Bio-Computation using Holliday junctions

Niall Murphy[1*] Damien Woods[2*] Thomas J. Naughton[1]
[1]Department of Computer Science,
National University of Ireland, Maynooth
http://www.cs.nuim.ie/~nmurphy and
http://www.cs.nuim.ie/~tnaughton

[2]Boole Centre for Research in Informatics, School of Mathematics,
University College Cork, Ireland
http://www.bcri.ucc.ie/~dw5

Abstract

We present a design for a novel computing machine composed of an artificial arrangement of DNA and proteins. We characterise the computational power of this construction by proving that its prediction problem is P-Complete.

Keywords: P-Complete, Holliday junction, biological computer, microtubule.

The natural world has always been an inspiration for computer scientists but biology in particular has been a very fertile source of new ideas. Examples of such are von Neumann's self reproducing automata [14], neural networks [9], evolutionary algorithms [7] and membrane computing [12].

Thomas Head was inspired by DNA and was among the first to explore the potential of DNA for computing [6]. However, it was Adleman who performed the first successful *in vitro* DNA computation [1]. Since then most DNA computers have utilised Watson-Crick complementarity to direct self assembling DNA molecules which encode both the input and the program. Sections of DNA are designed to clip together like a jigsaw which develops into a structure representing the progression or the result of a computation.

Although there is much exciting work being done [2, 3, 13] with biological computers, the move from the test-tube to a usable, practical system has yet to happen. We believe that the techniques of synthetic biology [4] will help in achieving the goal of a practical bio-computer. The method of synthetic biology is to apply the principles of engineering to the study of life. By engineering and reverse engineering living systems, synthetic biologists aim to learn about how biological systems and gene circuits operate. Synthetic biologists also aim to create custom new life for specific tasks by treating understood systems as components. By using this spirit and assembling our own machinery from cell parts we could possibly build a practical and useful bio-computer.

*Supported by the Irish Research Council for Science, Engineering and Technology

A Holliday junction arises when two homologous DNA double helices are aligned side by side [11]. Two strands of each helix partially unzip and are exchanged. The two joined DNA double helices then rotate to form a cross shape and proteins attach to separate and recombine the strands as shown in Figure 1. A Holliday junction will not form if the homologous DNA double helices meet perpendicularly. While each Holliday junction as an individual entity can be considered a computing device, we outline how it could be possible to artificially connect together individual junctions to construct a biological computer. The resulting chain of junctions would most likely never arise in nature. However, if proteins chosen for their function were arranged correctly it would be possible to artificially implement a Holliday junction computer. We call this computer a Holliday Framework.

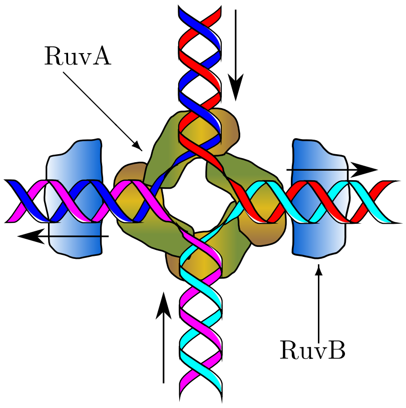

Figure 1: The Holliday junction illustrated with *Escherichia coli* proteins RuvA and RuvB. Arrows indicate the movement of the strands.

To join Holliday junctions together we design an artificial framework constructed of regular proteins and chemicals. The environment of the framework is balanced to produce optimal Holliday junction formation and consistent DNA movement. The framework consists of pairs of microtubule tracks arranged to construct a grid pattern. It is on these tracks that homologous DNA strands will be pulled by dynein motor proteins. Each pair of tracks acts like road where each track allows DNA traffic in one direction only. When two DNA double helices approach from opposite directions they will form a Holliday junction and will produce two double helices moving perpendicularly away from the junction site. These will then be able to form new junctions with any other homologous double helices they come across. We characterise the computational complexity of the Holliday Framework by showing that instances of Circuit Value Problem (CVP) [8] can be converted to instances of Holliday Framework Prediction by a logspace Turing Machine.

Definition 1 (Holliday Framework Prediction Problem). Given: A string x, a description $\overline{H}$ of a Holliday Framework H, and an integer t coded in unary. (More precisely, the input is the string $x\#\overline{H}\#^t$, where # is a delimiter character not otherwise present in the string.) **Problem:** Does M accept x within t timesteps?

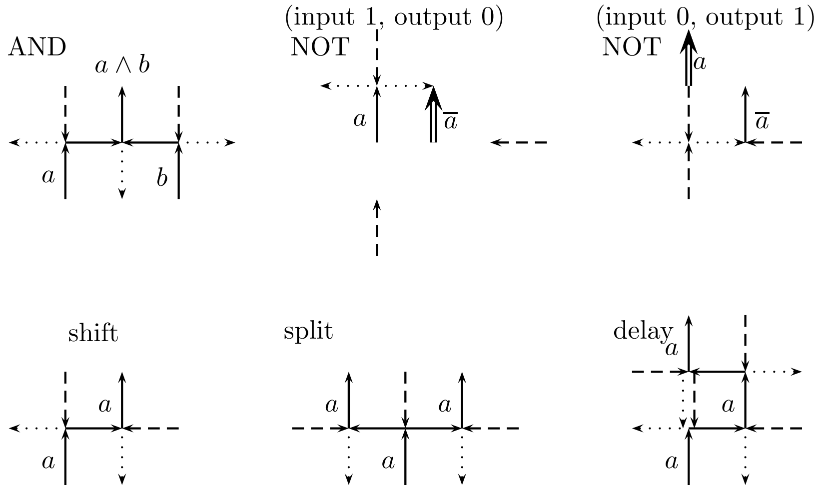

Figure 2: The widgets needed to make circuits. Solid arrows carry logic *true*, double lined arrows are logic *false*, dashed arrows are part of the widget, and dotted arrows are waste. The two NOT gates represent both possibilities for input, the second gate can be better understood as the next timestep of the first NOT gate, with a *false* input.

Our proof is similar to that used by Moore and Nordahl for lattice gases [10]. In our framework, the presence of a DNA double helix will represent a logical *true* and absence will indicate $false$. To represent logic gates we provide widgets, groups of several Holliday junctions that rely on the presence of truth signals which will only serve to provide a junction and produce waste signals. The waste signals are prevented from interfering with other junctions by staggering the computation so that there is only one junction in a row or column. The widgets for AND and NOT gates are shown in Figure 2, with these we can construct the universal NAND gates. We can connect our gate widgets together using the utility widgets; split, delay, and shift shown in Figure 2.

A logspace bounded Turing machine takes the CVP input and replaces all OR gates with AND and NOT gates via de Morgan's Laws. Then for each gate DNA strands are placed in position to create a Holliday junction at the correct time step and location. The resulting Holliday Framework has size and depth (in junctions) that is a constant times the respective size and depth of the circuit. So, since the instance of CVP has polynomial depth, the Holliday framework result of the reduction will also have polynomial depth. Thus Holliday Framework prediction is P-complete.

The task of deciding if a Turing Machine M will accept input x within t timesteps is known as the Generic Machine Simulation Problem (GMSP) [5] and is also P-complete. P-Complete problems are, by definition [5], logspace reductable to each other and so it is possible to encode an instance of GMSP as an instance of Holliday Framework Prediction using a logspace Turing Machine. Thus the Holliday framework is an efficient simulator of Turing Machines.

We have proposed a novel method for computing using the mechanical process of genetic combination then show that this method of computing is an efficient simulator of Turing Machines. The purpose of this paper is to inspire researchers to explore the possibilities of constructing artificial biological systems custom built to serve as biological computers and provide a simple tool, P-completeness, to prove that such a system is an efficient and useful computational device.

References

[1] L. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, 1994.

[2] Y. Benenson, T. Paz-Elizur, R. Adar, E. Keinan, Z. Livneh, and E. Shapiro. Programmable and autonomous computing machine made of biomolecules. *Nature*, 414:430–434, 2001.

[3] M. Cook, P. W. Rothemand, and E. Winfree. Self-assembled circuit patterns. In *DNA Computers 9*, volume 2943, pages 91–107. LNCS, 2004.

[4] D. Ferber. Synthetic biology: Microbes made to order. *Science*, 303(5655):158–161, jan 2004.

[5] R. Greenlaw, H. J. Hoover, and W. L. Ruzzo. *Limits to parallel computation:P-completeness Theory*. Oxford University Press, New York, Oxford, 1995.

[6] T. Head. Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors. *Bulletin of Mathematical Biology*, 47(6):737–759, 1987.

[7] J. H. Holland. *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, USA, 1975.

[8] R. E. Ladner. The circuit value problem is log space complete for P. *SIGACT News*, 7(1):18–20, 1975.

[9] W. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[10] C. Moore and M. Nordahl. Predicting lattice gases is P-complete. Institute Working Paper 97-04-034, Santa Fe Institute, 1997.

[11] H. Potter and D. Dressler. In vitro system from escherichia coli that catalyzes generalized genetic recombination. *Proceedings of the National Academy of Sciences of the United States of America*, 75(8):3698–3702, August 1978.

[12] G. Păun. Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143, 2000.

[13] P. Sa-Ardyen, N. Jonoska, and N. C. Seeman. Self-assembling DNA graphs. *Natural Computing*, 2(4):427–438, 2003.

[14] J. von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana and Landon, 1966.