Available online at www.sciencedirect.com

ScienceDirect

Physics of Life Reviews 29 (2019) 1–26

PHYSICS of LIFE reviews

www.elsevier.com/locate/plrev

Review

Does being multi-headed make you better at solving problems? A survey of *Physarum*-based models and computations

Chao Gao [a,1], Chen Liu [b,*,1], Daniel Schenz [c,1], Xuelong Li [d,*], Zili Zhang [a,*], Marko Jusup [e,f,*], Zhen Wang [g], Madeleine Beekman [h], Toshiyuki Nakagaki [e,i]

[a] *College of Computer and Information Science & College of Software, Southwest University, Chongqing 400715, China*
[b] *Center for Ecology and Environmental Sciences, Northwestern Polytechnical University, Xi'an 710072, China*
[c] *Graduate School of Life Science, Hokkaido University, Sapporo 060-0808, Japan*
[d] *Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, Shaanxi, China*
[e] *Center of Mathematics for Social Creativity, Research Institute for Electronic Science, Hokkaido University, Sapporo 060-0812, Japan*
[f] *Institute of Innovative Research, Tokyo Institute of Technology, Tokyo 152-8552, Japan*
[g] *School of Mechanical Engineering and Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University, Xi'an 710072, China*
[h] *School of Life and Environmental Sciences, The University of Sydney, NSW 2006, Australia*
[i] *Global Station for Soft Matter, Global Institution for Collaborative Research and Education, Hokkaido University, Sapporo 060-0815, Japan*

Received 7 March 2018; received in revised form 25 April 2018; accepted 4 May 2018
Available online 22 May 2018
Communicated by J. Fontanari

Abstract

Physarum polycephalum, a single-celled, multinucleate slime mould, is a seemingly simple organism, yet it exhibits quasi-intelligent behaviour during extension, foraging, and as it adapts to dynamic environments. For these reasons, *Physarum* is an attractive target for modelling with the underlying goal to uncover the physiological mechanisms behind the exhibited quasi-intelligence and/or to devise novel algorithms for solving complex computational problems. The recent increase in modelling studies on *Physarum* has prompted us to review the latest developments in this field in the context of modelling and computing alike. Specifically, we cover models based on (i) morphology, (ii) taxis, and (iii) positive feedback dynamics found in top-down and bottom-up modelling techniques. We also survey the application of each of these core features of *Physarum* to solving difficult computational problems with real-world applications. Finally, we highlight some open problems in the field and present directions for future research.

Keywords: Modelling and computation; Bio-inspired computing; *Physarum polycephalum*; Intelligent behaviour; Complex problem solving

[*] Corresponding author.
E-mail addresses: ChenLiu-Bean@hotmail.com (C. Liu), xuelong_li@opt.ac.cn (X. Li), zhangzl@swu.edu.cn (Z. Zhang), mjusup@gmail.com (M. Jusup).

[1] Equal contribution.

https://doi.org/10.1016/j.plrev.2018.05.002
1571-0645/© 2018 Published by Elsevier B.V.

1. Introduction

In the 1990s, computer scientists began to look to biological systems for inspiration when designing optimisation algorithms. For approximately 10 years, trail-laying ants were the paradigm, particularly with respect to shortest path problems [1,2]. Ant colony optimisation algorithms (ACOs) worked well in static environments. However, most optimisation problems are dynamic and require the algorithms to constantly update their solutions. Later, ACOs became increasingly less based on real biological systems because of the (as it turned out incorrect [3–5]) assumption that trail-laying ants are incapable of adjusting rapidly to changes in their foraging environment. As a consequence, attention turned to another organism, the true slime mould *Physarum polycephalum*, because of its ability to solve mazes and to develop adaptive and fault-tolerant networks [6,7].

Modelling has focused on two fundamental behaviours: **expansion** and **contraction**. The expansion behaviour is captured by models based on *Physarum*'s **morphology** and **taxis**, while the contraction behaviour is characterised by **positive feedback dynamics**.

Similar to research on other nature-inspired computational methods (e.g., artificial immune systems [8,9], ant colony systems [10], etc.), we will take two different perspectives, namely, *Physarum*-based modelling and *Physarum*-based computing. In the context of modelling, studies focus on *Physarum*'s foraging behaviour to help better capture its core features and the underlying mechanisms [11–13]. These features and mechanisms are then exploited in the context of *Physarum*-based computing to solve complex computational problems (e.g., travelling salesman problems (TSPs) [14] and community detection [15]). Knowing the advantages and the limitations of the existing models is critical to designing novel and more effective *Physarum*-based solutions. Such solutions are now particularly sought after because of the technological advances that have led to, on the one hand, cheap sensing techniques, and on the other hand, ubiquitous real-time data streaming. A comprehensive survey that summarises the research on *Physarum* from both a modelling and a computing perspective is therefore badly needed. Now is also the time to explore if *Physarum*-based optimization methods are superior to the methods that are based on trail-laying ants or whether the field has simply moved on to the next charismatic model organism.

Here we will first survey the existing *Physarum*-based research in Section 2 by using the latest published literature in the Web of Science database. This survey classifies the achievements of *Physarum*-based research and lays emphasis on the two time-evolving foraging behaviours (i.e., expansion and contraction) and the three features of these behaviours (i.e., morphology, taxis, and the positive feedback loop). Then, Section 3 details the typical *Physarum* models in terms of the three characteristic features from both top-down and bottom-up perspectives and illustrates how the core features of *Physarum*'s foraging process help solve the difficult problems that are encountered in complex systems research. The examples in this context include the design of man-made infrastructure networks [16,17], network organisation [18,19], path planning or finding [20,21], and hybrid optimisation [14,22]. Finally, Section 4 concludes with a list of unsolved problems in the field and a discussion of the potential to solve these problems by relying on *Physarum* as a source of inspiration.

2. Overview: *Physarum* in the scientific literature

2.1. Why study Physarum?

2.1.1. Physarum's plasmodium

Physarum polycephalum (literally, multi-headed slime mould) is a unicellular organism in the class Myxomycetes. *Physarum* feeds on microorganisms but also on larger food items such as fungi, and, conveniently, oat flakes in the lab [23]. All behaviours studied concerning adaptive networks are exhibited in the active vegetative stage of its complex life cycle, called the 'plasmodium'. In this stage the organism produces a large (up to 30×30 cm^2 in nature), multi-nucleated, but still unicellular, flat body extending over surfaces. During migration, the posterior exhibits tubular vein structures that bifurcate and merge towards the anterior to become a fan-like sheet at the extension front. Through these tubes the protoplasm is transported bi-directionally, oscillating in regular cycles of 1.5 to 3 minutes [24].

The body shape, and in particular the vein network, adapts to the organism's environment, such as the location of food sources and repellents, the geometry of the accessible surface, or brightness and humidity. This adaption process is not organised centrally or by any specialised tissue [25].

Despite the lack of central control, the organism can exhibit complex behaviour. For example, if spread out over an agar surface bounded to result in a maze of corridors, and then presented with food sources at two points in this maze, the organism will concentrate its body mass on the food sources, connecting them with a single vein in the almost shortest possible way through that maze [6]. Similarly the organism can reconstruct the Tokyo railway network when placed on an agar surface bounded to resemble the shape of the Tokyo bay area, with food sources placed in the same relative positions as the stations of the JR railway stations in that area. If *Physarum* is allowed to extend over this area, it will connect the food sources with a network of veins whose efficiency, cost, and robustness resemble and even surpass that of the human-made rail network [7].

2.1.2. Features of Physarum's foraging behaviour

Physarum polycephalum, in order to locate food sources or amicable environmental conditions, extends over surfaces by continuously adjusting its extension front. This is achieved by transporting protoplasm from the body's rear through its veins to the extension zone. Protoplasm is needed, too, to engulf and digest newly encountered food sources [26]. Therefore, if new food is discovered, body mass is concentrated there. Because total body mass is limited, it needs to be retracted from other areas. As a consequence, **extension** and **retraction** processes occur in the organism simultaneously but are locally separated from each other.

Extension is interesting in itself, because here the organism reacts to environmental stimuli, thus influencing **morphology** and causing the organism to exhibit **taxis**, i.e., the ability to adapt its movement to trends in the composition of its environment (e.g., concentration of diffusible food compounds, repellents, or brightness) [27].

Retraction, on the other hand, begs the question in which order tubes are resorbed to maintain a coherent body as well as to ensure that this process is energy-efficient. This has been explained with **feedback loop** dynamics [28], which can also explain the organism's ability to find shortest routes through a maze [6], recreate efficient transportation network layouts [7], and, in connection with the re-emergence of the extension front, adapt to changes in the environment [29].

These three features have been modelled individually and used for solving classical NP-hard problems (e.g., network design [18], hybrid optimisation [14,19], fuzzy shortest path problems [30], TSPs [14,31], and 0/1 knapsack problems [22]), improving the computational efficiency of existing nature-inspired evolutionary algorithms [15] (e.g., ant colony algorithms [14,31] and genetic algorithms [31]), and designing more robust and effective man-made transport networks [16,17,32,33].

2.2. Categories of Physarum-related research topics

To provide a comprehensive and quantitative classification and comparison of the existing *Physarum*-related research, we first survey papers since 2000 from the Web of Science based on keywords "*Physarum* model" and "amoeba model". To this end, we built a citation network based on the most cited 100 papers and their citing relationships. Finally, we use a community detection technique [34] to divide the entire network into several clusters. This clustering technique tries to maximise a graph's modularity, Q, which is a quantity defined and explained in Section 3.6.4. Each cluster is considered a main research domain. Fig. 1 shows the classifications and radar diagram of *Physarum*-related research.

There are two main ideas behind Fig. 1. First and rather obvious, we want to illustrate the development of the research field reviewed herein. Second and perhaps more subtle is the idea to take a realistic search example (i.e., search for terms anyone interested in *Physarum*-based algorithms may use), and then showcase how the results of such a search can automatically be sorted out to obtain an informative overview of the field (Fig. 1A) and of its development (Fig. 1B). Accordingly, we learn that searching for "*Physarum* model" or "amoeba model" yields the results classifiable into five heavily interrelated domains and one peripheral domain. We then see that some of these interrelated domains grew fast over the past eight or so years relative to the peripheral domain. All this happens without any human effort. We are just left with finding a common theme prevailing in each of the domains.

The six identified research domains are: contraction-inspired *Physarum* modelling, expansion-inspired *Physarum* modelling, applications of *Physarum* modelling, computing-oriented *Physarum* bio-experiments, other bio-experiments concerning *Physarum*, and testate amoeba-related research (peripheral). The statistical results shown in Fig. 1B reveal how much traction *Physarum*-based research gained over the past two decades, and in which direction. More

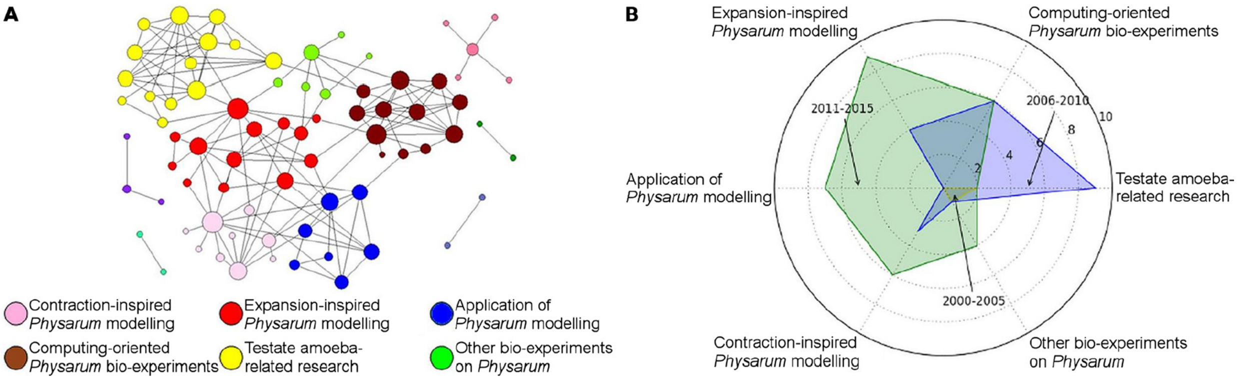

Fig. 1. Classification of *Physarum*-related papers. **A**, The results of community detection. **B**, The radar diagram of research domains.

Table 1
An overview of *Physarum*-based models.

Biological mechanisms and characteristics	Techniques	Typical model
Protoplasm concentrates on food sources, retracting body mass elsewhere while maintaining the minimum network to keep food sources connected	Cellular automata	CELL [36–38], VP-S [47,48]
Protoplasmic streaming exhibits a net movement along the concentration gradient of nutrients as a form of local behaviour	Cellular automata Agent-based system	ALCA [16,39,45] MAS-based model [11,40–42]
Physarum grows in the direction of food sources as a form of global behaviour	Agent-based system Differential equations	Gradient-based model [13] Oregonator-based model [43,44]
Positive feedback forms between the internal protoplasmic streaming and the thickness of the protoplasmic tubes	Differential equations	Current reinforcement (CR) model [7,21,28,46]

than half of the reviewed studies represent research on *Physarum*-based modelling and *Physarum*-based computing, which we expect to remain prominent research directions in relation to *Physarum* in the near future.

3. *Physarum*-based modelling

Modelling inspired by *Physarum*'s foraging behaviour has benefited from focusing on the above-mentioned characteristics, that is, *Physarum*'s morphology, tactical behaviour, and feedback loop dynamics. A general aim for *Physarum*-based and similar bio-inspired models is to achieve self-organised computability [35]. The modelling techniques to achieve this general aim can be quite different and include cellular automata [36–39], agent-based systems [13,40–42], and differential equations [7,21,43,44]. Table 1 summarises the representative literature, modelling techniques, biological mechanisms employed by *Physarum*, and the corresponding characteristics for computing.

3.1. Models based on morphology

The morphology of *Physarum polycephalum* adapts to changes in the environment [36]. Ref. [36] proposes a cellular automaton called the 'CELL model' in which the morphological behaviour is replicated. That is, when presented with food sources, an extended plasmodial sheet will move its body mass, via protrusions of protoplasm, to the food source while retracting it from other places, while still maintaining overall connectivity. In the CELL model (Fig. 2), the cells that correspond to the positions of food sources are defined as 'active zones', 'internal cells' represent the initial extent of the organism, and the active cells ('bubbles') that simulate the emerging protoplasm are generated within a grid of potential routes.

The bubble exhibits three types of behaviours: generating, moving, and replacing. Bubble generation randomly takes place in the external cells of an active zone as shown in Fig. 2C. Then, the bubble randomly moves to an internal

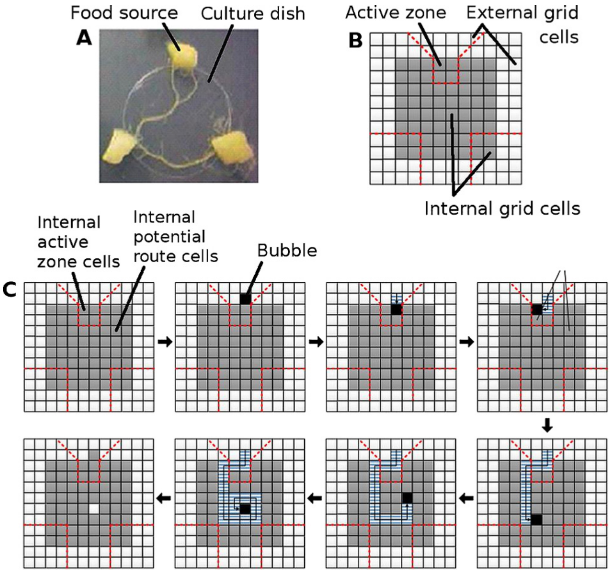

Fig. 2. Cellular automaton for simulating *Physarum*'s emerging protoplasm. **A**, Biological experiment to reveal the morphology resulting from the time-evolving behaviour of *Physarum polycephalum*. **B**, Basic setup of the CELL model. **C**, Life cycle of a bubble in the CELL model. Adapted from Ref. [36].

cell and keeps moving until there are no unvisited internal cells in the bubble's neighbourhood. Finally, the bubble replaces the state of the initial cell with the state of the final cell. The state of the initial cell is updated from external to internal, and the state of the final cell is updated from internal to external. The three behaviours—generating, moving, and replacing—are repeated until a route appears that connects the active zones. The final route, consisting only of internal cells, approximates an efficient network as produced by *Physarum polycephalum* [36,47].

Although highly effective networks emerge from the CELL model with bubbles, the model conserves the number of internal grid cells and hence cannot account for *Physarum*'s thickened protoplasm near food sources. An extension to resolve this problem is a vacant-particle model with shrinkage (VP-S), in which the number of internal cells decreases dynamically [37]. This is achieved by having a fraction of bubbles start from within the internal cells of an active zone in which case the state of the initial cell remains unchanged, but the state of the final cell, if it is outside of the active zones, is set to external. Slowly the number of internal grid cells outside the active zones decreases in favour of external cells, thus creating the effect of protoplasm concentrating around food sites. Such a feature is useful in designing efficient transport networks as described next.

3.2. *Computational applications of morphological models*

The problem of designing an effective transport network is one of the most debated problems in the study of computational intelligence. The reason for this is the problem's complexity arising from the fact that multiple factors, such as network efficiency, cost, and the tolerance to the accidental disconnection of routes, must be considered [49]. Exact conventional algorithms do construct efficient networks in a rigorous manner [50], but they are extremely time consuming [49]. Inspired by the foraging process of *Physarum polycephalum*, some evolving models, such as the CELL model [36] with bubbles and the VP-S model [37], have been used to design effective transport networks in an approximate, yet computationally economic manner.

Ref. [47] attempts to design an effective transport network using the VP-S model. The model's active zones represent an area of interest's cities (Fig. 3) and their relative positions. The model generates a number of solution networks, out of which the retained grid cells in the final version of the network are the ones that repeatedly get selected by the model, i.e., more often than a preselected threshold. Depending on the value of this threshold, the transport network designed by the VP-S model overlaps up to 75% with the real, man-made freeway network (Fig. 4). Moreover, a *post hoc* evaluation of the model performance shows that the model-generated network exhibits higher efficiency, shorter total

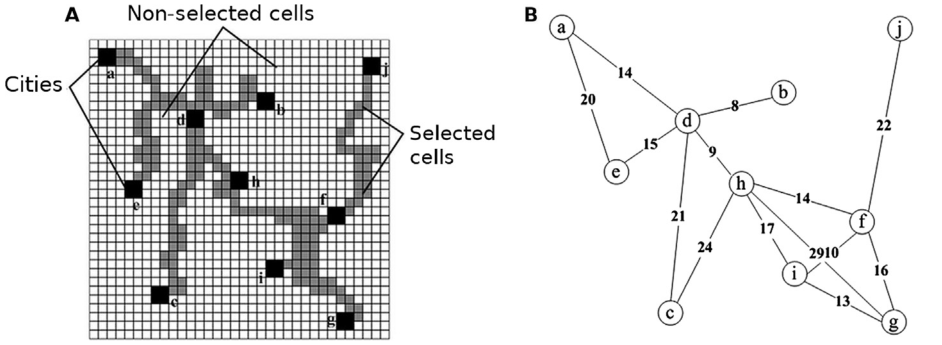

Fig. 3. Constructing an effective transport network with the VP-S model. **A**, Typical output from one model run. **B**, The final transport network generated by applying the threshold selection to many model runs. Adapted from Ref. [47].

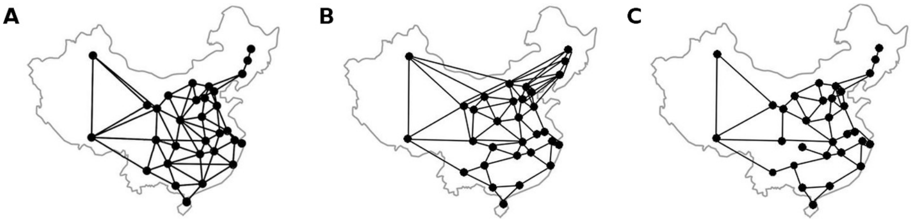

Fig. 4. Application of the VP-S model to the construction of a transport network. **A**, Topological structure of the real, man-made freeway network in China. **B**, Topological structure of the transport network constructed using the VP-S model. **C**, Intersection of networks in **A** and **B** with 54 edges. Adapted from Ref. [48].

length, and better robustness than the freeway network, which confirms the ability of the *Physarum*-inspired model to design effective transport networks.

3.3. Models based on taxis

The protoplasm of *Physarum polycephalum* preferentially moves in the direction of food sources [27]. This characteristic is called taxis of protoplasmic extension. Based on the hypothesis that *Physarum polycephalum* is able to detect the concentration of nutrients released from a food source, researchers aim to understand the underlying reason for taxis by characterising the relationship between *Physarum*'s global and local behaviour from top-down and bottom-up perspectives.

3.3.1. Top-down models

That *Physarum polycephalum* prefers to extend itself in the direction of food sources is a global phenomenon. Attempts to characterise this phenomenon resulted in multiple top-down modelling studies of which we showcase two: the oregonator-based model and the gradient-based model.

The oregonator-based model. This model simultaneously accounts for two processes: movement of the wave-shaped tip of the plasmodium and the subsequent formation of a trail of protoplasmic tubes [51]. To achieve the former, *Physarum*'s plasmodium is assumed to diffuse until a chemo-attractant (i.e., food) is encountered, upon which the diffusion process of the extension zone proximal (distal) to a higher concentration of the chemo-attractant is enhanced (suppressed). This enhancement is the strongest when the plasmodium's density is low and the food concentration is high. However, as the plasmodium's density increases beyond a certain threshold at a given location, food at this location becomes less attractive, thus allowing the plasmodium to keep following the chemo-attractant's positive gradient. Furthermore, locations without food quickly turn unattractive relative to locations with food, which is how the model suppresses diffusion in the direction of the chemo-attractant's negative gradient.

To simulate the formation of a trail of protoplasmic tubes, the oregonator-based model employs a simple threshold criterion [51]. Wherever the density of *Physarum*'s plasmodium exceeds the threshold value, the corresponding location is marked as occupied by the protoplasmic tube. Simulations are typically completed when the wave-shaped

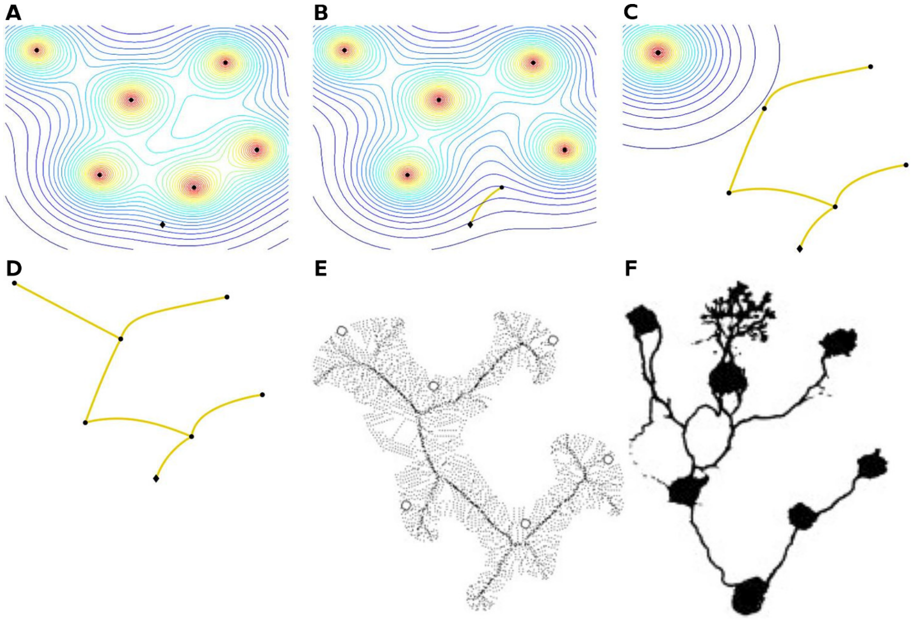

Fig. 5. Performance of the gradient-based model. **A**, The distribution of sources to be connected and the starting point of *Physarum*'s plasmodium. **B**, The plasmodium approaches the nearest source because all source intensities are the same. Once the nearest source is reached, its influence on the plasmodium vanishes, and movement to the next source begins. **C**, The process is repeated until all points are visited. **D**, The final spanning tree generated by the gradient-based model. **E**, The same spanning tree as in **D** but generated by the oregonator-based model. **F**, The same spanning tree as in **D** and **E** but formed by *Physarum polycephalum* itself. Adapted from Ref. [13].

tip of the plasmodium reaches the destination site. We will discuss an illustration of how the protoplasmic tube in the oregonator-based model evolves in time in the context of the model's computational applications below (see Fig. 9).

The gradient-based model. As in the oregonator-based model, the gradient-based model exploits the fact that food sources, such as oat flake and sugar, attract *Physarum* when inoculated on a nutrient-poor substrate [13]. In the model, a set of ellipsoid food sources, characterised by the point of origin $O_i = (x_i, y_i)$, intensity φ_i, and major a_i and minor b_i semi-axes, are assumed to exert their influence on *Physarum*'s plasmodium at point $P = (x, y)$ depending on distance $d(O_i, P) = [((x_i - x)/a_i)^2 + ((y_i - y)/b_i)^2]^{1/2}$. The influence of a single source is $f_i(P) = \varphi_i \exp[-d(O_i, P)]$, while the influence of multiple sources is simply a superposition of individual influences, $f(P) = \sum_i f_i(P)$. The unit gradient vector of field f at point P is defined by $\mathbf{N}(P) = (\partial_x f(P), \partial_y f(P))/\|\mathbf{N}(P)\|$, where $\|\cdot\|$ is the vector norm. The plasmodium at point P_t at moment t, will move to point P_{t+1} at time $t+1$ according to $P_{t+1} = P_t + \delta\mathbf{N}(P)$, where δ is a parameter controlling the speed of movement.

If all source intensities are the same, *Physarum*'s plasmodium grows in the direction of the nearest source until this source is successfully reached (Fig. 5A, B). Next, the influence of this source is removed from $f(P)$ and the plasmodium grows towards the next closest source. The process is repeated until all desired points are connected (Fig. 5C, D). The resulting spanning tree is comparable to the ones yielded by the oregonator-based model and *Physarum* itself (Fig. 5D–F).

3.3.2. Bottom-up models

The top-down models provide a macro-level explanation of the taxis of protoplasmic extension based on a set of differential equations, which ignores the micro-level interactions inside *Physarum*'s cell. Can the organism's tactical behaviour be simulated through the interactions of small, independently acting portions of *Physarum*'s plasmodium? Models attempting to answer this question positively have, in fact, been proposed and so far they have taken the form of multi-agent systems [52] and cellular automata [39].

Virtual plasmodium multi-agent system (MAS). By simulating the local behaviour of small portions of *Physarum*'s plasmodium, MAS-based models use a micro-level platform to simulate the time evolution of a *Physarum* network and explain the observed phenomena related to the taxis of protoplasmic extension [12,40–42,52,53]. In these models, *Physarum* is considered to comprise a population of particle-like agents. Each agent is randomly arranged

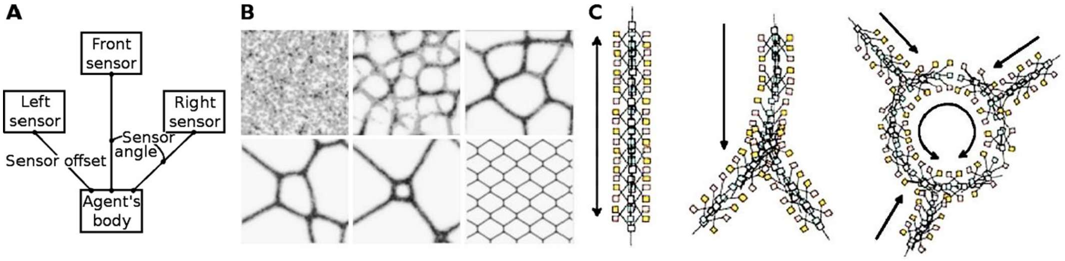

Fig. 6. Characteristics of the virtual plasmodium multi-agent system (MAS). **A**, An agent consists of three sensors and a body. The body determines the actual location of the agent, while the sensors enable movement in the direction of the increasing concentration of chemo-attractants. **B**, Time-evolving network condensation from a random initial state. With periodic boundary conditions (i.e., diffusion continues across the boundary to the opposite edge of the environment), a stable network forms and approximates a hexagonal pattern known for its minimal connectivity in two dimensions. **C**, Motifs appearing during network formation: bidirectional network path, zipping to minimise path distance, and closing of lacunas (left to right). Adapted from Ref. [41].

and works according to the same rules. The initial chaotic formation of agents can converge to a stable state and form a network that connects all data points (i.e., food sources).

The MAS-based model in Ref. [41] has three basic components: (i) the environment, (ii) agents, and (iii) the behavioural rules of agents. The concentration of nutrients released by food sources diffuses in the environment and may be damped to mimic decay. An agent uses three sensors to sense the concentration of nutrients around its position (Fig. 6A). This information is exploited in each time step to first rotate an agent in the direction of the sensor with the highest nutrient concentration, then moving it forward by one step if the new site is not occupied, and ultimately depositing a chemo-attractant trail at the new site. After a certain number of time steps, the entire system tends to form patterns that have not been encoded in the model (Fig. 6B, C).

To enhance the capabilities of MAS-based models, Ref. [42] proposes an extended multi-agent system with the following properties: (i) the number of sensors of each individual agent is reduced to two, (ii) the function of each sensor is extended to sample both nutrient and chemical trail, and (iii) a memory module is added to the architecture of an agent. Reducing the number of sensors is an improvement because the environmental information sensed by the left and the right sensor is enough to choose the agent's new direction. By distinguishing between the nutrient and the chemical trail, different weights can be attached to each of these attractants, thus adding more flexibility into the model. Finally, the memory module is in essence a motion counter which increases when the agent moves forward and decreases when the agent is stuck in place due to being surrounded by other agents. The assumption here is that more moveable agents are more important for the time evolution of the system, and hence can replicate. Less moveable agents, by contrast, are eliminated from the system. This assumption captures the basic idea behind current reinforcement-based models (see Section 3.5), adapted to agent-based systems.

The enhanced MAS is flexible enough to construct stable networks, including Steiner's minimum trees, cycles, and spanning trees. However, the agents in this system always move to the locations with higher chemo-attractants, which simulates only the contraction of a foraging *Physarum*. The other key behaviour (i.e., search) in the forming process of highly efficient *Physarum* networks [54] cannot be captured by the enhanced MAS. Nonetheless, the MAS-based models intuitively represent the process of network evolution in time, and have been successfully applied to path planning [20] and man-made networked infrastructure construction [32].

Amoeba-like cellular automaton (ALCA). To simulate the behaviour of small portions of *Physarum polycephalum*, Ref. [39] introduces an agent-based cellular automata model. The ALCA model simulates the tactical behaviour of protoplasmic extension using local interactions. This is achieved by combining three basic elements: (i) nutrient concentration in a lattice cells, (ii) the existence of protoplasm in a given lattice cell, and (iii) the moving behaviour of the protoplasm. Nutrients diffuse continuously from a lattice cell with a higher concentration to adjacent cells with lower concentrations. By contrast, protoplasm exchange takes place between two adjacent lattice cells, where the chosen direction is that of the cell with the highest nutrient concentration. If protoplasm and nutrients coexist in the same lattice cell, nutrients are consumed at a certain rate in each time step. Protoplasmic tubes are first formed after 100 time steps in accordance with the gradient of protoplasmic concentration starting from food sources.

The ALCA model can be used both for maze solving and network planning with varying degrees of success (Fig. 7). Of some interest is the direct hardware implementation of the model which results in high computational efficiency

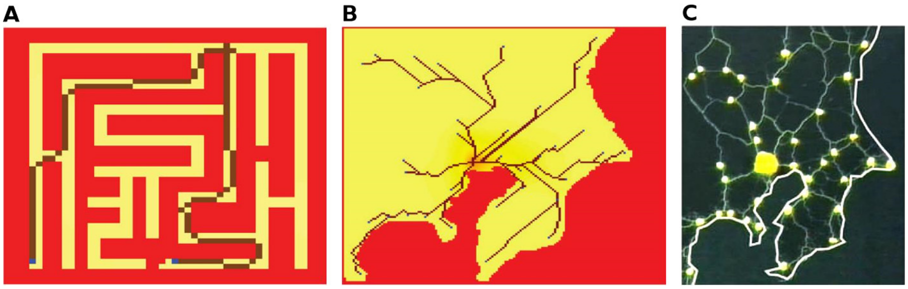

Fig. 7. Performance of the amoeba-like cellular automaton (ALCA) model. **A**, The ALCA model successfully solves mazes. **B**, In the context of network planning, while the model generates a network that spans all the desired origins/destinations, the generated network is overly centralised compared to realistic *Physarum*'s designs. Shown is the ALCA-generated "railway" network around Tokyo. **C**, For comparison with **B**, the "railway" network around Tokyo generated by *Physarum*. Adapted from Ref. [39].

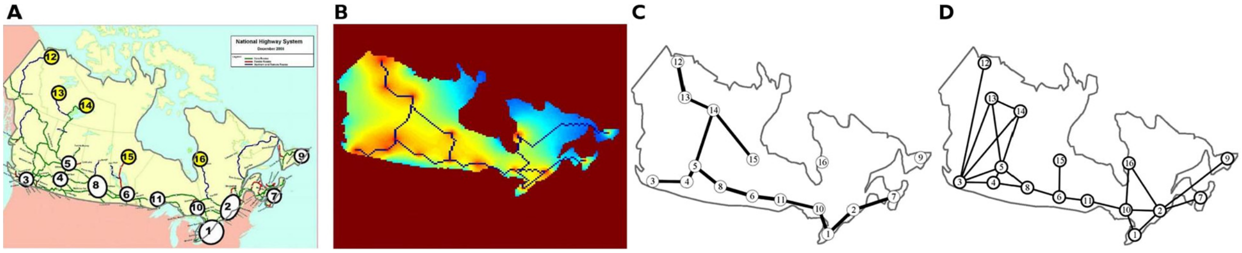

Fig. 8. Network planning with a variant of the amoeba-like cellular automaton (ALCA) model. **A**, Cities and transportation hubs of interest in Canada. **B**, The transportation network generated by the ALCA model. **C**, For comparison with **B**, the network generated by *Physarum* when the threshold probability is set to a high value (>0.74). Here, the experiment with *Physarum* is repeated many times, but in the final version of the network only edges that appear in approximately 75% of experiments are retained. **D**, Real, man-made motorway network in Canada. Adapted from Ref. [16].

[39,45]. In the context of network planning, however, the ALCA model mostly generates linear or tree-like structures, which are overly centralised relative to the real protoplasmic network (Fig. 7B, C).

3.4. *Computational applications of taxis-based models*

Computational applications of the taxis of protoplasmic extension are numerous and include network planning, route planning (maze solving and optimal route selection), and graph mining. Here, we illustrate some typical examples from the literature.

3.4.1. *Network planning*

Relying on the taxis of protoplasmic extension, Ref. [16] attempts to design efficient transportation networks using a variant of the ALCA model. The model is initialised with the geographical features of the area of interest (e.g., Canada in Fig. 8A). Food sources mimic the relative locations of major cities and transportation hubs, with different population sizes represented by the different food concentrations. The simulation results are then compared to the man-made motorway network as well as the networks generated by *Physarum* in a series of experiments (Fig. 8B–D). Interestingly, there is a considerable overlap between the model results and the relative neighbour graph. All connections that *Physarum* generates with high probability (>0.74) are produced by the model. However, the model still struggles to capture the decentralised design of man-made networks as evidenced by a considerably simpler topology of the network generated by the model relative to the motorway network.

3.4.2. *Route planning*

Here we consider two types of route planning problems: (i) solving a maze and (ii) optimal route selection. These problems are selected because they nicely illustrate the application of the tactical behaviour of *Physarum polycephalum*.

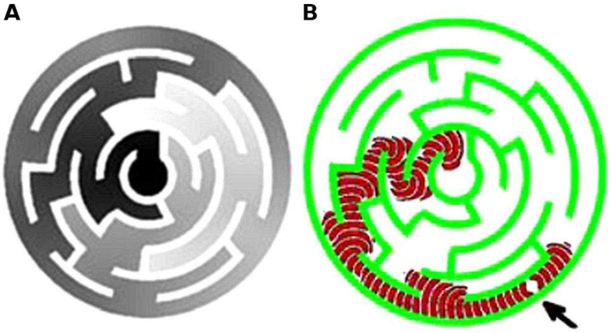

Fig. 9. Illustration of maze solving with the oregonator-based model. **A**, The maze to be solved after the nutrient has diffused. Shades of grey indicate nutrient concentration with black representing the highest concentration and white the lowest. **B**, Path traced by modelled *Physarum*'s plasmodium. Arrow indicates the starting point. Note that the diffusion of the plasmodium in the direction of decreasing nutrient concentration is quickly suppressed. Adapted from Ref. [44].

Maze solving. The task of solving a maze is typically used in demonstrating the computational capabilities of *Physarum polycephalum* in the context of complex searching problems [55]. One of the models that successfully tackle this problem is the oregonator-based model [44]. Because this is a top-down model in which *Physarum*'s plasmodium is assumed to follow the positive gradient of nutrients in the environment, mazes are solved in two steps. First the model is set up by placing a food source at the destination point. Nutrients are then assumed to leak and diffuse from this source (Fig. 9A). As the nutrient covers the maze, *Physarum*'s plasmodium is released from a starting point. The plasmodium also diffuses from this point, but the diffusion process is enhanced (suppressed) if in the direction of the positive (negative) nutrient gradient (Fig. 9B).

Multi-agent systems have also been applied to solving mazes and have even achieved better results than the oregonator-based model. An example is the *Physarum*-inspired multi-agent system (P-MAS) introduced in Ref. [11]. P-MAS is an enhanced version of the previously described model [42] in which agents always move towards locations with a higher concentration of chemo-attractants to simulate the contraction behaviour of *Physarum* during foraging. However, to solve mazes in experiments, *Physarum* first searches the environment for available food sources. To recreate this behaviour, P-MAS simulations contain two types of agents whose searching behaviour (Fig. 10A) is combined with contraction (Fig. 10B). The system readily finds all routes that solve the maze.

To explore the robustness of P-MAS, the maze is suddenly disturbed while the model keeps running. This disturbance is an extra wall that cuts off one of the maze-solving paths (Fig. 10C). Because the cut off path no longer connects the two food sources, agents located near the newly introduced wall discard this path and slowly gather around the other available path. This way, the system successfully adapts to the disturbance.

Optimal route selection. By incorporating contraction into MAS, a *Physarum*-based solution for optimal route selection emerges naturally [20]. This solution is suitable for route planning with multiple destinations or around obstacles. Here, we showcase an example in which the goal is to find a single, obstacle-avoiding path (Fig. 11). The arena contains two destination points to be connected by an efficient path. These points also act as the sources of chemo-attractant for MAS agents. To find the desired path, the arena is first completely covered by the virtual plasmodium, including the obstacles to be avoided in the final solution (Fig. 11B). Next, the plasmodium begins to contract. As the obstacles become partly uncovered, their uncovered parts start acting as sources of chemo-repellents, thus forcing the plasmodium to migrate away (Fig. 11C) and, in the process, preventing the formation of multiple paths (Fig. 11D). Eventually, MAS agents line up to form an efficient path between the two destination points.

Solving the travelling salesman problem (TSP)—which is NP-hard—is akin to finding a concave hull described below. To solve the TSP, it is necessary to identify the shortest Hamiltonian cycle of a graph. A Hamiltonian cycle (or circuit) is a closed loop through the graph which visits every node exactly once. A MAS-based algorithm to solve this problem starts from a convex hull (Fig. 12A; see below) of a given set of nodes [56]. This convex hull is filled with a dense population of agents who perform two types of actions. First, agents are attracted to nodes which act as the sources of a chemo-attractant. Second, the population of agents slowly decreases in size to mimic *Physarum*'s contracting behaviour. If attraction and contraction are carefully balanced, agents form a continuous region at all times. This region gradually morphs into an approximation of a concave hull of the initial set of nodes (see below),

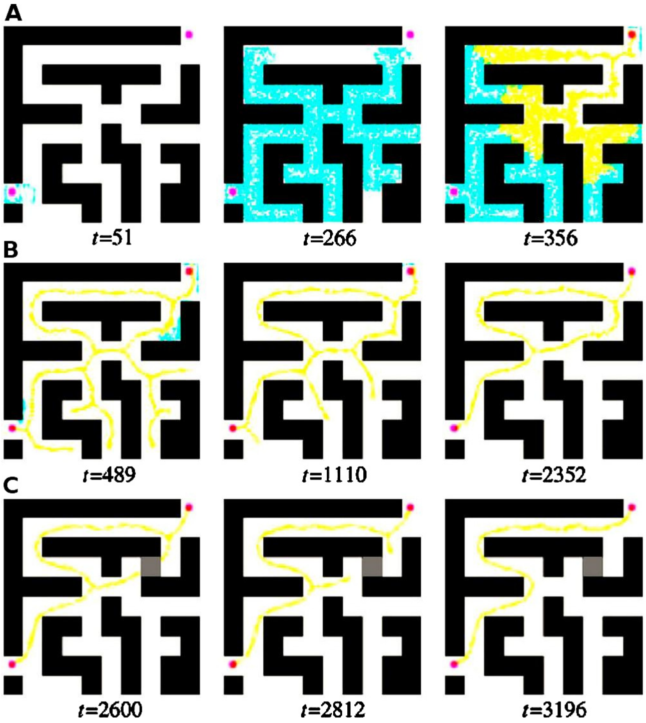

Fig. 10. Illustration of maze solving with the *Physarum*-inspired multi-agent system (P-MAS). **A**, Searching the maze for available food sources (cyan-coloured agents). Once the search completes, contraction begins (yellow-coloured agents). **B**, Contraction leads to the disappearance of plasmodium branches that have not reached any food sources. Black regions represent maze walls, whereas white regions are paths. The walls are inaccessible to agents and chemo-attractants alike. Two pink points represent the inlet and the outlet. **C**, Additional challenge to test the algorithm's robustness. At time step $t = 2600$, after the model has already solved the original maze, an extra wall (grey block) cuts off one of the maze-solving paths. Adapted from Ref. [11]. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

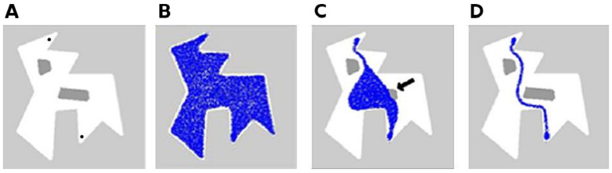

Fig. 11. Route planning with the multi-agent system (MAS). **A**, The arena (white area) contains two destination points (black) and two obstacles (dark grey). The destination points act as the sources of chemo-attractants from the beginning. **B**, Initially, the virtual plasmodium fills the arena. **C**, Contraction gradually uncovers the obstacles, whose uncovered parts then act as the sources of chemo-repellents (black arrow). **D**, A single, efficient path connecting the destination points is left as a result of the virtual plasmodium's contraction, the attracting action of the destination points, and the repelling action of the obstacles. Adapted from Ref. [20].

whereby nodes that were inside the convex hull at the beginning, one-by-one become outer (i.e., partly exposed) nodes of the final concave hull (Fig. 12B–D). When all internal nodes turn into outer ones (i.e., get partly exposed), the algorithm stops. The solution to the TSP, i.e., the shortest Hamiltonian cycle of the given set of nodes is then tracked in a clock-wise direction, starting from the uppermost node (Fig. 12E).

3.4.3. Graph mining

In addition to route and network planning, the taxis of protoplasmic extension is used to devise algorithms for graph mining. Examples in this context include Voronoi decomposition [57] and the construction of convex and concave hulls.

Given a set of points called seeds (also sites or generators), a Voronoi decomposition partitions a plane in such a way that for each seed, a corresponding region can be found consisting of all points closer to that seed than to

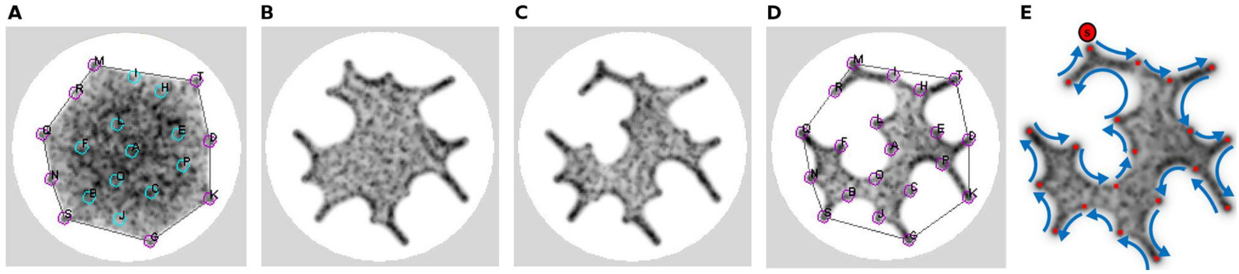

Fig. 12. Solving the travelling salesman problem (TSP) with the multi-agent system (MAS). **A**, The convex hull of a set of nodes for which the TSP is to be solved. Outer nodes (i.e., vertices of the initial convex hull) are purple-coloured, while the inner nodes are cyan-coloured. **B**–**D**, Carefully balancing the attraction of agents by nodes and the contraction of the virtual plasmodium, the MAS-based model eventually finds a fair approximation of the node set's concave hull at which point the algorithm stops. **E**, The shortest Hamiltonian cycle is read clock-wise, starting from the upper-most node. Adapted from Ref. [56].

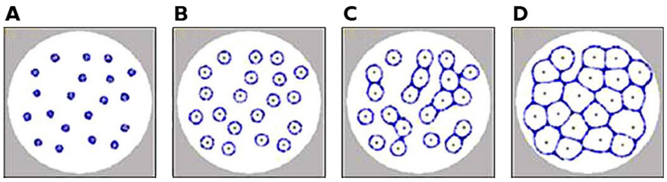

Fig. 13. Voronoi diagram generated by a multi-agent system (MAS). **A**, Plane is seeded with 20 repellent seeds. **B**, Agents of the MAS-based model disperse away from repellents. **C**, Dispersal fronts originating from different repellent seeds meet and merge as illustrated in Fig. 6C. **D**, By occupying positions as far as possible from seeds, agents form a network which partitions the plane exactly as demanded by the definition of a Voronoi decomposition. Adapted from Ref. [57].

any other seed. The MAS-based *Physarum* model naturally organises into a network that approximates the Voronoi decomposition by seeding the plane with point chemo-repellents, while a circular attractant is at the outer edge of the arena [57]. Because all points inside a single Voronoi cell are closer to this cell's seed (i.e., repellent) than the points at the cell's edge, agents are least exposed to the repellent if they spread out towards the edge (Fig. 13A–C). The more the agents accumulate at the cell's edge, the more attractant they release in their trail, thus attracting additional agents. The end result is a network whose edges are as far as possible from the seeds, which partitions the plane in a manner prescribed by the definition of the Voronoi decomposition. Attractant at the outer edge of the arena has the function of maintaining connectivity of the plasmodium network.

The convex hull of a set of points is the smallest convex polygon enclosing this set in such a way that all points of the set are on the boundary or in the interior of the polygon [25]. An intuitive analogy for visualising a convex hull is to imagine the shape assumed by a rubber band when wrapped around a number of vertically sticking nails. In constructing the convex hull of a set of points, the MAS-based model essentially exploits this rubber band analogy. A virtual plasmodium consisting of a large number of agents is initialised at the periphery of the given set of points with an attractant placed roughly at the centre (Fig. 14A). Because agents are attracted inwards, they eventually touch a point of the set whose convex hull is being computed. At this moment, nutrients start to emanate from the touched points, which strongly attract the agents nearby. The shrinking stops as soon as all points are reached by the inward motion of the virtual plasmodium (Fig. 14B). These points are then connected by bidirectional paths as shown in Fig. 6C, which form the desired convex hull.

When a set of points approximates a crescent shape, the convex hull is not a satisfactory enclosure for this set. Instead, the concave hull is preferred. The MAS-based model can be used to compute the concave hull provided that the algorithm starts from the convex hull of the set populated with a large number of agents (Fig. 14C). Subsequently, the release of attractant from the points whose concave hull is being computed is carefully balanced with the decrease in the agent population size. If the model is properly set up, the remaining virtual plasmodium after a while closely approximates the desired concave hull (Fig. 14D).

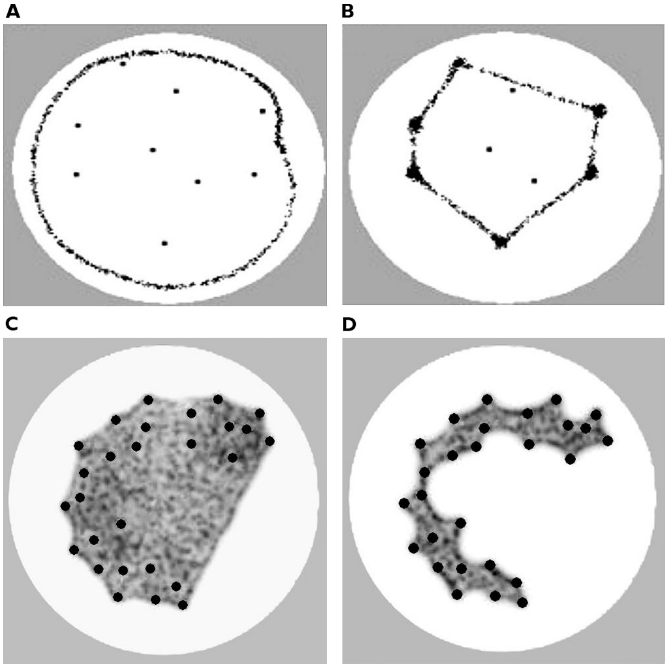

Fig. 14. Constructing convex and concave hulls with a multi-agent system (MAS). **A**, Virtual plasmodium with a large number of agents is initialised at the periphery of the set of points whose convex hull is to be computed. Agents are attracted inwards until they touch one of the preset points at which moment the touched point starts to emanate nutrients (i.e., chemo-attractant). **B**, Nutrients emanating from the touched points eventually override the inward movement of agents, causing the convex hull to form. **C**, When computing the concave hull, the model starts from a known convex hull which is densely populated with agents. **D**, Subsequently, the model carefully balances the attraction of agents by the set of points whose concave hull is being computed and the decrease in agent population size. Adapted from Ref. [25].

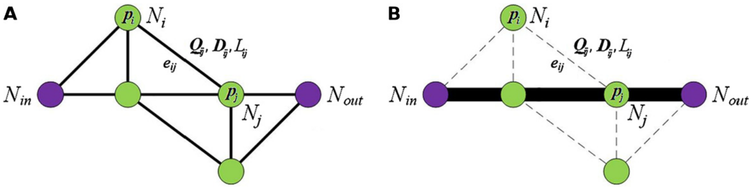

Fig. 15. Network representation of the current reinforcement model with one pair of inlet and outlet nodes. **A**, Network's initial state in which fluxes through all edges are the same. **B**, Steady state of the network in which non-zero fluxes exist only along the shortest path between the inlet and the outlet.

3.5. *Models based on current reinforcement dynamics*

In maze-solving biological experiments, the plasmodial pieces of *Physarum* are first deployed in a maze, and food sources are placed at both the inlet and the outlet of the maze. As the experiment progresses, the tubes that do not connect to food sources or do so inefficiently shrink and disappear. By contrast, the tubes that connect food sources efficiently become increasingly thicker. Eventually only the protoplasmic tubes comprising the shortest path between the inlet and the outlet remain [6].

A modelling assumption to capture the phenomena observed in maze-solving experiments is a feedback relationship between the flux inside the tubes and their thickness. Specifically, tube thickness is a consequence of the Hagen–Poiseuille flow through them. All other things being equal, shorter tubes carry more flow and, therefore, grow. The thus increased radius will allow for more flow through the tube, leading to a positive-feedback loop [28,58]. Models that exploit this positive feedback dynamic are called current reinforcement (CR) models.

The mathematical formalism behind a CR model is easier to understand by referring to a corresponding network representation (Fig. 15). In this representation, protoplasmic tubes are assumed to be the edges connecting a set of

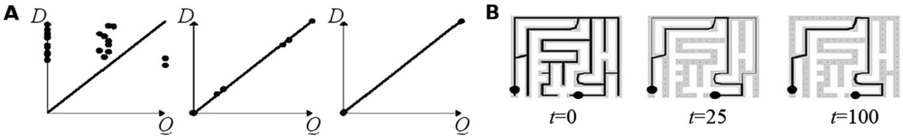

Fig. 16. How current reinforcement models solve mazes. **A**, Tube conductivities are randomly initialised, but with time they become increasingly polarised. Some conductivities converge to a maximum value, while others converge to zero. **B**, Network representation of *Physarum*'s plasmodium from **A** overlaying the maze to be solved. With time, only the tubes that span the shortest path between two food sources remain. Adapted from Ref. [21].

nodes of a graph. N_{in} and N_{out} represent the inlet and outlet nodes of the graph, respectively. D_{ij} measure conductivity (a function of the radius) of edge (i.e., tube) e_{ij} between nodes N_i and N_j. The tube conductivity increases from an initial value if the tube expands and decreases if the tube shrinks. Q_{ij} is the flux through e_{ij}. The relationship between D_{ij} and Q_{ij} is described by equation

$$Q_{ij} = \frac{D_{ij}}{L_{ij}}(p_i - p_j), \tag{1}$$

where L_{ij} represents the length of e_{ij}, and p_i and p_j are the pressures at N_i and N_j, respectively. To conserve mass, the sum of input fluxes at each node is taken to be equal to the sum of output fluxes. Only node N_{in} has a net inflow $-I_0$, while node N_{out} has a net outflow I_0. All this is summarised by the following equation

$$\sum_i Q_{ij} = \begin{cases} -I_0, & \text{if } j = in \\ I_0, & \text{if } j = out, \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

which specifies fluxes through all edges and pressures at all nodes. This information is then used to simulate the time evolution of conductivities according to

$$\frac{d}{dt}D_{ij} = f\big(|Q_{ij}|\big) - rD_{ij}. \tag{3}$$

New conductivities are in an iterative process fed back to Eqs. (1)–(3) to calculate new fluxes and pressures. This iterative process continues until the network finally converges to a steady state. The simplest CR model described here can be extended to incorporate multiple inlet or outlet nodes [31].

Function $f(|Q_{ij}|)$ can be any positive, monotone increasing function, but two forms are mainly used in the literature [28]. Simple $f(Q) = Q^u$, $u > 0$ performs better when searching for the shortest path, while more complex $f(Q) = (1+a)Q^u/(1+aQ^u)$ performs better when designing effective networks.

A CR model starts its search for the shortest path in a maze by having tube conductivities randomly initialised with relatively high values. Based on Eqs. (1)–(3), these conductivities converge to one of two equilibria: $D = 0$ or $D = D^* > 0$ (Fig. 16A). The tubes whose conductivities tend to zero gradually disappear from the maze (Fig. 16B). The first to disappear are dead ends. If two or more paths solve the maze, those that are longer eventually disappear as well, and the model ultimately converges to the shortest path.

Ref. [7] extends the CR model from maze solving to network design by iterating the described algorithm based on Eqs. (1)–(3), but with inflow and outflow points randomly selected in each iteration. During an iteration, the internal protoplasmic flow and the thickness of tubes are regulated until conductivities stabilise. The generated network resembles the protoplasmic network of *Physarum polycephalum* and exhibits a good performance in the context of efficiency and fault tolerance. By adjusting the parameters, the mathematical model has been successfully applied to re-creating Tokyo's rail system [7].

Current reinforcement dynamics is fully understood only when put into the biological context of the expanding organism [59]. In a natural setting, *Physarum* extends over a surface expanding in some directions, while retracting its body from others [60]. The expansion and contraction processes occur concurrently in different parts of the same organism. To understand how they influence each other, it therefore makes sense to bring these processes together in one comprehensive model.

The model consists of three parts representing key functions of *Physarum*: a reaction–diffusion model describes the expansion at the extension front, driven by the sol–gel transformation [61]; a Smith–Saldana model recreates the actomyosin contraction waves driving the transportation of protoplasm through the organism [62]; and a

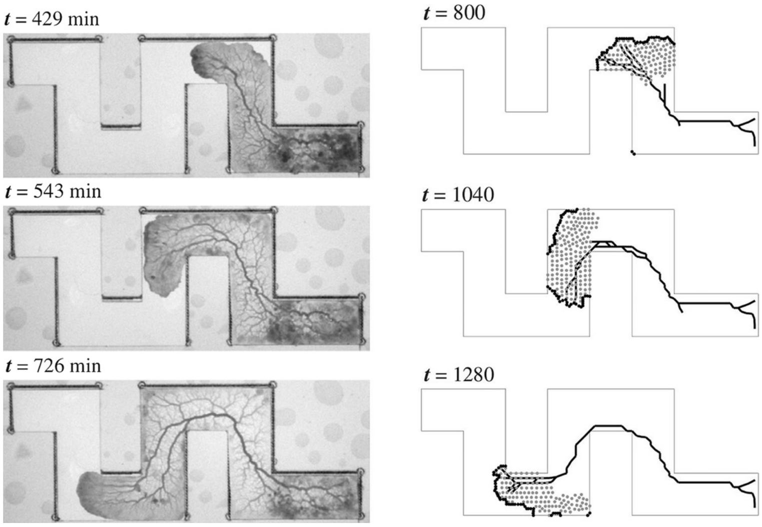

Fig. 17. Result of the numerical evaluation of the comprehensive *Physarum* model. Left column: stills from an experimental run at various times, for comparison. Right column: the model's state at various times chosen for their similarity to the experiment. Thick solid curve: position of the extension front. Medium solid line: major vein. Dotted area: area of high concentration of solation factors, representing the area of the organism's extension cone. Reproduced with permission from Ref. [59].

Tero–Kobayashi-type current reinforcement model represents the formation and behaviour of the organism's tube network [28]. The resulting set of eight inter-dependent differential equations are evaluated numerically on two layers: one continuous, handling the reaction–diffusion system and the contraction waves, the other a fine mesh, handling the evolution of the tube network.

The only input into the model, apart from setting the parameters, is the geometry of the arena. Given just this, the model reproduces the expanding behaviour of the real organism as well as the time-evolution of its vein network in a qualitatively and quantitatively accurate way (Fig. 17). The strength of this approach, therefore, is that it recreates the phenomenology of *Physarum*'s behaviour by modelling physiologically meaningful processes. Consequently, the model's results are interpretable in a biological context, e.g., the emergence of current reinforcement dynamics in *Physarum* as a means to transport body mass in an energy-efficient way during exploratory migration (see also Ref. [63]).

The comprehensive *Physarum* model assumes an infinite supply of body mass from the rear and cannot, therefore, describe the exploration of unbounded, open space [64]. In such a setting, as is true for *Physarum*, the limitation of body mass forces the organism to choose a direction, a choice that is side-stepped in the model. Also, the model does not incorporate dynamics in response to encountering food, and thus—despite its comprehensive nature—is not a full description of *Physarum*'s behaviour [27,65].

The CR model caused much hype due to its ability to find shortest paths without central control and to self-adjust network designs. Applications to various optimisation problems are ubiquitous and include the fuzzy shortest path problem [30,66,67], the linear transportation problem [19], the $0/1$ knapsack problem [14,22] and the TSP [31,68]. Although methods to prove the upper bound of the CR model's computational complexity have been proposed [46] and an estimate of $O(N^3)$ has been given by establishing a new path selection strategy and termination conditions [69], the exact computational time requirements of the CR model remain to be proven.

3.6. Computational applications of the current reinforcement model

Computational applications of the CR model are particularly strong in the context of network and route planning. An exciting recent development is network planning in expanding domains, especially because the approach to resolve this problem mimics *Physarum*'s physiology very closely [59]. The CR model can also be used to gain insight into the network structure, the result of which is an efficient immunisation strategy even in the face of incomplete information on nodes and links in the network [70]. Finally, the CR model can be combined with other algorithms to improve their performance.

3.6.1. Network and route planning

Network planning based on the CR model. The positive feedback loop in the wake of *Physarum*'s foraging process has been successfully characterised by the CR model [28]. This success inspired applications in the construction of highly effective and fault-tolerant networks, including transportation [32] and Bayesian networks [71,73].

The problem of designing a "good" transportation network can be formally expressed in the language of network science (or graph theory) [32]. Given a set of nodes (or vertices), V, the idea is to find a set of links (or edges), E, which satisfies certain optimality criteria. For example, efficiency can be measured as the inverse of the sum of minimum distances between all pairs of nodes in V. A "good" network would then maximise such a measure of efficiency. Furthermore, the total cost of building the network is another important quantity, which is usually expressed in terms of the total length of all edges in $E \subseteq V^2$. By knowing the average economic cost of building per unit of length, the total cost can be compared to the available budget. In constructing a "good" network, therefore, the efficiency is not blindly maximised. Instead, efficiency maximisation is performed under the constraint of keeping the total cost below the predetermined budget. Finally, achieving high fault tolerance (or robustness) is another desirable property. An appropriate measure in this context is the probability of the network remaining connected after a single link is removed, where connectedness is defined as the existence of a path between any pair of nodes. Here, the more robust the network, the higher is the probability that the network remains connected.

Transportation networks are usually weighted because connecting a metropolis with another metropolis is presumably more important in terms of demand than connecting two local hubs. A popular method to assess demand is the gravity model [32]. In this model, the outward traffic flow from city i to city j is given by $F_{ij} = GM_i^{\alpha} M_j^{\beta} / C_{ij}^{\gamma}$, where M_i and M_j measure city sizes (population, annual budget, contribution to GDP, etc.), C_{ij} is travel cost (for which a good proxy is distance), and G is an empirically determined proportionality constant. Exponents α, β, and γ depend on what kind of city size data (demographic vs. economic) and what means of transport (roads vs. rails vs. flights) are considered. Once demand is estimated using the gravity model, these results are fed into the CR model for network construction. Specifically, $I_0 = F_{ij}$ in Eq. (2).

The CR model is first run for each pair of cities separately for a total of k iterations to produce conductivity matrices D_{ij}^k, which are then summed to yield one global conductivity matrix $D^k = \sum_i \sum_j D_{ij}^k$. Matrix D^k is renormalised by its maximum element to keep conductivities in the range between 0 and 1. The CR model is then re-run to update the global conductivity matrix until the values in this matrix stabilise. The obtained conductivity matrix defines the desired transportation network.

Another domain in which the CR model has been successfully applied is the construction of Bayesian networks. Here, the key is to infer the network's conditional probability distribution. Specifically, a node in a Bayesian network represents an event in the probability space, whereas an edge between two nodes i and j represents the conditional probability, denoted $P(X_i|X_j)$, that event X_i will occur given that event X_j occurred. Taking advantage of the computational capability of the CR model, Ref. [71] proposes a bio-inspired method for constructing a Bayesian network.

First, a set of features is defined (e.g., blood counts in a certain type of patient). This set is assumed to form a fully connected graph, but the length of the connections is defined as a decreasing function of the Pearson correlation coefficient [72]. Two highly correlated features are thus close to one another, while two weakly correlated features are relatively far. The CR model is then run with every possible pair of nodes taken as a source and a sink in order to estimate the shortest indirect path between this pair. After each iteration of the CR model, a score of the surviving edges, i.e., those with non-zero diffusivities, is incremented. Finally, the highest scored connections are combined to form a Bayesian network. Put more intuitively, the algorithm finds indirect paths that explain where the correlation between two nodes comes from.

We singled out some of the more successful applications of the CR model. However, our exposition is by no means complete. For instance, the CR model has been successfully applied to real-world problems such as routing in wireless sensor networks [74], power network design [33], the minimal exposure network [75], and the Steiner problem [78].

Route selection based on the CR model. *Physarum*-inspired computations are widely used to resolve route selection problems, including the fuzzy shortest path [30,66,67], the constrained shortest path [76,77], and the multi-objective shortest path [68,79]. In fact, some classic optimisation problems are solvable with *Physarum*-based models by casting them into a route selection form. Ref. [22], for example, reformulates the $0/1$ knapsack problem as a shortest path search and proceeds to perform this search in a fast manner with the help of the CR model.

A particularly well-known application of the CR model in the context of route selection is the multi-objective travelling salesman problem (TSP) [68]. The multi-objective TSP searches for a Pareto-optimal Hamiltonian cycle in a complete graph. Because different objectives may be better satisfied by different routes, Pareto optimality is needed to single out a route which cannot be improved upon with respect to any one objective without performing worse with respect to another objective. An algorithm proposed in Ref. [68] considers two opposing objectives. On the one hand, the algorithm is trying to minimise the total road length that needs to be traversed in order to visit all nodes and return to the node of origin. On the other hand, the algorithm is also trying to minimise road traffic, which is assumed to be inversely proportional to road length because travellers are likely to choose a shorter road over a longer alternative. If we denote the length between nodes i and j with L_{ij}, and traffic load with T_{ij} (where by definition $T_{ij} \propto 1/L_{ij}$), then the link connecting these two nodes is characterised by pair (L_{ij}, T_{ij}). If we furthermore focus on node i and examine links to all its neighbours $j \in N_i$, where N_i is the set representing i's neighbourhood, we can define dominance index I_{ij} as follows: $I_{ij} = 0$ if $L_{ij} \leq L_{ik}$ and $T_{ij} \leq T_{ik}$ for all $k \in N_i$; $I_{ij} = 1$ if there is exactly one $k \in N_i$ for which $L_{ij} > L_{ik}$ or $T_{ij} > T_{ik}$; $I_{ij} = 2$ if there are exactly two $k \in N_i$ for which $L_{ij} > L_{ik}$ or $T_{ij} > T_{ik}$; and so on. Dominance index I_{ij} plays a prominent role in the algorithm because the goal is to find a route with minimal I_{ij}. Note that there may be multiple solutions which minimise I_{ij}. These are collectively called the Pareto front.

The algorithm in Ref. [68] attempts to identify the Pareto front by starting from a single randomly generated solution, i.e., a random Hamiltonian cycle. Nodes along this path are seeded with agents, which at each time step can choose either (i) to move to the next node on their current path or (ii) to explore the space by creating a new link to any of the nodes outside of their current path. This link creation connects nodes i and j with probability proportional to $1/(1 + I_{ij})$, where j can now be any node which is presently unconnected to i. If an agent decides to move on their current path, i.e., without creating a new link, then the probability of choosing a particular direction is guided by the results of the CR model. When all agents make their decisions, the algorithm updates the Pareto front with newly discovered Pareto-optimal paths, if any. This is followed by an iteration of the CR model to help guide agents in their subsequent movements.

In the CR model, each path is assumed to have certain radius r_{ij}—an analogue to diffusivity in Eq. (1)—which is dilated or shrunk in accordance with Eq. (3). Dominance indices are used to transform radii into fluxes, i.e., in Eq. (1) L_{ij} is replaced with $1 + I_{ij}$, where one is added to avoid zeros in denominators when $I_{ij} = 0$. Finally, the probability of moving in the direction of node j from node i is calculated from fluxes by $Q_{ij}/\sum_{k \in Ni} Q_{ik}$. In practice, the algorithm employs two instances of *Physarum* simultaneously in order to "exchange information" between them and retain only the links that are important in both instances.

Another interesting application of the CR model has been proposed in Ref. [80] which examines the characteristics of a novel *Physarum*-inspired routing protocol for wireless sensor networks (WSNs). Such networks consist of a large number of sensor nodes covering a certain domain of interest. A typical sensor node is a low-cost, short-range wireless transceiver, equipped with a low-end processor for basic computations and a battery for supplying power. In many practical situations, sensor nodes are intended to function with a single battery pack for several years. Especially problematic in this context is the many-to-one communication between nodes in a typical WSN, which may cause power failure near or at the sink node because information from all nodes is routed through the sink. Further consideration in designing WSNs is the efficient routing of information in multihop transmission sequences because a typical WSN may contain hundreds of sensor-nodes (Fig. 18).

The algorithm proposed in Ref. [80] seeks to minimise energy depletion for transmitting information through WSNs, while maximising the routing efficiency. In doing so, the algorithm takes into account the connection quality, the connection distance, the packet loss rate, and the deviation angle between the sink node and the next-in-line node through which transmission takes place. Specifically, when working with WSNs, pressure differences in the standard CR model are replaced with (i) connection qualities. Tube lengths become (ii) physical lengths between the source and the possible next-in-line nodes modified to account for the packet loss rate. Finally, diffusivities are substituted with (iii) projections of the distance between the source and the possible next-in-line nodes in the direction of the sink node, which are obtained using the mentioned deviation angle, and interpreted as a node's potential for transmission. Quantities (i) to (iii) are inserted into an analogue of Eq. (1) to calculate virtual communication fluxes Q_{ij}. These fluxes are then plugged in Eq. (3) to identify the next node along the gradient of the transmission potential. The process is repeated until the sink node is reached (Fig. 18). The algorithm can be slightly modified by making virtual

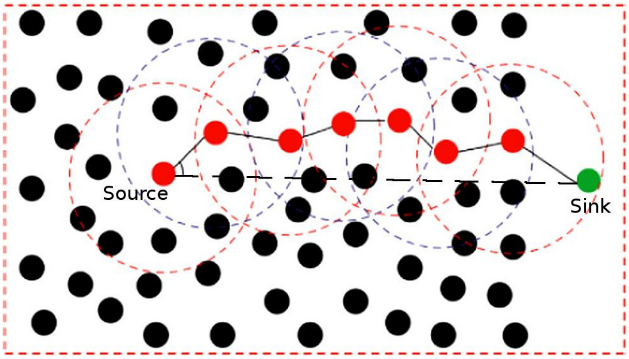

Fig. 18. Signal routing through a wireless sensor network (WSN) by means of a current reinforcement (CR) model. Communication in a typical WSN is many-to-one because any of the black coloured nodes can be sending a signal to the sink node. WSNs are furthermore characterised by multihop transmission meaning that the signal sent out by a source hops from node to node to reach the sink. A signal receiving node always seeks in its neighbourhood (dashed circles) a suitable next-in-line node. Selection is such that the next-in-line node is in the direction of the gradient of the transmission potential. This gradient approximates the line connecting the source and the sink (dashed line) as closely as possible given the location and the residual energy of possible next-in-line nodes. Adapted from Ref. [80].

communication fluxes dependent on the residual energy of the possible next-in-line nodes. In this way, nodes with low battery charge can be avoided, thus preventing data losses and extending the operational life of the WSN.

3.6.2. Efficient networks in expanding domains

Finding efficient network configurations for a given set of nodes is a challenging task. An added difficulty is encountered when a semi-static network has to be constructed in an expanding domain. That is, in a situation where edges cannot be freely added or shifted at any time, but can be added only in a region newly acquired, it becomes conceptually difficult to construct an efficient network, as the final layout of nodes is unknown at the time of network construction. This characterisation is valid for many real-world infrastructural networks, e.g., train and road networks or the internet, as the expansion of these networks stimulates the emergence of new hubs (e.g., Ref. [81]), yet it is impractical to re-connect the network *ex post* to adapt to the new circumstances. How can efficiency be achieved in such circumstances? This is a much more realistic question than assuming total knowledge of the position of the hubs, but allowing for their *de novo* re-connection (as in Refs. [7] and [82]).

Physarum is, once again, a good system to look at, as its vein system is similarly constructed during exploratory migration. The organism needs its veins to move its body mass around and will use the same veins to connect to food sources, but the location of food sources is unknown at the time of the veins' construction. In addition, constructed veins can be disassembled if unused, but within the area already spanned, new veins are not—at least for some considerable time—systematically built anew [83].

Physarum's vein network development and extension front dynamics are interdependent [59]. A corollary to this statement is that it should be possible to infer one from the knowledge of the other. For example, one should be able to predict the spatiotemporal development of the vein network given only the history of the extension front expansion. An algorithm that achieves this working only with the positions of the extension front at regular time intervals is presented in Ref. [59] (Fig. 19). This algorithm is based on the idea that, directly behind the organism's extension front, proto-veins are formed by protoplasmic flow and the flow follows a path of least resistance. That is, the proto-veins follow the shortest paths connecting points in a extension front's previous and current positions. Then, incorporating the quintessence of current reinforcement dynamics, veins formed in this manner survive only if they connect to veins formed in later time steps. Veins that hit dead ends disappear and take all preceding vein segments with them, as they are supposed to not carry any further flow.

To speed up computation, the algorithm evaluates the time series of extension front positions from the last to the first frame. That is, the algorithm first connects the points of the extension front in the last frame with the closest points in the penultimate extension front position. Then, only the points thus arrived at are considered and connected with the points closest to them in the extension front position just before that. This procedure is repeated backwards until the first frame. In this way, veins that do not belong to the set of finally surviving veins do not have to be considered, while the end result is the same.

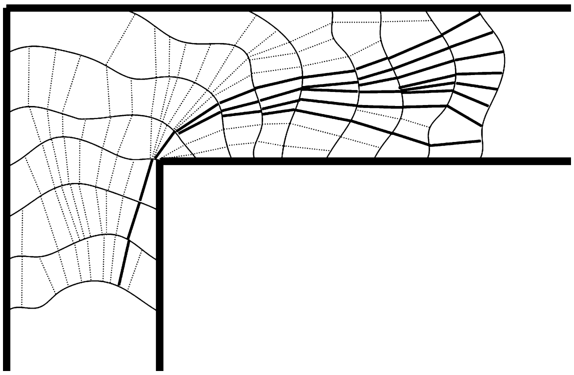

Fig. 19. Schematic illustration of the vein prediction algorithm. Thick solid lines: outlines of the experimental arena. Thin solid lines: extension fronts at regular time intervals (experimental data). Thin broken lines: shortest connections between points in consecutive extension fronts that do *not* connect to the final extension front position. Medium solid lines: shortest connections between points in consecutive extension fronts that *do* connect to the final extension front position. Note the emergence of a single main vein at some distance behind the extension front, as well as its centre-in-centre trajectory. Also note that the same result can be obtained by running the process from the last to the first position.

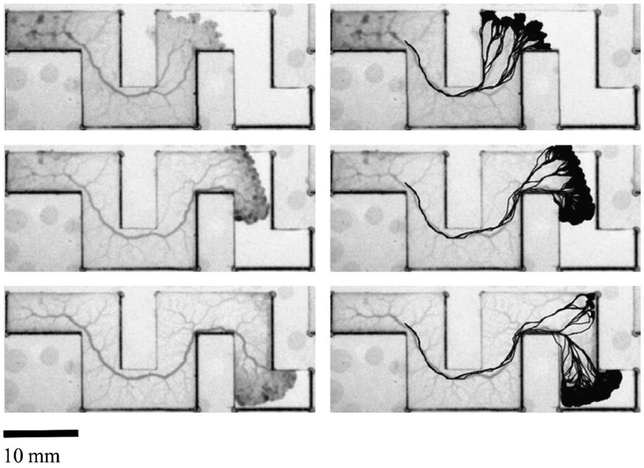

Fig. 20. Results of the vein prediction algorithm. Left column: stills of experimental run at various times. Right column: predicted vein (in black) superimposed over experimental footage, obtained by connecting with straight lines the points closest to each other in the extension fronts at regular time intervals starting with the time point of the still to the right and working backwards in time. Reproduced with permission from Ref. [59].

The result of the described algorithm is a vein network resembling a tree, with its single trunk where the organism started, branching out at some distance before the final extension front position, and then continuing to bifurcate until its smallest branches end at the point of the organism's farthest extent. This prediction is qualitatively very similar to *Physarum*'s vein pattern and is quantitatively very accurate (Fig. 20).

What is most remarkable about the resulting vein trajectory, though, is its overall efficiency: In the studied arena shapes, the predicted and real veins were only 6 percent longer than the globally shortest possible path, which would have required prior knowledge of the arena's total layout to construct. Therefore, local optimisation combined with current reinforcement dynamics promise to yield good solutions for the task of finding efficient networks in expanding domains, a task that is closely related to the development of real-world infrastructural networks.

3.6.3. Hybrid optimisation algorithms with the current reinforcement model

The CR model may be unsuitable for some optimisation problems on its own, but it still can enhance the existing algorithms. In particular, the positive feedback loop in the wake of *Physarum*'s foraging process, which is successfully captured by the CR model, can be integrated into heuristic and meta-heuristic algorithms (e.g., ant colony [31], genetic [15], and random walk [84] algorithms). Such integration has been shown to improve robustness and search ability of

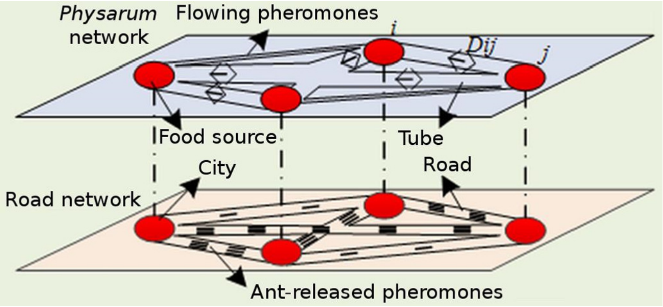

Fig. 21. Combining ant colony optimisation (ACO) with the current reinforcement (CR) model. The usual ACO algorithm in the lower plane operates by having ants deposit pheromones on traversed paths, where the shorter paths are favoured over the longer ones. Simultaneously, the CR model in the upper plane selects only the "efficient" paths from all paths spanned by a given set of points. This selection is fed back into the ACO algorithm to improve the algorithm's performance. Adapted from Ref. [14].

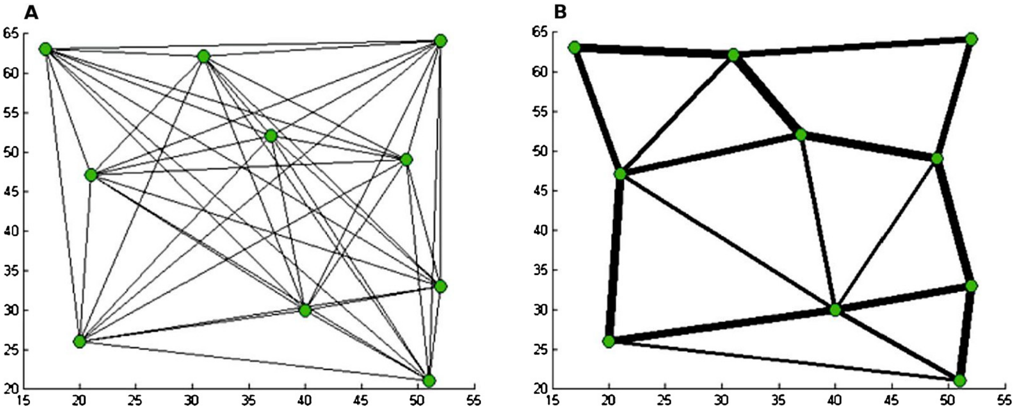

Fig. 22. Role of the current reinforcement (CR) model in aiding ant colony optimisation (ACO). **A**, The full graph spanned by the set of points (i.e., locations to be visited) for which the shortest Hamiltonian cycle is sought. **B**, Trimmed graph of "efficient" paths after the CR model converges. These "efficient" paths are given extra pheromone on top of the amount deposited by ants who traversed them. Adapted from Ref. [31].

the original algorithms, and to accelerate their convergence speed. Here, we first illustrate how the CR model aids the ant colony optimisation (hereafter ACO) techniques [85–87] in solving the travelling salesman problem (Fig. 21).

In an iteration of the ACO algorithm, a number of ants move from their current location to a previously unvisited one, where the probability of selecting a particular destination depends on the level of pheromones deposited between the point of origin and the destination point, as well as the distance between these two points. The process continues until all ants complete a Hamiltonian cycle. Once this is done, each ant is assumed to deposit a fixed amount of pheromone on the traversed path. Pheromones are further assumed to decay with time. Accordingly, the amount of pheromone on a path between locations i and j, τ_{ij}, at time $t+1$ is given by $\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t)$, where ρ is the pheromone decay rate, and the change due to newly deposited pheromones, $\Delta\tau_{ij}(t)$, is dependent on a particular implementation of the ACO algorithm.

Because traversing large distances is discouraged in all ACO implementations, ants are supposed to gradually increase the amount of pheromones along relatively short Hamiltonian cycles. The process continues until the shortest cycle is identified. In reality, however, the algorithm's output is dependent on the initial distribution of ants and their chosen directions, often causing the algorithm to end up in a local minimum. Combining the ACO algorithm with the CR model is an attempt to resolve this problem (Fig. 21).

The CR model is run with the purpose of trimming the full graph spanned by a given set of points (i.e., locations to be visited) in such a way that only "efficient" paths remain (Fig. 22). These "efficient" paths receive an extra amount of pheromones on top of what ants deposit while visiting different locations. In mathematical terms, $\Delta\tau_{ij}(t) = \Delta\tau_{ij}^{ACO}(t) + \Delta\tau_{ij}^{CR}(t)$, where the first summand is the amount of pheromones deposited by ants and the second summand is the addition along paths selected by the CR model.

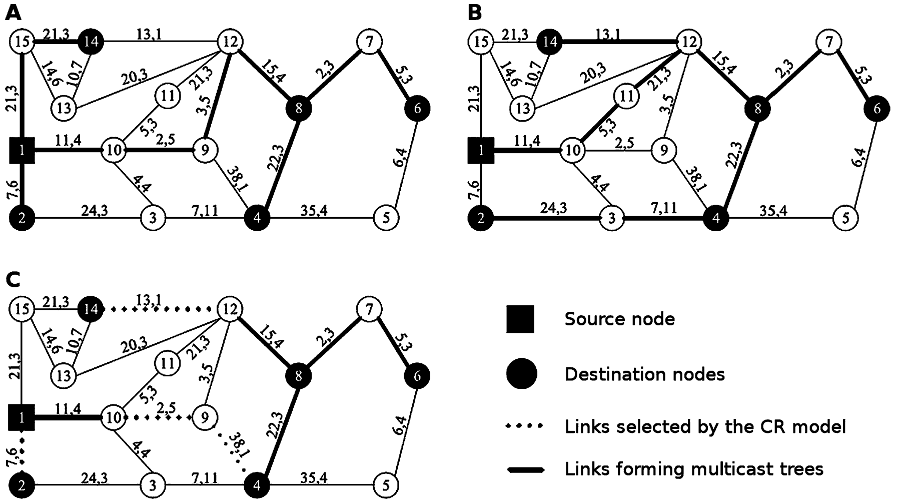

Fig. 23. Role of the current reinforcement (CR) model in aiding the genetic algorithm. **A**, An example of a multicast tree is shown in which bold links connect the source node (black square) with the destination nodes (black circles). Each link is marked with a pair of numbers indicating the cost of sending a signal through and the corresponding delay. This multicast tree is used as the first chromosome in the genetic algorithm. **B**, Another multicast tree used as the second chromosome in the genetic algorithm. **C**, Links reserved by the genetic algorithm, i.e., those that are common to both chromosomes (in **A** and **B**), generally fail to produce a multicast tree. To form such a tree while keeping the reserved links, a fast variant of the CR model is used. In this case, the CR model selects the dotted links as the ones that secure the quality of service and yield the minimum operational cost. Adapted from Ref. [88].

Another example of hybrid optimisation is the genetic algorithm combined with the CR model. This hybrid optimisation technique has been proposed in the context of securing the quality of service in mobile *ad hoc* networks [88]. When a set of mobile devices maintains connectivity by functioning cooperatively and without any aid from fixed infrastructure, these devices are said to have self-organised into a mobile *ad hoc* network. Securing the quality of service in such networks implies guaranteeing (i) a certain minimum bandwidth and (ii) a certain maximum delay, while keeping the operational cost of the network as low as possible.

Mobile *ad hoc* networks operate by forming multicast trees which connect a single source with a number of targeted destinations (Fig. 23A). Some variation of a genetic algorithm is often used to minimise the operational cost. The genetic algorithm attempts to combine the properties of two multicast trees (i.e., two parent chromosomes) with the lowest cost (i.e., the highest fitness) in order to generate an even better multicast tree (i.e., an offspring chromosome with even higher fitness). In the present example, a way to generate the offspring tree is to first "reserve" those links that appear in both parent trees (Fig. 23). However, the reserved links are often not enough to form the whole offspring tree (Fig. 23C). Because the strength of the CR model lies precisely in finding the shortest, most economical routes, the missing links are filled with a fast variant of this model.

3.6.4. *Community detection*

Community detection is a long-standing problem in the structural analysis of complex networks, aiming to help us understand and predict the characteristics and functionalities of such networks. The underlying idea is to divide network nodes according to the principle that they are sparsely connected between communities, but relatively densely connected within communities. Because the definition of a community is somewhat arbitrary, and in complex networks there may be a large number of constituent nodes, designing a high-accuracy, low-computational-cost algorithm remains a matter of continuous interest.

Ref. [15] represents an attempt to improve community detection with the help of the CR model. The main idea in this context is that, instead of considering a single inlet and a single outlet as in Eq. (2), there are multiple outlets accompanying each inlet (see also Ref. [70]). Specifically, each node i in the network is selected once as an inlet, while all the other nodes serve as outlets. The CR model is then run to generate i's diffusivity matrix D_i^T, where T is the number of model iterations. After a similar procedure is performed for all network nodes (the number of which is $|V|$), an average diffusivity matrix is calculated using $D^T = \sum_i D_i^T/|V|$. This diffusivity matrix is then used to aid community detection by, for instance, a genetic algorithm or an ant colony optimisation algorithm [15,89].

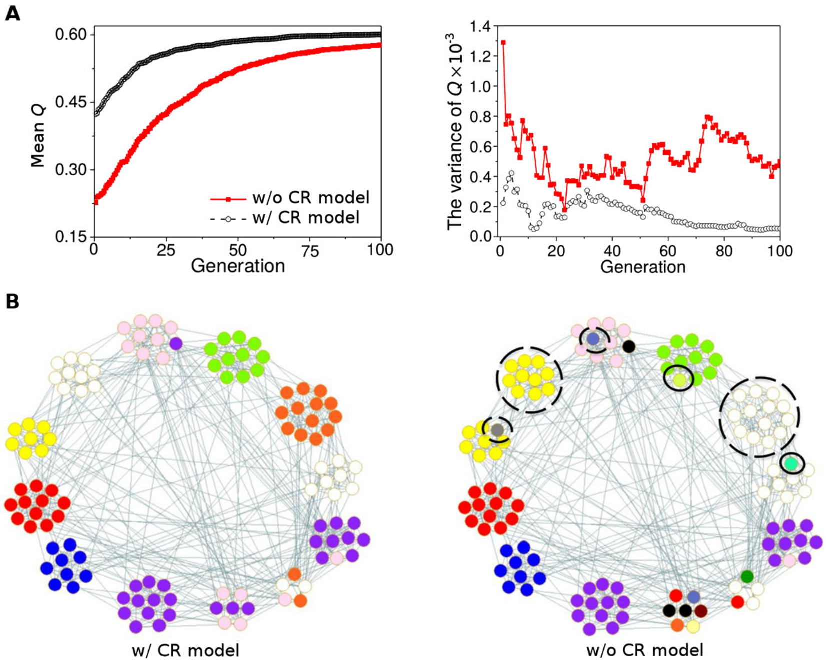

Fig. 24. Using CR model for community detection. **A**, Initialising the genetic algorithm with the CR model results in higher modularity both initially and after a large number of generations, thus indicating faster convergence and better accuracy. **B**, Some differences between community detection with (left) and without (right) the CR model. Circles highlight where the two approaches differ the most. Adapted from Ref. [15].

Community division of a complex network is achieved by forming matrix C, which classifies the network's nodes as follows: $C_{ij} = -1(C_{ij} = 1)$ if the link connecting nodes i and j is an intra-community (inter-community) connection; $C_{ij} = 0$ if there is no link connecting nodes i and j. A typical initialisation of matrix C makes all links inter-community ones, i.e., $C = A$, where A is the adjacency matrix of the considered network. However, information on diffusivities from the CR model can be used to improve upon this initialisation. Specifically, a certain number of network nodes is randomly assigned to the network's core. Then for each core node, and each of its links, the initialisation algorithm checks (by reading from matrix D^T) if the link's diffusivity is in the top $R\%$ of all diffusivities. If yes, this is a strong indication that the link is an intra-community—rather than the inter-community—link, as it was in the naïve initialisation, $C = A$.

Community detection algorithms attempt to maximise some kind of performance measure, e.g., modularity Q which is defined by $Q = \sum_i \sum_j (A_{ij} - d_i d_j/2|V|) \times \delta_{ij}/2|V|$, where d_* is the node degree and $\delta_{ij} = 1$ if nodes i and j belong to the same community or $\delta_{ij} = 0$ otherwise. This definition contrasts the actual fraction of intra-community links, $\sum_i^c A_{ii}/2|V|$ (c is the number of communities), to the expected fraction of intra-community links, $\sum_i^c (d_i/2|V|)^2$, in a random network. If these two fractions were equal—i.e., if $Q = 0$—the network would just be a random network without any real communities. If $Q > 0$ (maximum $Q = 1$), the network has an unexpectedly high fraction of intra-community links, much higher than prescribed by pure chance, thus indicating the presence of real communities.

If, for example, the genetic algorithm for maximising Q is initialised using information from the CR model, the initial value of Q is much higher, while the variance of Q is much lower (Fig. 24A). These advantages are preserved over any number of generations, indicating that the use of CR model provides both faster convergence and/or more accurate results. Some differences in community detection with and without the use of CR model are highlighted in Fig. 24B.

4. Conclusion and outlook

Is there a need for yet another organism to base bio-inspired optimisation algorithms on? And if so, is *Physarum* the best alternative? The initial choice of ants as inspiration for optimisation algorithms logically followed from studies on the organisation of ant colonies. By studying the behaviour of individual ants and translating that behaviour into colony-level behaviour, often with the use of mathematical models, we achieved a good understanding of how self-organisation leads to emergent behaviour in insect societies [90,91]. Insect societies are perfect model systems because they are amenable to experimental manipulation and we can easily observe the individual insects as they go about their business. Basing optimisation algorithms on insect behaviour, or more precisely ant behaviour, had an immediate appeal, as evidenced by the large number of publications based on ACO (8571 publications in Web of Science on March 2, 2018 mention 'Ant Colony Optimisation Algorithm'). The downside of ant-inspired optimisation algorithms is the way information is transferred: indirectly, via the pheromone trail. As a result, ACOs are not suitable to solve dynamic optimisation problems, unless the algorithm steers away from real ant behaviour. Moreover, the sort of problems real ants solve is rather limited, at least to the extent that such problems are relevant to optimisation problems. Hence, real ants soon ceased to inspire computer scientists. One of us (M.B.) explored the possibility of using honeybees as a model organism for bio-inspired optimisation algorithms. Honeybees communicate the location of food sources directly to their nestmates, via the famous dance language [92]. 'Second-generation' ACOs already incorporated direct information transfer to improve the solutions found even though such direct information transfer is scarce in real ant. So why not base optimisation-algorithms on insects that naturally use direct communication? Such direct communication makes the foraging behaviour of honeybees much more flexible than that of ants. In addition, honeybees are well-adapted to dynamic changes in their foraging environment [93]. However, it is almost impossible to model what exactly honeybees optimise. The colony will never converge onto a single food source, instead exploiting multiple high quality food sources simultaneously. Hence, an early bee-inspired optimisation algorithm is based on nest site selection, where there is a single solution [94,95]. For other approaches, not necessarily based on nest site selection, we refer the reader to Ref. [96] and further references therein.

Physarum polycephalum thus seems to fill a void. Information transfer is direct, the organism solves a range of different optimisation problems and its behaviour is well-studied and continues to be studied. It is such an easy organism to work with that even non-biologists have taken the plunge and started experimenting with the organism, which leads to a more direct feedback between experimental outcomes and models. Hence, newly developed models and algorithms can relatively quickly be verified and modified where required. But as with everything, there are limitations as a *Physarum*-inspired algorithm can only be applied to optimisation problems that can be mapped onto a shortest path problem. Applications to other types of problems are perhaps possible, but at a cost of requiring more abstract algorithms that are increasingly disconnected from their biological inspiration. We have already seen examples of this in hybrid algorithms in which *Physarum*-based part takes a secondary role and/or becomes a caricature of the original, bio-inspired version (e.g., fast variant of the CR model in Ref. [88]). While *Physarum*'s initial appeal perhaps was mainly based on the fact that the organism is a rather peculiar entity, it has clearly shown to be an extremely fruitful inspiration.

We have summarised the many ways in which the behaviour of *Physarum* can be modelled, but other aspects of the organism's decision making remain more elusive. We know that the organism can habituate to an unfavourable stimulus [97] and even transfer information when part of the habituated plasmodium is fused with a naïve plasmodium [98]. It is clear that as the slime mould experiences its environment, information about that environment is stored and this information can be transferred to other individuals. Stored information also allows the organism to anticipate periodic events [99], but how and where that information is stored remains a mystery. If we can identify the mechanism behind information storage in *Physarum*, we will not only have found the most primitive form of information storage, but perhaps even the basal means by which all living organisms learn and store information. We therefore believe that the humble slime mould will continue to inspire scientists for years to come.

Acknowledgements

We would like to acknowledge the support from the National Natural Science Foundation of China (Nos. 61403315, 61402379) and the Fundamental Research Funds for the Central Universities (No. XDJK2016A008) to C.G. and Z.Z.; Ministry of Education, Culture, Sports, Science & Technology (MEXT) Scholarship (No. 152503) to D.S.; the Research Grant Program of Inamori Foundation to M.J.; the National 1000 Young Talent Plan (No. W099102), the Fundamental Research Funds for the Central Universities (No. G2017KY0001), and China Computer Federation–Tencent Open Fund (No. IAGR20170119) to Z.W.; and the Australian Research Council (FT120100120) to M.B.

This work was performed under the Cooperative Research Program of "Network Joint Research Center for Materials and Devices".

References

[1] Dorigo M, Maniezzo V, Colorni A. Ant system: optimization by a colony of cooperating agents. IEEE Trans Cybern 1996;26(1):29–41.

[2] Bonabeau E, Dorigo M, Theraulaz G. Inspiration for optimization from social insect behaviour. Nature 2000;406(6791):39–42.

[3] Reid CR, Sumpter DJ, Beekman M. Optimisation in a natural system: Argentine ants solve the Towers of Hanoi. J Exp Biol 2011;214(1):50–8.

[4] Reid CR, Latty T, Beekman M. Making a trail: informed Argentine ants lead colony to the best food by U-turning coupled with enhanced pheromone laying. Anim Behav 2012;84(6):1579–87.

[5] Latty T, Beekman M. Keeping track of changes: the performance of ant colonies in dynamic environments. Anim Behav 2013;85(3):637–43.

[6] Nakagaki T, Yamada H, Tóth Á. Intelligence: maze-solving by an amoeboid organism. Nature 2000;407(6803):470.

[7] Tero A, Takagi S, Saigusa T, Ito K, Bebber DP, Fricker MD, et al. Rules for biologically inspired adaptive network design. Science 2010;327(5964):439–42.

[8] De Castro LN, Timmis J. Artificial immune systems: a new computational intelligence approach. Springer Science & Business Media; 2002.

[9] Read M, Andrews PS, Timmis J. An introduction to artificial immune systems. In: Handbook of natural computing. Berlin, Heidelberg: Springer; 2012. p. 1575–97.

[10] Dorigo M, Blum C. Ant colony optimization theory: a survey. Theor Comput Sci 2005;344(2–3):243–78.

[11] Liu Y, Gao C, Zhang Z, Wu Y, Liang M, Tao L, et al. A new multi-agent system to simulate the foraging behaviors of *Physarum*. Nat Comput 2017;16(1):15–29.

[12] Jones J. Characteristics of pattern formation and evolution in approximations of *Physarum* transport networks. Artif Life 2010;16(2):127–53.

[13] Wu Y, Zhang Z, Deng Y, Zhou H, Qian T. A new model to imitate the foraging behaviour of *Physarum polycephalum* on a nutrient-poor substrate. Neurocomputing 2015;148:63–9.

[14] Liu Y, Gao C, Zhang Z, Lu Y, Chen S, Liang M, et al. Solving NP-hard problems with *Physarum*-based ant colony system. IEEE/ACM Trans Comput Biol Bioinform 2017;14(1):108–20.

[15] Gao C, Liang M, Li X, Zhang Z, Wang Z, Zhou Z. Network community detection based on the *Physarum*-inspired computational framework. IEEE/ACM Trans Comput Biol Bioinform 2016. https://doi.org/10.1109/TCBB.2016.2638824.

[16] Tsompanas MAI, Sirakoulis GC, Adamatzky AI. Evolving transport networks with cellular automata models inspired by slime mould. IEEE Trans Cybern 2015;45(9):1887–99.

[17] Houbraken M, Demeyer S, Staessens D, Audenaert P, Colle D, Pickavet M. Fault tolerant network design inspired by *Physarum polycephalum*. Nat Comput 2013;12(2):277–89.

[18] Zhang X, Mahadevan S. A bio-inspired approach to traffic network equilibrium assignment problem. IEEE Trans Cybern 2017;48(4):1304–15.

[19] Gao C, Yan C, Zhang Z, Hu Y, Mahadevan S, Deng Y. An amoeboid algorithm for solving linear transportation problem. Physica A 2014;398:179–86.

[20] Jones J. A morphological adaptation approach to path planning inspired by slime mould. Int J Gen Syst 2015;44(3):279–91.

[21] Tero A, Kobayashi R, Nakagaki T. *Physarum* solver: a biologically inspired method of road-network navigation. Physica A 2006;363(1):115–9.

[22] Zhang X, Huang S, Hu Y, Zhang Y, Mahadevan S, Deng Y. Solving 0–1 knapsack problems based on amoeboid organism algorithm. Appl Math Comput 2013;219(19):9959–70.

[23] Jacobson DN, Dove WF. The amoebal cell of *Physarum polycephalum*: colony formation and growth. Dev Biol 1975;47(1):97–105.

[24] Kessler D. Plasmodial structure and motility. In: Aldrich HC, Daniel JW, editors. Cell Biology of *Physarum* and *Didymium*, vol. I. 1982. p. 145–208.

[25] Jones J. Applications of multi-agent slime mould computing. Int J Parallel Emerg Distrib Syst 2016;31(5):420–49.

[26] Dussutour A, Latty T, Beekman M, Simpson SJ. Amoeboid organism solves complex nutritional challenges. Proc Natl Acad Sci 2010;107(10):4607–11.

[27] Kincaid RL, Mansour TE. Chemotaxis toward carbohydrates and amino acids in *Physarum polycephalum*. Exp Cell Res 1978;116(2):377–85.

[28] Tero A, Kobayashi R, Nakagaki T. A mathematical model for adaptive transport network in path finding by true slime mold. J Theor Biol 2007;244(4):553–64.

[29] Reid CR, Beekman M. Solving the towers of Hanoi—how an amoeboid organism efficiently constructs transport networks. J Exp Biol 2013;216(9):1546–51.

[30] Zhang X, Wang Q, Adamatzky A, Chan FT, Mahadevan S, Deng Y. A biologically inspired optimization algorithm for solving fuzzy shortest path problems with mixed fuzzy arc lengths. J Optim Theory Appl 2014;163(3):1049–56.

[31] Zhang Z, Gao C, Liu Y, Qian T. A universal optimization strategy for ant colony optimization algorithms based on the *Physarum*-inspired mathematical model. Bioinspir Biomim 2014;9(3):036006.

[32] Zhang X, Adamatzky A, Chan FT, Deng Y, Yang H, Yang XS, et al. A biologically inspired network design model. Sci Rep 2015;5:10794.

[33] Watanabe S, Takamatsu A. Transportation network with fluctuating input/output designed by the bio-inspired *Physarum* algorithm. PLoS ONE 2014;9(2):e89231.

[34] Rosvall M, Bergstrom CT. Mapping change in large networks. PLoS ONE 2010;5(1):e8694.

[35] Liu J. Autonomy-oriented computing (AOC): the nature and implications of a paradigm for self-organized computing. In: Fourth international conference on natural computation; 2008. p. 3–11.

[36] Gunji YP, Shirakawa T, Niizato T, Haruna T. Minimal model of a cell connecting amoebic motion and adaptive transport networks. J Theor Biol 2008;253(4):659–67.

[37] Gunji YP, Shirakawa T, Niizato T, Yamachiyo M, Tani I. An adaptive and robust biological network based on the vacant-particle transportation model. J Theor Biol 2011;272(1):187–200.

[38] Liu Y, Zhang Z, Gao C, Wu Y, Qian T. A *Physarum* network evolution model based on IBTM. In: Tan Y, Shi Y, Mo H, editors. Advances in swarm intelligence. Lect notes comput sci, vol. 7929. Berlin, Heidelberg: Springer; 2013. p. 19–26.

[39] Tsompanas MAI, Sirakoulis GC. Modeling and hardware implementation of an amoeba-like cellular automaton. Bioinspir Biomim 2012;7(3):036013.

[40] Jones J. The emergence and dynamical evolution of complex transport networks from simple low-level behaviours. Int J Unconv Comput 2010;6(2):125–44.

[41] Jones J. Influences on the formation and evolution of *Physarum polycephalum* inspired emergent transport networks. Nat Comput 2011;10(4):1345–69.

[42] Wu Y, Zhang Z, Deng Y, Zhou H, Qian T. An enhanced multi-agent system with evolution mechanism to approximate *Physarum* transport networks. In: Thielscher M, Zhang D, editors. AI 2012: advances in artificial intelligence. Lect notes comput sci, vol. 7691. Berlin, Heidelberg: Springer; 2012. p. 27–38.

[43] Adamatzky A. If BZ medium did spanning trees these would be the same trees as *Physarum* built. Phys Lett A 2009;373(10):952–6.

[44] Adamatzky A. Slime mold solves maze in one pass, assisted by gradient of chemo-attractants. IEEE Trans Nanobiosci 2012;11(2):131–4.

[45] Dourvas N, Tsompanas MA, Sirakoulis GC, Tsalides P. Hardware acceleration of cellular automata *Physarum polycephalum* model. Parallel Process Lett 2015;25(01):1540006.

[46] Becchetti L, Bonifaci V, Dirnberger M, Karrenbauer A, Mehlhorn K. *Physarum* can compute shortest paths: convergence proofs and complexity bounds. In: Fomin FV, Freivalds R, Kwiatkowska M, Peleg D, editors. Automata, languages, and programming. Lect notes comput sci, vol. 7966. Berlin, Heidelberg: Springer; 2013. p. 472–83.

[47] Liu Y, Gao C, Liang M, Tao L, Zhang Z. A *Physarum*-inspired vacant-particle model with shrinkage for transport network design. In: Tan Y, Shi Y, Buarque F, Gelbukh A, Das S, Engelbrecht A, editors. Advances in swarm and computational intelligence. Lect notes comput sci, vol. 9140. Cham: Springer; 2015. p. 74–81.

[48] Adamatzky A, Yang XS, Zhao YX. Slime mould imitates transport networks in China. Int J Intell Comput Cybern 2013;6(3):232–51.

[49] Tero A, Yumiki K, Kobayashi R, Saigusa T, Nakagaki T. Flow-network adaptation in *Physarum* amoebae. Theory Biosci 2008;127(2):89–94.

[50] Minoux M. Discrete cost multicommodity network optimization problems and exact solution methods. Ann Oper Res 2001;106(1–4):19–46.

[51] Adamatzky A. From reaction–diffusion to *Physarum* computing. Nat Comput 2009;8(3):431–47.

[52] Jones J. Approximating the behaviours of *Physarum polycephalum* for the construction and minimisation of synthetic transport networks. In: Calude CS, Costa JF, Dershowitz N, Freire E, Rozenberg G, editors. Unconventional computation. Lect notes comput sci, vol. 5715. Berlin, Heidelberg: Springer; 2009. p. 291–308.

[53] Jones J. Towards programmable smart materials: dynamical reconfiguration of emergent transport networks. Int J Unconv Comput 2011;7:423–47.

[54] Nakagaki T, Yamada H, Toth A. Path finding by tube morphogenesis in an amoeboid organism. Biophys Chem 2001;92(1–2):47–52.

[55] Pershin YV, Di Ventra M. Solving mazes with memristors: a massively parallel approach. Phys Rev E 2011;84(4):046703.

[56] Jones J, Adamatzky A. Computation of the travelling salesman problem by a shrinking blob. Nat Comput 2014;13(1):1–16.

[57] Jones J. Multi-agent slime mould computing: mechanisms, applications and advances. In: Adamatzky A, editor. Advances in *Physarum* machines. Emergence, complexity and computation, vol. 21. Cham: Springer; 2016. p. 423–63.

[58] Tero A, Kobayashi R, Nakagaki T. A coupled-oscillator model with a conservation law for the rhythmic amoeboid movements of plasmodial slime molds. Physica D 2005;205(1–4):125–35.

[59] Schenz D, Shima Y, Kuroda S, Nakagaki T, Ueda KI. A mathematical model for adaptive vein formation during exploratory migration of *Physarum polycephalum*: routing while scouting. J Phys D, Appl Phys 2017;50(43):434001.

[60] Zhang S, Guy RD, Lasheras JC, del Alamo JC. Self-organized mechano-chemical dynamics in amoeboid locomotion of *Physarum* fragments. J Phys D, Appl Phys 2017;50(20):204004.

[61] Ueda KI, Takagi S, Nishiura Y, Nakagaki T. Mathematical model for contemplative amoeboid locomotion. Phys Rev E 2011;83(2):021916.

[62] Smith DA, Saldana R. Model of the Ca^{2+} oscillator for shuttle streaming in *Physarum polycephalum*. Biophys J 1992;61(2):368–80.

[63] Akita D, Kunita I, Fricker MD, Kuroda S, Sato K, Nakagaki T. Experimental models for Murray's law. J Phys D, Appl Phys 2016;50(2):024001.

[64] Rodiek B, Hauser MJB. Migratory behaviour of *Physarum polycephalum* microplasmodia. Eur Phys J Spec Top 2015;224(7):1199–214.

[65] Latty T, Beekman M. Food quality affects search strategy in the acellular slime mould, *Physarum polycephalum*. Behav Ecol 2009;20(6):1160–7.

[66] Wang Q, Zhang Z, Zhang Y, Deng Y. Fuzzy shortest path problem based on biological method. J Inf Comput Sci 2012;9(5):1365–71.

[67] Zhang Y, Zhang Z, Deng Y, Mahadevan S. A biologically inspired solution for fuzzy shortest path problems. Appl Soft Comput 2013;13(5):2356–63.

[68] Masi L, Vasile M. A multi-directional modified *Physarum* algorithm for optimal multi-objective discrete decision making. In: Schuetze O, et al., editors. EVOLVE – a bridge between probability, set oriented numerics, and evolutionary computation III. Stud comput intell, vol. 500. Heidelberg: Springer; 2014.

[69] Wang Q, Lu X, Zhang X, Deng Y, Xiao C. An anticipation mechanism for the shortest path problem based on *Physarum polycephalum*. Int J Gen Syst 2015;44(3):326–40.

[70] Liu Y, Deng Y, Jusup M, Wang Z. A biologically inspired immunization strategy for network epidemiology. J Theor Biol 2016;400:92–102.

[71] Schön T, Stetter M, Tomé AM, Puntonet CG, Lang EW. *Physarum* learner: a bio-inspired way of learning structure from data. Expert Syst Appl 2014;41(11):5353–70.

[72] Davenport EC Jr, El-Sanhurry NA. Phi/phimax: review and synthesis. Educ Psychol Meas 1991;51(4):821–8.

[73] Schön T, Stetter M, Lang EW. Structure learning for Bayesian networks using the *Physarum* solver. In: 11th international conference on machine learning and applications; 2012. p. 488–93.

[74] Li K, Torres CE, Thomas K, Rossi LF, Shen CC. Slime mold inspired routing protocols for wireless sensor networks. Swarm Intell 2011;5(3–4):183–223.

[75] Liu L, Song Y, Zhang H, Ma H, Vasilakos AV. *Physarum* optimization: a biology-inspired algorithm for the steiner tree problem in networks. IEEE Trans Comput 2015;64(3):818–31.

[76] Zhang X, Zhang Y, Hu Y, Deng Y, Mahadevan S. An adaptive amoeba algorithm for constrained shortest paths. Expert Syst Appl 2013;40(18):7607–16.

[77] Wang H, Lu X, Zhang X, Wang Q, Deng Y. A bio-inspired method for the constrained shortest path problem. Sci World J 2014;2014:271280.

[78] Tero A, Nakagaki T, Toyabe K, Yumiki K, Kobayashi R. A method inspired by *Physarum* for solving the Steiner problem. Int J Unconv Comput 2010;6(2):109–23.

[79] Zhang Z, Gao C, Lu Y, Liu Y, Liang M. Multi-objective ant colony optimization based on the *Physarum*-inspired mathematical model for bi-objective traveling salesman problems. PLoS ONE 2016;11(1):e0146709.

[80] Zhang M, Xu C, Guan J, Zheng R, Wu Q, Zhang H. A novel *Physarum*-inspired routing protocol for wireless sensor networks. Int J Distrib Sens Netw 2013;9(6):483581.

[81] Cervero R. Road expansion, urban growth, and induced travel: a path analysis. J Am Plan Assoc 2003;69(2):145–63.

[82] Adamatzky A, Jones J. Road planning with slime mould: if *Physarum* built motorways it would route M6/M74 through Newcastle. Int J Bifurc Chaos 2010;20(10):3065–84.

[83] Nakagaki T, Kobayashi R, Nishiura Y, Ueda T. Obtaining multiple separate food sources: behavioural intelligence in the *Physarum* plasmodium. Proc R Soc Lond B, Biol Sci 2004;271(1554):2305–10.

[84] Ma Q, Johansson A, Tero A, Nakagaki T, Sumpter DJ. Current-reinforced random walks for constructing transport networks. J R Soc Interface 2013;10(80):20120864.

[85] Colorni A, Dorigo M, Maniezzo V. Distributed optimization by ant colonies. In: Proc. ECAL '91, first European conference on artificial life. Paris: Elsevier; 1991. p. 134–42.

[86] Dorigo M, Gambardella LM. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1997;1(1):53–66.

[87] Stützle T, Hoos HH. MAX–MIN ant system. Future Gener Comput Syst 2000;16(8):889–914.

[88] Liang M, Gao C, Zhang Z. A new genetic algorithm based on modified *Physarum* network model for bandwidth-delay constrained least-cost multicast routing. Nat Comput 2017;16(1):85–98.

[89] Liang M, Gao C, Li X, Zhang Z. A *Physarum*-inspired ant colony optimization for community mining. In: Kim J, Shim K, Cao L, Lee JG, Lin X, Moon YS, editors. Advances in knowledge discovery and data mining. Lect notes comput sci, vol. 10234. Cham: Springer; 2017. p. 737–49.

[90] Bonabeau E, Theraulaz G, Deneubourg JL, Aron S, Camazine S. Self-organization in social insects. Trends Ecol Evol 1997;12(5):188–93.

[91] Camazine S. Self-organization in biological systems. Princeton University Press; 2003.

[92] Von Frisch K. The dance language and orientation of bees. Harvard University Press; 1967.

[93] Seeley TD. The wisdom of the hive: the social physiology of honey bee colonies. Harvard University Press; 2009.

[94] Diwold K, Beekman M, Middendorf M. Bee nest site selection as an optimization process. Artificial life XII. In: Fellerman H, Dörr M, Hanczyc MM, et al., editors. Proceedings of the 12th international conference on the synthesis and simulation of living systems. The MIT Press; 2010. p. 626–33.

[95] Diwold K, Himmelbach D, Meier R, Baldauf C, Middendorf M. Bonding as a swarm: applying bee nest-site selection behaviour to protein docking. In: Proceedings of the 13th annual conference on genetic and evolutionary computation; 2011. p. 93–100.

[96] Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 2007;39(3):459–71.

[97] Boisseau RP, Vogel D, Dussutour A. Habituation in non-neural organisms: evidence from slime moulds. Proc R Soc B 2016;283(1829):20160446.

[98] Vogel D, Dussutour A. Direct transfer of learned behaviour via cell fusion in non-neural organisms. Proc R Soc B 2016;283(1845):20162382.

[99] Saigusa T, Tero A, Nakagaki T, Kuramoto Y. Amoebae anticipate periodic events. Phys Rev Lett 2008;100(1):018101.