Project no. 248992

Project acronym: NEUNEU

Project title: Artificial Wet Neuronal Networks from Compartmentalised Excitable Chemical Media

Small or medium-scale focused research project (STREP)

Deliverable 3.4 - Report or publication on signal dynamics within many droplet systems

Period covered: from 1.2.2010 to 15.2.2012 Date of preparation: 18.9.2009

Start date of project: 1.2.2010 Duration: 36 months

Project coordinator name: Dr. Peter Dittrich
Project coordinator organisation name: Friedrich Schiller University Jena

Symbol Representations in Evolving Droplet Computers

Gerd Gruenert, Gabi Escuela, Peter Dittrich

March 1, 2012

Abstract

We investigate evolutionary computation approaches as a mechanism to program networks of excitable chemical droplets. For this aim, we first concentrate on the characteristics of symbol representing signals in this kind of coupled excitable systems when assigned a specific task. Considering a given Boolean function like Identity, OR, AND, NAND, XOR, XNOR or the half-adder as the target functionality, 2D networks composed of 10×10 droplets. Three different setups were tested: Evolving network structures with fixed on/off rate coding signals, coevolution of networks and signals, and network evolution with fixed but pre-evolved signals. Evolutionary computation served in this work not only for designing droplet networks and input signals but also to estimate the quality of a symbol representations: We assume that a signal leading to faster evolution of a successful network for a given task is better suited for the droplet computing infrastructure. Results show that complicated functions like XOR can evolve using only rate coding and simple droplet types, while other functions involving negations like the NAND or the XNOR function evolved slower using rate coding. Furthermore we discovered symbol representations that performed better than the straight forward on/off rate coding signals for the XNOR and AND Boolean functions. We conclude that our approach is suitable for the exploration of signal encoding in networks of excitable droplets.

1 Introduction

In an excitable medium the propagations and collisions of waves of chemical activity can be used for computation [1]. We refer to droplets as small amounts of excitable medium floating in oil that are covered with a layer

of lipid molecules. The lipids stabilise the droplets against merging but still allow two adjacent droplets to communicate when the lipid molecules form a bilayer similar to that of biological cells [2]. Excitation waves can be transmitted through droplets but can also interfere with one another, dependent on their timing and on the chemical properties of the droplets and the medium within. Hence, droplets arranged in a network form a potential chemical computer [7, 14, 9].

In a droplet based computer, the spatiotemporal dynamics of the excitation waves determine the computation, therefore the topology of the coupled droplets plays a decisive role when "programming" such devices. Additionaly we can also look at the signals representation in order to discover an adequate and efficient interpretation for them. Here we refer to "programming" in the broadest sense of specifying the desired functionality of a computing device in contrast to the typically understood exact algorithmic specification of data manipulation. Examples for this unconventional sense of programming could be evolutionary algorithms, functional programming languages, amorphous computing, spatial computing, collision computing, chemical computing, membrane computing, natural computing, neural computing [3] and liquid state machines [11].

In this study, we consider evolutionary algorithms [10, 15, 6] as a mechanism to infer adequate symbol representations when building logic gates with droplet networks. Given an optimisation problem, an evolutionary algorithm selects for good individuals in a population of solutions that changes over time via genetic operators. Starting with a randomly generated population and guided by the fitness function, the evolutionary algorithm gives us after several generations an approximating solution to the problem. The use of evolutionary algorithms to design logic gates and circuits has been studied specially in the context of genetic programming [10] and evolvable harware [12]. The idea is to implement computers based on reconfigurable hardware that are able to adapt to a changing environment.

We are not aiming at building a single droplet network design that could act as a universal computer, solving any kind of computable problem. But it appears feasible and useful to build droplet devices that compute results for different instances of a problem. Therefore, given a problem instance, input data needs to be specified in some way. This could either happen through the initial state of the droplet system or during the runtime, most probably through external stimulation of certain droplets. In either way it is an important design decision which encoding is used to feed inputs into the droplet network. Most probably the optimal encoding will depend on a number of factors like the type of task, the number of used symbols, parameters of the computing substrate and the applied quality measure. From

the neurosciences we know for example the coding techniques rate coding, population coding and temporal coding [4]. While rate coding uses the oscillation frequency to distinguish different meanings, different populations of active neurons are meant by population coding. Temporal coding, in contrast, utilises the timing differences between droplets as information carrier. These coding schemes might be candidates for excitable droplets as well.

To find adequate symbol representations for droplet computers, we start by evolving droplet networks that fulfil a functionality, given by simple Boolean functions. Similar to evolutionary algorithms or to genetic programming the evolved droplet network topology can be seen as the definition for a program that can be executed on the droplet computing architecture. Then we explore the coevolution of the droplet network topologies with different symbol encoding options for two symbols and basic Boolean logic functions.

2 Methods

Droplet Networks

We generate simulations of droplet networks in a 10×10 grid of simulation droplets that are connected in a von Neumann neighbourhood, such that all directly adjacent cells can excite each other. Up to four different kinds of cells are used, which represent empty cells, normal droplets, droplets of lower excitability and droplets with longer oscillation periods. Furthermore, there are two fixed input droplets and two fixed output droplets defined on the network grid. They can be used to dynamically feed a stream of excitations into and out of the droplet network. We represent a specific droplet network instance as an n by n matrix as visualised in Figure 1a:

$$N = \begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,n} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n,1} & d_{n,2} & \cdots & d_{n,n} \end{pmatrix}$$

$$d_{i,j} \in \{\varnothing, d_{Norm}, d_{LowEx}, d_{Slow}, d_{In0}, d_{In1}, d_{Out0}, d_{Out1}\}$$

A von Neumann neighbourhood around each droplet $d_{i,j}$ defines the connectivity of the droplets, i.e. droplets that are at the positions $d_{i,j-1}$, $d_{i,j+1}$, $d_{i-1,j}$ or $d_{i+1,j}$ are connected and can excite $d_{i,j}$ or be excited by it.

To allow fast simulations while being able to fine-tune the droplet timing parameters and noise levels, we simulate the droplet networks using our discrete-event simulation approach [8] with the following parameters: Normal droplets d_{Norm} as well as input and output droplets are modelled with

an expected oscillation period of 16 s, which is composed of 10 s responsive time τ_{res}, 1 s excited time τ_{ex} and 5 s refractory time τ_{ref}. Signal propagation delays τ_{prop} are 1 s. Droplets self-excite after not being triggered into an excitation externally after the responsive time τ_{res} of about 10 s. The exact timing parameters for each phase are sampled using normal distributions with a standard deviation of 0.05 s around the mean values given before. Less excitable droplets d_{LowEx} use the same timing distributions but require at least two adjacent droplets to be excited at the same time to trigger an excitation. For the droplets with longer periods, all timing mean values as well as the standard deviations are multiplied by a factor of $\frac{3}{2}$.

Signal Encoding

When representing binary signals by rate coding, we stimulate droplets as much as possible for a symbol '1' and not at all for a symbol '0'. When droplets are maximally stimulated, the oscillation time will be $\tau_{ex} + \tau_{ref} =$ 6 s. Normal droplets that are left alone do not stop oscillating but their frequencies are lower with periods of $\tau_{ex} + \tau_{ref} + \tau_{res} = 16$ s.

To allow more complex symbol representations, we use a timing pattern that determines which input droplet is stimulated from the outside at which times. We divide the length T of the stimulation pattern up into m small intervals $\{I_1...I_m\}$, each of the length $\triangle t = \frac{T}{m}$. Hence, interval I_j is defined between the times $(j-1) \cdot \Delta t$ and $j \cdot \Delta t$. We define a channel's pattern as a Boolean vector, which states if droplets are stimulated in the interval I_j or not. To describe meaningful symbols, Δt should be small in comparison to a droplet's oscillation period, resulting in a fine temporal resolution. Meanwhile, the total length T of the symbol should probably be long in comparison to the droplet's oscillation to allow symbols to consist of more than a single excitation.

$$S = (a_{I_1}, a_{I_2}, ..., a_{I_m})'$$

$$a_i \in \{0,1\}, \quad S \in \{0,1\}^m, \quad m \cdot \Delta t = T$$

Besides stimulating exactly one specific droplet, we can imagine that it might prove sensible to treat a larger number of droplets equally in larger droplet networks. We will denote such a set of equally treated droplets as a *droplet channel* for input and output signals. To communicate a symbol to the network, we could use single or multiple channels.

Consequently, for symbols composed of many channels C, we can extend the signal definition S to a matrix S' that stores the activation state a_{c,I_j} of each channel $c \in C$ for each interval I_j:

		Input			
	Task	**0 0**	**0 1**	**1 0**	**1 1**
	Identity	0 0	0 1	1 0	1 1
	OR	0	1	1	1
Expected	AND	0	0	0	1
Output	NAND	1	1	1	0
$\tilde{o}_{cp}$	XOR	0	1	1	0
	XNOR	1	0	0	1
	Half-adder	0 0	1 0	1 0	0 1

Table 1: Boolean functions that were used as fitness criteria in evolution. Two input and up to two output channels were used.

$$S' = \begin{pmatrix} a_{1,I_1} & a_{1,I_2} & \cdots & a_{1,I_m} \\ a_{2,I_1} & a_{2,I_2} & \cdots & a_{2,I_m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{|C|,I_1} & a_{|C|,I_2} & \cdots & a_{|C|,I_m} \end{pmatrix}$$

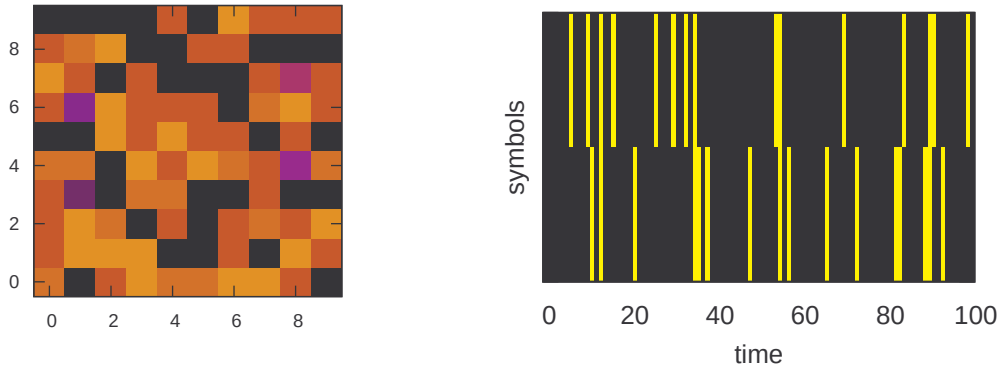
Task Definition

To evaluate the quality of a droplet network and of different symbol encodings, we define Boolean functions that should be fulfilled in terms of their truth tables. As displayed in Table 1, we tested seven different functions with up to two input and output channels.

Fitness Evaluation

Ultimately, the aim of these experiments is to find symbols that can be used by the network internally as input as well as for output. But to evaluate the fitness of a droplet network for binary operations using arbitrary symbols, a metric that determines the similarity between an input symbol and a recorded output excitation stream would be necessary. As discussed in Section 4, choosing an appropriate metric is not trivial. Consequently, we are evolving complex symbol representations to feed into the network but we do not yet expect the network to reproduce these complicated symbols as outputs. Instead we use simple rate coding for the outputs: high activity is interpreted as symbol '1' and low activity as symbol '0'.

The evaluation is divided into distinct phases p by assigning each combination of input symbols to one phase, resulting in four phases for two binary inputs. For each phase p, the system is simulated with the appropriate input



(a) Droplet Network

(b) Input Symbol Representation

Figure 1: Individuals for the Evolutionary Algorithm:
(a) Rendering of an evolved 10×10 droplet network instance. Each square represents a droplet on a two dimensional array. Black cells represent empty spaces that are not excitable. Dark red, orange and yellow cells represent normal, less excitable and long period droplet respectively. The four blue cells represent fixed input / output droplets; input droplets on the left and output droplets on the right. A von Neumann neighbourhood around each cell defines the connectivity of the droplets, i.e. droplets that are directly on top, bottom, left or right are connected and can excite one another.
(b) Example of two symbols that evolved together with a network instance to realise the XOR function. The lower row of the image represents symbol '0' while the upper row represents symbol '1'. Time advances left to right over 100 frames with a time step of 0.5 s, leading to a total length of 50 s per symbol. The input droplets are stimulated only in the intervals that are represented by yellow vertical bars and are left alone where the black vertical bars are rendered. The symbols are fed into the droplet network repeatedly, recapitulating the stimulation pattern every 50 s. At least three oscillation cycles are completed per symbol repetition because the simulated droplets' self-excitation periods are around 16 s. Since droplets are modelled with refractory times, not every yellow stimulation bar will actually lead to an excitation in the droplet but can as well be disregarded in the droplets refractory phases, especially when two excitations follow each other closely.

signals for a fixed time and the number of received excitations at the droplets of output channel c are stored in o_{cp}. We denote the maximal and minimal peak numbers of any channel-phase pair as o_{max} and o_{min}. The symbol that is expected at the output droplets for the channel-phase pair is referenced as $\tilde{o}_{cp} \in \{0, 1\}$ instead.

The final fitness F is influenced by two different aspects, F_1 and F_2, of the output behaviour. First, the normed difference between highly activated and less activated channel-phase pairs should be maximised to allow some kind of discrimination. We define the difference between the maximum and minimum peak numbers divided by the maximum peak number as F_1. F_1 is zero if all peak numbers are equal and at most one when the minimum value is zero. Second, the truth table should be fulfilled, leading to a function F_2. Here, the worst case channel-phase pair defines the overall fitness. Each channel-phase pair peak number should lie as close as possible to the minimum or maximum peak number, dependent on the expected output $\tilde{o}_{cp}$. Finally, if a minimum discriminability is exceeded and also the Boolean function is fulfilled, the distance between minimum and maximum rates should further be expanded.

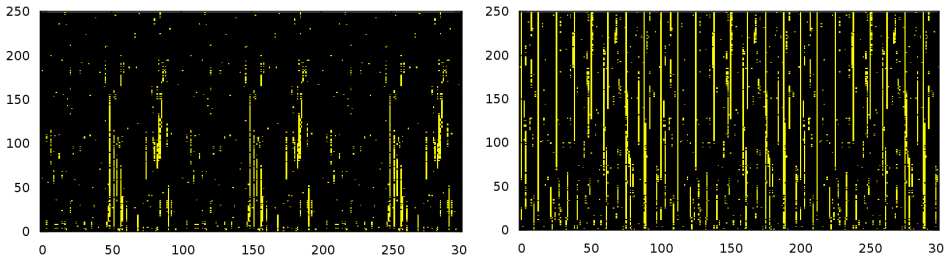
$$F = \begin{cases} F_1 & \text{if } \; F_1 < 0.2 \\ F_2 + 1.0 & \text{if } \; F_1 \geq 0.2 \;\; \text{and} \;\; F_2 < 0.9 \\ F_1 + 2.0 & \text{if } \; F_1 \geq 0.2 \;\; \text{and} \;\; F_2 \geq 0.9 \end{cases} \tag{1}$$

$$F_1 = \frac{o_{max} - o_{min}}{o_{max}} \tag{2}$$

$$F_2 = \min_{c,p} \begin{cases} 1 - \frac{o_{cp} - o_{min}}{o_{max} - o_{min}} & \text{if } \tilde{o}_{cp} = 0 \\ \frac{o_{cp} - o_{min}}{o_{max} - o_{min}} & \text{if } \tilde{o}_{cp} = 1 \end{cases} \tag{3}$$

Experimental Setup

We employed an evolution strategy of the type $(8, 30) - ES$, meaning a comma strategy with 8 parents and 30 children, running for 250 generations where the parents of each generation are discarded. The best symbol representation of each generation of a single experiment is displayed in Figure 2. For each experiment, we ran a batch of 50 evolutionary optimisations to build mean values. In total, we conducted 35 experiments for all the combinations of the seven target functions from Table 1 and the five experimental variations: Network only evolution with three or four droplet types, network and signal coevolution with three or four droplet types and network only evolution with pre-evolved symbol representations. The symbol representation for the pre-evolved signals was taken from the coevolution experiment that achieved the



(a) Symbol '0' (b) Symbol '1'

Figure 2: Evolutionary trajectory of two symbol representations over 250 generations coevolution with a droplet network. The y-axis denotes the evolutionary generation while the x-axis represents the stimulation interval for each fitness evaluation similar to the signal plot in Figure 1b. The regularities that can be observed along the x-axis in both graphics are not evolved regularities but result from the repetition of the pattern: As the pattern of 100 intervals is fed into the simulator during fitness evaluation in a repeated manner, three repetitions of the input signal are plotted over 300 time frames.

best fitness. Using four droplet types means using empty droplets, normal droplets, less excitable droplets and long period droplets, while the latter is discarded for the three droplet type experiments.

For mutating the droplet network, the probability of switching an arbitrary position is 0.05. When using four droplet types, the probabilities for changing to an empty cell, to a normal droplet, to a low-excitability droplet and to a long-period droplet are 0.4, 0.4, 0.1 and 0.1 respectively. For the runs without the long-period droplet type, the remaining probabilities read $\frac{4}{9}$, $\frac{4}{9}$ and $\frac{1}{9}$. Single point crossover recombination is applied with an uniformly chosen position in the row-by-row linearised representation of the droplet network. For mutations of the input signal, the probability of switching an arbitrary position is 0.025. When a mutation occurs, the probability for generating a '1' is 0.1 while a '0' is generated with probability 0.9. Single point crossover recombination is applied with an uniformly chosen position.

3 Results

Small droplet systems of about 100 droplets were arranged by means of evolutionary algorithms to satisfying the Boolean functions Identity, OR, AND, NAND, XOR, XNOR and half-adder. Obviously, some target functions are easier to evolve than others. Using rate coding only, the OR and AND func-

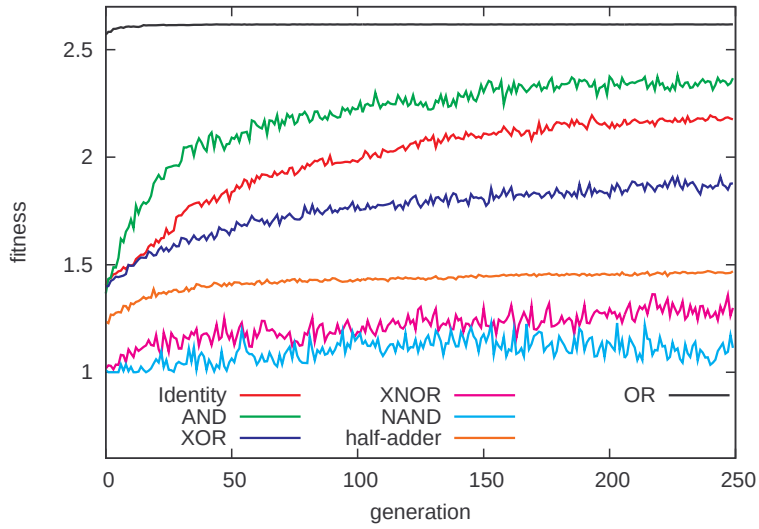
tions evolve fastest, followed by the identity function. Based on the slower fitness increase, functions that involve inversions like the XOR, the half-adder, the XNOR, and NAND functions (cf. Figure 3a) are more difficult.

A network successfully implementing the half-adder functionality did not evolve in our experiments so far. The problem of inverting signals should easily be resolved when using multi-channel symbol representations. Despite these difficulties, even a complicated function like XOR were evolved, even for single channel rate coding signal inputs, albeit not as fast as a simple OR or AND functions. Interestingly, the identity function, meaning a mere connection between both inputs and outputs, is not a simple task compared to AND or XOR when coevolving input signals (cf. Figure 3b). Apparently coevolving networks and symbol representations for the identity function is almost as hard as evolving the half-adder. While using rate coding, in contrast, the identity function evolved faster than the XOR function. Evolution with and without the third droplet type with long oscillation periods did not result in significantly worse evolution progress.

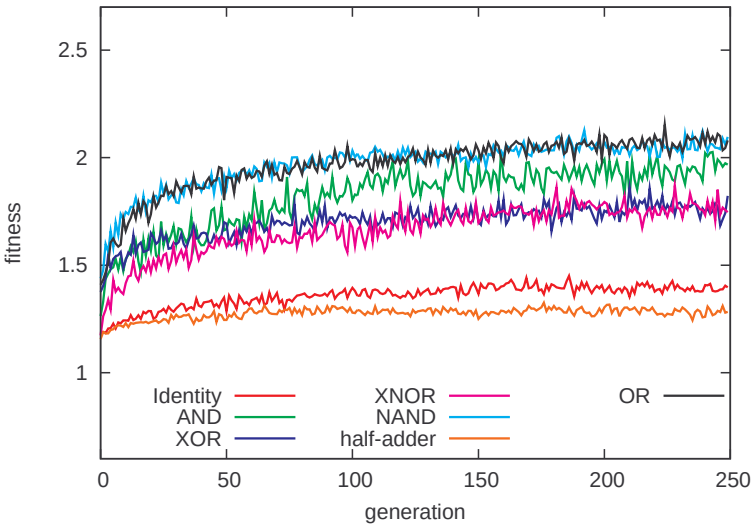
Shown in Figure 4, at least in the case of the AND and XNOR functions, pre-evolved signals exist (cf. Figure 5) that are clearly leading to a faster evolution of droplet networks than simple rate coding. Here droplet networks and signals were originally coevolved. Then one of the best evolved symbol representations was used consistently through a full network-only evolution run. The evolved signal looks similar to rate coding signals but a single activation peak remained for the '0' symbol that might lead to better synchronisation. Further experiments will be necessary to investigate if evolved signals will show the same characteristics repeatedly. An extreme rise in evolution effectivity was observed for the NAND function. However, this is most probably only due to a crosswise substitution of the signals for symbols '0' and '1', such that the problem is reduced to a rate coded OR function. Problems that did not benefit significantly from pre-evolved symbol representations were the OR, the XOR, the Identity function and the half-adder. Nonetheless, the pre-evolved symbols never led to worse evolution trajectories in our experiments.

4 Discussion

Besides designing droplet network structures and symbol encodings, evolutionary algorithms also served another purpose in this work: To some extent, evolutionary algorithms also offer a measure of complexity, telling us whether a problem is simple or hard to solve. Or, given two distinct symbol encodings, which of them makes searching for a solving network structure easier.



(a) Network only Evolution



(b) Network and Signal Coevolution

Figure 3: Average fitness of population's best individual over 50 experiments for evolving different target functions from Table 1. Generally, all fitness values are lower for the signal and network coevolution because of the higher dimensional search space. Exceptions are those functions that benefit from a simple swapping of rate coding signals, i.e. the NAND and XNOR functions.

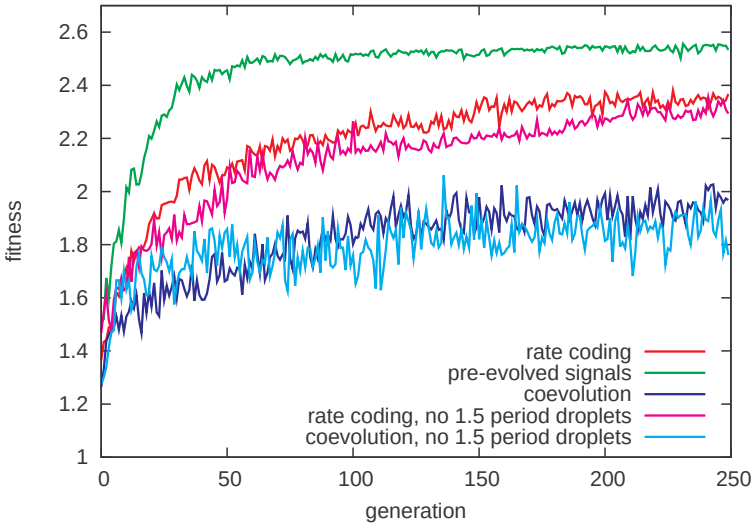
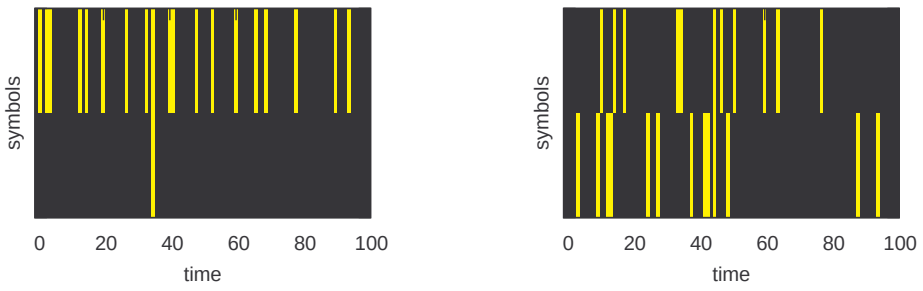


Figure 4: Average fitness of populations' best individual over 50 experiments for evolving the AND function using rate coding, coevolution and pre-evolved symbol representation. Purple and cyan curves correspond to evolution experiments that ignored the fourth droplet type with longer oscillation periods.



(a) Evolved AND Symbols

(b) Evolved XNOR Symbols

Figure 5: Evolved symbol representations for the AND and the XNOR functions that performed better than rate coding. (a) While the AND symbol looks very similar to rate coding symbols, there is one peak included for symbol '0' that might serve as a helper for synchronisation. (b) For the XNOR signals, both symbols are represented by a series of about 30 seconds activation followed by ca. 20 s rest. The difference between both symbol representations could be either in the shift of the active phases of about 10-20 s or in the exact pattern of each signal.

A straight forward construction of two adequate symbols might be to maximise the distance between them. The problem here is to define the distance metric that would heavily influence the result of the maximisation. Ideally these experiments would only depend on the properties of the computing substrate itself and not on arbitrary definitions that are put in from the outside. But any kind of metric like the Hamming distance or the spike train similarity measures from the neurosciences [5] seem sensible but artificial with respect to the computing droplet substrate. A meaningful alternative would be to run a nested evolution of a droplet network simulation as distance metric - the easier it is to evolve a network that discriminates both signals, the larger the distance between both symbols. Still, the computational efforts for a single evaluation of the fitness function appear immense. This led us to the different approach of coevolving signals and droplet networks for simple binary problems at first.

Even though simple logic functions were evolved here, the automatic construction of larger, more complex systems might be hard, especially when fitness functions cannot provide enough gradient for the optimisation algorithm to follow.

The "multi-step" fitness functions that we used in Equation 1 tries to focus different aspects of generating the network functions at different times, dependent on how close to perfect the solution is. But since it is generally impossible to find all non-dominated solution candidates by mapping multiple fitness criteria onto a single scalar value, we will transition to using Pareto optimisation for future experiments [13, 16].

Generally the influence of the droplet network dimensions should be interesting - especially how few droplets can generate the sought-after behaviour, what number of droplet species are essential, is there a preferential length for droplet signal patterns and how many input channels should be used per symbol? Also the aspect of robustness has not yet been in the focus of this work. Nonetheless it appears important if a droplet network and symbol representation led to a high score accidentally or if the performance can be sustained under different initial conditions and with noise.

Acknowledgements

The research was supported by the NEUNEU project (248992) sponsored by the European Community within FP7-ICT-2009-4 ICT-4-8.3 - FET Proactive 3: Bio-chemistry-based Information Technology (CHEM-IT) program.

References

[1] A. Adamatzky. *Computing in nonlinear media and automata collectives.* IOP Publishing Ltd., Bristol, UK, 2001.

[2] S. Aghdaei, M. Sandison, M. Zagnoni, N. Green, and H. Morgan. Formation of artificial lipid bilayers using droplet dielectrophoresis. *Lab Chip*, 8(10):1617–1620, 2008.

[3] J.-P. Banâtre, P. Fradet, J.-L. Giavitto, and O. Michel, editors. *Unconventional Programming Paradigms, International Workshop UPP 2004, Le Mont Saint Michel, France, September 15-17, 2004, Revised Selected and Invited Papers*, volume 3566 of *Lecture Notes in Computer Science*, 2005. Springer.

[4] E. N. Brown, R. E. Kass, and P. P. Mitra. Multiple neural spike train data analysis: state-of-the-art and future challenges. *Nat Neurosci*, 7 (5):456–461, May 2004.

[5] J. Dauwels, F. Vialatte, T. Weber, and A. Cichocki. On similarity measures for spike trains. In *Proceedings of the 15th international conference on Advances in neuro-information processing-Volume Part I*, pages 177–185. Springer-Verlag, 2008.

[6] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing (Natural Computing Series).* Springer, 2008.

[7] J. Gorecki, K. Yoshikawa, and Y. Igarashi. On chemical reactors that can count. *The Journal of Physical Chemistry A*, 107(10):1664–1669, 2003.

[8] G. Gruenert, J. Szymanski, J. Holley, G. Escuela, A. Diem, B. Ibrahim, A. Adamatzky, J. Gorecki, and P. Dittrich. Multi-scale modelling of computers made from excitable chemical droplets. *NEUNEU Technical Report*, 2012.

[9] J. Holley, I. Jahan, B. Costello, L. Bull, and A. Adamatzky. Logical and arithmetic circuits in belousov zhabotinsky encapsulated discs. *Physical Review E*, 84(5):056110, 2011.

[10] J. R. Koza. Hierarchical genetic algorithms operating on populations of computer programs. In N. S. Sridharan, editor, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*

IJCAI-89, volume 1, pages 768–774, Detroit, MI, USA, 20-25 Aug. 1989. Morgan Kaufmann.

[11] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, Nov. 2002.

[12] J. F. Miller, D. Job, and V. K. Vassilev. Principles in the evolutionary design of digital circuits—part i. *Genetic Programming and Evolvable Machines*, 1:7–35, 2000. 10.1023/A:1010016313373.

[13] J. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st international conference on genetic algorithms*, pages 93–100. L. Erlbaum Associates Inc., 1985.

[14] J. Szymanski, J. N. Gorecka, Y. Igarashi, K. Gizynski, J. Gorecki, K.-P. Zauner, and M. D. Planque. Droplets with information processing ability. *International Journal of Unconventional Computing*, 2011.

[15] K. Weicker. *Evolutionäre Algorithmen.* Vieweg+Teubner, 2002.

[16] E. Zitzler, M. Laumanns, and S. Bleuler. A tutorial on evolutionary multiobjective optimization. *Metaheuristics for Multiobjective Optimisation*, pages 3–37, 2004.