Slime Mould Foraging: An inspiration for algorithmic design

Anthony Brabazon

Smurfit School of Business
University College Dublin, Dublin, Ireland
E-mail: anthony.brabazon@ucd.ie

Seán McGarraghy

Smurfit School of Business
University College Dublin, Dublin, Ireland
E-mail sean.mcgarraghy@ucd.ie

Abstract: The metaphor of 'foraging as search' provides a rich source of inspiration for the design of optimisation algorithms. An extensive literature has resulted in computer science over the past twenty years based on this, with algorithmic families such as ant colony optimisation and honeybee optimisation amongst others, being successfully applied to a wide range of real-world problems. Of course, all organisms must forage to acquire necessary resources and in recent years, increasing attention has been paid to the mechanisms by which nonneuronal organisms, in other words organisms without a central nervous system, forage. The vast majority of living organisms, encompassing some 99.5% of all biomass on earth, are nonneuronal. In this paper we introduce the plasmodial slime mould *Physarum polycephalum*. This nonneuronal organism is formed when individual amoebae swarm together and fuse, resulting in a large bag of cytoplasm encased within a thin membrane which acts a single organism. Inspiration has been drawn from some of its foraging behaviours to develop algorithms for graph optimisation and exemplars of these algorithms along with some suggestions for future research are presented in this paper.

Keywords: Slime mould algorithms; Foraging-inspired algorithms; Graph optimisation; Nonneuronal organisms.

Reference to this paper should be made as follows: Brabazon, A., McGarraghy, S. (2019) 'Slime Mould Foraging: An inspiration for algorithmic design', *International Journal of Innovative Computing and Applications,*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Prof. Brabazon's research interests concern the development of natural computing algorithms and their application to real-world problems. He is co-founder and co-director of the Natural Computing Research & Applications Group at UCD (see http://ncra.ucd.ie). Anthony has published in excess of 200 peer-reviewed studies and has authored / edited sixteen books.

Dr. McGarraghy is Director of the UCD Centre for Business Analytics. His specialist areas of interest include analytics, operational research (particularly graph methods) and applying natural computing approaches to problems arising in business, such as supply chain management. He has published extensively in these and other areas.

1 Introduction

Computation abounds in nature. This realisation has led to the development of the field of natural computing in which inspiration is taken from natural processes in order to design powerful algorithms for diverse applications including optimisation, classification, clustering, design and model induction (Brabazon et al., 2015b). Well-known subfields of natural computing include neural networks, evolutionary algorithms, artificial immune systems and particle swarm optimisation.

Algorithms deriving their design inspiration from the foraging activities of various organisms form an important subdomain of natural computing (Fig. 1). In this paper we focus on one of the lesser known of these families, specifically slime mould foraging algorithms, initially providing some background on these organisms and then outlining how inspiration can be drawn from their foraging behaviours in order to design algorithms for optimisation. An interesting aspect of slime mould behaviour is that it provides an exemplar of sophisticated decision making by a nonneuronal organism and consequently, challenges conventional wisdom that only organisms with a central

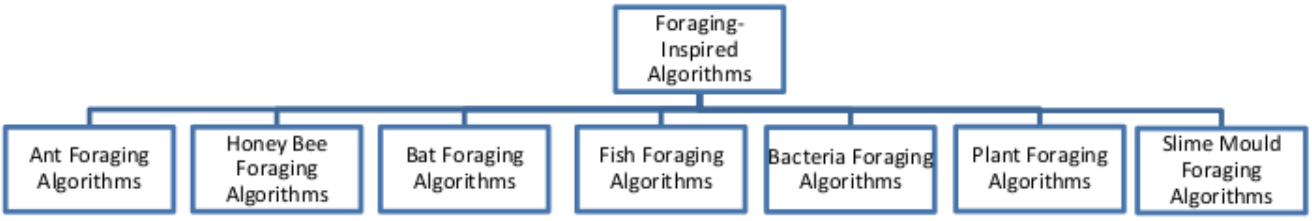

Figure 1 Taxonomy of some of the common families of foraging-inspired algorithms

nervous system are capable of complex behaviours in response to their environment.

1.1 Structure of Paper

The remainder of this contribution is organised as follows. Section 2 introduces the domain of foraging and a number of algorithms that have been inspired by the foraging behaviours of various organisms. Slime moulds are introduced in Section 3, including a short overview of slime mould physiology and foraging behaviours. Relevant behaviours include how slime moulds move in order to efficiently access and harvest food resources, the capability of slime moulds to assess food quality, and their capability to create a memory of already-visited areas when making foraging decisions. Section 4 provides a synopsis of some recent work on the application of slime mould as a 'biocomputer' and outlines a mathematical model of slime mould foraging behaviour which has been used to develop a series of computational algorithms. Section 5 discusses the application of derived algorithms for the purposes of graph optimisation. Section 6 compares slime mould algorithm performances on graph problems against classical graph algorithms and considers whether they have other properties that could aid their robustness and widen their applicability to real-world problems; this adds to the critical appreciation of these algorithms. Finally, some opportunities for future work are discussed in Section 7.

2 Foraging

All animals require resources such as food, shelter and mates. Success in foraging for these is critical to survival and reproduction. Indeed, the need to acquire resources and to avoid predators is a key driver of the physical morphology and of the sensory and cognitive capabilities of organisms. Given the enormous diversity of life forms and ecological niches in nature, it is unsurprising that we see a corresponding diversity of foraging strategies.

Perhaps the most commonly-known foraging behaviours in nature are those of animals which we can easily observe, such as large predators actively searching an environment for prey items. However, even the simplest creatures also need to 'earn a living'. Foraging activity can range from solitary foraging, where an individual forages on its own, to group foraging where foraging is a social behaviour. The essence of social foraging is that there must be some communication among organisms. Communication about resource finds may take place between individuals, via direct interaction, indirect interaction such as environmental marking or via a broadcast mechanism at a communal nest or hive (Sumpter and Brannstrom, 2008).

2.1 Foraging Algorithms

The observation that foraging typically requires organisms to undertake a search process has in turn led to the design of several families of search algorithms which draw metaphorical inspiration from a range of real-world foraging behaviours. These include ant colony optimisation algorithms (Bonabeau et al., 1999; Dorigo, 1992; Dorigo and DiCaro, 1999; Dorigo et al., 1996; Dorigo and Stützle, 2004; Ibri et al., 2010; Pedemonte and Cancela, 2010) and honeybee algorithms (Bansal et al., 2013; Chong et al., 2006; Nakrani and Tovey, 2004; Karaboga, 2005; Karaboga and Akay, 2009; Pham et al., 2006; Yang, 2005; Gao et al., 2011; Wang et al., 2014; Brabazon et al., 2013).

In the case of both ant and honeybee foraging, several species of the insects are *central place foragers*, in that they return to a colony or hive in order to store food. Therefore they can interact with colony members and potentially pass on information about food finds. A notable aspect of their interaction is that successful foragers seek to *recruit* other conspecifics to food resources that they have found.

Social transmission of information is emphasised in the majority of search algorithms inspired by foraging processes. Typically, the algorithms emphasise the importance of sensing (i.e., assessment of the local environment around an agent) and of communication processes between the agents. This leads, in turn, to a discussion of what the agents 'know' and how information is propagated or 'spread' between individual nodes or agents in the population. In some real-world instances, individual foraging agents in the population have relatively limited information-processing capabilities, but as a result of direct or indirect communication mechanisms between these agents, an 'informationally-connected emergent creature' or *super-organism* results (Goldfield, 2018). Hence, sophisticated information processing can result from physical and

informational linkages between apparently simple individual agents, not just as a result of agent's having complex individual cognitive capabilities.

Other significant families of foraging algorithms include those inspired by the echolocation process of bats, with the *bat algorithm* being developed by Yang (2010). The bat algorithm has produced very competitive results on both benchmark optimisation problems and across a variety of applications, encompassing constrained optimisation (Gandomi et al., 2013; Yang and Gandomi, 2012), multimodal landscapes (Cai et al., 2015), multiobjective optimisation (Bora et al., 2012; Niknam et al., 2013), binary-valued representations (Nakamura et al., 2012), and clustering (Rui et al., 2012). A detailed review of applications of the bat algorithm is provided in Yang (2013) and some extensions of the canonical algorithm are discussed in Cui et al. (2015).

A variety of animals, including some species of birds, engage in social roosting whereby large numbers of conspecifics gather together to roost, either overnight or for longer periods. It has been claimed that these roosts can serve as *information centres* (Zahavi, 1971) to spread knowledge concerning the location of food resources in the environment. Brabazon et al. (2016) draw inspiration from this behaviour in designing the raven roosting optimisation (RRO) algorithm.

A number of studies have employed a fish school metaphor to develop algorithms for optimisation and clustering. Amintoosi (2007); Bastos Filho et al. (2008); He et al. (2009); Li et al. (2002); Tsai (2011); Zhou et al. (2009) provide a sample of this work. Algorithms adopting this approach include fish school search (FSS) (Bastos Filho et al., 2008), the artificial fish swarm algorithm (AFSA) (Li et al., 2002) and the fish algorithm (Brabazon et al., 2015).

Dominance hierarchies amongst group-living animals can influence their decision-making and foraging behaviours. One example of this is provided by wolves, and a number of studies have drawn inspiration from wolf pack foraging behaviours to design optimisation algorithms (Yang et al., 2007; Mirjalili, 2015; Mirjalili et al., 2014; Li et al., 2016). Obviously it is not possible to provide a detailed coverage of the entire literature on foraging-inspired algorithms in a single paper, and readers requiring a comprehensive overview of this domain are referred to (Brabazon et al., 2018).

2.2 Nonneuronal Organisms

From the brief review above, it is evident that an array of optimisation algorithms derived from the foraging behaviours of organisms, including mammals, birds, fish and insects, have been explored. In all of these cases, the organisms possess a nervous system with varying degrees of complexity and plasticity.

Of course, the foraging activities of most living things on earth are not driven by neuronal-mediated decision processes. As noted by Reid et al. (2015) and Tang (2018), the vast majority of life forms, including the plant, bacteria, fungi and protist kingdoms of life, do not have a brain or other nervous system hardware. Recent work by Bar-on et al. (2018) estimates that the vast majority of the earth's biomass is found in nonneuronal organisms such as plants (accounting for 82% of total biomass) and bacteria (13%), with less than 0.45% of total biomass being comprised of animals with neuronal tissue.

Nonneuronal organisms live in environments that are no less complex than those faced by organisms with a brain. They face the same basic challenges as animals in foraging for food and other resources, and in dealing with competitors, predators and pathogens. Key questions which arise concern the nature of the mechanisms which allow these organisms to forage successfully in their individual dynamic and challenging environments.

Since the turn of the century, a significant research effort has emerged concerning algorithms inspired by the foraging activities of nonneuronal organisms, including bacterial foraging algorithms (Passino, 2000, 2002; Müller et al., 2000; Gao and Xu, 2014; Cai et al., 2015), plant-inspired foraging algorithms (Müller et al., 2000; Mehrabian and Lucas, 2006; Premaratne et al., 2009; Salhi and Fraga, 2011; Tong et al., 2005; Cai et al., 2008; Zhang et al., 2012; Brabazon et al., 2015a, 2018) and—the focus of this paper—slime mould foraging algorithms. To date, the majority of slime mould foraging algorithms have been applied to graph optimisation applications and next we provide a short introduction to this domain.

2.3 Graphs

Essentially, a *graph* is a mathematical structure that models *connectivity*: the *connections* between things. The things are called the *vertices* or *nodes* of the graph and the connections are called *edges*, *arcs* or *links*. The connections capture binary relationships. Graphs are also commonly referred to as *networks*, but we will reserve the word network for real world structures and the word graph for a mathematical model thereof. The set of vertices is commonly denoted by V and its order $|V|$ by n; the set of edges is commonly denoted by E and its order $|E|$ by m.

Given vertices i and j, if there is an edge ij between these vertices, then we say that vertices i and j are *adjacent*. Nonadjacent vertices may be *(path-)connected*: if i and j are vertices then we say that there is a *path* between i and j if there is an ordered sequence of distinct vertices $i, a_1, \ldots, a_k, j$ such that each vertex is adjacent to the next in the sequence (that is, there is an edge connecting i and a_1, an edge connecting a_1 and a_2, and so on).

The graph is called *weighted* if there is a numerical *weight* on each edge, representing a distance, cost or some other number associated with the edge. Figure 2 shows an example of a weighted graph.

2.3.1 Applications of Network Models

The graph model is simple and general: it has uses ranging from analysis of road networks, to social network analysis, to data clustering approaches and other applications. A multitude of problems in operational research and/or combinatorial (discrete) optimisation can be modelled as graphs and thus solved. A graph generally allows multiple

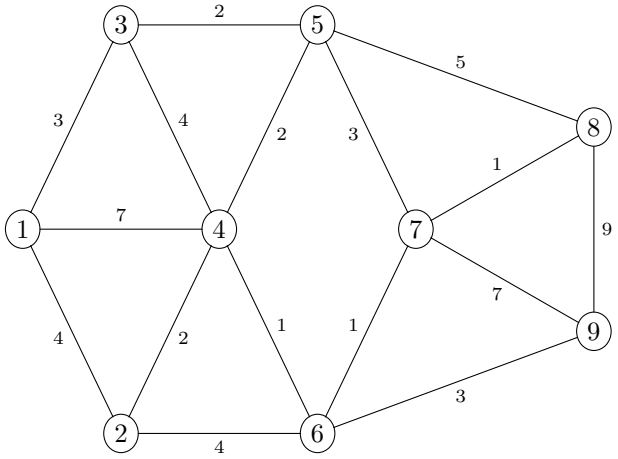

Figure 2 An example of a weighted graph with $n = 9$ vertices and $m = 16$ edges. Vertices 1 and 2 are adjacent and the weight on edge 12 is 4. Vertices 1 and 9 are path-connected: an example of a path from vertex 1 to vertex 9 is 1–2–6–7–9; another such is 1–4–5–7–8–9.

paths between two given vertices i and j; thus, natural questions arising include:

- which path(s) are shortest (the *shortest path problem*) by some measure such as distance, travel time, number of edges, or some other cost or objective function? There are well-established classical graph algorithms for solving this problem: in the case where all edge lengths are positive, the original Dijkstra algorithm using an unordered array of vertices is $O(n^2)$, Tarjan's version (Tarjan, 1983) using a heap is $O(m \log n)$ which is faster on sparse graphs, while Fredman and Tarjan's version (Fredman and Tarjan, 1987) using a Fibonacci heap is $O(m + n \log n)$ and is asymptotically the fastest known single-source shortest path algorithm;
- if one or more paths were removed from the graph, would it still be possible to get from i to j (that is, is the graph *resilient* or *fault tolerant*)?
- how can we design a graph (e.g., a transport or communication network) to be resilient while still meeting certain cost and/or connectivity requirements? There are many kinds of network design problem, for example requiring at least two paths between certain pairs of vertices to enhance robustness of the design, and they are typically NP-hard (Carroll et al., 2013).
- what is the shortest tour (a closed path containing all nodes, starting and finishing at the same node) such that each node is visited exactly once? (This is the famous *travelling salesperson problem* or TSP.)

Another standard application of graph models is in the study of *flow* or *flux*, e.g., a fluid, or a good transported in a supply logistics network). The kinds of questions asked here include: given one or more vertices (called *sources*) where flow enters the graph and other vertices (called *sinks*) where flow leaves the graph, and a maximum throughput or *capacity* on each edge:

- how does flow move through the graph from source(s) to sink(s)?
- what is the maximum flow possible from source(s) to sink(s)? Classical max-flow/min-cut algorithms such as the Ford-Fulkerson approach are efficient if all capacities are rational numbers (in any situation modelled on a finite-precision machine, all numbers represented are perforce rational): the Ford-Fulkerson method has complexity $O(mf)$ where f is the maximum flow (Cormen et al., 2001, Section 26.2); and the Edmonds-Karp implementation of Ford-Fulkerson using breadth-first search has complexity $O(m^2 n)$ (Cormen et al., 2001, Lemma 26.8, Theorem 26.9).

Another optimisation problem that can arise in a graph setting is that of finding a Steiner minimum tree (SMT): the set of edges of minimum total weight connecting a prechosen subset U of the vertex set V. If the subset $U = V$, this reduces to finding a minimum spanning tree (MST) in the graph (a subset of edges which contains no cycles, but which path-connects any pair of vertices, and is of least possible weight given these requirements). The MST problem can be solved by standard algorithms in $O(m \log n)$ time or less (Cormen et al., 2001, 23.2), but in general the Steiner tree problem is NP-hard. However, this problem can also be defined in a setting where only the vertices are given, and the problem becomes that of identifying and constucting the edges of the Steiner minimum tree, possibly including new intermediate vertices. For example, given three vertices in the plane, each the same distance from the other two (so they are the vertices of an equilateral triangle), the Steiner minimum tree can be shown to be the tree got by adding a new vertex c at the centre of gravity of the triangle, and connecting each of the three original vertices to c by an edge; but no two of the original

three vertices are directly connected to each other. The Steiner tree problem in graphs is an example of a network design problem.

Biological systems often require extensive networks to transport resources and information (along paths). These networks may be internal or external. Examples include the vasculature of animals or plants, ant trails, and the interconnected tubular networks of foraging slime mould. Transport networks are ubiquitous in both human-designed and biological systems. It is plausible that networks in both will share common features such as short path length between source and destination nodes, and resilience to faults.

It is important to note that (as with any model) a graph model is an abstraction and may omit certain details of a modelled real world situation; for example, the precise topographical layout of a road segment in a road network may be reduced to its start and end points, and its total length. It is also plausible that when modelling some real networks, the model may require more detail than a "pure" graph model as described above would capture, in order to be an adequate representation. Thus, a problem definition and solution method incorporating more domain knowledge may be preferable to a pure graph approach, despite having inferior runtime: see Section 6.

3 Slime Mould

The term *slime mould* is a shorthand name for a number of unrelated eukaryotic organisms which have similarities in their life cycles, with their primary reproductive stage arising on the formation of a fruiting body and the release of spores. The organisms can live freely as single amoebae, aggregating together for spore production and dispersal. The phrase 'slime mould' refers to their appearance during the aggregative phase of their life cycle, in which they look like a gelatinous slime.

Perhaps the best-studied slime mould is is the plasmodial slime mould, *Physarum polycephalum*. Plasmodial (or acellular) slime moulds are formed when individual amoebae swarm together and fuse, resulting in a *plasmodium*: a large bag of cytoplasm encased within a thin membrane which acts a single organism. Unusually, the organism in this state consists of a large single cell with thousands of nuclei (i.e. it is a unicellular, multinucleated eukaryote (Boisseau et al., 2016)). About 875 species of plasmodial slime mould are known to exist. Plasmoida can grow to a considerable size, extending up to 930 cm^2 in the case of *Physarum polycephalum* (Latty and Beekman, 2009). During the motile phase of their life cycle, plasmodia are capable of movement by means of cytoplasmic streaming (the internal movement of cell contents) at rates of up to 5 cm/h (Latty and Beekman, 2009).

3.1 Foraging Behaviours of Slime Mould

As with all organisms, a key behaviour of slime moulds is foraging for food resources. Slime moulds feed on dead plant material, living bacteria, yeasts and fungi, by engulfing their prey, secreting enzymes and digesting the prey item (*phagocytosis*). To do this effectively, they need to be able to process sensory information and integrate this with internal state information in order to decide for how long to exploit already-discovered food resources, and when and where to move to next. In the following subsections we consider a number of characteristics of *Physarum polycephalum*, including its ability to move and its ability to assess food quality.

3.1.1 Mobility

Under idealised conditions of plentiful resources, *Physarum polycephalum* plasmodia are sedentary and grow steadily. If they are located on non-nutrient-bearing substrates they search for food resources by migrating at a rate of up to a few cm per hour directed by external stimuli such as gradients of sugar and proteins (Dussutour et al., 2010). When a slime mould chemotactically senses attractants such as food via the binding of chemicals to receptor molecules presented on its outer membrane surface, cytoplasm flows towards the attractant, thereby inducing movement of the organism in the direction of the food (Reid et al., 2012).

When a slime mould comes into contact with a food item, it fully or partly engulfs it by covering it with biomass, later resuming exploration by extending pseudopodia into the surrounding environment while remaining in physical contact with the initial food source (Latty and Beekman, 2009).

Physarum polycephalum is able to assess the quality of current food resources and use this information to determine how much time to allocate to exploring the environment for new resources, and to determine what search strategy to use during exploration. The quality of a food source influences subsequent search behaviour, with higher quality food sources promoting intensification of local search around that food source, and lower quality food sources encouraging explorative search of the environment elsewhere (Latty and Beekman, 2009).

3.2 Foraging Networks in Physarum polycephalum

If a number of small sources of food are presented at various positions on a surface to a starved plasmodium of *Physarum polycephalum*, it endeavours to reach them all. In nutrient-poor environments, the plasmodium grows by extending pseudopodia towards nearby chemoattractant sources, producing a parallel search process guided by the chemoattractant gradients in the local environment. In nutrient-rich conditions, the plasmodia grows radially in all directions.

The organism attempts to optimise the shape of its biomass in order to facilitate the harvesting of food resources. As the plasmodium has a limited mass, it has to decide how much of this mass to place over each food sources in order to absorb nutrients (the more biomass placed over a food source, the faster it can be absorbed) and how much to allocate to the construction of tubes between the food sources in order to maintain connectivity and intracellular communication throughout the organism (Nakagaki, Kobayashi et al., 2004).

3.2.1 Spatial Memory

The ability to navigate efficiently through an environment is crucial to most organisms' ability to survive (Smith-Ferguson et al., 2017). It is clearly important to be able to find, for example, a nest or to relocate a previously-discovered food resource. Conversely, the ability to avoid areas already known to contain no resources is also of importance. The presence of a memory system will typically enhance the foraging success of an organism.

Given the nature of the food resources consumed by a slime mould, it would take some time for a previously harvested area to regenerate new food resources (Reid et al., 2013b). As *Physarum polycephalum* moves in its environment, it deposits nonliving extracellular slime. It has been found that *Physarum polycephalum* preferentially avoids areas if they are found to be so marked (Reid et al., 2012). This response is adaptive, as such regions are likely to have been previously harvested of resources and therefore represent a poor choice of foraging location. The deposit of extracellular slime creates a form of *external memory*, somewhat akin to the pheromone deposits of trail-marking ants.

The use of an external spatial memory frees the forager from having to internally store information on which areas have already been searched, and it has been speculated that external memory processes may be a functional precursor to the development of internal memory systems, allowing biological organisms with primitive information-processing systems to solve complex tasks requiring spatial memory (Reid et al., 2012).

4 Slime Mould Computation

An extensive literature on the computational capabilities of slime mould has emerged over the past two decades. At a high level, this literature can be divided into two groupings, the first where slime mould is employed as a biological computer (biocomputer), and the second where elements of slime mould behaviours are used to inspire the design of computational algorithms.

4.1 Slime Mould as a Biocomputer

Although slime moulds are relatively simple organisms, they are capable of both sensory and locomotive behaviours, and are also capable of adapting the morphology of their protoplasmic tube network in response to environmental conditions.

These capabilities have resulted in slime moulds such as *Physarum polycephalum* being used as a living computational material (Adamatzky, 2010), which can be 'programmed' by appropriate placement of external stimuli such as food, or repellents such as chemicals or light (Jones, 2016), with the result of the program being the structure of the final slime mould network.

Graph and network design problems are a natural fit for slime mould computing applications, as the growth and adaptation of a slime mould naturally forms edges (protoplasmic tubes of slime mould) between discrete nodes (which can be located on nutrients such as food flakes) (Jones, 2016). This has led to multiple studies which have applied slime moulds as a biocomputer to solve these problems.

In Nakagaki, Kobayashi et al. (2004) three separate food sources were presented to *Physarum polycephalum*, located at the vertices of a triangle. In repeated experiments, the resulting tubular networks connecting the food sources were often similar in design to (and an approximation of) a Steiner minimum tree (SMT): the set of edges of minimum total length connecting all of the vertices. Further investigation of the ability of the slime mould to construct smart networks indicated that it can simultaneously meet multiple requirements of a smart network, trading off total length of connections against resilience in the sense of tolerance of accidental disconnection of the tubes (Nakagaki, Yamada et al., 2004; Reid et al., 2013a).

Recently (Zhu et al., 2018), *Physarum polycephalum* was applied as a biocomputer to solve the TSP, and it was found that the time taken by the organism to find a reasonably high-quality TSP solution grows linearly for small problem size up to eight vertices, and that the solution quality is approximately constant in spite of the enormous expansion of the search space size from $\frac{1}{2}4! = 12$ to $\frac{1}{2}8! = 20160$.

4.1.1 Solving Maze Problems

Physarum polycephalum has also demonstrated a capability to solve maze problems, which are a standard test of intelligence in animal psychology (Hickey and Noriega, 2008). In an experiment (Nakagaki, Yamada et al., 2000), a large piece of plasmodium was cut into a number of smaller pieces, which were then distributed at intervals within a physical, two-dimensional maze structure. Initially, the pieces coalesced to form a single organism that filled the entire maze but when nutrient blocks of oat flakes (technically, it is the bacterial colonies on the oat flakes, rather than the oat flakes themselves, which are harvested by the slime mould) were placed at two locations in the maze, the pseudopodia reaching dead ends shrank resulting in a single thick pseudopodium spanning the shortest path between the nutrient blocks, indicating the capability of the slime mould to solve the maze by adapting its physical structure.

4.1.2 Reproducing Human-Designed Transport Networks

An interesting variant on the above studies investigated the capability of *Physarum polycephalum* to design transport networks, with the results being benchmarked against existing human-designed transport networks. In Tero et al. (2010), food sources were placed on a flat wet dish at locations corresponding to 36 major cities in the Greater Tokyo area and bright light was shone at appropriate places in the environment in order to simulate the challenges posed to network designers by geographic features such as mountains and waterways (*Physarum polycephalum* is negatively phototropic and preferentially avoids bright light).

The slime mould initially filled the entire space with plasmodia and then thinned to describe a network linking the food sources. In repeated experiments, many of the networks bore notable similarity to the actual rail system linking Tokyo and the outlying cities. Analysis of the produced networks indicated that they had characteristics similar to those of the rail network in terms of cost, transport efficiency and fault tolerance (Tero et al., 2010). A study by Adamatzky et al. (2017) explored the capability of *Physarum polycephalum* to design transport networks which bore similarity to the French motorway network.

The previous studies used a two-dimensional representation of the transport nodes / hubs (cities) and an open question was whether the results would carry over to a more complex three-dimensional representation of the terrain between transport nodes. This issue was investigated by Adamatzky (2014) using plastic three-dimensional terrain models of both the US and Europe and the results obtained illustrated *Physarum polycephalum*'s capability to approximate Route 20, the longest road in the United States (running from Boston to Newport, Oregon, this road is 5,415 km in length), and Autobahn 7 (running from Flensburg to Füssen, this road is 963 km in length), the longest national motorway in Europe. In these experiments, a plasmodium of *Physarum polycephalum* was placed at one end of the road and nutrient sources were placed at the other endpoint and the plasmodium was allowed to explore the three-dimensional terrain, producing solutions which bore considerable similarity to the actual road networks.

4.2 Slime Mould Algorithms

In addition to using a slime mould as a physical computing substrate (computing *in vivo*), a number of researchers have sought to mathematically model aspects of the behaviour of slime mould. Although the underlying molecular mechanisms which produce problem solving behaviour in slime mould are not completely understood (Marwan, 2010), these simplified models have proven to be high-quality problem solvers in their own right. The resulting algorithms provide an example of computing *in silico*.

4.2.1 Tero Model

One of the most commonly-used models is based on the work of Tero et al. (2007). This model of the evolution of transport networks within *Physarum polycephalum* is based on an assumption that protoplasmic flux through network veins produces an autocatalytic effect whereby increases in the flux in a protoplasmic tube causes it to change structure (becoming wider), in turn facilitating even greater levels of flux.

The starting point for the Tero model, as applied to shortest-path-type problems, is a complete network of simulated protoplasmic tubes which connect nodes (metaphorically, nutrient sources) which act as sources and sinks for flux. An adaptation process, as outlined in the model, is simulated, producing an optimal (shortest) path through the network. In Sect. 5 we outline the detail of the model's implementation and describe two algorithms for combinatorial optimisation which are derived from this.

Other approaches to deriving inspiration from slime mould behaviours to design optimisation algorithms includes the cellular automaton (CA) model of Gunji et al. (2008), which considers the processes of plasmodial growth and amoeboid movement, developing an algorithm for application to maze solving and shortest path problems based on these processes. A multiagent approach to reproduce behaviors of slime mould is outlined by (Jones, 2016). This paper also provides a useful review of other recent works on slime mould computing.

5 Graph Optimisation Using Slime Mould

The majority of published work on slime-mould-inspired algorithms concerns graph optimisation applications, particularly shortest path and close variants of this problem. In this section two exemplar algorithms from this work are introduced.

The first algorithm, the 'improved *Physarum polycephalum* algorithm', employs the Tero model but modifies this through the addition of an energy parameter. The resulting algorithm is used for a series of graph optimisation problems. The second algorithm, the *Physarum*-based ant colony system, illustrates how the model from Tero et al. (2007) can be used to help modify the pheromone matrix in an ant colony system application.

5.1 Improved Physarum polycephalum *Algorithm*

As described in earlier sections, plasmodia of *Physarum polycephalum* in a starved state state can find the shortest path between two points in a maze, where each end of the maze is marked with food. In this scenario, as the explorative (foraging) phase progresses, tubular pseudopodia that are not on the shortest path will shrink and eventually disappear, whereas pseudopodia on the shortest path between discovered food sources will be reinforced and become thicker. In essence, *Physarum polycephalum* self-adapts, reconfiguring its biomass morphology to resource availability in its environment.

In this section we introduce the improved *Physarum polycephalum* algorithm (IPPA) developed by Zhang, Wang et al. (2014). In this study the authors note that basic *Physarum*-inspired algorithms can suffer from a low rate of convergence, resulting in slow performance when applied to shortest path problems. The IPPA adapts the canonical *Physarum*-inspired shortest path algorithm based on the Tero model (Tero et al., 2007) by adding a new energy parameter in order to improve the speed at which good solutions are found. The authors argue that the addition of an energy component to the model is plausible from a biological perspective as energy is consumed during the process of tube expansion and during slime mould movement, whereas, at the same time, these processes can facilitate the capture of new resources. Therefore, dynamic adaptation of the morphology of a slime

mould involves a trade-off between the consumption and absorption of energy.

Initially, the canonical *Physarum* model for determination of the shortest path across a graph is described, and this is followed by a description of the modifications made to this in the IPPA.

5.1.1 Canonical Physarum Algorithm for Shortest Path

The canonical *Physarum polycephalum* algorithm for calculation of the shortest path in an undirected graph is based on Tero et al. (2007). In this model it is assumed that the network formed by the *Physarum* is represented by a graph, in which a plasmodial tube is an edge and the junction between two tubes is a node. A key feature in the model is a positive feedback phenomenon, based on hydrostatic pressure, between flux and tube thickness. As noted in Tero et al. (2007), hydrodynamic theory implies that short thick tubes are the most effective for internal transportation. If a slime mould can form short, thick, tubes it can enhance its survival as this enables it to place most biomass over discovered food, and to transport nutrients and chemical signals effectively across its structure.

Tubes in the network become thicker in a given direction when streaming of protoplasm persists in that direction for a period of time, and thicker tubes enable greater conductance as the resistance to flow is reduced as the tube widens. In turn, greater conductance enables a greater flux (flow), leading to a further thickening of the relevant tubes. Therefore, tubes with a large flux tend to grow, while those with a small flux tend to disappear.

Below we describe the model following the description provided in (Tero et al., 2007) and (Zhang, Wang et al., 2014).

Let N_1 and N_2 be the source (starting) and sink (ending) nodes, respectively, of the graph, let the edge (tube) between node N_i and N_j be expressed as ij, and let Q_{ij} be the flux in tube ij. If the flow on this tube is approximately a Poiseuille flow (a laminar flow of an incompressible fluid induced by a constant positive pressure difference or pressure drop in a pipe), then the flux Q_{ij} can be calculated as:

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

where p_i is the pressure at node N_i, D_{ij} is the conductivity of tube ij and L_{ij} is its length. As the inflow and outflow at each node must be balanced (the principle of *conservation of flow*), we have

$$\sum_{j \neq 1,2} Q_{ij} = 0. \tag{2}$$

For the source node (N_1) and the sink node (N_2) the following hold:

$$\begin{aligned} \sum_i Q_{i1} + I_0 &= 0, \\ \sum_i Q_{i2} - I_0 &= 0, \end{aligned} \tag{3}$$

where I_0 is the flux flowing out of the source node (a fixed value; equivalently, I_0 is the total flux flowing into the sink node). It is assumed that conductivity D_{ij} changes over time as the flux Q_{ij} changes, such that the evolution of $D_{ij}(t)$ can be expressed as

$$\frac{d}{dt}D_{ij} = f(|Q_{ij}|) - rD_{ij}, \tag{4}$$

where r is a decay parameter for a tube. The conductivity of a tube will vanish over time if there is no flux along it.

From (1)–(3), the network equation for the pressure can be deduced (Zhang, Wang et al., 2014):

$$\sum_i \frac{D_{ij}}{L_{ij}}(p_i - p_j) = \begin{cases} +1 & \text{for } j = 1, \\ -1 & \text{for } j = 2, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

By setting $p_2 = 0$ as a basic pressure level, all p_i can be determined from (5) and in turn Q_{ij} can be calculated.

Following the assumption in Tero et al. (2007) that $f(Q) = |Q|$ and with the flux calculated, the conductivity can be calculated using (6):

$$\frac{D_{ij}^{n+1} - D_{ij}^{n}}{\delta t} = |Q| - D_{ij}^{n+1}. \tag{6}$$

A more detailed description of the above is provided in Tero et al. (2007). In addition to its application to standard shortest path problems, a variant on this model has been applied to solve constrained shortest path problems (Zhang, Wang et al., 2014).

5.1.2 Improved Physarum polycephalum Algorithm

In the improved *Physarum polycephalum* algorithm (IPPA), the above mathematical model is modified to incorporate an energy constraint. Under this modification it is assumed that tubes need to consume energy in order to be maintained, and that this energy is obtained from nutrients in the flow through the tubes. If the net energy gain is positive, the tubes grow and conductivity increases, otherwise the tubes wither and vanish over time. As the tubes alter their physical state, the flux of each tube also changes. Over time, the tubes in the network tend to converge to a steady state. The IPPA is described as follows in Zhang, Wang et al. (2014).

The energy E, the flux Q, and the conductivity D are defined as below:

$$E_1 = f(Q), \tag{7a}$$

$$E_2 = g(D), \tag{7b}$$

$$\Delta D = h(E_3), \tag{7c}$$

where E_1 is the energy that can be provided by the tube when its flux reaches Q; E_2 is the energy consumed by the tube when its conductivity is equal to D and ΔD shows how the conductivity changes when the remaining energy (i.e. the energy provided by the flux less the energy consumed by the tube) is E_3. Therefore, (6) becomes:

$$\Delta D_{ij} = h(f(Q_{ij} - g(D_{ij}))) \, \Delta t, \tag{8}$$

or, taking the derivative,

$$\frac{d\,\Delta D_{ij}}{dt} = h(f(Q_{ij} - g(D_{ij}))). \tag{9}$$

Next, the functions f, g and h are modified to render them operational. In order to preserve the law of conservation of energy (as noted in Zhang, Wang et al. (2014) the formulation in (6) does not obey this law) it is assumed that the total energy provided by the flux from the start to the end node is constant, and is therefore independent of path. Therefore, f can be defined as

$$f(Q_{ij}) = \frac{Q_{ij}(p_i - p_j)}{p_s - p_e}, \tag{10}$$

where s and e are the starting and ending node respectively, and p_i and p_j are the pressure at nodes i and j respectively.

It is assumed that the energy required to maintain tubes is a function of the conductivity and length of the underlying tube, and g is defined as

$$g(D_{ij}) = D_{ij}L_{ij}. \tag{11}$$

The function h which relates the level of net energy in a tube to the change in conductivity is defined so as to take tube length into account:

$$h(E_3) = \frac{E_3}{L_{ij}}. \tag{12}$$

The longer a tube is, the more energy it requires for its maintenance.

Combining (10)–(12), the following can be obtained:

$$\frac{dD_{ij}}{dt} = \frac{Q_{ij}(p_i - p_j)}{L_{ij}(p_s - p_e)} - D_{ij}. \tag{13}$$

The pseudocode for the IPPA is presented in Algorithm 1. As illustrated by Zhang, Wang et al. (2014), the IPPA can produce competitive results when compared with ant colony optimisation for a shortest path problem.

Recently, this approach has been extended to directed networks (where movement may only happen in one direction along an arc, not both), in the context of traffic flow problems possibly involving one-way streets (Zhang and Mahadevan, 2018).

Algorithm 1: Improved *Physarum Ploycephalum* Algorithm

Let N be the number of nodes in the network;
Let L be the $N \times N$ matrix whose (i, j) entry L_{ij} is the length between node i and j;
Let s and e be the start and end nodes;
Let $D_{ij} \in (0, 1]$ for all $i, j = 1, 2, \ldots, N$;
Let $Q_{ij} := 0$ for all $i, j = 1, 2, \ldots, N$;
Let $p_i := 0$ for all $i = 1, 2, \ldots, N$;
Set iteration counter $t = 0$;
while *termination criteria not met* **do**
  Let $p_e := 0$ (pressure at ending node e) ;
  Calculate the pressure of every node in the network solving (5);
  Let $Q_{ij} := D_{ij}(p_i - p_j)/L_{ij}$ using (1);
  Let $D_{ij} := \frac{1}{2}\left(\frac{Q_{ij}(p_i - p_j)}{L_{ij}(p_s - p_e)} + D_{ij}\right)$ using (6);
  Let $t = t + 1$;
end
Output the best solution (shortest path) found;

5.2 Physarum-Based Ant Colony System

In this section we highlight work of Lu et al. (2014) and Zhang, Gao et al. (2014), who combined the Tero model with an ant colony system algorithm to develop a hybrid *Physarum*-Based Ant Colony System (denoted as PM-ACS).

Canonical versions of ant colony optimisation (ACO) algorithms can be prone to premature convergence, and a multitude of variant algorithms have been developed to overcome this issue, for example by limiting the rate of pheromone build-up on arcs. These algorithms attempt to maintain continual exploration of novel solutions over time and therefore avoid undue exploitation of already discovered tours. A key suggestion from Lu et al. (2014) and Zhang, Gao et al. (2014) is that a hybrid PMM-ACO algorithm, where the updating strategy for the pheromone matrix is partially based on a PMM (*Physarum*-inspired mathematical model), can improve the efficiency and robustness of ACO algorithms.

The PMM used in this algorithm focusses on feedback regulation of the thickness of each tubular pseudopodia (*tube* in the network) arising from changes in the internal protoplasmic flow. Higher rates of protoplasmic flow stimulate an increase in the tube diameter (i.e., tubes get bigger as internal streaming rates increase) and low flow rates lead to a reduction and eventual disappearance of tubes (Fig. 3).

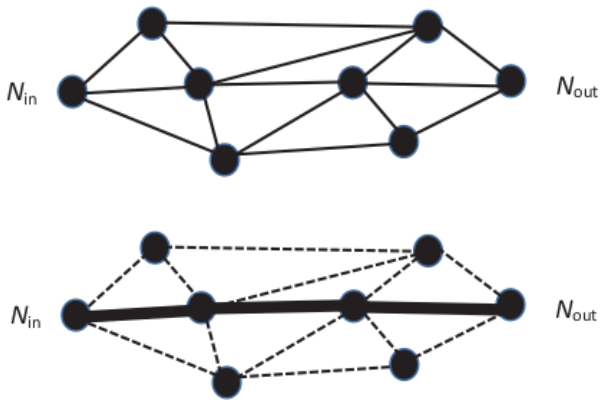

Figure 3 PMM: initial network (top); final network after shortest path has been reinforced (bottom). N_{in} and N_{out} are the inlet and outlet nodes

Below, we outline the workings of the algorithm drawing on the description provided in Lu et al. (2014).

5.2.1 Algorithm

The PM-ACS algorithm assumes that there is a *Physarum* network with simulated pheromone flowing along the tubes in

the network. Applying the model to the TSP, nodes represent cities and the tubes of the *Physarum* network are the paths connecting the cities.

As the algorithm iterates, the quantity of pheromone in each tube of the *Physarum* network dynamically changes. The update of the global pheromone matrix considers both the pheromone released by ants as they construct tours and the simulated pheromone flows in the *Physarum* network which also models these cities and their connections. The PMM acts to modify the *Physarum* network over time, reinforcing (thickening) the arcs on the shorter tours of the cities. The classical formulation of the TSP is adopted and it is assumed that the object is to construct the shortest tour of n cities, such that each city is visited exactly once, and the tour ends in the city in which it started.

As in the canonical ACS algorithm, the ants construct tours using the information in the pheromone matrix to guide their choice of exit path at each node (city) as they build their tour. In canonical ACS, the deposit and evaporation steps which determine the updates in the pheromone matrix are governed by

$$\tau_{ij}(t+1) = \tau_{ij}(t)(1-p) + p\,\Delta\tau^*_{ij}, \tag{14}$$

where only the arcs traversed by the best-so-far ant (on tour T^*) participate in the pheromone deposit/evaporation process. The term $\Delta\tau^*_{ij}(t)$ is equal to $1/L^*$, where L^* is the length of the best-so-far tour.

Under PM-ACS, the global pheromone matrix update rule from ACS is modified by adding an additional term which considers the quantity of pheromone in the *Physarum* network on all the arcs in the best-so-far solution, as follows:

$$\tau_{ij}(t+1) = \tau_{ij}(t)(1-p) + p\,\Delta\tau^*_{ij} + \epsilon\frac{Q_{ij}M}{I_0}, \quad \text{for all arcs } ij \text{ on } T^*, \tag{15}$$

where M is the number of tubes (arcs) between the cities in the TSP problem, Q_{ij} is the flux through a tube connecting city i and j, and I_0 is the flux (assumed to be a fixed quantity) between the inlet node to a network and the outlet node of the network (i.e. it represents the flow across the network). The parameter ϵ determines the effect of the flowing pheromone in the *Physarum* network on the final update in the pheromone matrix. The value of ϵ is calculated using

$$\epsilon = 1 - \frac{1}{1+\lambda^{t_{\text{PMM}}/2-(t+1)}}. \tag{16}$$

In (16), t_{PMM} is the total number of steps of iteration affected by the PMM process, t is the current iteration number (time step) and $\lambda \in (1, 1.2)$. As t gets large, the value of ϵ becomes smaller, therefore, the impact of flows in the *Physarum* network reduces as the PM-ACS algorithm iterates and as the ants converge on a good solution.

The relationship between conductivity and flux in each tube is modelled as:

$$Q_{ij} = \frac{1}{M}\sum_{m=1}^{M}\left|\frac{D_{ij}}{L_{ij}}(p_i^m - p_j^m)\right|, \tag{17}$$

where Q_{ij} represents the flux through a tube connecting node i and j, L_{ij} is the length of tube connecting nodes (i, j), D_{ij} is a measure of the conductivity of the tube connecting nodes i and j, and the pressure at node i is p_i^m. The flux through a tube is related to its conductivity, its length and the pressure at the node on each end of the tube.

The conductivity can be considered as the flow capacity of a tube and is related to the tube's thickness (diameter). All tubes have an initial assigned value for D_{ij} and if a tube subsequently becomes thicker, its conductivity will be enhanced. In turn, as conductivity increases, all other things being equal, so does the rate of flux.

In an iteration of the PMM, each pair of nodes connected by a tube can be selected as inlet / outlet nodes. The flux input to a node must equal the flux output from that node under an assumption of conservation of flow. As above, I_0 is a fixed quantity, being the flux between inlet node to a network and the outlet node to the network. When two nodes a and b connected by the m^{th} tube are selected as inlet and outlet nodes respectively, the pressure on each node p_i^m is calculated using Kirchhoff's Law as follows:

$$\sum_i \frac{D_{ij}}{L_{ij}}(p_i^m - p_j^m) = \begin{cases} -I_0 & \text{for } j = a, \\ I_0 & \text{for } j = b, \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

The above process iterates until all pairs of nodes in each tube have been selected as inlet or outlet nodes. The flux Q_{ij} is calculated using (17). The conductivity of a tube adapts according to the flux based on

$$\frac{dD_{ij}}{dt} = \frac{|Q_{ij}|}{1+|Q_{ij}|} - D_{ij} \tag{19}$$

The conductivities at the next iteration step are fed back to (18), and the flux is updated using (17). Based on the positive feedback mechanism between conductivity and flux, the shorter tubes (called *critical tubes*) become wider and are maintained as connections, while other tubes become narrower and eventually disappear.

The pseudocode for the PM-ACS algorithm is outlined in Algorithm 2. Aspects of the ACS algorithm which remain unchanged under PM-ACS, such as the processes by which ants construct a new tour and the local pheromone matrix update step, are not discussed here.

5.2.2 Discussion

The application of the PM-ACS algorithm requires the setting of several parameters. The parameter values used by Lu et al. (2014) are outlined in Table 1, with N being the number of cities. A sensitivity analysis concerning the impact on the results from different choices of parameter settings is provided in Zhang, Gao et al. (2014).

The TSP is representative of a wide array of graph problems, and algorithms for efficient solution to TSP problems are of general interest. A feature of slime moulds when foraging is their capability to allocate biomass efficiently in order to capture resources. One aspect of this is their ability to stream protoplasm to pseudopodia which have encountered resources and away from locations without resources.

Algorithm 2: PM-ACS Algorithm

Set values for $\alpha, \beta, p, s, q_0, \lambda, I_0, t_{\text{PMM}}, t_{\max}$;
Set initial values for pheromone levels on each arc $\tau_{ij}(0)$ and conductivity of each tube D_{ij};
Set iteration counter $t = 0$;
while $t < t_{\max}$ **do**
  for $k = 1$ **to** s *(all s ants)* **do**
    Construct a tour by ant k using approach in canonical ACS algorithm;
    Update the local pheromone matrix;
  end
  Let $k_{\text{best}} =$ global best ant (shortest tour found);
  Let $S_{\min} =$ length of tour generated by k_{best} ant;
  Calculate flowing pheromone in the *Physarum* network using (17)–(18) and update the conductivity of each tube using (19);
  Update the global pheromone matrix using (15);
  Let $t = t + 1$;
end
Output the best solution (best tour) found;

Table 1 Parameter values used in Lu et al. (2014)

Parameter	Description	Value
α	Relative importance of pheromone trail	1
β	Relative importance of heuristic information	2
p	Pheromone evaporation rate	0.8
s	Number of ants	N
q_0	Parameter in range $[0, 1]$	0.1
λ	Parameter impacting ϵ	1.05
I_0	Fixed flux flowing in Physarum network	20
t_{PMM}	Total steps of iteration affected by PMM	300
$t_{\max}$	Maximum number of algorithm iterations	300
$\tau_{ij}(0)$	Initial pheromone level on edge ij	1
D_{ij}	Initial conductivity in tube ij	1

In the PMM-ACS algorithm, a model of this streaming process is used to contribute to the updating of the pheromone matrix which is used by the ants in constructing their solutions, tours in the case of the TSP.

6 Critique of Slime Mould Algorithms

In this section, we compare slime mould approaches against other graph based algorithms and contribute new perspectives on slime mould algorithms, in addition to the literature to date.

As discussed in Section 2.3 on classical graph algorithms, there is a large body of work on graph optimisation in pure graph models — generic models consisting only of vertices and (weighted) edges linking them. For example, there exist very efficient specialisations to graphs with particular properties, e.g., sparsity. Slime mould algorithms, in common with many other natural computing algorithms, may have a higher computational cost for the same quality of solution (whether an exact or — as here — approximate solution). In particular, IPPA has a runtime of $o(n^3)$ (where n is the number of vertices) and appears to be at least an order of magnitude more costly than Dijkstra's algorithm (Zhang, Wang et al., 2014, p. 5, Figure 5), and even more costly compared to approaches using special data structures such as (Fredman and Tarjan, 1987).

The question then naturally arises as to whether a new paradigm, such as slime mould algorithms, has value in its own right: what other strengths and/or weaknesses might they have, and can these outweigh the higher computational cost?

An important factor (and indeed to some extent a design decision) is how close to nature the natural computing algorithm is. A crucial choice in all biologically-inspired computing algorithms is which of the currently-understood aspects of the modelled organism to incorporate in the problem/algorithm definition, and which to omit (Brabazon et al., 2015a).

6.1 Strengths of slime-mould-inspired approaches

We first consider what strengths may arise from using an approach inspired by our current understanding of nature, in particular, foraging behaviours of slime moulds.

Compared to pure graph approaches, natural computing (including slime mould approaches) may be compared and contrasted in at least three ways:

Robustness The only purpose of an organism is to survive (Brabazon et al., 2015a) and so biological systems are not strict optimisers in just one area: robustness is more important as it means no life-threatening weaknesses. Slime mould biocomputers exhibit resilience in the sense of tolerance of accidental disconnection of the tubes (Nakagaki, Yamada et al., 2004; Reid et al., 2013a). A biologically-inspired algorithm — if well-designed, well-implemented and leveraging survival traits — should exbibit robustness, for instance, to network changes or faults in a network design problem. The IPPA finds all shortest paths at the same time (Zhang, Wang et al., 2014), whereas the classical Dijkstra algorithm finds just one; and the one path found depends on the ordering of nodes in the selection set (although Dijkstra's algorithm can be modified to find all such shortest paths, at an extra runtime cost).

Multiple objectives Because of the survival value of robustness, slime mould behaviours are fundamentally multiobjective: they find a good or optimal main objective, but also seek robustness.

Classical graph algorithms such as Dijkstra's and variants of it are good at finding an optimum shortest path spanning tree but without significant reengineering are not able to handle multiple competing objectives. However, as mentioned in Section 4, slime moulds do not just seek a shortest path, but trade off total length of connections against resilience (Nakagaki, Yamada et al., 2004; Reid et al., 2013a). Thus a potential strength of slime-mould-inspired algorithms is the handling of multiple objectives.

Dynamic environments Robustness and survivability imply ability to adapt to dynamic changes. Thus it is plausible that a biologically-inspired algorithm may show better behaviour in dynamic environments. This has been used in applying slime-mould-based algorithms to network flow problems (Zhang and Mahadevan, 2018). Classical max-flow algorithms are efficient (Section 2.3), but these approaches assume that the arc capacities are static. However, Physarum-inspired algorithms can be adaptable, because by adjusting tube thicknesses, the simulated slime mould can dynamically react to environmental changes, such as changes in flows in the network (e.g., traffic network) being modelled (Zhang and Mahadevan, 2018).

6.2 Choosing among metaheuristics

Next, we consider if and when one should choose slime mould approaches over other meta-heuristics. In the specific comparison of PM-ACO algorithms with traditional ant-colony algorithms, Zhang, Gao et al. (2014) report that experimental results in synthetic and real networks show the PM-ACO algorithms are more efficient and robust than the standard ACO algorithms, and are adaptable to solve the TSP with single or multiple objectives. The strong performance of slime-mould-based algorithms on shortest path problems, both undirected (Zhang, Wang et al., 2014) and directed (Zhang and Mahadevan, 2018), as well as on TSP problems, gives reason to believe these approaches may transfer well to a range of network problems.

The situation is more complex regarding the broader question of when slime-mould-based algorithms should be preferred over other metaheuristics. Recent work, e.g., (Antosiewicz et al., 2013; Gutierrez-Rodríguez et al., 2019; Kanda et al., 2016) has shown that on the TSP and

related problems, it is typical that the "best" algorithm may fundamentally depend on the characteristics of the problem instance being addressed. In fact, Kanda et al. (2016) and Gutierrez-Rodríguez et al. (2019) have used machine learning approaches to decide, on an instance-by-instance basis, which algorithm or metaheuristic to apply for the TSP or its super-problem, the Vehicle Routing Problem. An extension of this is the field of *hyperheuristics* (see, for example, (Burke et al., 2013)), where the solver may swap heuristics or metaheuristics in and out on the fly, during the run, according to the current status of the problem or stage of the search.

With this in mind, it is clearly very difficult to make a generalisation as to when one metaheuristic should be preferred over another. However, we may make some high-level comments, based on the above discussion of properties of slime-mould-based algorithms. Firstly, a slime-mould-inspired algorithm may be considered where robustness or resilience is a primary or secondary consideration, where this can be incorporated in a multi-objective function. Secondly, if the environment is dynamic, and network weights are changing, a slime-mould-inspired algorithm is capable of adjusting to these. However, as the work of Kanda et al. (2016), Gutierrez-Rodríguez et al. (2019) and others has shown, the choice of appropriate metaheuristic is an active area of research and one that is rapidly evolving, with no firm conclusions thus far.

6.3 Making slime-mould-inspired algorithms more faithful to real-world slime mould behaviour

Finally, we consider whether a better understanding of slime moulds may give new strengths to slime-mould-inspired algorithms.

In the slime-mould-inspired algorithms disussed here, some aspects of slime mould behaviour are deliberately omitted or simplified by designer choice, and so the algorithms are not completely faithful to nature. These design choices represent the algorithm designer's belief, given experimental evidence, that the benefit in simplicity and tractability outweighs the loss of accuracy of representation.

However, an issue with current slime-mould-inspired approaches is that real-world slime mould behaviours are imperfectly understood. In addition to the simplifications made to slime-mould-inspired algorithms by choice of the designer, it is possible that they miss some essential real-world element of slime mould foraging behaviour and so lack faithfulness to nature in another way. It is unclear at present whether this happens and, if so, to what degree. What can be said is that new developments in biological understanding do not always transfer immediately to the domain of foraging-inspired algorithms.

Recently, new hypotheses have been proposed on the physical processes underlying slime mould movement and tube diameter increase or reduction, for example, that cytoplasmic flows are organised in peristaltic waves of wavelength matching the organism size: "Contrary to long-lasting speculations about localized pumps driving pressure difference, the common understanding now is that flows arise through network-wide, self-organized contractions of the actin cortex" (Alim, 2017). These wave mechanisms, if implemented in algorithms instead of the Tero model, could plausibly give new algorithmic behaviours and capabilities, and so extend the range of slime-mould-inspired algorithms. It may be that more closely mimicking slime moulds can lead to better algorithms because of adoption of some crucial fact.

As mentioned in Section 2.3, many network design problems where the sum of weights of a certain subset of edges must be minimised, subject to constraints enforcing robustness, are NP-hard: such problems offer opportunities for slime-mould-inspired approaches, given the capabilities of slime moulds as discussed in Section 4. In particular, Zhu et al. (2018) found promising results in terms of both computational cost and solution quality when applying *Physarum polycephalum* as a biocomputer to solve small instances of the TSP.

Note that the three-dimensional representations discussed in Section 4.1.2 are more detailed models (involving topography) than the graph models introduced in Section 2.3; the topography itself acts as an external memory. This application together with the others in Section 4 demonstrate that slime mould computation is applicable to a range of problems, including but not restricted to "pure" graph models. This indicates that slime-mould-inspired algorithms may not run as fast as "pure" graph algorithms, but this may be compensated for by a wider breadth of applicability, and by the fact that they may be able to capture more aspects of the real world problem/domain being modelled than the simpler "pure" graph model can.

7 Conclusions

We now realise that even apparently 'simple' nonneuronal organisms are capable of quite sophisticated behaviours. This suggests that brains and neuronal networks are not prerequisites for complex decision making (Reid et al., 2015). All organisms are faced with a multitude of decisions when foraging, including searching for resources, efficient allocation of resources between exploitation of current food resources and exploration for new resources, anticipation of periodic environmental events, and making risk–return tradeoffs when foraging. In this paper, we have provided a brief introduction to some of the foraging behaviours the plasmodial slime mould *Physarum polycephalum* and described a number of algorithms which have drawn inspiration from these behaviours in order to solve graph optimisation problems. A range of avenues are open for further exploration in this domain with Gao et al. (2018) providing a good taxonomy for considering these.

Much of the work in this area so far has concerned 'proof of concept' of the relevant algorithms and further work is required to comprehensively assess their scalability and effectiveness on benchmark problems. Despite the relative immaturity of the algorithms it is interesting to note that a stream of literature is emerging which applies these algorithms to real-world problem domains including supply-chain network design (Zhang et al., 2017a,b), road network design (Zhang and Mahadevan, 2018) and electrical

grid load shedding (Gao et al., 2017). Clearly, many real-world problems can be encompassed in a graph-optimisation framework and accordingly, if their effectiveness and efficiency can be successfully verified, there will be plentiful opportunities to apply the algorithms to practical problems.

Of course, it is also important to note that the developed algorithms are very simplified representations of (the imperfectly understood) real-world foraging behaviours of *Physarum polycephalum* and other slime moulds and doubtless future biological research concerning these organisms will open up new avenues of investigation.

Taking a wider perspective, there are striking similarities between the group decision-making processes of organisms such as social bacteria and slime moulds, and those of social insects. In each case the colony-level decision is based on information gathered by individual organisms (or cells), in a bottom-up, emergent process with feedback loops playing an important role. The investigation of slime mould behaviours could therefore cast light on a wider class of group decision-making processes. Indeed, it is speculated that similar mechanisms may govern the decision-making processes in higher animals. Marshall et al. (2009) compare the decision-making processes of social insect colonies and the brain and notes that both individual ants and individual neurons are relatively simple information processors with incomplete information on the environment. For example, sensory information may be ambiguous and time varying, therefore individual information processors have local rather than global information concerning the environment. In insect societies, slime mould, bacterial colonies and brains, decisions are not made by individual information processors but rather as a result of an emergent process with information being accumulated concerning alternative decision choices. Future work is required in order to better integrate these currently distinct research areas.

References

Adamatzky, A. (2010) *Physarum Machines: Computers from Slime Moulds*, World Scientific Publishing Company, Singapore.

Adamatzky, A. (2014) 'Route 20, Autobahn 7, and Slime Mold: Approximating the Longest Roads in the USA and Germany with Slime Mold on 3-D Terrains', *IEEE Transactions on Cybernetics*, Vol. 44, No. 1, pp. 126–136.

Adamatzky, A., Allard, O., Jones, J. and Armstrong, R. (2017) 'Evaluation of French motorway network in relation to slime mould transport networks', *Environment and Planning B: Urban Analytics and City Science*, Vol. 44, No. 2, pp. 364–383.

Alim, K. (2018) Fluid flows shaping organism morphology. *Phil. Trans. R. Soc. B* Vol. 373: 20170112, pp. 1–5.

Amintoosi, M., Fathy, M., Mozayani, N. and Rahmani, A. (2007) 'A Fish School Clustering Algorithm: Applied to Student Sectioning Problem', In: *Proceedings of 2007 International Conference on Life System Modelling and Simulation (LSMS)*, Published as a supplementary volume to Dynamics of Continuous Discrete & Impulse Systems, series B: Applications and Algorithms, Vol. 2, pp. 696–699, Watam Press.

Antosiewicz, M., Koloch, G. and Kamiński, B. (2013). 'Choice of best possible metaheuristic algorithm for the travelling salesman problem with limited computational time: quality, uncertainty and speed', *Journal of Theoretical and Applied Computer Science*, Vol. 7, No. 1, pp. 46–55.

Bansal, J. C., Sharma, H., Arya, K. V. and Nagar, A. (2013) 'Memetic search in artificial bee colony algorithm', *Soft Computing*, Vol. 17, No. 10, pp. 1911–1928.

Bar-On, Y., Phillips, R. and Milo, R. (2018) 'The biomass distribution on Earth', *Proceedings of the National Academy of Sciences*, Vol. 115, No. 25, pp. 6506–6511.

Bastos Filho, C., de Lima Neto, F., Lins, A., Nascimento, A. and Lima, M. (2008) 'A Novel Search Algorithm Based on Fish School Behavior', In: *Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 2646–2651, IEEE Press.

Bonabeau, E., Dorigo, M. and Theraulaz, G. (1999) *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, Oxford.

Boisseau, R., Vogel, D. and Dussutour A. (2016) 'Habituation in non-neural organisms: evidence from slime moulds', *Proceedings of the Royal Society B*, Vol. 283, No. 1829, pp. 20160446.

Bora, T., Coelho, L. and Lebensztajn, L. (2012) 'Bat-inspired Optimization Approach for the Brushless DC Wheel Motor Problem', *IEEE Transactions on Magnetics*, Vol. 48, No. 2, pp. 947–950.

Brabazon, A., Cui, W. and O'Neill, M. (2013) 'Examining the Role of Perception, Social and Private Information in Honey Bee Foraging Algorithms', *International Journal of Innovative Computing and Applications*, Vol. 5, No. 4, pp. 240–261.

Brabazon, A., Cui, W. and O'Neill, M. (2015) 'Information Propagation in a Social Network: The Case of The Fish Algorithm', In: Dariusz Krol, Damien Fay, Bogdan Gabrys (eds). *Propagation Phenomena in Real World Networks*, pp. 27-51, Springer, Berlin.

Brabazon, A., McGarraghy, S. and Agapitos, A. (2015a) 'Plant Propagation-Inspired Algorithms', In: Sean Washington (eds). *New Developments in Evolutionary Computation Research*, Nova Science Publishers, New York.

Brabazon, A., O'Neill, M. and McGarraghy, S. (2015b) *Natural Computing Algorithms*, Springer, Berlin.

Brabazon, A., Cui, W. and O'Neill, M. (2016) 'Raven Roosting Optimisation Algorithm', *Soft Computing*, Vol. 20, No. 2, pp. 525–545.

Brabazon, A. and McGarraghy, S. (2018) *Foraging-Inspired Algorithms for Optimisation*, Springer, Berlin.

Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., and Qu, R. (2013). 'Hyper-heuristics: a survey of the state of the art', *Journal of the Operational Research Society*, Vol. 64, No. 12, pp. 1695–1724. doi: 10.1057/jors.2013.71.

Cai, W., Yang, W. and Chen, X. (2008) 'A Global Optimization Algorithm Based on Plant Growth Theory: Plant Growth Optimization', In: *Proceedings of 2008 International Conference on Intelligent Computation Technology and Automation (ICICTA 2008)*, pp. 1194–1199, IEEE Press.

Cai, X., Li, W., Kang, Q., Wang, L. and Wu, Q. (2015) 'Bat algorithm with oscillation element', *International Journal of Innovative Computing and Applications*, Vol. 6, No. 3/4, pp. 171–180.

Carroll, P., Fortz, B., Labbé, M. and McGarraghy, S. (2013) 'A Branch and Cut Algorithm for the Ring Spur Assignment Problem', *Networks*, Vol. 61, No. 2, pp. 89–103. DOI: 10.1002/net.21495.

Chong, C., Low, M., Sivakumar, A. and Gay, K. (2006) 'A Bee Colony Optimization Algorithm to Job Shop Scheduling', In: *Proceedings of the 2006 Winter Simulation Conference (WinterSim 2006)*, pp. 1954–1961, IEEE Press.

Cormen, T., Leiserson, C., Rivest, R. and Stein, C. (2001) *Introduction to Algorithms*, MIT Press, Cambridge, MA. Second edition.

Cui, W., Brabazon, A. and Agapitos, A. (2015) 'Extending the Bat Foraging Metaphor for Optimisation Algorithm Design', *International Journal of Metaheuristics*, Vol. 4, No. 1, pp. 1–26.

Dorigo, M. (1992) *Optimization, Learning and Natural Algorithms*, Ph.D. Dissertation, Politecnico di Milano.

Dorigo, M. and DiCaro, G. (1999) 'Ant colony optimization: a new meta-heuristic', In: *Proceedings of IEEE Congress on Evolutionary Computation (CEC 1999)*, pp. 1470–1477. IEEE Press.

Dorigo, M. and Stützle, T. (2004) *Ant Colony Optimization*, MIT Press, Cambridge, Massachusetts.

Dorigo, M., Maniezzo, V. and Colorni, A. (1996) 'Ant system: optimization by a colony of cooperating agents', *IEEE Transactions on Systems, Man, And Cybernetics - Part B: Cybernetics*, Vol. 26, No. 1, pp. 29–41.

Dussutour, A., Latty, T., Beekman, M. and Simpson, S. (2010) 'Amoeboid organism solves complex nutritional challenges', *Proceedings of the National Academy of Sciences*, Vol. 107, No. 10, pp. 4607–4611.

Fredman, M. L. and Tarjan, R. E. (1987) 'Fibonacci Heaps and Their Uses in Improved Network Optimization Algorithms', *Journal of the ACM*, Vol. 34, No. 3, pp 596–615.

Gandomi, A., Yang, X.S., Alavi, A. and Talatahari, S. (2013) 'Bat algorithm for constrained optimization tasks', *Neural Computing and Applications*, Vol. 22, No. 6, pp. 1239–1255.

Gao, C., Liu, C., Schenz, D., Li, X., Zhang, Z., Jusup, M., Wang, Z., Beekman, M. and Nakagaki, T. (2018) 'Does being multi-headed make you better at solving problems? A survey of Physarum-based models and computations', *Physics of Life Reviews*, https://doi.org/10.1016/j.plrev.2018.05.002

Gao, H., Liu, Y. and Diao, M. (2011) 'Robust multi-user detection based on quantum bee colony optimisation', *International Journal of Innovative Computing and Applications*, Vol. 3, No. 3, pp. 160–168.

Gao, H. and Xu C. (2014) 'Quantum-inspired cultural bacterial foraging algorithm for direction finding of impulse noise', *International Journal of Innovative Computing and Applications*, Vol. 6, No. 1, pp. 44–54.

Gao, C., Chen, S., Li, X., Huang, J. and Zhang, J. (2017) 'A Physarum-inspired optimization algorithm for load-shedding problem', *Applied Soft Computing*, Vol. 61, pp. 239–255.

Goldfield, E. (2018) *Bioinspired Devices: Emulating Nature's Assembly and Repair Processes*, Harvard University Press, Cambridge, MA.

Gunji, Y., Shirakawa, T., Niizato, T. and Haruna, T. (2008) 'Minimal model of a cell connecting amoebic motion and adaptive transport networks', *Journal of Theoretical Biology*, Vol. 253, No. 4, pp. 659–667.

Gutierrez-Rodríguez, A. E., Conant-Pablos, S. E., Ortiz-Bayliss, J. C., Terashima-Marín, H. (2019). 'Selecting meta-heuristics for solving vehicle routing problems with time windows via meta-learning', *Expert Systems With Applications*, Vol. 118, pp. 470–481.

He, D., Qu, L. and Guo, X. (2009) 'Artificial Fish-school Algorithm for Integer Programming', In: *Proceedings of IEEE International Conference on Information Engineering and Computer Science (ICIECS)*, pp. 1–4, IEEE Press.

Hickey, D. and Noriega, L. (2008) 'Insights into Information Processing by the Single Cell Slime Mold *Physarum Polycephalum*', In: *Proceedings of UKACC Control Conference*, Manchester, UK, September 2–4, 2008, http://fusion-edu.eu/HAM/Papers_files/p246.pdf

Ibri, S., Drias, H. and Nourelfath, M. (2010) 'A parallel hybrid ant-tabu algorithm for integrated emergency vehicle dispatching and covering problem', *International Journal of Innovative Computing and Applications*, Vol. 2, No. 4, pp. 226–236.

Jones, J. (2016) 'Applications of Multi-Agent Slime Mould Computing', *International Journal of Parallel, Emergent and Distributed Systems*, Vol. 31, No. 5, pp. 420–449.

Kanda, J., de Carvalho, A., Hruschka, E., Soares, C., Brazdil, P. (2016). 'Meta-learning to select the best meta-heuristic for the Traveling Salesman Problem: A comparison of meta-features', *Neurocomputing*, Vol. 205, pp. 393–406

Karaboga, D. (2005) 'An idea based on honeybee swarm for numerical optimization', *Technical Report TR06, Engineering Faculty, Computer Engineering Department, Erciyes University*, http://mf.erciyes.edu.tr/abc/pub/tr06_2005.pdf.

Karaboga, D. and Akay, B. (2009) 'A survey: algorithms simulating bee intelligence', *Artificial Intelligence Review*, Vol. 31, No. 1-4, pp. 61–85.

Latty, T. and Beekman, M. (2009) Food quality affects search strategy in the accellular slime mould, *Physarum polycephalum*. *Behavioral Ecology*, Vol. 20, No. 6, pp. 1160–1167.

Li, H., Xiao, R. and Wu, H. (2016) 'Modelling for combat task allocation problem of aerial swarm and its solution using wolf pack algorithm', *International Journal of Innovative Computing and Applications*, Vol. 7, No. 1, pp. 50–59.

Li, X., Shao, Z. and Qian, J. (2002) 'An optimizing method based on autonomous animats: fish swarm algorithm', *Systems Engineering Theory and Practice*, Vol. 22, pp. 32–38 (in Chinese).

Lu, Y., Liu, Y., Gao, C., Tao, L. and Zhang, Z. (2014) 'A Novel *Physarum*-Based Ant Colony System for Solving the Real-World Travelling Salesman Problem', In: *Proceedings of 5th International Conference on Swarm Intelligence (ICSI 2014)*, LNCS 8794, pp. 173–180, Springer, Berlin.

Marshall, J., Bogacz, R., Dornhaus, A., Planque, R., Kovacs, T. and Franks, N. (2009) 'On optimal decision-making in brains and social insect colonies', *Journal of the Royal Society Interface*, Vol. 6, No. 40, pp. 1065–1074.

Marwan, W. (2010) 'Amoeba-Inspired Network Design', *Science*, Vol. 327, No. 5964, pp. 419–420.

Mehrabian, A. and Lucas, C. (2006) 'A novel numerical optimization algorithm inspired from weed colonization', *Ecological Informatics*, Vol. 1, No. 4, pp. 355–366.

Mirjalili, S. (2015) 'How effective is the Grey Wolf Optimizer in training multi-layer perceptrons', *Applied Intelligence*, Vol. 43, No. 1, pp. 150–161.

Mirjalili, S., Mirjalili, S, M. and Lewis, A. (2014) 'Grey Wolf Optimizer', *Advances in Engineering Software*, Vol. 69, pp. 46–61.

Müller, S., Airaghi, S., Marchetto, J. and Koumoutsakos, P. (2000) 'Optimization algorithms based on a model of bacterial chemotaxis', In: *Proceedings of the 6th International Conference on the Simulation of Adaptive Behavior: From Animals to Animats (SAB 2000)*, pp. 375–384, Cambridge, Massachusetts: MIT Press.

Nakagaki, T., Iima, M., Ueda, T., Nishiura, Y., Saigusa, T., Tero, A., Kobayashi, R. and Showalter, K. (2007) 'Minimum-risk Path Finding by an Adaptive Amoebal Network', *Physical Review Letters*, Vol. 99, No. 6, pp. 068104.

Nakagaki, T., Kobayashi, R., Nishiura, Y. and Ueda, T. (2004) 'Obtaining multiple separate food sources: behavioural intelligence in the *Physarum* plasmodium', *Proceedings of the Royal Society London B*, Vol. 271, No. 1554, pp. 2305–2310.

Nakagaki, T., Yamada, H. and Hara, M. (2004) 'Smart network solutions in an amoeboid organism', *Biophysical Chemistry*, Vol. 107, No. 1, pp. 1–5.

Nakagaki, T., Yamada, H. and Toth, A. (2000) 'Maze-solving by an amoeboid organism', *Nature*, Vol. 407, No. 6803, pp. 470.

Nakamura, R., Pereira, L., Costa, K., Rodrigues, D., Papa, J. and Yang, X.S. (2012) 'BBA: A Binary Bat Algorithm for Feature Selection', In: *Proceedings of XXV SIBGRAPI Conference on Graphics, Patterns and Images*, pp. 291–297, IEEE Press.

Nakrani, S. and Tovey, C. (2004) 'On Honey Bees and Dynamic Server Allocation in Internet Hosting Centres', *Adaptive Behavior*, Vol. 12, No. 3-4, pp. 223–240.

Niknam, T., Azizipanah-Abarghooee, R., Zare, M. and Bahmani-Firouzi, B. (2013) 'Reserve Constrained Dynamic Environmental/Economic Dispatch: A New Multiobjective Self-Adaptive Learning Bat Algorithm', *IEEE Systems Journal*, Vol. 7, No. 4, pp. 763–776.

Passino, K. (2000) 'Distributed Optimization and Control Using Only a Germ of Intelligence', In: *Proceedings of the IEEE International Symposium on Intelligent Control*, pp. 5-13, IEEE Press.

Passino, K. (2002) 'Biomimicry of Bacterial Foraging for Distributed Optimization and Control', *IEEE Control Systems Magazine*, Vol. 22, No. 3, pp. 52–67.

Pedemonte, M. and Cancela, H. (2010) 'A cellular ant colony optimisation for the generalised Steiner problem', *International Journal of Innovative Computing and Applications*, Vol. 2, No. 3, pp. 188–201.

Pham, D., Ghanbarzadeh, A., Koc, E., Otri, S., Rahim, S. and Zaidi, M. (2006) 'The Bees Algorithm - A novel tool for complex optimisation problems', In: *Proceedings of International Production Machines and Systems (IPROMS 2006)*, pp. 454–459, Elsevier, UK.

Premaratne, U., Samarabandu, J. and Sidhu, T. (2009) 'A New Biologically Inspired Optimization Algorithm', In: *Proceedings of the Fourth International Conference on Industrial and Information Systems (ICIIS 2009)*, pp. 279–284, IEEE Press.

Reid, C. and Beekman, M. (2013a) 'Solving the Towers of Hanoi — how an amoeboid organism efficiently constructs transport networks', *The Journal of Experimental Biology*, Vol. 216, No. 9, pp. 1546–1551.

Reid, C., Beekman, M., Latty, T. and Dussutor, A. (2013b) 'Amoeboid organism used extracellular secretions to make smart foraging decisions', *Behavioral Ecology*, Vol. 24, No. 4, pp. 812–818.

Reid, C., Garnier, S., Beekman, M. and Latty, T. (2015) 'Information integration and multiattribute decision making in non-neuronal organisms', *Animal Behaviour*, Vol. 100, pp. 44–50.

Reid, C., Latty, T., Dussutor, A. and Beekman, M. (2012) 'Slime mold uses an externalized spatial memory to navigate in complex environments', *Proceedings of the National Academy of Sciences*, Vol. 109, No. 43, pp. 17490–17494.

Rui, T., Fong, S., Yang, X.S. and Deb, S. (2012) 'Nature-inspired Clustering Algorithms for Web Intelligence Data', In: *Proceedings of the IEEE, WIC, ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pp. 147–153, IEEE Press.

Salhi, A. and Fraga, E. (2011) 'Nature-inspired Optimisation Approaches and the New Plant Propagation Algorithm', In: *Proceedings of 2011 International Conference on Numerical Analysis and Optimization (ICeMATH 2011)*, pp. K2–1:K2–8.

Smith-Ferguson, J., Reid, C., Latty, T., and Beekman, M. (2017) 'Hänsel, Gretel and the slime mould—how an external spatial memory aids navigation in complex environments', *Journal of Physics D: Applied Physics*, Vol. 50, No. 41, pp. 414003.

Sumpter, D. and Brannstrom, A. (2008) 'Synergy in social communication', In: D'Ettorre and Hughes (Eds.) *Social Communication*, Oxford University Press.

Tang, S. K.Y., Marshall, W. F. (2018) 'Cell Learning', *Current Biology*, Vol. 28, R1171-R1189.

Tarjan, R. E. (1983), *Data Structures and Network Algorithms*, CBMS-NSF Regional Conference Series in Applied Mathematics, 44, Society for Industrial and Applied Mathematics.

Tero, A., Kobayashi, R. and Nakagaki, T. (2007) 'A mathematical model for adaptive transport network in path finding by true slime mold', *Journal of Theoretical Biology*, Vol. 244, No. 4, pp. 553–564.

Tero, A., Takagi, S., Saigusa, T., Ito, K., Bebber, D., Fricker, M., Yumiki, K., Kobayashi, R. and Nakagaki, T. (2010) 'Rules for Biologically Inspired Adaptive Network Design', *Science*, Vol. 327, No. 5964, pp. 439–442.

Tong, L., Wang, C., Wang, W. and Su, W. (2005) 'A global optimization bionics algorithm for solving integer programming—plant growth simulation algorithm', *Systems Engineering — Theory and Practice*, Vol. 25, No. 1, pp. 76–85.

Tsai, H-C. and Lin, Y-H. (2011) 'Modification of the fish swarm algorithm with particle swarm optimization formulation and communication behavior', *Applied Soft Computing*, Vol. 11, pp. 5367–5374.

Wang, H., Wang, W., Zhao, J., Zhu, H. and Sun, H. (2014) 'A hybrid artificial bee colony based on differential evolution for production scheduling problems', *International Journal of Innovative Computing and Applications*, Vol. 6, No. 2, pp. 55–62.

Yang, C., Tu, X. and Chen, J. (2007) 'Algorithm of Marriage in Honeybees Optimization Based on the Wolf Pack Search', In: *Proceedings of IEEE International Conference on Intelligent Pervasive Computing*, pp. 462–467, IEEE Press.

Yang, X.S. (2005) 'Engineering Optimization via Nature-Inspired Virtual Bee Algorithms', In: Mira J, Álvarez J (Eds.) *Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach*, pp. 317–323, Springer, Berlin.

Yang, X.S. (2010) 'A New Metaheuristic Bat-Inspired Algorithm', In: *Proceedings of Fourth International Workshop on Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, pp. 65–74, Springer, Berlin.

Yang, X.S. and Gandomi, A. (2012) 'Bat Algorithm: A Novel Approach for Global Engineering Optimization', *Engineering Computations*, Vol. 29, No. 5, pp. 464–483.

Yang, X.S. (2013) 'Bat Algorithm: Literature Review and Applications', *International Journal of Bio-Inspired Computation*, Vol. 5, No. 3, pp. 141–149.

Zahavi, A. (1971) 'The function of pre-roost gatherings and communal roosts', *Ibis*, Vol. 113, pp. 106–109.

Zhang, H., Zhu, Y. and Chen, H. (2012) 'Root Growth Model for Simulation of Plant Root System and Numerical Function Optimization', In: *Proceedings of 8th International Conference, Intelligent Computing Technology (ICIC 2012)*, Lecture Notes in Computer Science 7389, pp. 641–648, Springer, Berlin.

Zhang, Z., Gao, C., Liu, Y. and Qian, T. (2014) 'A universal optimization strategy for ant colony optimization algorithms based on the *Physarum*-inspired mathematical model', *Bioinspiration and Biomimetics*, Vol. 9, p. 036006.

Zhang, X., Wang, Q., Adamatzky, A., Chan, F., Mahadevan, S. and Deng, Y. (2014) 'An Improved Physarum polycephalum Algorithm for the Shortest Path Problem', *The Scientific World Journal*, Vol. 2014, Article ID 487069.

Zhang, X., Adamatzky, A., Chan, F., Mahadevan, S. and Deng, Y. (2017) 'Physarum solver: a bio-inspired method for sustainable supply chain network design problem', *Annals of Operations Research*, Vol. 254, No. 1–2, pp. 533–552

Zhang, X., Chan, F., Adamatzky, A., Mahadevan, S., Yang, H. and Zhang, Z. (2017) 'An intelligent physarum solver for supply chain network design under profit maximization and oligopolistic competition', *International Journal of Production Research*, Vol. 55, No. 1, pp. 244–263.

Zhang, X. and Mahadevan, S. (2018) 'A Bio-Inspired Approach to Traffic Network Equilibrium Assignment Problem', *IEEE Transactions On Cybernetics*, Vol. 48, No. 4, pp. 1304–1315.

Zhou, Y. and Liu, B. (2009) 'Two Novel Swarm Intelligence Clustering Analysis Methods', In: *Proceedings of the 5th International Conference on Natural Computation*, pp. 497–501, IEEE Press.

Zhu, L., Kim, S.-J., Hara, M. and Aono, M. (2018) 'Remarkable problem-solving ability of unicellular amoeboid organism and its mechanism', *R. Soc. open sci.* Vol. 5: 180396, pp. 1–13.