Logical Gates and Circuits Implemented in Slime Mould

Andrew Adamatzky, Jeff Jones, Richard Mayne, Soichiro Tsuda and James Whiting

Abstract We overview families of Boolean logical gates and circuits implemented in computer models and experimental laboratory prototypes of computing devices made of living slime mould *Physarum polycephalum*. These include attraction gates, based on chemo-tactic behaviour of slime mould; ballistic gates, employing inertial movement of the slime mould's active zones and a repulsion between growing zones; repellent gates, exploited photo avoidance of *P*. polycephalum; frequency gates, based on modification of electrical potential oscillations frequency in protoplasmic tubes; fluidic gates, where a tactical response of the protoplasmic tubes is used for the actuation of two- and four-input logical gates and memory devices; and circuits based on quantitative transformations which completely avoids spatial propagation, branching and crossings in the design of circuits.

1 Introduction

We overview several families of Boolean gates and circuits: attraction gates [13, 33], ballistic gates [2], repellent gates [19], frequency gates [35], fluidic gates [5], and quantitative transformation circuits [14].

A. Adamatzky (✉) · J. Jones · R. Mayne · J. Whiting
Unconventional Computing Centre, University of the West of England, Bristol, UK
e-mail: andrew.adamatzky@uwe.ac.uk

J. Jones
e-mail: jeff.jones@uwe.ac.uk

R. Mayne
e-mail: richard.mayne@uwe.ac.uk

J. Whiting
e-mail: james.whiting@uwe.ac.uk

S. Tsuda
School of Chemisty, University of Glasgow, Glasgow, UK
e-mail: Soichiro.Tsuda@glasgow.ac.uk

A. Adamatzky (ed.), *Advances in Physarum Machines*, Emergence, Complexity and Computation 21, DOI 10.1007/978-3-319-26662-6_3

In attraction gates [33] the chemo-tactic behaviour of slime mould to sugar gradient is employed to construct Boolean logic gates. Cells are grown and propagated on the agar gel with glucose gradient, just as electrons flow through a wire in the case of electric circuits. If a plasmodial cell comes into contact with another cell, the Physarum plasmodium tend to show following behaviour. If there are opportunities to escape, the plasmodium changes the growth direction and avoids contact with another plasmodium. Otherwise, i.e. no space to escape, it merges with another cell and behave as one single plasmodium cell. This is due to excreted "slime" from another plasmodium that acts as a weak repellent. By combining these two chemotactic behaviours of Physarum slime mould (attraction to sugar and repulsion from excreted slime), slime mould-based AND, OR, and NOT gates are constructed.

In designs of ballistic gates [2] we employ inertia of the Physarum growing zones. On a non-nutrient substrate the plasmodium propagates as a traveling localization, as a compact wave-fragment of protoplasm. The plasmodium-localization travels in its originally predetermined direction for a substantial period of time even when no gradient of chemo-attractants is present. We utilize this property of Physarum localizations to design a two-input two-output Boolean logic gates $\langle x, y\rangle \rightarrow \langle xy, x + y\rangle$ and $\langle x, y\rangle \rightarrow \langle x, \overline{x}y\rangle$. We verify the designs in laboratory experiments and computer simulations. We cascade the logical gates into one-bit half-adder and simulate its functionality.

In experimental laboratory prototypes of repellent gates [19], active growing zones of slime mould representing different inputs interact with other by electronically switching light inputs and thus invoking photo avoidance in each other.

The electrical activity of the tubes oscillates, creating a peristaltic like action within the tubes, forcing cytoplasm along the lumen; the frequency of this oscillation controls the speed and direction of growth. External stimuli such as light and food cause changes in the oscillation frequency. We demonstrate that using these stimuli as logical inputs we can approximate logic gates using these tubes and derive combinational logic circuits by cascading the gates, we can call them frequency gates [35], with software analysis providing the output of each gate and determining the input of the following gate.

Tactile response of the protoplasmic tubes is used for the actuation of two- and four-input logical fluidic gates and memory devices [5]. The tube-based logical gates display results of logical operation by blocking flow in mechanically stimulated tube fragments and redirecting the flow to output tube fragments. We demonstrate how XOR and NOR gates are constructed. We also exemplify circuits of hybrid gates and binary memory devices. The slime mould based fluidic gates are non-electronic, simple and inexpensive, several gates can be realised simultaneously at the sites where protoplasmic tubes merge.

Simulations of more complex combined logic gates and half-adder circuits are demonstrated using a multi-agent model of slime mould [13]. These simulation experiments demonstrated the limiting factors affecting the foraging behaviour of the model plasmodium, particularly at junctions within the gate pattern where choice of growth direction and timing of propagation may be affected. These limitations are compounded when more complex circuits such as the half-adder are used, or when

gates are cascaded. The logical circuits based on quantitative transformations [14] are based on the following ideas. Computing devices are based on spatial arrangements of simple fundamental logic gates. These gates may be combined to form more complex adding circuits and, ultimately, complete computer systems. Implementing classical adding circuits using unconventional, or even living, substrates is made difficult and impractical by the challenges of branching fan-out of inputs and regions where circuit lines must cross without interference. We explore mechanisms to avoid spatial propagation, branching and crossing completely in the design of adding circuits. We analyse the input and output patterns of a single-bit full adder circuit. A simple quantitative transformation of the input patterns which considers the *total number* of bits in the input string allows us to map the respective input combinations to the correct outputs patterns of the full adder circuit, reducing the circuit combinations from a 2:1 mapping to a 1:1 mapping. The mapping of inputs to outputs also shows the same incremental progression, suggesting its implementation in a range of physical systems. We demonstrate an example application, in simulation, inspired by oscillatory dynamics of the true slime mould *P. polycephalum*. This simple transformation may enrich the potential for using unconventional computing substrates to implement digital circuits.

2 Attraction Gates

The first implementation of Physarum plasmodium logic gates [33] employ attracting behaviour of the slime mould [17]. A uni-directional concentration gradient of glucose is formed in the agar media where cells are grown and propagate. The logic gate paths are constructed by limiting the area that plasmodial cells can grow using cut-outs of transparent plastic film. As slime moulds tend to prefer wet regions over dry ones, cells thus grow only in the region that agar is exposed (i.e. logic gate paths). Under this condition, cells are attracted towards areas with higher sugar concentration and interact with other cells in the logic gates, as shown in Fig. 1.

When a Physarum cell comes into contact with another cell, it tends to avoid contacts with other plasmodial cells as "slimes" (gel-like material excreted from a plasmodium) works as weak repellent to other cells. We exploit these two chemotactic behaviours of Physarum slime mould, i.e. attraction to sugar gradient and repulsion from excreted slimes, to construct chemo-attractant/repellent-based logic gates. The attraction logic gates are designed based on following rules:

Rule 1 Physarum cells tend to move towards an area with higher sugar concentration.
Rule 2 When a cell comes into contact with another cell, the cell tends to change migrating directions and avoid contact with another cell if there is a space that have not been occupied by other cells.
Rule 3 Otherwise, i.e. no space to escape, the active growing zone fuses with another cell and becomes one single plasmodium afterwards.

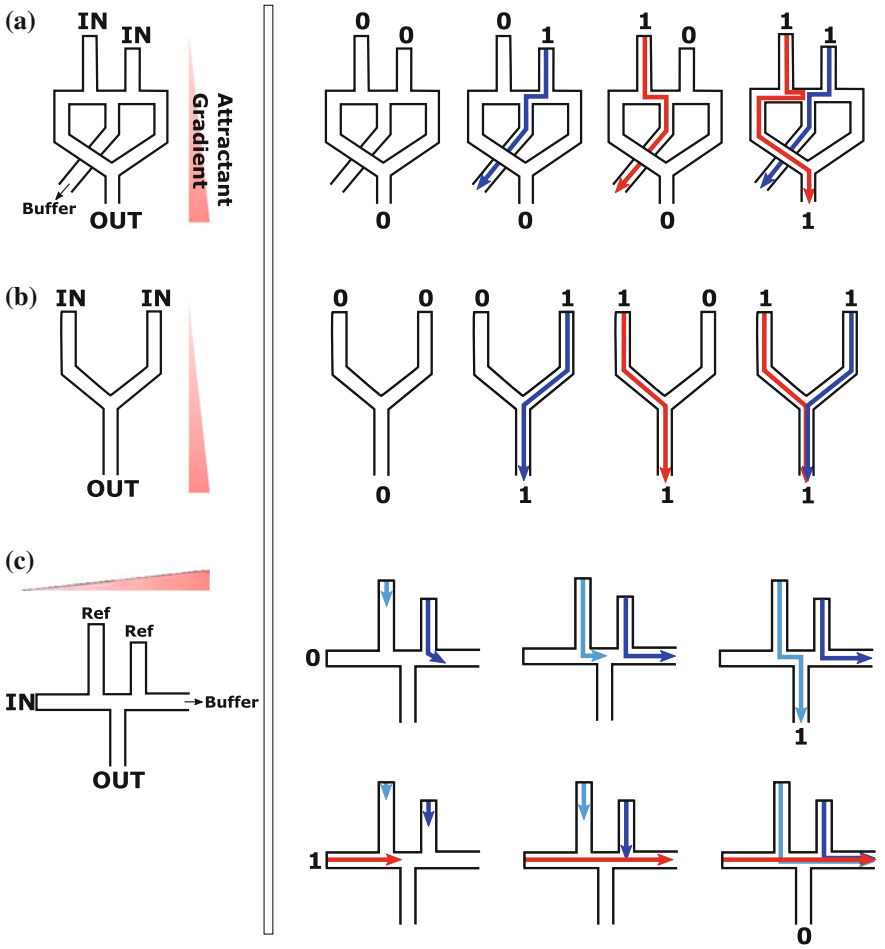

Fig. 1 Physarum attraction logic gates

AND gate (Fig. 1a) produces 1 (i.e., logical True) only when both inputs are 1. Here, 1 and 0 of input or output correspond to the presence and absence of Physarum plasmodium in a specific location, respectively. In the case of inputs (0, 1) or (1, 0), an inoculated cell at an input location migrates along the sugar gradient (Rule 1) and enters a path to a buffer zone. Cells entered in the region are discarded and will not contribute to any computation. The gate is designed that Physarum cells take the route to buffer as it is shortest path to a higher concentration region than the other diverted route, which lead to the output. Thus, (0, 1) and (1, 0) gives 0 as output. In the case of input (1, 1), two inoculated cells migrates in the logic gate paths. However, as one of the input paths has a shorter path (right-hand side in Fig. 1a) than the other, a cell in the path (indicated as arrow in blue) occupies the route to buffer.

When another cell (indicated in red) arrives at the junction, it tends to take a path leading to the output as the path to buffer is already occupied by the first cell (Rule 2). As a result, the input (1, 1) is mapped to output 1, as expected of the operation of AND gate.

OR gate (Fig. 1b) has a rather simple design. The Y-shaped logic gate takes two inputs from the top and has one output from the bottom. If a Physarum plasmodium is present in either one of the two inputs, it just migrates downwards along with the sugar gradient and gives 1 as output. Even when cells are present in both inputs, they merge at the junction as there is no space to escape (Rule 3) and gives 1 as well.

NOT gate (Fig. 1c) is an inverter that gives an output opposite to the input. To implement this gate with the Physarum plasmodia, two additional cells as 'reference" are required. In the case of input 0 (Fig. 1c upper row), a reference cell in the left (arrow in light blue) first arrives at the junction and takes up the path to the buffer (note that the sugar gradient is higher in the right-hand side). When another reference cell (blue) reaches the junction, it can only take the route to the output. On the other hand, in the case of input 1, the input Physarum cell (red) is inoculated first and it migrates straight to the buffer. This blocks the paths for two reference cells, the possible action for the reference cells is only to merge with the input cell (Rule 3) and therefore no output is given.

The attraction logic gates can operate with over 80 % success rate [33]. It was also observed that plasmodium cells changes the tactic behaviour (Rule 1–3) when the gates are broken.

3 Ballistic Gates

Given cross-junction of agar channels, cut from 2–3 mm thick agar plate, and plasmodium inoculated in one of the channels, the plasmodium propagates straight through the junction [2]; the speed of propagation may increase if sources of chemoattractants are present. An active zone, or a growing tip, of plasmodium propagates in the initially chosen direction, as if it has some kind of inertia. Based on this phenomenon we designed two Boolean gates with two inputs and two outputs, see Fig. 2a, b. Input variables are x and y and outputs are p and q. Presence of a plasmodium in a given channel indicates TRUTH and absence—FALSE. Each gate implements a transformation from $\langle x, y\rangle \rightarrow \langle p, q\rangle$. Experimental examples of the transformations are shown in Fig. 2.

Plasmodium of Physarum implements two-input two-output Boolean gate P_1: $\langle x, y\rangle \rightarrow \langle xy, x + y\rangle$.

Plasmodium inoculated in input y of P_1 propagates along the channel yq and appears in the output q (Fig. 2c). Plasmodium inoculated in input x of P_1 propagates till junction of x and y, 'collides' with the impassable edge of channel yq and appears in output q (Fig. 2d). When plasmodia are inoculated in both inputs x and y of P_1 they collide with each other and the plasmodium originated in x continues along the route xp. Thus the plasmodia appear in both outputs p and q (Fig. 2e).

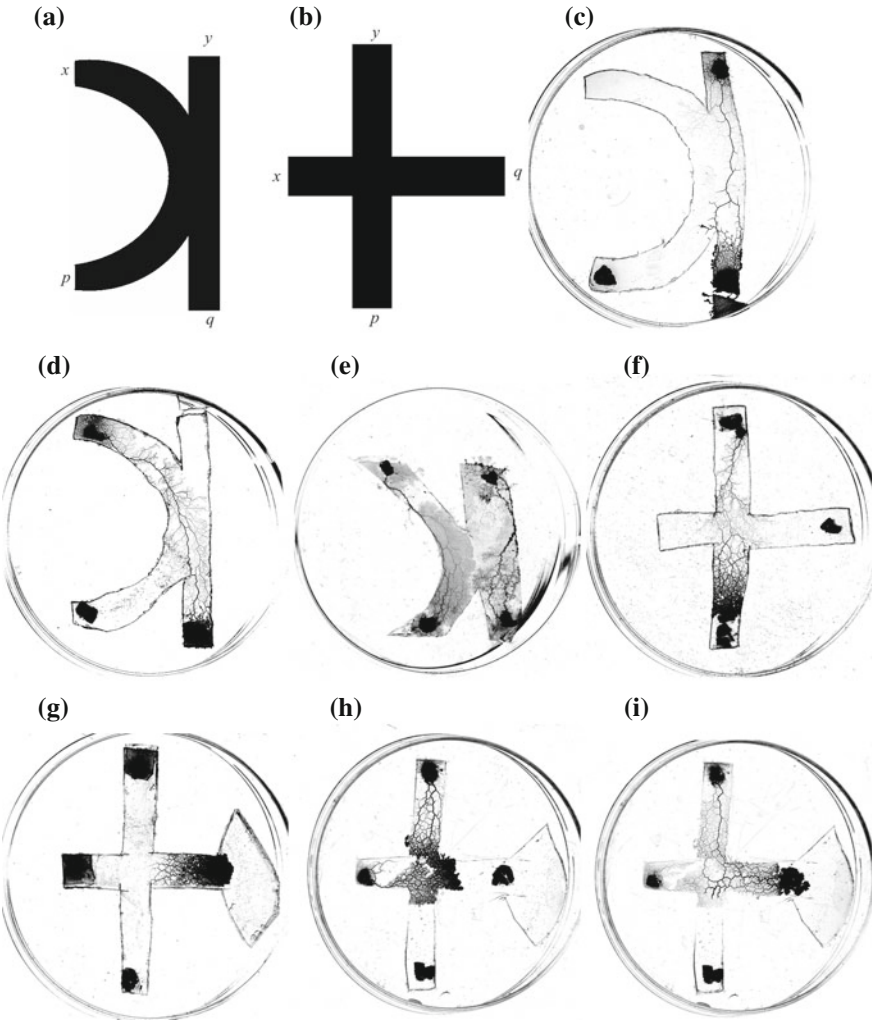

Fig. 2 Physarum ballistic gates. **a**, **b** Geometrical structure of Physarum gates P_1 (**a**) and P_2 (**b**) x and y are inputs, p and q are outputs. **c–e** Experimental examples of transformation $\langle x, y\rangle \rightarrow \langle p, q\rangle$ implemented by Physarum gate P_1. **c** $\langle 0, 1\rangle \rightarrow \langle 0, 1\rangle$. **d** $\langle 1, 0\rangle \rightarrow \langle 0, 1\rangle$. **e** $\langle 1, 1\rangle \rightarrow \langle 1, 1\rangle$. **f–i** Experimental examples of transformation $\langle x, y\rangle \rightarrow \langle p, q\rangle$ implemented by Physarum gate P_2. **f** $\langle 0, 1\rangle \rightarrow \langle 1, 0\rangle$. **g** $\langle 1, 0\rangle \rightarrow \langle 0, 1\rangle$. **h**, **i** Two snapshots (taken with 11 h interval) of transformation $\langle 1, 1\rangle \rightarrow \langle 0, 1\rangle$

Plasmodium of Physarum implements two-input two-output gate P_2: $\langle x, y\rangle \rightarrow \langle x, \overline{x}y\rangle$.

If input x is empty, plasmodium placed in input y of P_2 propagates directly towards output p (Fig. 2f). Plasmodium inoculated in input x of P_2 (when input y is empty) travels directly towards output q (Fig. 2g). Thus transformations $\langle 0, 1\rangle \rightarrow \langle 1, 0\rangle$ and $\langle 1, 0\rangle \rightarrow \langle 0, 1\rangle$ are implemented. The gate's structure is asymmetric, x-channel is

shorter than y-channel. Therefore the plasmodium placed in input x of P_2 usually passes the junction by the time plasmodium originated in input y arrives at the junction (Fig. 2h). The y-plasmodium merges with x-plasmodium and they both propagate towards output q (Fig. 2i). Extension of gel substrate after output q does usually facilitate implementation of the transformation $\langle 1, 1\rangle \rightarrow \langle 0, 1\rangle$.

One-bit half-adder is a logical circuit which takes two inputs x and y and produces two outputs: sum $\overline{x}y + x\overline{y}$ and carry xy. To construct a one-bit half-adder with Physarum gates we need two copies of gate P_1 and two copies of gate P_2. Cascading the gates into the adder is shown in Fig. 3a. Signals x and y are inputted in P_2 gates. Outputs of P_2 gates are connected to inputs of P_1 gates. We did not manage to realise a one-bit half-adder in experiments with living plasmodium because the plasmodium behaved differently in the assembly of the gates than in isolated gates. Therefore we simulated the adder using the Oregonator model, see details in [2]. To simulate inputs $x = 0$ and $y = 1$ we initiated plasmodium's active zones near the entrances to the channels, marked y and arrow in Fig. 3a. The active zones propagated along their channels (Fig. 3b).

For input values $x = 1$ and $y = 0$ active zones are originated at sites marked x and arrow in Fig. 3a. The active zone starting in the left x-input channel propagated towards the $x + y$-output of the adder. The active zone originating in the right x-input channel traveled towards $\overline{x}y + x\overline{y}$ (Fig. 3c). When both inputs are activated, $x = 1$ and $y = 1$, an active zone originated in left y-input channel is blocked by active zone originated in left x-input channels. The plasmodium traveling in the right x-input channel is blocked by the active zone traveling in the right y-input channel. The active zones representing $x = 1$ and $y = 1$ enter top-right gate P_1 and emerge at its outputs xy and $x + y$ (Fig. 3d). The Physarum adder was also implemented in chemical laboratory experiments with excitable chemical system employing Belousov-Zhabotinsky reaction [8].

4 Repellent Gates

As a departure from previously described ballistic logics in which bits—migrating plasmodia—interact with each other in order to perform computation, slime mould may also be adapted into functional electrical logical gates more akin to those found in a conventional computer, i.e. where data interacts with the solid components of the device in order to achieve computation.

This may be achieved relatively easily by capitalising upon the organism's migratory behaviour and resilience to insulting stimuli. More specifically, conventional logical operations may be implemented by conditionally routing plasmodial growth with optical (repellent) inputs between live electrodes: migration of the organism between two electrodes causes an output circuit within the device to become closed (as Physarum is tolerant to having a mild electrical current passed through it), resulting in the device's output equating to TRUTH; when no electrical output is resultant from the slime mould's migratory behaviour, the output is FALSE.

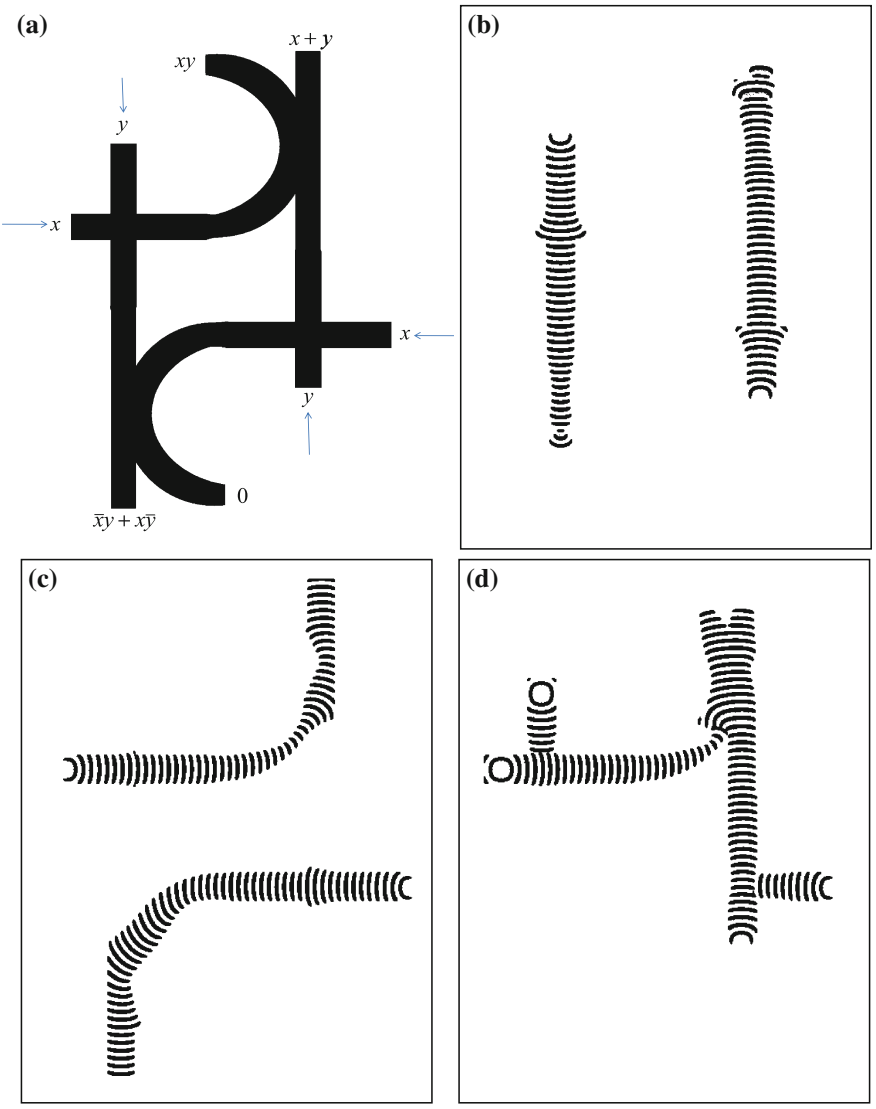

Fig. 3 Simulation of Physarum one-bit half-adder using numerical integration of two-variable Oregonator equations, see details in [2]. **a** Scheme of one-bit half-adder made of gates P_1 and P_2. Inputs are indicated by arrows. Outputs $\overline{x}y + x\overline{y}$ and xy are sum and carry values. Outputs 0 and $x + y$ are byproducts. **b–d** Time-lapse images of plasmodium's active zones traveling in channels of one-bit half-adder. Dynamics of growth is shown for input values **b** $x = 0$ and $y = 1$, **c** $x = 1$ and $y = 0$, **d** $x = 1$ and $y = 1$

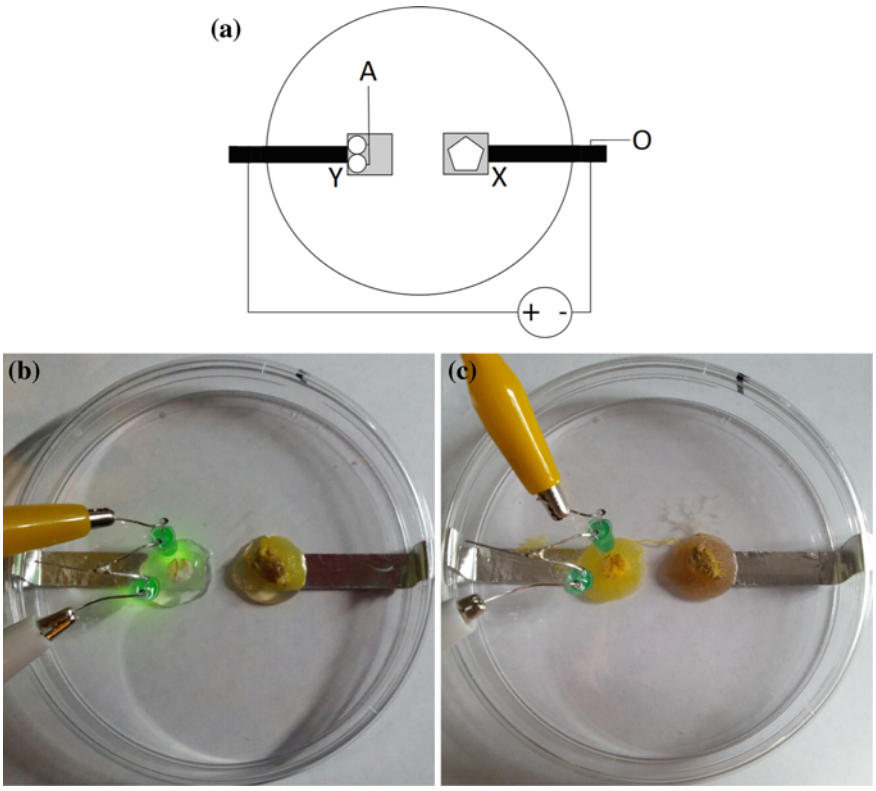

Fig. 4 The PNOT gate. **a** Schematic representation where electrodes (*black rectangles*) and accompanying agar islands (*grey square*) $\langle X, Y \rangle$ are connected to a live output circuit ('O'). Plasmodial homogenate (pentagon) is placed on electrode X and *left* to propagate: if input LED array A is illuminated, it prevents the plasmodium from propagating to electrode Y and closing the circuit. **b–c** Photographs of experimental implementation of PNOT gate. **b** $\langle 0 \rangle \rightarrow \langle 1 \rangle$. Note how the organism has oriented its self about the farthest pole of its agar island away from the repellent. **c** $\langle 1 \rangle \rightarrow \langle 0 \rangle$. The organism has, in the absence of a repellent, migrated to electrode Y

Following a brief scoping study in which the most repellent variety of LED-generated[1] light was ascertained to be green (568 nm, 40 mcd; see Ref. [19] for further details), laboratory prototypes of Physarum NOT and NAND (PNOT/PNAND) gates were fabricated in accordance with these principles. Schemes for each with photographs of functional prototypes are shown in Figs. 4 and 5.

The PNOT gates works as follows. A fresh plasmodial homogenate is added to a 0.5 ml agarose gel (agar) 'island' overlying a 90 × 10 mm aluminium tape electrode stuck to the base of a plastic Petri dish. Another agar island and electrode are present 10 mm away which is loaded with a chemoattractant. An array of two green LEDs—input A—is mounted through the lid of the Petri dish directly above the unoccupied

[1] LEDs were chosen as the repellent input due to their comparative energy efficiency, long life span and low cost of manufacture, all of which are key properties for alternative computing technologies.

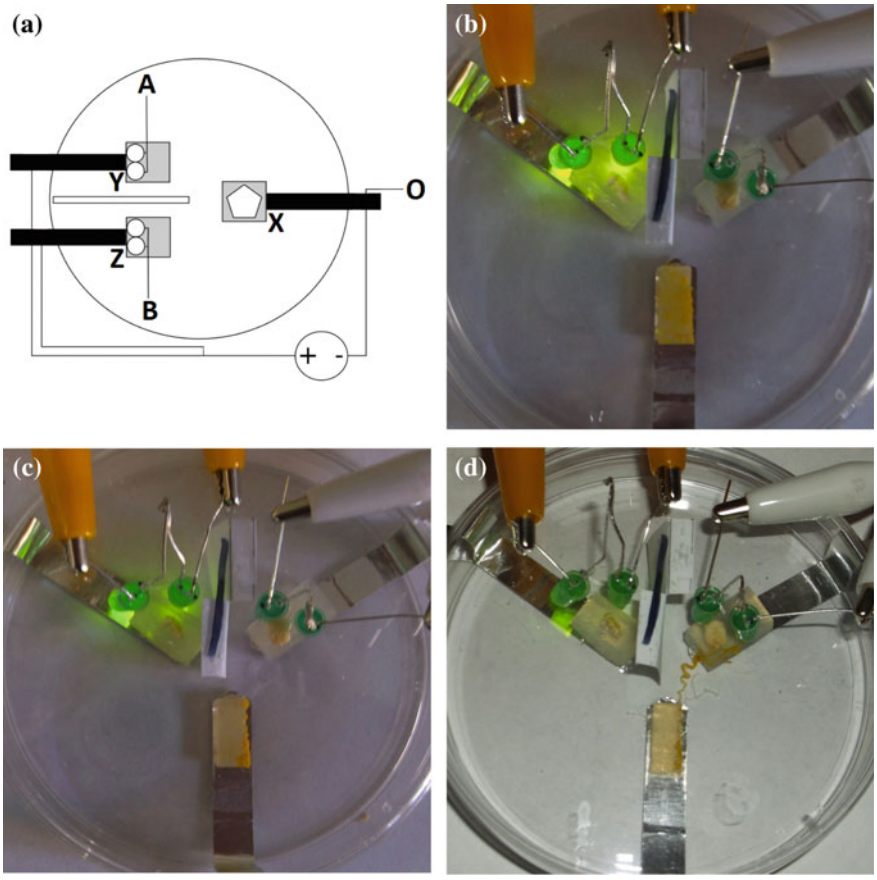

Fig. 5 The PNAND gate. **a** Schematic representation; as in Fig. 5, but with a second input LED array, B, divider to separate LED inputs (*white rectangle*) and third electrode/agar island, Z, present. **b–d** Photographs of device completing $\langle 0, 1\rangle \rightarrow \langle 1\rangle$ operation **b** Time = 0 h. **c** Time = 6 h, the plasmodium has shifted to the right of its agar island away from the illuminated left electrode. **d** Time = 12 h, the plasmodium has migrated to the unilluminated electrode and completed the circuit

agar/electrode. Both electrodes are connected to a separate 'output' circuit being supplied with a constant 9 V, 0.1 A. When input $A = 0$, the plasmodium is free to propagate across the gap between the electrodes and hence closes the output circuit, such that the operation $\langle 0\rangle \rightarrow \langle 1\rangle$ is completed. When $A = 1$, the LEDs illuminate and repel the organism, preventing it from propagating across and closing the circuit, resulting in the operation $\langle 1\rangle \rightarrow \langle 0\rangle$. The device's functionality is therefore equal to that of a conventional NOT gate, i.e. $\langle A\rangle \rightarrow \langle \overline{A}\rangle$.

The PNAND gate operates on the same principles but differs in that it has a second input LED array, input B, and a third agar/electrode device which is wired into the common output circuit. The electrodes are in a spatial arrangement such that the tips of each electrode form the nodes of an equilateral triangle. A card divider is

mounted onto the Petri dish lid to physically separate the LED arrays, thus isolating each input from the other. When both inputs are at 0 state, the plasmodium is free to migrate towards an electrode and complete the output circuit, $\langle 0, 0\rangle \rightarrow \langle 1\rangle$—note that in such an instance, the electrode to which the organism will migrate is entirely random. Equally, when both inputs are at 1, the plasmodium is repelled from both and will not complete the output circuit, resulting in the operation $\langle 1, 1\rangle \rightarrow \langle 0\rangle$. Crucially, if only one input is at 1, the plasmodium will migrate towards the other electrode and complete the output circuit, such that $\langle 0, 1\rangle, \langle 1, 0\rangle \rightarrow \langle 1\rangle$. The device's truth table is therefore identical to that of a NAND gate, i.e. $\langle A, B\rangle \rightarrow \langle \overline{A \cdot B}\rangle$.

To discuss the detriments of this approach to slime mould logic, these devices suffer from extremely long propagation delays which consequently makes cascading extremely difficult to implement. Furthermore, whilst their operation is reasonably consistent from a biological perspective (circa 75 % success rate), they fall far short of the repeatability requirements of electrical components. It is also pertinent to mention that the plasmodium, whilst technically electrically conductive, has a high resistance, which limits the usefulness of any electrical signal passed through the organism and implies energy inefficiency.

PNOT and PNAND gates are not without redeeming features, however. Aside from their value as intellectual curiosities—indeed, it is the singular joy of an unconventional computer scientist to observe a natural system's behaviour in the language of computing—these logical gates demonstrate a slime mould can, via 'programming' with optical inputs, be used to implement computationally universal logic. The gates are made of extremely cheap, readily available materials that utilise virtually no hazardous waste. They were also found to be resettable—i.e. carry out subsequent operations—within a limited time frame. Finally, they are also extremely tolerant to aberrations in certain parameters: for example, increasing the voltage to over 30 Volts was found to have no effect on the devices' operation or the health of the organism. Although they cannot be described as true electrical logical gates due to the disparity in the media through which the 'data' is carried—i.e. the electrical-optical-biomechanical-electrical transitions—they nevertheless exploit the key features of slime mould that make it an ideal unconventional computing substrate; distributed sensing, decision making, actuation and resilience to unfavourable conditions.

With these findings, we may begin to imagine a new generation of computing devices built upon these principles. If, for example, our devices are to continue to rely on plasmodial migration, it has been found that the organism may be hybridised with a range of metallic nanoparticles in order to significantly decrease its electrical resistance [21]. But, when one considers the propagation delay as the major failing of the devices presented, it is clear that the most effective step would be to minimise the role of physical movement in future devices. This could be achieved via automated computer interpretation of the bioelectrical phenomena that ensue following plasmodial stimulation, as was capitalised upon in Ref. [22], in which a basic tactile sensor was realised via an FPGA-based interface which measured and interpreted the alterations in membrane potential that result from insulting stimuli which were, crucially, extremely rapid when compared to the measurement of migratory behaviour.

5 Frequency Gates

The plasmodial phase of the organism was cultured using the same method as mention in the previous section, using 2 % non-nutrient agar, and daily feeding with rolled oat flakes; enough stock culture could be produced on several Petri-dishes worth of agar as long as they were maintained.

Data on Physarum's response to stimuli was collected for previous research and is described in full in [36]. The data collected is processed and presented here (Table 6) in order to derive additional Physarum based gates. Figure 6 shows the experimental set up in order to produce and measure a single protoplasmic tube. 1 ml of non-nutrient Agar is placed on each of the aluminium electrodes (Farnell, UK) in a customised 9 cm Petri dish (Fisher Scientific, UK) to form a cell interface. A Physarum inoculated oat flake from culture is placed on 1 agar hemisphere while a bare oat flake is placed on the remaining agar hemisphere. The agar acts as a growth medium for the organism on the electrode. After a minimum of 5 h and maximum of 12 h, a single protoplasmic tube grows between the two electrodes, allowing recording of the surface potential of the tube. Electrical measurement of the protoplasmic tube were performed by connecting the aluminium electrodes to a PicoLog ADC-24 high resolution analogue-to-digital data logger (Pico Technology, UK) connected via USB to a laptop installed with PicoLog Recorder software for data capture. The PicoLog ADC-24 recorded ± 39 mV at 1 Hz for the duration of the experiment, with a 24 bit resolution; the originally inoculated agar hemisphere was connected to ground, while the newly connected agar hemisphere was connected to an analogue recording channel. Stimulation of the organism was performed by adding an oat flake on the recording electrode or by heating the recording electrode to 10 °C above room temperature using a 1.4 W Peltier element (RS Components, UK) placed underneath the Petri dish at the site of the recording electrode. Simultaneous heating and oat flake addition was also performed. The 10 min period before stimulation was used as the baseline frequency measurement (fpre) and the 10 min after stimulation had started was the frequency change (fpost); relative frequency change (Δf) was calculated and expressed as a percentage.

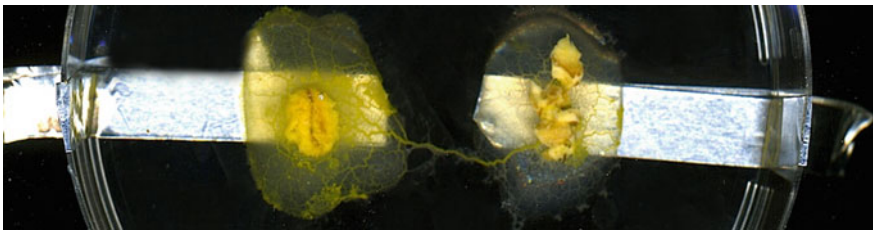

Fig. 6 An example of Physarum protoplasmic tube grown between agar hemispheres

Logic gate approximation as described in [35] uses frequency change of shuttle streaming to determine a logic true or false output. A logic gate is a single protoplasmic tube of Physarum (Fig. 6), which is stimulated with combinations of inputs, light, oat flakes or heat. The frequency change is measured by custom Matlab software which performs a frequency analysis before and after the stimuli, determining the logical output; the type of gate used determines the thresholds for logic 1 or 0, as shown in Table 1. Using the principals of combinational logic, this paper uses the basic gates to produce derived gates NOR, NAND, XOR and XNOR, as well as more complex combinational logic circuits. Combinational logic circuits are cascaded manually, with the software output detailing the input for the following gate which is performed manually; in the future it is envisioned that this process can be totally automated. For the logic gate inputs A and B, the stimuli heat and oat flake was applied respectively. While A and B are both false, or logic L0, neither the heat nor the oat flake is applied, while A and B are both true, or logic L1, both stimuli are simultaneously applied. The relative frequency change for each gate and the classification for the previously produced simple logic gates are shown in Table 1.

The exclusive OR (XOR) gate is commonly used in binary adders and other logic circuits; the output is high if either input is true but not both, otherwise the output is false. A frequency change system can be deduced using two thresholds, in a similar manner to that proposed in [35]. Frequency change of between 4.9 and 32 % (inclusive) is logical True or 1, a change of either less than 4.9 % or greater than 32 % is a logical False or 0 (Table 1).

While the AND, OR and XOR gates calculate specific outputs, they can be inverted by, producing NAND, NOR and XNOR gates respectively. These gates normally have NOT gates at each input or a single NOT gate at the output, so are in essence combinational logic. Hence the frequency system for each gate can be simply modified by inverting the threshold categories (Table 1), for example, an OR gate is high when the frequency change is greater or equal to 10 %, and low when less than 10 %; a NOR gate is low when the frequency change is greater or equal to 10 %, and high when less than 10 %. Alternatively the inputs can be inverted, and the inputs to the OR gate are high when present, that is, when an oat flake and heat are on, whereas for the NOR gate, the inputs are high when the oat flake and heat are not present; this becomes more useful when some inputs are inverted and others are not as in the 2–4 bit decoder (Fig. 7).

Table 1 Type of logic gate is determined by the upper and lower frequency change threshold

Gate type	Lower threshold	Upper threshold
OR	$\Delta f < 10\,\%, L0$	$\Delta f \geq 10\,\%, L1$
AND	$\Delta f < 24\,\%, L0$	$\Delta f \geq 24\,\%, L1$
NOT	$\Delta f < -5.5\,\%, L0$	$\Delta f \geq -5.5\,\%, L1$
NOR	$\Delta f < 10\,\%, L1$	$\Delta f \geq 10\,\%, L0$
NAND	$\Delta f < 24\,\%, L1$	$\Delta f \geq 24\,\%, L0$
XOR	$\Delta f < 4.9\,\%, L1$	$\Delta f \geq 32\,\%, L1$
XNOR	$\Delta f < 4.9\,\%, L1$	$\Delta f \geq 32\,\%, L0$

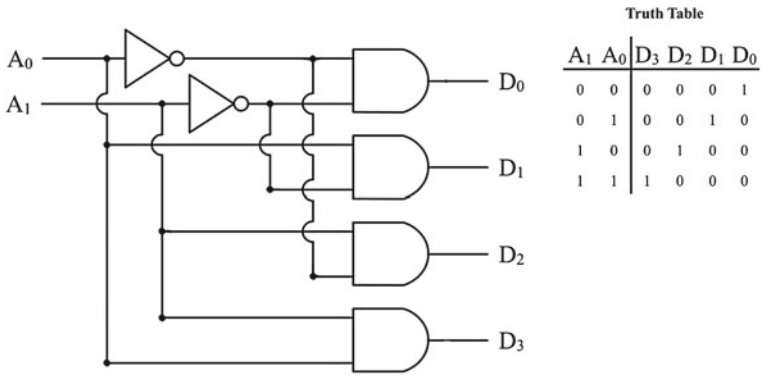

Fig. 7 The 2–4 bit decoder logic circuit with truth table

With the production of NAND and NOR gates, functional completeness is achieved using Physarum frequency gates (PFGs); any possible combination of logic gates may be produced using a combination of NAND (and NOR) gates. Combinational circuits comprising several individual PFGs could be produced; the input stimuli can be automatically controlled using a microprocessor. The output of the PFG is measured using custom Matlab software consisting of data handling and a fast Fourier transform; this could be implemented on a microprocessor with analogue to digital converter and appropriate software to automatically determine frequency change hence logic output of one PFG and subsequently control the input of the next gate in the sequence. In this instance however, the frequency change was calculated using the Matlab software and the subsequent inputs were controlled manually, although frequency analysis software or hardware could be tasked with this process in the future. The response for each stimuli type used for logic gate approximation was tested for normality. With the knowledge of frequency change distribution for each input derived from previous data [35], the probability and hence accuracy of the NOR, NAND, XOR and XNOR gates were calculated using the distributions and likelihood of either type I or II error for each input; this method accounted for the variation in response to the stimuli. In addition to the derived gates, accuracy of the half adder, full adder and 2–4 bit decoder combinational logic circuits were also calculated. For the gates which had one inverted input and one normal (non-inverted) input such as in the 2–4 bit decoder (Fig. 7), one stimuli/input state was inverted while the other one was not.

The accuracy for the inverted gates is demonstrated in Table 2; With the same accuracy as the non-inverted gates presented previously [35] as the frequency change boundaries were inverted. PFGs and their inverted input have identical accuracy because only the thresholds were inverted. The accuracy of the half adder, full adder and 2–4 bit decoder are listed in Table 2, with the least accurate being the 2–4 bit decoder.

Table 2 Accuracy of combinational logic using multiple PFGs

Logic operation	Correct output (%)	Number of PFGs required 1
OR/NOR	90	1
AND/NAND	77.8	1
NOT	91.7	1
XOR/XNOR	70.8	1
24 decoder	57.5	4
Half adder	65	2
Full adder	58.8	5

The frequency change when exposed to the heat and oat stimuli is repeatable and of similar magnitude (Fig. 6) we can use this reliable change to approximate Boolean logic with logic 0 and 1 if the frequency change from these stimuli is within certain value ranges. Boolean logic operation OR is implemented with a threshold of 10 % increase in frequency while AND uses a threshold of 24 % increase in frequency. A NOT gate can be implemented when light is used as an input, with white light representing the input and using a threshold of −5.5 % frequency change. This information is summarised in Table 1. The logic OR, AND and NOT gates derived in this paper demonstrate that logic functions in the slime mould can be performed accurately, orders of magnitude faster than the growth based logic implementations. Tsuda originally implemented the growth based Physarum logic gates with OR, AND and NOT gates giving 100, 69 and 83 % accuracy respectively, values which are comparable to the accuracy of frequency based logic operations presented in this chapter (Table 1). Until now, Physarum logic computation used growth and migration [13, 31]. These computations took several hours to complete due to the slow rate of organism growth. Frequency-change based logic implementation is significantly faster than any performed using growth as the calculation, with calculations lasting between 20 and 30 min. The main advantage of this electrically recorded implementation is the speed of processing; it has been previously reported that while the migration response of Physarum to stimuli is slow with speeds of up to 5 cm per hour [6], electrically recorded responses to chemical, mechanical and optical stimuli are immediate [3, 4, 35, 36]. The Physarum logic gates are designed by interpretation of results where Physarum processes inputs and closely describes the output by way of frequency change. OR and AND logic gates may closely be approximated when using the combined effects of Oat and Heat as inputs x and y, the NOT logic gate uses Light as an input this is tested using obtained data of frequency response using light as a single stimuli. Thresholds are implemented in order to divide the categories of logic 1 and logic 0. With a marginal overlap of the frequency changes of different stimuli, there is some error when approximating the logic outputs. Table 2 highlights the accuracy of the logic operations when using the frequency change as a basis for Boolean logic operation and using the threshold, as defined above for each gate. The accuracy was determined by the number of correct outputs using Physarum's

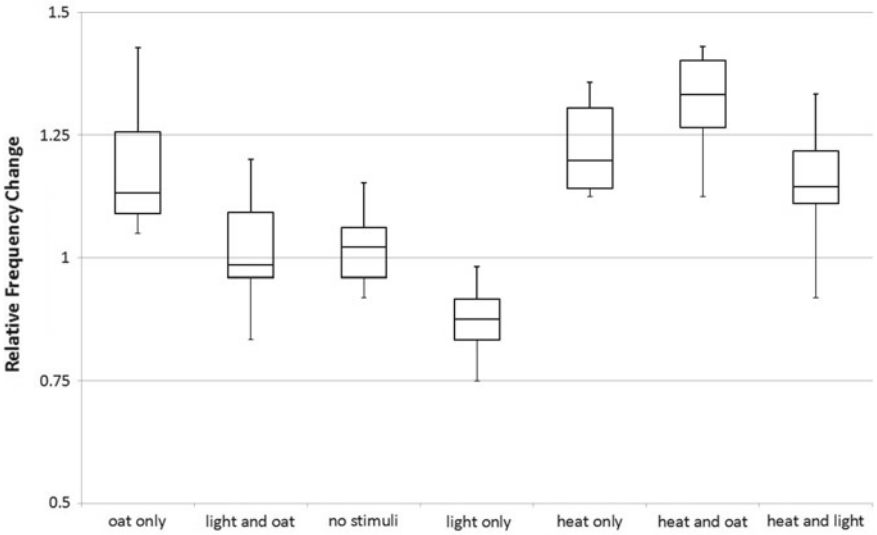

Fig. 8 A box-and-whisker plot of frequency changes when testing different stimuli

frequency change, displayed as a percentage, for all 4 input combinations of x and y; each combination was repeated 12 times for each gate type. A correct output was given when the frequency change as a result of the stimuli fell the correct side of the threshold (Fig. 8).

With the addition of the NAND, NOR, XOR and XNOR gates, there is now a complete database of basic and derived logic gates using frequency change. The number of inputs to these gates is limited to 2, due to the number of tested stimuli, however both NAND and NOR gates have functional completeness, which is to say they can be combined to produce any other logical operation, including single gates with more than 2 inputs. Multi-NAND ICs are often only used in practical systems to limit the number of different chips required in a system, as multiple gates of the same type are produced on CMOS or TTL chips. The fact that the architecture of a PFG is the same regardless of gate type used, means they are effectively programmable logic gates, with the gate type being determined purely by the boundary conditions of the frequency change. The PFGs only have two inputs, however gates with more than 2 inputs can be approximated with more gates.

The number of PFGs used to solve a logical operation correlates with accuracy as demonstrated by Fig. 9; the 2–4 decoder and full adder use 4 and 5 PFGs respectively and have significantly more error than the logic with 1 or 2 PFGs. The trend is not linear as the error is cumulative in a system, it is evident that logic operations with gates higher than these presented would have an accuracy no better than tossing a coin. The layout of the gates also plays a role in accuracy, as a full adder which has 1 more gate than a 2–4 decoder is marginally more accurate, this is due to the series layout of the logic gates; an error produced from one gate has a chance of being

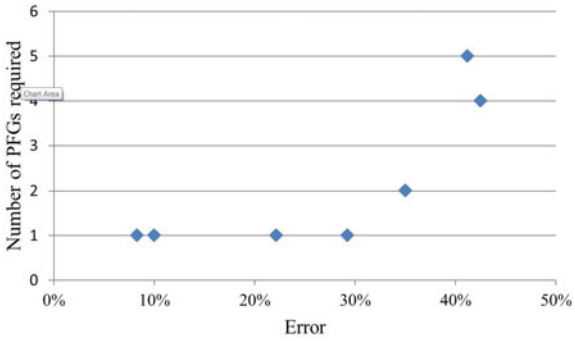

Fig. 9 The correlation between the number of PFGs required to calculate a logical problem and the error rate

coincidentally corrected by another error in the subsequent gate whereas parallel gates such as those in the 2–4 decoder are only correct if all gates produce the correct answer. In a logic system, a correct output is most important, even if the correct operation throughout the circuit is not, as in this system. The speed of processing of a PFG is 30 min for a single gate, decreasing the computation time by approximately 30 times compared to morphological, or growth based, gates [13, 31]. We have shown that the basic logic gates are as accurate as those shown in those previous studies [13, 31]. The half adder shown is similarly accurate (65 %) to that simulated by Jones and Adamatzky (63 %) [13, 31].

6 Fluidic Gates

The protoplasmic networks developed by Physarum are living self-growing microfluidic systems [16, 18, 27] capable of intake and controllable delivery of biocompatible materials [1, 20]. The slime mould microfluidic systems can range in size from a few millimetres to meters of complex protoplasmic networks with hundreds of interconnected tube fragments. To be used efficiently the protoplasmic networks must be controlled and a flow of cytoplasm transporting objects must be programmed. In 2004 Vestad, Marr and Munakata [18, 34] constructed logical gates by changing functional properties of a fluidic system without resorting to non-linear properties of a liquid. They showed that by dynamically changing resistance of individual channels in a microfluidic system it is possible to direct overall relative system of flow rates, independently of the pressure of the liquid. Their logical gates are actuated by depressing one channel of the system and reconfiguring the network [34]. Being inspired by Vestad-Marr-Munakata results we conducted laboratory experiments with slime mould Physarum and found that when a fragment of protoplasmic tube is mechanically stimulated a cytoplasmic flow in this fragment halts and thus resistivity increases. The cytoplasmic flow is then directed through adjacent protoplasmic tubes. We explored this phenomenon to construct several logical gates and a memory device [5].

An undisturbed Physarum exhibits more or less regular patterns of oscillations of its surface electrical potential. The electrical potential oscillations are more likely controlling a peristaltic activity of protoplasmic tubes, necessary for distribution of nutrients in the spatially extended body of Physarum [10, 25]. A calcium ion flux through the membrane triggers oscillators responsible for controlling the dynamic of contractile activity [9]. Physarum surface electrical potential oscillates with an amplitude of 1 to 10 mV and period 50–200 s, associated with shuttle streaming of cytoplasm [11, 15]. Oscillations of the electrical potential and the corresponding peristaltic activity are due to calcium waves propagating along protoplasmic tubes. These waves, and associated electrical charges and a difference in electrical potential leads to a flow of cytoplasm.

In any given tube cytoplasmic flow reverses its direction approximately every 54 s [5]. We can speculate this is because calcium and peristaltic waves propagate from a root, an inoculation site, of a Physarum tree towards its leaves (growth zones) and then back. That is, a protoplasmic tree is polarised and its polarisation is reversed almost every minute.

When a segment of a protoplasmic tube, between two junctions, or branching points, is touched with a hair a flow of cytoplasm inside this fragment becomes blocked. The blockage of a cytoplasmic flow could be due to K^+ channel activation, increase in intracellular Ca^{2+}, temporary increase in concentration of inositol trisphosphate, activation of adenylyl cyclase. A mechanically stimulated fragment restores its conductivity and flow of cytoplasm in 54–59 s after the stimulation.

Plasmodium of Physarum is cultivated in plastic containers, on paper towels sprinkled with distilled water and fed with oat flakes (Alnatura Haferflocken, Feinblatt, Germany). Experimental substrate is 2 % non-nutrient agar gel (Agar-Agar, Kobe I, pulv. Carl Roth, Germany) poured in 9 cm plastic Petri dishes. In each experiment an oat flake colonised by plasmodium is placed in the centre of the Petri dish. Protoplasmic tubes were mechanically[2] stimulated with a human hair approximately 50 μm in diameter, 4–5 cm in length. A tip of hair was forced into a wall of a protoplasmic tube till temporary invagination and/or immediate stoppage of cytoplasmic flow occurred. Videos of cytoplasmic flows were recorded using digital high-resolution microscope Keyence VH-Z20R (KEYENCE Microscope Europe) at zoom × 200.

When a fragment of a tube becomes blocked, a flow of cytoplasm is directed through auxiliary, or second-order, bypassing tubes. Main, or first order, protoplasmic tubes have diameter c. 100 μm while auxiliary, second order, tubes have diameter 30–40 μm. In intact Physarum tree, a flow of cytoplasm is directed along a route with lowest resistance, i.e. along first-order tubes whose diameter is large. Tubes with a small diameter act as a reserve, or emergency, route for situations when large diameter tubes are damaged or a flow is blocked. We use this phenomenon to design logical gates. A detailed example of an XOR gate is shown in Fig. 10 and its scheme in Fig. 11a–d.

[2]Experiments are done by Theresa Schubert, Bauhaus University, Weimar, Germany.

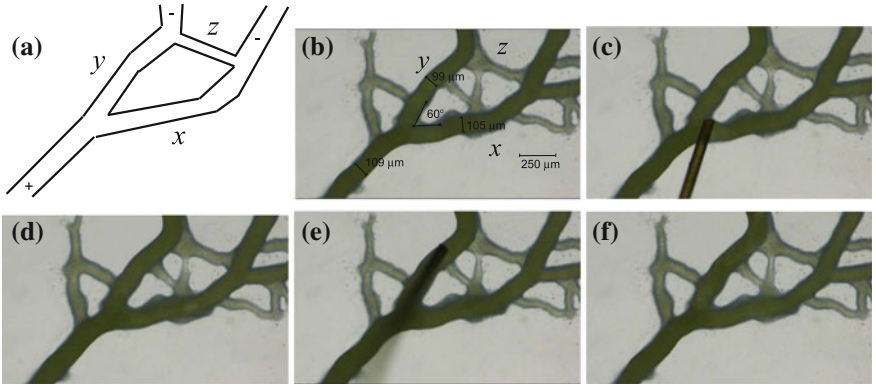

Fig. 10 Implementation of XOR gate in Physarum. **a** scheme of the gate, input tubes x and y and output tube z are shown; '+' and '−' indicate polarity of cytoplasm flow, the polarity is changing almost every minute. **b** snapshot of the living gate before stimulation: $x = 1$ and $y = 1$, $z = 0$; some parameters of the junction are indicated. **c** mechanical stimulation of tube x, $x = 0$. **d** gate after stimulation, $x = 0$, $y = 1$ and $z = 1$. **e** mechanical stimulation of tube y, $y = 0$. **f** gate after stimulation $x = 1$, $y = 0$, $z = 1$. Experimental photos courtesy of Theresa Schubert. From [5]

Flow is directed from '+' to '−' and then reversed from '−' to '+' (Figs. 10a and 11a). First order tubes x and y represent input Boolean variables. Second order tube z represents an output variable (Figs. 10a and 11a). If there is a flow of cytoplasm in a tube the tube represents state TRUE, if there is no flow state FALSE. In an intact, or resting, state the gate's inputs are in state '1', tubes x and y exhibit flow of cytoplasm and tube z does not exhibit a flow: $x = 1$, $y = 1$, $z = 0$ (Figs. 10b and 11a). This is because tube z's diameter, c. 30 µ, is nearly three times smaller than the diameter of tubes x and y, c. 100 µm.

When tube x is touched, the moment of this mechanical stimulation is shown in Figs. 10c and 11b, tube x becomes 'non-conductive' and flow through the tube x stops. Subsequently a pressure in the cytoplasm increases and the cytoplasm is directed through tube z, which diameter increases to 70 µm due to pressure from the passing cytoplasm (Figs. 10d and 11b).

The gate remains in such state for 54 s in average and then tube x restores its conductivity. Flow of cytoplasm is then directed through tubes x and y, tube z becomes unused, shrinks due to elasticity and its diameter returns to a resting value 30 µm. This is somewhat analogical to an automated adjustment employed a microfluidic implementations of Wheatstone bridge [30]. State of the gate after mechanical stimulation of tube y (Fig. 10e) is shown in (Figs. 10f and 11c). The gate restores its original state in less than a minute after mechanical stimulation. When a flow stops in x and y at the same time the flow may not occur in the tube z because the tube becomes isolated from an upper part of protoplasmic network. Therefore we assume that $z = 0$ if $x = 0$ and $y = 0$. Thus XOR gate is implemented $z = x \oplus y$ (Fig. 11a–d).

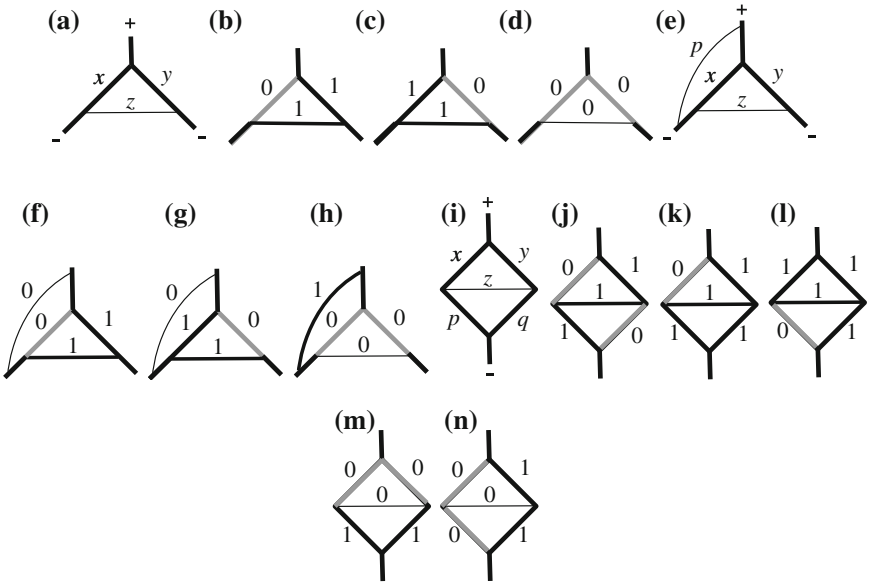

Fig. 11 Schematics of gates implementable with Physarum tubes. Tubes x and y are *solid black* when represent logical TRUE, $x = 1$, $y = 1$, and *grey* when represent logical FALSE, $x = 0$, $y = 0$. Tubes z and p are thin when represent FALSE, $z = 0$ and $p = 0$, and they are sick when represent TRUE, $z = 1$ and $p = 1$. Symbols '+' and '−' indicate polarity of cytoplasm flow, the polarity is changing almost every minute. **a–d** XOR gate, discussed in Fig. 10, $z = x \oplus y$, where **a** $x = 1$, $y = 1$, **b** $x = 0$, $y = 1$, **c** $x = 1$, $y = 0$, **d** $x = 0$, $y = 0$. **e–h** XOR and NOR gates: $z = x \oplus y$ and $p = \overline{x + y}$, where **e** $x = 1$, $y = 1$, **f** $x = 0$, $y = 1$, **g** $x = 1$, $y = 0$, **f** $x = 0$, $y = 0$. **i–n** combined gate: $z = \overline{x}yp + x\overline{y}q + x\overline{p}q + yp\overline{q}$, where **i** $x = 1$, $y = 1$, $p = 1$, $q = 1$, **j** $x = 0$, $y = 1$, $p = 1$, $q = 0$, **k** $x = 0$, $y = 1$, $p = 1$, $q = 1$, **l** $x = 1$, $y = 1$, $p = 0$, $q = 1$. **m** $x = 0$, $y = 0$, $p = 1$, $q = 1$, **n** $x = 0$, $y = 1$, $p = 0$, $q = 1$. From [5]

By adding one more second order tube to gate XOR (Fig. 11a–d) we produce a gate with two inputs and two outputs (Fig. 11e–h). The gate is shown in (Fig. 11e–h). It computes exclusive disjunction and negated disjunction in parallel. Output tube $z = x \oplus y$ acts in a manner similar to XOR gate (Fig. 11a–d). Output tube $p = \overline{x + y}$ connects inlet to tube x, just before junction of x and z, to outlet of junction of the tubes x and y. Cytoplasmic flow is directed via tube p, $p = 1$ only if tubes x and y are blocked, $x = 0$ and $y = 0$ (Fig. 11h).

By adding two more first order tubes to gate XOR (Fig. 11a–d) we produce a gate with four inputs and one output (Fig. 11i–n). The gate $z = \overline{x}yp + x\overline{y}q + x\overline{p}q + yp\overline{q}$ (Fig. 11i–n) responds with value TRUE only when one input tube is blocked yet two of its neighbouring input tubes are unblocked. Examples are as follows. Tubes x and q are blocked, $x = 0$ and $q = 0$, tubes y and p are unblocked, $y = 1$ and $p = 1$ (Fig. 11j), flow is directed via tube z. Tubes y, p, q are unblocked, $y = 1$, $p = 1$, $q = 1$, tube x is blocked, $x = 0$ (Fig. 11k), flow is directed via tube z. Tubes x, y and q are unblocked and tube p is blocked (Fig. 11l), flow is directed via tube z. For all other combinations of input tuples output is FALSE. Examples are as follows. Tubes

x and y are blocked and tubes p and q are unblocked (Fig. 11m), there is no flow of cytoplasm through the gate and thus $z = 0$. Tubes x and p are blocked and tubes y and q are unblocked (Fig. 11n), cytoplasm is flowing through tubes y and q and thus $z = 0$.

7 Gates and Circuits Implemented Only in Simulation

7.1 Modelling Complex Logical Gates

In the paper by [31] some output channels of Physarum gates were considered as buffers. Let us now slightly redesign the gates in [31] and interpret all outputs of the gates as Boolean logic values [13].

Consider G_1 gate in (Fig. 12a). Physical structure of the gate satisfies the following constraints $|xb| = |yc|$ and $|bd| > |bc| + |ce|$ (Fig. 12a). Chemoattractants are placed in sites d and e. We assume strength of attraction to d (e) at point p is proportional to distance $|pd|$ ($|pe|$) (Fig. 12a).

Situations corresponding to input values (0, 0), (0, 1) and (1, 0) are simple. When no plasmodia are inoculated in x and y nothing appears at outputs d and e (Fig. 12b). When plasmodium is placed only in site y the plasmodium follows the route $(yc)(ce)$ (Fig. 12c). If plasmodium inoculated only in site x the plasmodium follows the route $(xb)(bc)(ce)$ (Fig. 12d).

The main novelty of the gate is in how input values $x = 1$ and $y = 1$ are handled. The plasmodia are inoculated in sites x and y (Fig. 12d). The plasmodium growing from site y follows route $(yc)(ce)$. The plasmodium growing from site x tends to follow route $(xb)(bc)(ce)$, however part of the route (ce) is already occupied by another plasmodium. Therefore the plasmodium, starting in x, grows along the route $(xb)(bd)$ (Fig. 12d).

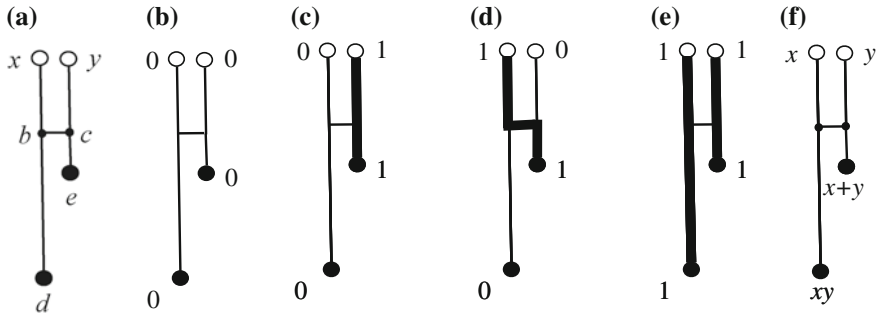

Fig. 12 Scheme of G_1 gate: **a** landmark points are shown; **b–e** configuration of plasmodia in gates for all combinations of input values—$x = 0$, $y = 0$ **b**, $x = 0$, $y = 1$ **c**, $x = 1$, $y = 0$ **d**, $x = 1$, $y = 1$ **e**, the plasmodia bodies are shown by *thick lines*; **f** input-output logical function realized by the gate. Chemoattractants are placed in sites marked by *solid black discs*

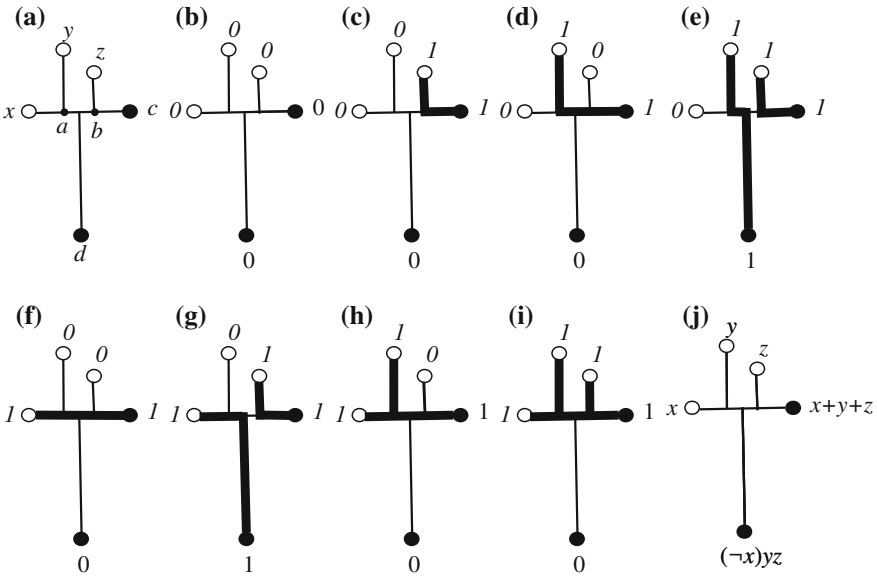

Fig. 13 Scheme of G_2 gate. **a** landmark points are shown; **b–i** configuration of plasmodia in gates for all values of input tuple $\langle x, y, z\rangle$: **b** $\langle 000\rangle$, **c** $\langle 001\rangle$, **d** $\langle 010\rangle$, **e** $\langle 011\rangle$, **f** $\langle 100\rangle$, **g** $\langle 101\rangle$, **h** $\langle 110\rangle$, **i** $\langle 111\rangle$, the plasmodia bodies are shown by *thick lines*; **j** input-output logical function realized by the gate. Input are marked with *circles*, outputs with *solid discs*. Chemoattractants are placed in sites marked by *solid black discs*

A table of transformation $\langle x, y\rangle \rightarrow \langle d, e\rangle$ shows that the gate G_1 (Fig. 12f) implements logical conjunction and logical disjunctions $\langle x, y\rangle \rightarrow \langle xy, x + y\rangle$ at the same time but on two different outputs.

Geometrical structure of G_2 gate is shown in Fig. 13. Chemoattractants are placed in sites c and d and plasmodia can be inoculated in sites x, y and z (Fig. 13a). Lengths of channels in the gate satisfy the following conditions: $|xc| < |xd|$, $|ac| < |ad|$, $|bc| < |bd|$, and $|zb| + |bc| < |ya| + |ac|$.

In [31] input channels y and z (Fig. 13a) were assigned to constant TRUTH inputs an output channel c to a buffer (unused output to collect 'excess' of plasmodium). Let consider scenario when all three input can take values '0' and '1' and both outputs have a meaning.

If plasmodium placed in site z it propagates toward closest attractant-site c (Fig. 13c); similarly a plasmodium inoculated in site y propagates towards attractant-site c (Fig. 13d). When plasmodia are placed in sites y and z simultaneously, the plasmodium from the site z follows the route $(zb)(bc)$ and thus blocks the way for plasmodium propagating from y (Fig. 13e). Therefore the plasmodium originating in y moves to attractant-site d (Fig. 13e). The situations sketched in Fig. 13g–j can be described similarly. Considering the transformations $\langle x, y, \rangle \rightarrow \langle c, d\rangle$ we find that the gate implements the following logical function $\langle x, y\rangle \rightarrow \langle x, x\overline{y}\rangle$. If y- and z-iputs are constant TRUTH, $y = 1$ and $z = 1$, the gate G_2 is a negation (this how it was initially designed in [31]).

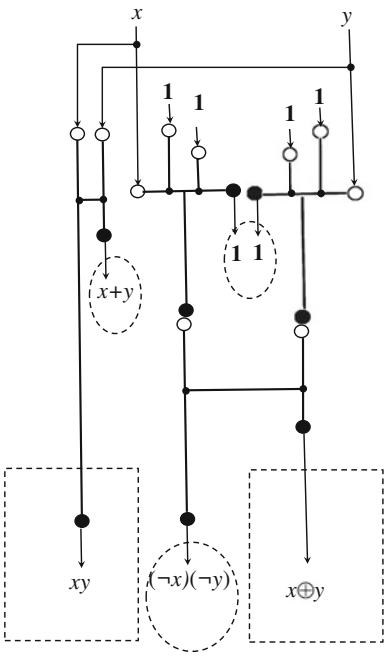

Fig. 14 Scheme of Physarum one-bit half-adder. Input variables are x and y, **1** on input channels represent constant TRUTH. Carry value xy and sum $x \oplus y$ are highlighted by *dotted rectangle*, unused outputs $x + y$, **1** and $(\neg x)(\neg y)$ by *dotted ellipses*

Physarum gates G_1 and G_2 can be cascaded by linking output gel-channels of one gate to input gel-channels of another gate. An example of such cascading in a form of one-bit half adder is shown in Fig. 14. Four pieces of plasmodium are fed in input channels as constant TRUTH. The plasmodia representing Boolean variables x and y are multiplied or branched and fed into gate G_1 and two copies of gate G_2. Output channels of gates G_2 are fed into data channels of another gate G_1. In addition to results we are looking for—xy and $x \oplus y$—the circuit (Fig. 14) produces several byproducts: $x + y$, $(\neg x)(\neg y)$ and two copies of constants TRUTH. These signals can be used further down in the chain of computation or routed in the buffer zones (plasmodium pool). Plasmodia representing constant TRUTH can be also rerouted back to control inputs of gates G_2.

To model the Physarum gate behaviours the three physical criteria identified in [31] and utilised in the design of the logic gates need to be implemented. The criteria can be summarised as:

1. Physarum grows and moves towards nutrient chemoattractant gradients.
2. If two plasmodium fragments encounter each other, they will avoid contact where other routes exist.
3. If two plasmodium fragments cannot avoid contact, the plasmodia will fuse.

The environment is represented by a greyscale image where different values correspond to different environmental features (for example, habitable areas, inhabitable areas, nutrient sources). The particles move about their environment

(a two-dimensional lattice) and sample sensory chemoattractant data from an isomorphic diffusion map. When particles move about their environment they deposit chemoattractant to the same structure. Chemoattractant gradients were represented by projection of chemoattractant to the diffusion map at the locations indicated on the gate schematic illustrations. The projection weight was set at 20 multiplied by the chemoattractant pixel value (255). The weight factor is high as chemoattractant is deemed to be completely absorbed when it encounters the edges of the chamber and a large weight value is necessary to ensure the required propagation distance. The diffusion kernel was a 7×7 window for all experiments. Diffusion was achieved by the mean of the local window at each location in the diffusion map and damped at 10^{-4} (i.e. new value is equal to the mean multiplied by $1 - 10^{-4}$). We assumed that diffusion of chemoattractant from a nutrient source was suppressed when the source was engulfed by particles. The suppression was implemented by checking each pixel of the food source and reducing the projection value (concentration of chemoattractants) by multiplying it by 10^{-3} if there was a particle within a 9×9 neighbourhood surrounding the pixel. Particle sensor offset was 5 pixels, angle of rotation set to 45°, and sensor angle was 45°.

Growth and shrinkage states are iterated separately for each particle and the results for each particle are indicated by tagging Boolean values to the particles. The growth and shrinkage tests were executed every three scheduler steps and the method employed is specified as follows. If there are 1 to 10 particles in a 9×9 neighbourhood of a particle, and the particle has moved forwards successfully, the particle attempts to divide into two (i.e. a new particle is created) if there is an empty location in the immediate neighbourhood surrounding the particle. If there are 0 to 20 particles in a 5×5 neighbourhood of a particle the particle survives, otherwise it is annihilated.

7.2 Modelling Individual Gates

To implement the gates using the model, the schematic illustrations in Figs. 12 and 13 were transformed into the spatial representations shown in Fig. 15. The spatial pattern and greyscale encoding (boundaries, nutrient sources) is used to configure the diffusive map.

Particles were introduced (depending on logical input conditions) at the areas indicated by solid circles at the top of the gates. Strong sources of chemoattractant were introduced at the outputs indicated as enclosed by dashed circles. The chemoattractant diffused from the output locations along channels etched into the gate configurations (white areas) and chemoattractant was removed immediately on contact with boundaries of the channels (light grey areas). The particle population was inoculated at identical times at the inputs, sensing, growing and moving towards the propagating diffusion gradients. To ‘anchor’ the growing paths to the start positions a very small amount of chemoattractant was also deposited at the respective start positions (the amount chosen was the lowest level needed to anchor the position

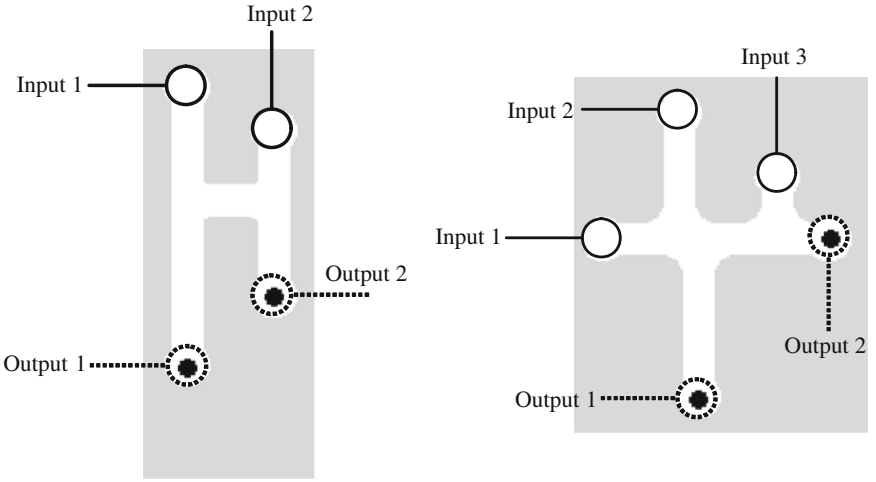

Fig. 15 Spatial implementation of logic gates G_1 and G_2 used in the particle model

without affecting the actual gate computation). Population inoculation and chemoattractant diffusion occurred at the same time and there was little or no directed growth of the population until the chemoattractants reached the source of inoculation.

The operation of the gates occurs due to the complex interactions between the chemoattractant diffusion gradients. Because there is a quantitative aspect to the chemoattractant gradient (i.e. particles sense not only the presence but also the concentration of the diffusion gradient), the gradient concentration is affected by the length and width of the gate channels [12]. The point at which the competing wave fronts meet is a spatial interface which delineates path choices in a similar way to those observed in chemical reaction-diffusion computations [26]. Thus, the environment is partially responsible for the initial selection of path choice. This 'background processing' by the environment satisfies the first of the three aforementioned criteria for plasmodium gate construction.

Two more factors add to the complexity of gradient interactions. Firstly when the particle representation of the plasmodium engulfs a food source, the diffusion of chemoattractant from that source is suppressed (reduced by a factor of one thousand). This alters the concentration of the gradient field from the engulfed source and the interface position where competing fronts meet shifts to reflect the new gradient field. Secondly, the collective movement of the particle population also results in local chemoattractant deposition along the path (this deposition is responsible for the local recruitment of particles by positive feedback and also acts to maintain the cohesiveness of the particle swarm). The local deposition of chemoattractant is also subject to the same diffusion as that which affects the food sources (in fact it is represented computationally as the same 'substance') and the diffusion away from the particle population also acts to generate a dynamical interface which competes with the food source gradients.

Suppression of food source gradients and local modification of gradients by the particle collective represents a highly dynamical spatial computation in which both local and distant sources of information (food source location, path availability) are integrated by both environmental and collective swarm computation. It can also be seen that the local modification of the gradient by the particle collective indirectly satisfies the second criterion for plasmodium gate construction—attempted avoidance of local plasmodia. The dynamical gradient interface represents a fragile boundary between two separate swarms, two separate food gradients or a combination of both swarm and food gradients. The third criterion—fusion of plasmodia can be represented in the particle model when movement of separate particle paths is limited and perturbation of the dynamic boundary occurs. This can result in fusion of network paths which corresponds to fusion of plasmodia.

The complex evolution of gradient fields can be seen in an example run of G_2 with the inputs 011 in Fig. 16. The top row shows the particle positions and the bottom row shows the chemoattractant gradient field enhanced by a local method of dynamic contrast enhancement. The first column shows the propagation of chemoattractant

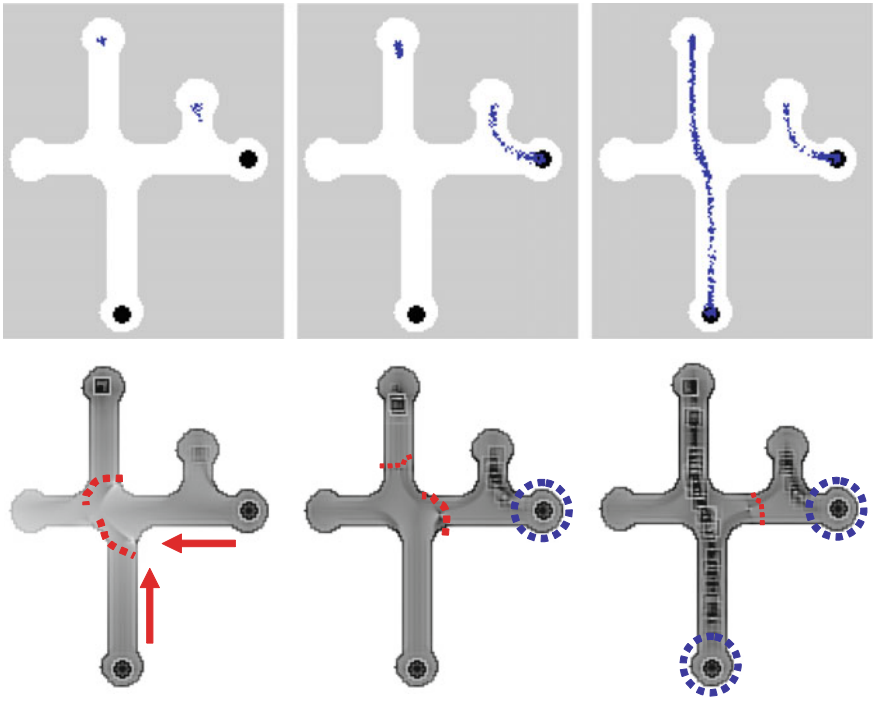

Fig. 16 Evolution of 'plasmodium' positions and interaction fronts in the particle model for the G_2 gate with inputs 011. *Top Row* Particle positions. *Bottom Row* Chemoattractant gradient. Arrows indicate propagation of gradient from food sources. *Dashed arcs* represent boundary regions separating competing gradients. *Dashed circles* represent diffusion from food sources suppressed by engulfment. See text for explanation

gradient from the two food sources and the interfacial region (dashed arcs). Note that the gradient from the right suppresses the gradient from the bottom source. The second column shows the effect of suppression of the rightmost food source when engulfed by the particle population which has migrated towards it. Because the bottom food source is not suppressed the gradient from this source is stronger than the right side and the interface boundary shifts to the right of the T-junction. Note that there is also a weaker interface boundary between the diffusion gradient emanating from the bottom food source and the chemoattractant deposition from the particle population in the long vertical column. The third column shows the result of the competition between the food gradient and the population gradient—the food gradient is stronger and the population grows and migrates downwards to the food node.

When the bottom node is suppressed the two separate paths remain stable and do not fuse. A fragile interfacial boundary can be seen between the two network paths (dashed arc) and, as long as the particles do not cross the 'buffer' space between the two paths, the paths will not fuse.

Results using the particle model for gates G_1 and G_2 are shown in Figs. 17 and 18. The G_1 gate achieved 90 % reliability and the G_2 gate achieved 98.57 % reliability. The input conditions 0–0 were not included with the results because the output result for these inputs is guaranteed regardless of gate design. For the G_1 gate we see that the shorter path to the right food source attracts the simulated plasmodium in

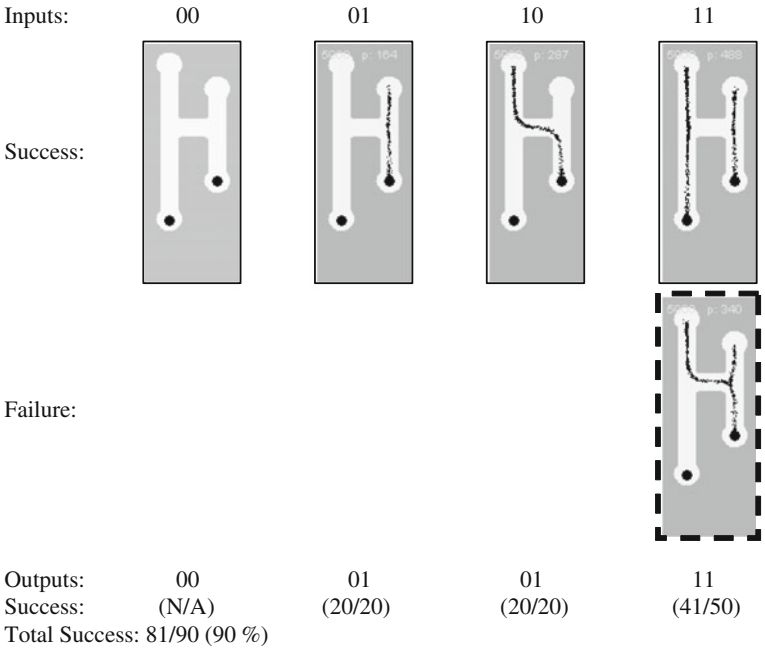

Fig. 17 Summary of results for particle approximation of Physarum based logic gate G_1

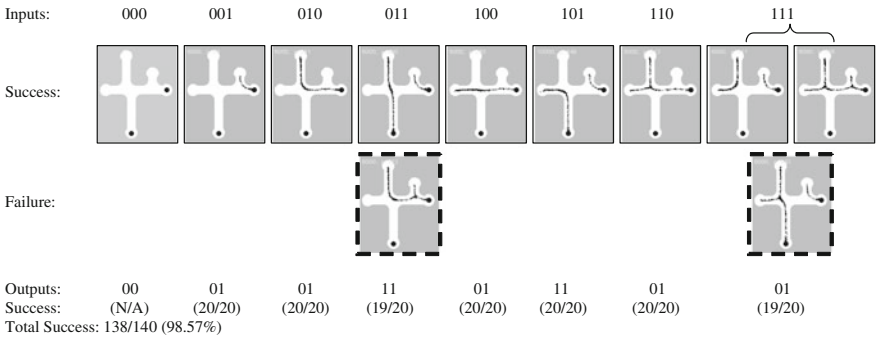

Fig. 18 Summary of results for particle approximation of Physarum based logic gate G_2

both 0–1 and 1–0 condition. Note that no branching occurs from the plasmodium to the left nutrient source when the right source is connected. This is because the movement of particles (and their deposition to the diffusion map) creates a local diffusion field around the particle collective. The strength of this locally generated field is enough to suppress the field emanating from the left food source and no branching is observed. If the strength of the local field were less than that of the nutrient source then branching and growth to the left nutrient source would indeed occur.

The errors in the G_1 gate all occurred in the 1–1 input condition. The 'pattern' of the error is that the left particle stream did not continue downwards to the food source, but fused with the right side particle stream (indicated by dashed box). Analysis of all of the results found that whenever the growing particle plasmodium encountered a junction in a gate an apparent 'hesitation' was seen. The growth tip appeared to be indecisive as to which direction to take. When a direction was eventually chosen the growth speed increased when the growth tip moved past the junction. The hesitation, and indeed some of the gate errors, was caused by disturbances in the diffusion field near the tip of the growing plasmodium. The diffusion gradient emanating from the nutrient sources is relatively uniform whereas the gradient from the plasmodium tip is more intermittent in quality (because the tip growth is non uniform and changeable in form). In contrast the gradient from a moving straight part of the particle plasmodium was more uniform. The fragility of the gradient field at the growth tip was further perturbed by the spatial changes in the environment at the junctions. This, coupled with increased possible choices of directions, led to what we describe as junctional errors. The junctional errors are characterised by failures in searching of the growing plasmodium tip and were responsible for all of the failure instances of the G_1 gate.

The G_2 gate, although more complex in design, was more reliable than G_1 and the only errors which occurred were a single junctional error in the 011 input condition and an error in the 111 input condition. This error was classed as a timing error and was caused by different growth rates from the two left-side inputs. Ideally the two particle streams should meet and fuse but differences in the growth of the two separate streams led to non fusion and errors in output.

7.3 Modelling the Half Adder

To implement the half adder based on gates G_1 and G_2 with the particle model the scheme of the half adder in Fig. 14 was slightly modified as shown in Fig. 19. The G_2 gate combination was simplified by 'sharing' the food source between both gates. Constant TRUTH inputs ('**1**') were provided as some of the gate inputs to implement the desired function. The outputs of the combined G_2 gates were fed to act as inputs to the lower G_1 gate. To ensure that the particle population continued to the input positions of the lower gate synthetic chemoattractant stimuli (small dots) were placed to guide any plasmodium along the channel to the input positions. The '$G_2G_2G_1$' triplet combination acted as the XOR (summation) part of the half adder. The AND section of the half adder (carry computation) was implemented as a single G_1 gate (Fig. 19, left). In the simulations the branching of initial X and Y signals to provide the inputs to both sections of the half adder was not implemented in an effort to simplify the design and the relevant X and Y inputs were introduced to the gate manually.

The use of constant TRUTH inputs to the half adder introduces errors in gate output when inputs are 0–0. This is because the outermost truth signals at the inputs of the G_2G_2 gates travel down through the gates and into the lower G_1 gate. This would result in the 'no input' condition actually causing an erroneous output. Apart from redesigning the gate this presents an opportunity to consider possible use of error checking signals in the gate design. One possible error checking signal is the 'EA' output in the left side of the circuit (Fig. 19, left). It can be seen that this flag should be set whenever any of the inputs are set to true. It would therefore be possible to use the absence of the EA output to indicate a 0–0 input to the half adder, and thus indicate erroneous output from the constant TRUTH inputs to G_2G_2. Another possible use of outputs to indicate error conditions is the 'EB' output from the left G_1 portion of

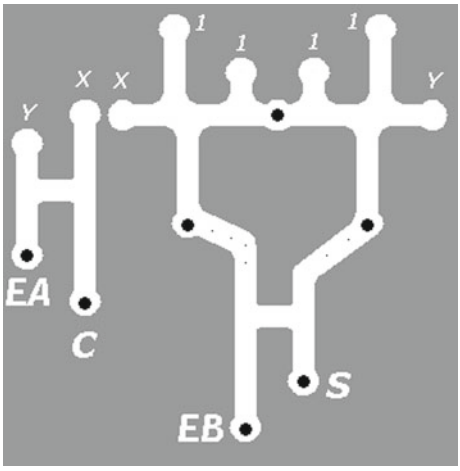

Fig. 19 Spatial representation of half adder based on combinations of G_1 and G_2. X and Y: Inputs to half adder, 1: constant TRUTH signals, S: Sum output, C: Carry output. *Solid discs* are food sources and *small dots* are small food sources to feed outputs towards lower gate inputs. EA and EB: Error checking flags (see text)

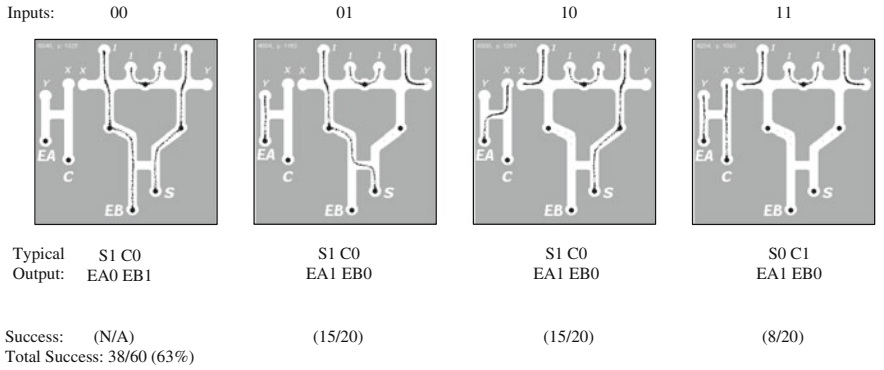

Fig. 20 Examples of input and output conditions for the particle approximation of the half adder

the $G_2G_2G_1$ triplet (Fig. 19, bottom). It can be seen (Fig. 20) that the EB flag should never be set unless the 0–0 condition caused by constant TRUTH inputs occurs. This flag could be combined with the lack of EA output to indicate errors. When the EB flag is set without the presence of EA then a fault can be assumed to have occurred within the half adder $G_2G_2G_1$ triplet. Of course the addition paths and mechanisms to make use of these error checking flags adds another layer of complexity to the circuitry which is out of the scope for this research. The results of the half adder approximation can be seen in Fig. 20.

The failure rate for the half adder approximation, even when not including the difficulty posed by the 0–0 configuration, was significantly higher than for the single gates. The majority of the failures were caused by timing errors, which occurred when the outermost inputs to the G_2G_2 combined gate did not fuse correctly with the constant TRUTH inputs and, instead, travelled down towards the lower gate. Junctional errors also occurred three times in the left G_1 gate for the 1–1 input condition.

The combination and extension of the individual gates appeared to compound the errors in the individual gates. Although no definitive answer can be given as to why the unreliability increased, we speculate that the combining of the gates subtly affected the propagation and profile of the chemoattractant gradients.

7.4 Quantitative Transformation of Adder Circuits

The full adder circuit enables the addition of two binary inputs, together with a carry input (for example, from a previous calculation). The carry output may be cascaded to other adder circuits so that larger strings of bits may be added. However, the complex spatial routing of signals from the input channels and from combined outputs of individual logic gates is difficult to implement reliably with living substrates such as the Physarum plasmodium. In [14] we explored whether it was possible to

avoid spatial propagation, branching and crossing completely in the design of adder circuits. A simple quantitative transformation of the input patterns which considered the *total number* of bits in the input string allowed us to map the respective input combinations to the correct outputs patterns of the full adder circuit, reducing the circuit combinations from a 2:1 mapping to a 1:1 mapping. The mapping of inputs to outputs shows an incremental linear progression, suggesting its implementation in a range of physical systems. We give a brief overview of the approach below.

The truth table for the full adder circuit is shown in Table 3. In binary terms the full adder may be seen as a 2:1 mapping of the 8 (3-bit) possible input combinations (X, Y, C_{in}) into 4 (2-bit) possible output combinations (S, C_{out}). If we examine the three inputs to the full adder, together with their corresponding outputs, a pattern can be seen if we ignore the binary values of the inputs, and instead concentrate on the *total* number of '1' bit digits in the input patterns (Table 3, column: number of bits). The number of '1' bits in each combination ranges from zero to three, giving four possible combinations, matching the number of output pattern combinations.

Furthermore, combinations of different inputs which share the *same number of bits* all share the same output configuration. For example (referring to Table 3) the input patterns which all contain only a single '1' bit (decimal value of patterns 1, 2 and 4) all map to the same output pattern of 0 (C_{out}) and 1 (S). Similarly, input patterns which contain two '1' bits (decimal value of patterns 3, 5 and 6) all map to the output patterns of 1 (C_{out}) and 0 (S).

This mapping of quantitatively transformed input pattern bits to output pattern 'bins' is clarified in Table 4 which shows the mapping of quantitatively transformed inputs into decimalised interpretations of the output patterns. The result is a 1:1 mapping of (transformed) input to output values. By encoding the number of bits instead of the pattern of bits we have reduced the complexity of the input by half (8 possible combinations to 4). The mapping removes the need for spatial re-routing of separate components of the adder circuit. Importantly, this mapping of input to output values is also identical in that they both follow an incremental progression. This suggests that it might be possible to use a physical mechanism to perform the computation of the adder circuit. That is, the transformed input could be encoded in a single signal line carrying a weighted non-binary signal.

Table 3 Single-bit full adder truth table including decimal value of input combinations and the quantitative number of 1 bits in each input combination

Decimal	Number of bits	Input X	Input Y	Input C_{in}	Output C_{out}	Output S
0	0	0	0	0	0	0
1	1	0	0	1	0	1
2	1	0	1	0	0	1
3	2	0	1	1	1	0
4	1	1	0	0	0	1
5	2	1	0	1	1	0
6	2	1	1	0	1	0
7	3	1	1	1	1	1

Table 4 Mapping quantitatively transformed input combinations into truth table outputs

Number of bits	C_{out}	S	Decimalised outputs
0	0	0	0
1	0	1	1
2	1	0	2
3	1	1	3

Left column shows the total number of bits in the X,Y and C_{in} inputs. *Middle columns* (C_{out} and S) are binary outputs. *Right column* shows the decimal values of the two binary outputs C_{out} and S

A physical implementation of the adder circuit must relate the number of input bits to an increase in some physical value. This value could be changes in light intensity, an increase in pressure, and so on… To return the output to the binary domain the relevant output map 'bin' must be transformed into binary values in order to pass the S and C_{out} on to the next gate. This conversion adds complexity at the interface of the circuit, but this is balanced by the increased simplicity within the adder circuit itself.

7.5 Implementation of the Quantitative Mechanism in a Model of P. polycephalum

As an illustrative example we demonstrate how a multi-agent model of slime mould can be used to implement the quantitative adder. The multi-agent approach is inspired by the simple components of the Physarum and is composed of a population of simple mobile particles indirectly coupled within a diffusive lattice. Agents loosely correspond to aggregates of overlapping actin filaments. Their collective structure of the population corresponds to the pattern of the tube networks of the plasmodium and the collective movement of particles corresponds to the sol flux within the plasmodium. A full description of the model is given in [14] and below we concentrate on the transformations of binary values at the inputs to the adder circuit and the subsequent interpretation of the changes in behaviour of the model as the outputs of the circuit.

We represent the changing physical quantities of the transformed binary inputs by geometrically constraining the model plasmodium within a narrow tube-like arena of 323 × 20 pixels (Fig. 21). The model plasmodium (5000 particles) is inoculated within the habitable region of this arena. As the individual particles move they deposit a generic chemoattractant trail within the lattice at their new site. Particles are also attracted by the local concentration of trails. If particles cannot move or collide, no trail is deposited. After a short period of time the initially uniform distribution of trail within the lattice becomes unevenly distributed, causing local oscillations of trail concentration. These oscillatory domains grow and become entrained (see [28, 32] for a full description of the emergence of oscillatory domains in the model and real plasmodium respectively). A small sampling window 20 × 20 pixels at the left of the arena records the mean flux of trails within the lattice within this region at every 5 scheduler steps.

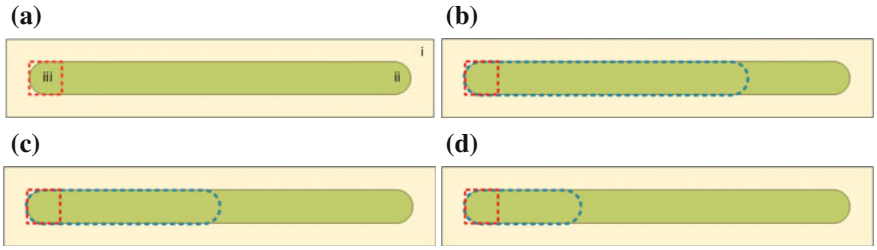

Fig. 21 Geometrically constraining arena of the model plasmodium. **a** schematic of model arena with non-habitable area (*i*), habitable region (*ii*) and sampling window (*iii*, *dashed red*, online) indicated, **b–d** Constrained habitable region (*dashed blue*, online) of 0.75, 0.5 and 0.25 the original arena length. **a** Arena length 1. **b** Arena length 0.75. **c** Arena length 0.5, **d** Arena length 0.25

The quantitative input values mapped from 0 to 3 (from inputs of Table 4) are transformed into a physical representation by only allowing movement of particles which occupy smaller regions of the arena (elongated dashed regions in Fig. 21b–d). These regions represent decreasing fractions of the original length of the arena.

If the geometric constraints of the arena represent the binary input transformation, how is the physical behaviour which generates the individual output bins represented? We can represent this in the model plasmodium by measuring the frequency of oscillations within the arena. The Physarum plasmodium, when confined in a suitably shaped arena, exhibits regular oscillations of thickness at each end of the arena [29]. These thickness oscillations emerge from initially random contractions of the plasmodium strand which propel sol throughout the strand. There is a reciprocal relationship between contractile activity and strand thickness (presumably due to the stretch activation phenomenon of the tube strand) as contraction of the tube results in transport of sol away from the contraction site and a subsequent decrease in strand thickness.

The same reciprocal relationship emerges in the model. As particles move they deposit chemoattractant in the lattice (greyscale brightness corresponds to concentration in Fig. 22). The increased flux attracts local particles, causing further increases in population density. Eventually the occupancy in this region is too high to allow free particle movement and flux decreases. At more distant sites the relative decrease in population density (caused by previous particle efflux) allows greater freedom of movement and so flux increases in these regions. The result is a gradual emergence of small oscillatory domains Fig. 22a which fuse and become entrained Fig. 22b until a single large oscillation traverses the arena from end to end Fig. 22c, d. The dominant frequency of this stable oscillation Fig. 22e can be measured by FFT transform and spectral analysis.

When the model plasmodium is geometrically constrained using the patterns in Fig. 21 the dominant oscillatory frequency increases as arena length decreases. An example plot of the oscillatory patterns for decreasing arena length is shown in Fig. 23a. When the oscillation data (10 runs at each arena length fraction) is analysed

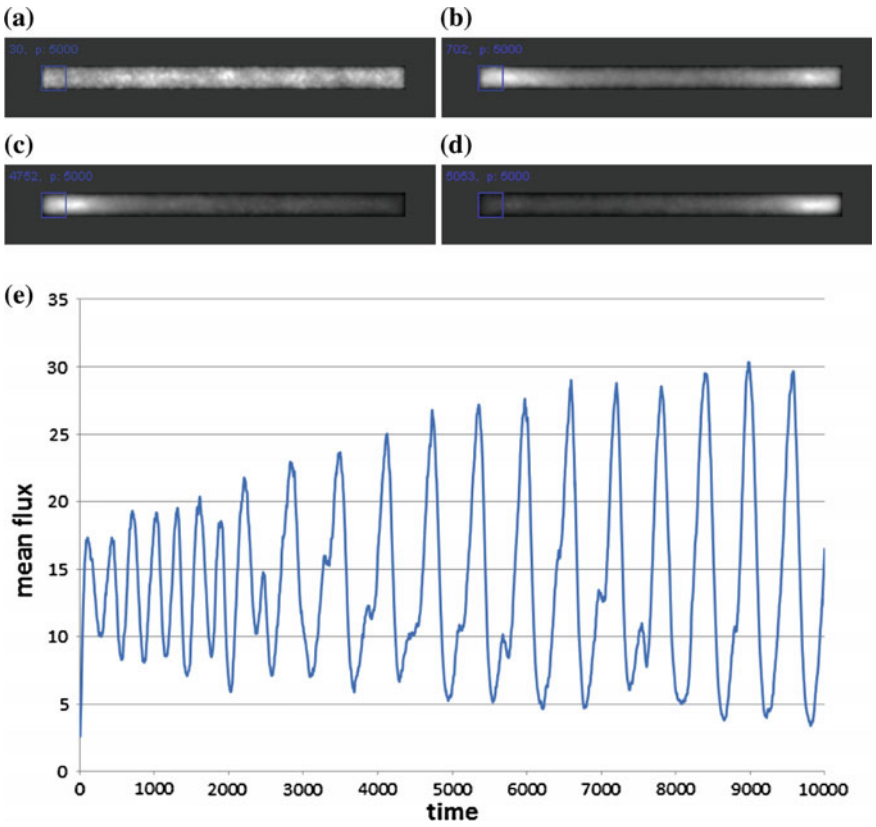

Fig. 22 Emergence of regular oscillations of flux within model the plasmodium on full width arena. **a** flux of particle trails at t = 30 shows relatively uniform distribution (increasing grey brightness corresponds to greater flux), **b** at early stages there are two oscillatory domains at each end of the arena, **c**, **d** at later stages there is a single reciprocating oscillatory pattern, **e** plot of mean flux within measuring window shows initial oscillatory domains fusing by entrainment at approximately t = 2000 to form a regular reciprocating oscillation pattern. **a** t = 30, **b** t = 702, **c** t = 4752, **d** t = 5053

in the frequency domain, the frequency of oscillations does indeed increase as the arena length decreases. Although the relationship is not perfectly linear (Fig. 23b), a simple thresholding of oscillation frequency should be sufficient to represent the output bins of the quantitative adder.

Although this example uses changing oscillation frequency in a model of Physarum plasmodium to represent the physical system, the quantitative concept is applicable to any physical system with approximately linear progression. For example, constraining the geometry of the model plasmodium is suggestive of changing the dominant frequency of a resonant cavity by altering its size.

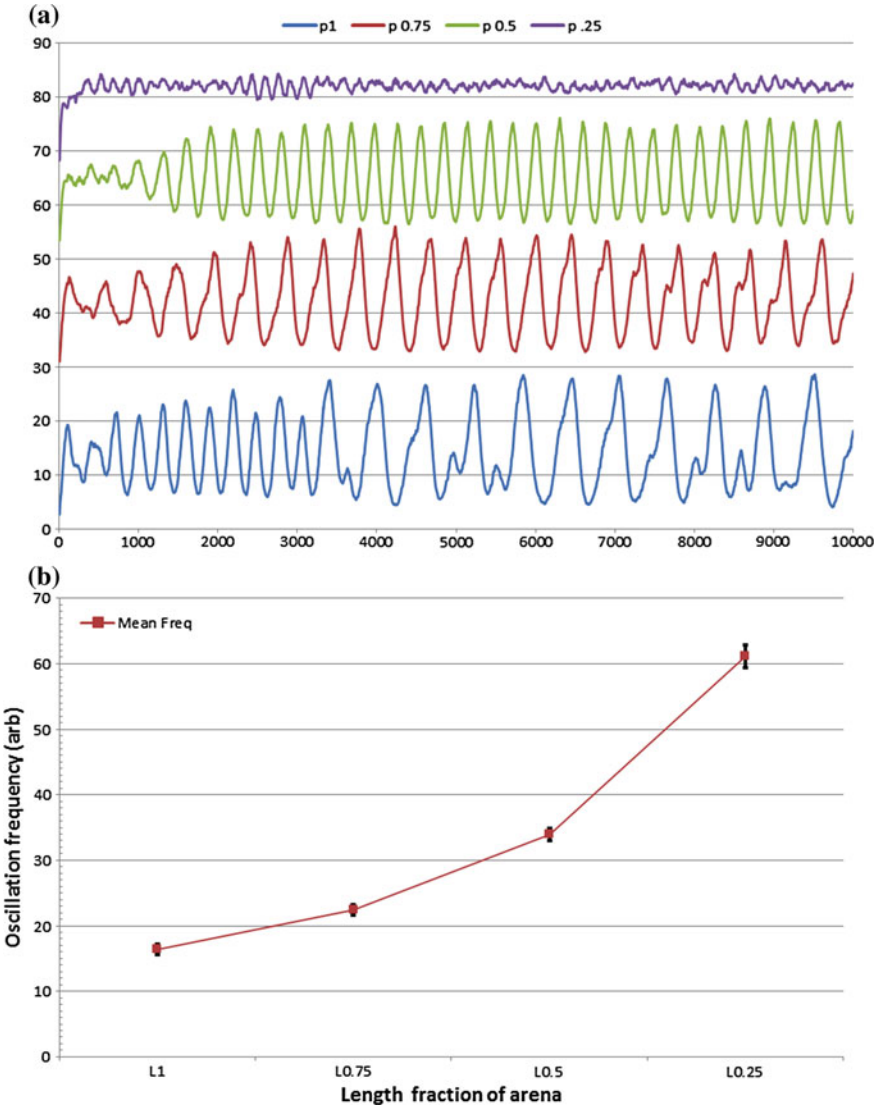

Fig. 23 Oscillation frequency and arena length. **a** Oscillation frequency increases as arena length decreases, offset plots show example oscillatory activity at arena fractions of (*top* to *bottom*) 0.25, 0.5, 0.75 and 1. **b** Plot showing increase in oscillation frequency as arena length fraction decreases (10 runs per arena length fraction, standard deviation shown in error bars)

8 Discussion

Experimental prototype of attraction, ballistic and repellent gates suffer from low speed. The gates are based on physical propagation of the slime and therefore computation might take hours if not days, depending on size of the gates. Reliability of experimental Physarum gate is limited by 69 % for gate P_1 and 59 % for gate P_2. This is because behavior of plasmodium is determined by too many environmental factors—thickness of substrate, humidity, diffusion of chemo-attractants in the substrate and in the surrounding air volume, and physiological state of plasmodium during each particular experiment. Increasing reliability of Physarum gates might be a scope of further studies.

Frequency gates employ oscillations of electrical potential in the slime mould's protoplasmic tubes. They are faster, approximately 30 times faster, than growth based gates, yet still require 20–30 mins to perform computation, because it is necessary to calculate frequencies of the potential oscillation before and after stimulation. Functional completeness of the frequency gates is demonstrated with the development of the NOR and NAND logic gates. NOR, NAND, XOR and XNOR derived gates have also been demonstrated. Basic gates OR, AND and NOT were correct 90, 77.8 and 91.7 % of the time respectively. Derived logic circuits XOR, half adder and full adder were 70.8, 65 and 58.8 % accurate respectively. Increasing the number of frequency gates in circuits correlates with an increase in error of the combinational logic circuit; the error is proportional to the number of gates. The results shown here demonstrate a significant advancement in organism-based computing, providing a solid basis for hybrid computers of the future.

With regards to fluidic gates, a need for a mechanical control is a hereditary of many micro-fluidic circuits and slime mould circuits do not make an exclusion. Physarum fluidic gates can be cascaded using opto-mechanical coupling, e.g. when a width of an output tube or a speed of flow in the tube is detected by optical means and then input tubes of next gates in a circuits are stimulated mechanically. This is not an optimal solution thought. We should think on how to exploit mechanical properties of slime mould tubes to make device-embedded flow switching functions.There are some published results showing that embedded check valve and switch valve (with pressure-dependent states), analogous to diode and p-channel JFET transistor, can be implemented in layered elastomeric materials [23]. Another route to explore could be to map designs of hybrid fluidic-electronic-mechanical universal logical gates [37] onto patterns of living slime mould devices. With respect to delivery of e.g. encapsulated substances by plasmodial tubes, changes in a cytoplasmic resistance in tubes containing capsules can be exploited to construct logical gates similar to droplet based gates [7] and bubble logic [24]. Physarum microfluidic gates are slow, their speed is an order of seconds, much slower than silicon gates. However, slime mould microfluidic gates are self-growing and self-repairing and can be incorporated in a hybrid wetware-hardware devices for sensing and analysing of non-lethal substances, and detection of molecules or certain types of living cells.

The simulation findings for spatially implemented gates suggest that, although such circuits can indeed be built, the presence of both timing errors and junctional (search) errors would severely limit the effectiveness and practicality with the cascading towards more complex circuits. The matter of errors of the gate operations (timing errors and junctional errors) requires further consideration. The term 'error' depends on the perspective taken. From an experimental viewpoint the occasionally unreliable operation of the gates is erroneous. But the notion of externally applied—by the experimenter—environment conditions and metrics of success cannot be easily applied to the behaviour of a living (or even simulated) collective organism, whose sole imperative is the location and connection of nutrient sources for survival. By following the biological imperative, the organism is not actually doing anything 'wrong', even though this may not result in reliable logical operations. The quantitative encoding of the adder circuit results in a linear transformation between input and output conditions. Although this would require additional complexity in transformation of inputs and outputs, it reduces the potential for errors caused by branching of signals and bridging of signals and may be implementable in a range of physical systems.

References

1. Adamatzky, A.: Manipulating substances with Physarum polycephalum. Mater. Sci. Eng. C **30**(8), 1211–1220 (2010)
2. Adamatzky, A.: Slime mould logical gates: exploring ballistic approach (2010). arXiv preprint arXiv:1005.2301
3. Adamatzky, A.: Slime mould tactile sensor. Sens. Actuators B Chem. **188**, 38–44 (2013)
4. Adamatzky, A.: Towards slime mould colour sensor: recognition of colours by Physarum polycephalum. Org. Electron. **14**(12), 3355–3361 (2013)
5. Adamatzky, A., Schubert, T.: Slime mold microfluidic logical gates. Mater. Today **17**(2), 86–91 (2014)
6. Aldrich, H.: Cell Biology of Physarum and Didymium V1: Organisms, Nucleus, and Cell Cycle. Elsevier, Amsterdam (2012)
7. Cheow, L.F., Yobas, L., Kwong, D.-L.: Digital microfluidics: droplet based logic gates. Appl. Phys. Lett. **90**(5), 054107 (2007)
8. De Lacy, B., Costello, A.A., Jahan, I., Zhang, L.: Towards constructing one-bit binary adder in excitable chemical medium. Chem. Phys. **381**(1), 88–99 (2011)
9. Fingerle, J., Gradmann, D.: Electrical properties of the plasma membrane of microplasmodia of Physarum polycephalum. J. Membr. Biol. **68**(1), 67–77 (1982)
10. Heilbrunn, L.V., Daugherty, K.: The electric charge of protoplasmic colloids. Physiol. Zool. **12**(1), 1–12 (1939)
11. Iwamura, T.: Electric impedance of the plasmodium of a slime mold, Physarum polycephalum. Cytologia **17**(4), 322–328 (1952)
12. Jones, J.: Passive vs active approaches in particle approximations of reaction-diffusion computing. Int. J. Nanotechnol. Mol. Comput. **1**(3), 37–63 (2009)
13. Jones, J., Adamatzky, A.: Towards Physarum binary adders. Biosystems **101**(1), 51–58 (2010)
14. Jones, J., Whiting, J.G.H., Adamatzky, A.: Quantitative transformation for implementation of adder circuits in physical systems. Biosystems **134**, 16–23 (2015)
15. Kamiya, N., Abe, S.: Bioelectric phenomena in the myxomycete plasmodium and their relation to protoplasmic flow. J. Colloid Sci. **5**(2), 149–163 (1950)

16. Kirby, B.J.: Micro-and Nanoscale Fluid Mechanics: Transport in Microfluidic Devices. Cambridge University Press, Cambridge (2010)
17. Knowles, D.J., Carlile, M.J.: The chemotactic response of plasmodia of the myxomycete Physarum polycephalum to sugars and related compounds. J. Gen. Microbiol. **108**(1), 17–25 (1978)
18. Marr, D.W.M., Munakata, T.: Micro/nanofluidic computing. Commun. ACM **50**(9), 64–68 (2007)
19. Mayne, R., Adamatzky, A.: Slime mould foraging behaviour as optically-coupled logical operations. Int. J. Gen. Syst. **44**(3), 305–313 (2015)
20. Mayne, R., Adamatzky, A.: Toward hybrid nanostructure-slime mould devices. Nano LIFE **5**(01), 1450007 (2015)
21. Mayne, R., Adamatzky, A.: Towards hybrid nanostructure-slime mould devices. Nano LIFE **5**(1), 140007 (2015)
22. Mayne, R., Tsompanas, M.-A., Sirakoulis, G.C., Adamatzky, A.: Towards a slime mould-FPGA interface. Biomed. Eng. Lett. **5**(1), 51–57 (2015)
23. Mosadegh, B., Kuo, C.-H., Tung, Y.-C., Torisawa, Y.-S., Bersano-Begey, T., Tavana, H., Takayama, S.: Integrated elastomeric components for autonomous regulation of sequential and oscillatory flow switching in microfluidic devices. Nat. Phys. **6**(6), 433–437 (2010)
24. Prakash, M., Gershenfeld, N.: Microfluidic bubble logic. Science **315**(5813), 832–835 (2007)
25. Seifriz, W.: A theory of protoplasmic streaming. Science **86**(2235), 397–398 (1937)
26. Steinbock, O., Tóth, Á., Showalter, K.: Navigating complex labyrinths: optimal paths from chemical waves. Science **210**, 868–868 (1995)
27. Tabeling, P.: Introduction to Microfluidics. Oxford University Press, Oxford (2010)
28. Takagi, S., Ueda, T.: Emergence and transitions of dynamic patterns of thickness oscillation of the plasmodium of the true slime mold Physarum polycephalum. Physica D **237**, 420–427 (2008)
29. Takamatsu, A., Fujii, T., Yokota, H., Hosokawa, K., Higuchi, T., Endo, I.: Controlling the geometry and the coupling strength of the oscillator system in plasmodium of Physarum polycephalum by microfabricated structure. Protoplasma **210**(3–4), 164–171 (2000)
30. Tanyeri, M., Ranka, M., Sittipolkul, N., Schroeder, C.M.: A microfluidic-based hydrodynamic trap: design and implementation. Lab Chip **11**(10), 1786–1794 (2011)
31. Tsuda, S., Aono, M., Gunji, Y.-P.: Robust and emergent physarum logical-computing. Biosystems **73**, 45–55 (2004)
32. Tsuda, S., Jones, J.: The emergence of synchronization behavior in Physarum polycephalum and its particle approximation. Biosystems **103**, 331–341 (2010)
33. Tsuda, S., Aono, M., Gunji, Y.P.: Robust and emergent Physarum logical-computing. Biosystems **73**(1), 45–55 (2004)
34. Vestad, T., Marr, D.W.M., Munakata, T.: Flow resistance for microfluidic logic operations. Appl. Phys. Lett. **84**(25), 5074–5075 (2004)
35. Whiting, J.G.H., de Lacy Costello, B.P.J., Adamatzky, A.: Sensory fusion in Physarum polycephalum and implementing multi-sensory functional computation. Biosystems **119**, 45–52 (2014)
36. Whiting, J.G.H., de Lacy Costello, B.P.J., Adamatzk, A.: Towards slime mould chemical sensor: mapping chemical inputs onto electrical potential dynamics of Physarum polycephalum. Sens. Actuators B Chem. **191**, 844–853 (2014)
37. Zhang, R., Dalton, C., Jullien, G.A.: Two-phase ac electrothermal fluidic pumping in a coplanar asymmetric electrode array. Microfluid. Nanofluid. **10**(3), 521–529 (2011)