we achieved in the past:

1. The new computer does not use ATP [Adenosine triphosphate, an organism's natural source of energy] and is fueled by its input; the old one relied on ATP as its energy source.
2. The new computer recycles its software and hardware; the old one consumed one software molecule per transition.
3. The new computer is 50 times faster and performs 8000 times more operations in parallel than the old one.
4. The new computer can process much longer input strings. We could routinely compute on inputs 12 symbols long but also got results with 20-symbols long strings. The yield of a single step rose above 98%. The old one could only process 6-symbol inputs on a good day, with about 50% single step yield.
5. The new computer has a completely redesigned structure, making possible all the improvements summarized above.

DH: What other work has been done in this field?

ES: The field started in 1994 with the publication of a paper by Len Adleman. It described how DNA can be used to solve combinatorial problems, such as the so-called "traveling salesman" problem, in which the aim is to find an optimal route through several cities, given the distances between them. The hope was that the parallel nature of DNA manipulation could be used to outperform electronic computers in solving such problems.

YB: Another line of research is "semi-autonomous molecular computation", which is pursued by Hagiya and Sakamoto in Tokyo University, and by Winfree, Reif and Seeman in the USA. This approach aims at creating systems that evolve in a programmed fashion to perform a computation, while minimizing the external intervention. These researches were successful in performing very simple computations without much interference except regulating the temperature during the process. Still, experimentally demonstrated computation capabilities of these systems are substantially lower than that of our devices, and they're not completely autonomous.

Document: Done