2014 47th Hawaii International Conference on System Science

Slime Mold Inspired Evolving Networks under Uncertainty (SLIMO)

Erik Kropat, Silja Meyer-Nieberg
Department of Computer Science
Universität der Bundeswehr München
Werner-Heisenberg-Weg 39, 85577 Neubiberg, Germany
Email: Erik.Kropat@unibw.de, Silja.Meyer-Nieberg@unibw.de

Abstract

Slime molds have received much attention in the recent years. Ever since it was shown that the organism can solve network problems, researchers have worked to transfer the working principles to optimization algorithms. This paper introduces an innovative network evolving approach which enables the use for challenging tasks where information on the state of edges may be scarce, uncertain, and changing. We present a slime mold-based optimization algorithm (SLIMO) that integrates time-dependent changes of the uncertainty layer. In addition, we study the adaptation of the slime mold evolution and the corresponding single path or multi path solutions of the shortest path problem on a grid. Examples of potential applications include important topics in disaster and crisis relief and in sensor networks.

Keywords

slime mold, evolving networks, natural computing, disaster relief, uncertainty

1. Introduction

Operations research in the context of disaster and crisis relief has to deal with uncertain and dynamically changing information. For an illustration, consider a supply network for relief operations during a flood. Usually, there are supply centers with support materials, possibly some distribution centers, and the operative sites. The rising of the water level, however, changes the topology of the network that can be used for transportation. Pathways that were still passable moments ago may become flooded and information on the state of the roads may be scarce and unreliable. The problem is aggravated by new operative sites which may come into existence while the emergency relief is ongoing. The information available for planning must therefore be regarded as uncertain and subject to change. This requires specialized algorithms which compute safe and short pathways efficiently and can –if necessary– re-route the transport. Similar tasks, although usually not as critical as during relief operations, present themselves to routing algorithms which have to take changing traffic conditions into account. Finding fast routes through a city during the rush hour is a demanding task since the system changes continually. Traffic jams and road blocks occur which cause cascading effects inside the network contributing to a nearly chaotic situation – especially in megacities.
In a first step, this paper presents a slime mold inspired approach for finding short and safe paths between sites. Experiments with slime molds (*Physarum polycephalum*), a plasmodium, have revealed the single cell organism is capable of finding shortest paths between food sources. A slime mold searches for nutrients by moving and spreading over the area. It uses a network of tubes for transportation of signals, nutrients, and metabolites. If a tube connects to food sources, it thickens to enable better transport while tubes in areas with low nutrient concentrations wither away.
Slime molds have been shown to be able to solve shortest path problems in mazes and the Steiner tree problem [10, 11].
Further experiments have revealed that they can be brought to reflect the structure of public transport network [15] and are even able to cope with disruptions and disasters [2].
The experiments with slime molds and their optimization capabilities motivated the development of mathematical models describing the behavior as a deterministic dynamical system [14]. The model is explained in more detail in section 2.
There are even earlier approaches to utilize principles from single cell organisms for solving optimization problems. [16] introduced an amoeba behavior searching model for solving the Euclidian travelling salesman problem. However, the algorithm differs strongly from the principles introduced in [10, 11].
Since the early studies, several applications and theoretical results have been obtained. For example, [4] presented a proof of Physarum solving shortest-path problems. While the first applications considered transportation networks, later research also focussed on wireless sensor networks, developing for example routing algorithms inspired by slime mold behavior [8, 9]. Other researchers addressed fault tolerant wireless sensor networks [3] or the problem of minimal exposure [13].
In [5] a very interesting approach in the area of transportation problems was presented. They proposed fault tolerant networks by changing to some extend the structure of the network. Nodes may change positions, i.e., intermediate nodes move towards large fluxes which is called migration. Additionally, if the flow required for transportation cannot be achieved by the paths currently used stimulation of additional pathways occurs.
In [7] a slime mold based approach was presented for linear programming. First, the authors proved the convergence of the algorithm to the optimal solution for continuous problems. Additionally, they introduced a discrete Physarum solver and discussed its application to linear assignment problems.

It should be noted that slime mold based algorithms are usually deterministic. This is in contrast to other natural computing methods which are usually stochastic. This raises the question whether a slime mold based approach may benefit from stochastic elements.

In [12], a method modelling the forward and backward flow of protoplasm was introduced and coupled with stochastic components. However, the experiments revealed no significant performance differences between the two classes.

This paper differs from the approaches presented in several ways. We propose to use an *evolving graph* structure instead of the usually static construction graph. Generally, aside from [5], the slime mold covers the complete network and only changes the topology by changing the tube conductance. Here, the slime mold grows from the sinks and connects to the nutrient sources by following the strongest concentrations. The approach proposed is in parts oriented after the findings in [1] for real slime molds. In [1] it was shown that slime molds follow the gradient of chemo-attractant of nutrients. Introducing a growth phase enables us to control the spread of the slime mold. If measures of uncertainty or risk are incorporated in the algorithm, the mold can be steered away from insecure areas into currently safe regions. A risk avoidance behavior has also been observed in real slime molds [6]. Usually, changes of the information will occur over time. Therefore, the slime mold has to be able to react. We propose to adapt the changing rule of the conductance by incorporating a measure for the state of the edge (representing for example a road) or for the uncertainty of the information. Degrading conditions will result in a shrinkage of the affected tubes thus changing the connections of the slime mold. Since this can result in lost pathways between sources and sinks, the slime mold has to grow again.

This paper introduces a new approach for slime mold inspired evolving networks according to the available information. It is structured as follows: First, the algorithm developed is described in detail. It consists of several phases. Each can be realized using different options. Afterwards, the algorithm is applied to representative tasks on grids and the findings are presented before coming to the conclusions and giving an outlook on future research.

2. THE SLIMO-ALGORITHM

In this section, we introduce the slime mold algorithm (SLIMO) for *network optimization under uncertainty*. The algorithm consists of four phases: Phase A: preprocessing, Phase B: slime mold evolution, Phase C: slime mold tube dynamics, and Phase D: Network Adaptation. First, the notation is explained.

2.1. Notation

In our study, we consider a connected planar grid $G = (V, E)$ that may for example represent an urban road network. The set of vertices is composed of three types of nodes: *sources* $q_1, \ldots, q_K$ (demand nodes), *sinks* $s_1, \ldots, s_L$ (supply nodes), and *additional nodes* $a_1, \ldots, a_N$ (crossroads), i.e., $V := \{q_1, \ldots, q_K,\, s_1, \ldots, s_L,\, a_1, \ldots, a_N\}$. Since this paper focusses on the determination of paths, fixed nodes and additional nodes are not considered further. They will come into play in future research when supply chains are considered. The network under interest is usually a part of a much larger grid that is contained in domain $\Omega \subset \mathbb{R}^2$ with $V \subset \Omega$ which defines our search space. In network applications, the sources can represent demand nodes with assigned demand $Q_1, \ldots, Q_K$. Similarly, the sinks stand for supply nodes with supply values $S_1, \ldots, S_L$. The set of neighbors of a node $v \in V$ is denoted by $\mathcal{N}_v = \{w \in V \mid (v, w) \in V\}$.
We are considering network and graph problems under uncertainty. The time-dependent *uncertainty layer* given by the uncertainty function (or indicator function) $\mu_t : \Omega \to [0, 1]$ associates nodes and networks branches with uncertainty values and influences the working of our proposed algorithm.
Our approach is applicable to any kind of curvilinear grids. However, for simplicity we are focussing on regular grids where $\mathrm{dist}(v, w) = \|v - w\|_1$ is the L_1-distance of nodes $v, w \in V$.
Examples for networks that can be addressed by our approach are among others humanitarian supply chains. Here, the sources stand for the demand nodes (refugee camps) and the sinks represent the supply nodes like ports and harbors. The additional nodes are the crossroads of the underlying transportation network. The uncertainty layer comprises all available (fuzzy) information the state of the road network and the availability of roads and bridges. Other potential applications are telecommunication networks, sensor networks, and navigation systems. All these networks types have to operated with uncertain and time-dependent information.

2.2. Phase A: Preprocessing

In a preliminary step of preprocessing we assign some additional information $\mathcal{I}$ to the nodes $v \in V$ of the network G. This heuristic information is useful for the solution of the problem as it leads to a faster and more precise growth of the slime mold with regard to the uncertainty layer μ and the sources and/or sinks.
The additional information depends on the type of problem under consideration and may be related to the position of sources and/or sinks as well as the supply/demand at these nodes.
We propose four different preprocessing strategies:

- Strategy A: Position of sources,
- Strategy B: Position and strength of sources,
- Strategy C: Position of sources and sinks,
- Strategy D: Position and strength of sources and sinks.

Strategies A and C are suitable for shortest path problems whereas strategies B and D are designed for use in supply chain models. The strategies depend on the distance ($\mathcal{D}$) and the strength ($\mathcal{Q}$) of the sources and sinks. For each source $i \in \{1, \ldots, K\}$, we define

$$\mathcal{D}_i^+(q_i) = 1, \quad \mathcal{Q}_i^+(q_i) = Q_i$$

and

$$\mathcal{D}_i^+(v) = \frac{1}{\text{dist}(q_i, v) + 1}, \mathcal{Q}_i^+(v) = \frac{Q_i}{\text{dist}(q_i, v) + 1}$$

for $v \in V \setminus \{q_i\}$.
Similarly, for each sink $i \in \{1, \ldots, L\}$, we set

$$\mathcal{D}_i^-(s_i) = -1, \quad \mathcal{Q}_i^-(s_i) = -Q_i$$

and

$$\mathcal{D}_i^-(v) = -\frac{1}{\text{dist}(s_i, v) + 1}, \mathcal{Q}_i^-(v) = -\frac{Q_i}{\text{dist}(s_i, v) + 1}$$

for $v \in V \setminus \{s_i\}$.
In addition, we have

$$\mathcal{D}^+ = \sum_{i=1}^{K} D_i^+(v), \quad D^- = \sum_{i=1}^{L} D_i^-(v),$$
$$\mathcal{Q}^+ = \sum_{i=1}^{K} Q_i^+(v), \quad Q^- = \sum_{i=1}^{L} Q_i^-(v).$$

The combination of these parameters with the information from the uncertainty layer leads to the following preprocessing strategies:

Strategy A - Position of sources
Strategy A reflects the position of the sources:

$$\mathcal{I}_A(v) = \mu(v) \cdot \mathcal{D}_i^+(v), \ v \in V.$$

Strategy B - Position and strength of sources
Strategy B depends on the position and strength of the sources:

$$\mathcal{I}_B(v) = \mu(v) \cdot \mathcal{Q}_i^+(v), \ v \in V.$$

Strategy C - Position of sources and sinks
Strategy C integrates the position of sources and sinks.

$$\mathcal{I}_C(v) = \mu(v) \cdot \Big\{\mathcal{D}^+(v) + \mathcal{D}^-(v)\Big\}, \ v \in V.$$

Strategy D - Position and strength of sources and sinks
Strategy D integrates the position and strength of sources and sinks.

$$\mathcal{I}_C^-(v) = \mu(v) \cdot \Big\{\mathcal{Q}^+(v) + \mathcal{Q}^-(v)\Big\}, \ v \in V.$$

Remark: We can replace $\frac{1}{dist(v,w)+1}$ by a function of the distance, i.e., $f\big(\frac{1}{dist(v,w)+1}\big)$. Typically, a sigmoidal function can be applied here.

2.3. Phase B: Slime Mold Evolution

In Phase B, the slime mold grows outward starting from the sinks. Several strategies are applied in order to connect new edges to the slime mold. The evolution process stops when all sources are part of the slime mold and, thus, connected to the sinks.
We represent the slime mold by the graph $MG^t = \big(V_{MG}^t, E_{MG}^t\big)$, $t \in \mathbb{N}_0$. The initial node set is given by the sinks, i.e., $V_{MG}^0 = \{s_1, \ldots, s_L\}$, and the initial set of edges is empty, i.e., $E_{MG}^0 = \emptyset$.
In Phase B, the information from the uncertainty layer takes influence on the growth of the slime mold network. This information is encoded in $\mathcal{I}$.
Different strategies can be applied, e.g., deterministic and stochastic approaches.
In each step of Phase B, a set of ramification nodes at the outer boundary of the slime mold is determined by

$$\overline{MG} = \{v \in V_{MG}^t \,|\, (\exists w \in N_v) : \ w \notin V_{MG}^t\}.$$

These nodes can be used for further growth. The ramification nodes are connected to a subset of their neighbors. Let $v \in V_{MG}^t$ be a ramification node. The set of feasible neighbors of v is given by

$$N_f(v) = \{w \in V \,|\, (v, w) \notin E_{MG}^t\}.$$

The set $N_f(v)$ comprises all the neighbors of v that are not connected by an edge to the slime mold (however, the node w can be an element of V_{MG}^t).
We use the information $\mathcal{I}$ to decide on the set of new neighbors, $\mathcal{N}^t(v)$, to be connected to the ramification node v in step t. We follow two strategies:

Strategy A (greedy)
A ramification node v is connected to the feasible neighbor $w \in \mathcal{N}_f(v)$ with the highest value $\mathcal{I}(w)$:

$$\mathcal{N}^t(v) = \{w \in \mathcal{N}_f(v) \,|\, \mathcal{I}(w) \geq \mathcal{I}(u) \ (\forall u \in \mathcal{N}_f(v))\}.$$

Strategy B (average value)
A ramification node is connected to all feasible neighbors that have a higher information value than the average of the information values of all feasible neighbors. We define a threshold

$$\vartheta = \frac{1}{|\mathcal{N}_f(v)|} \cdot \sum_{w \in \mathcal{N}_f(v)} \mathcal{I}(w).$$

Then,

$$\mathcal{N}^t(v) = \{w \in \mathcal{N}_f(v) \,|\, \mathcal{I}(w) > \vartheta\}.$$

We note, that also stochastic approaches can be applied with a random choice of ramification nodes and their connections. Phase B ends when all source nodes are part of the slime mold network and, thus, connected to the sinks. The stopping time $t_{max} \in \mathbb{N}$ is reached when $q_1, \ldots, q_K \in V_{MG}^{t_{max}}$.

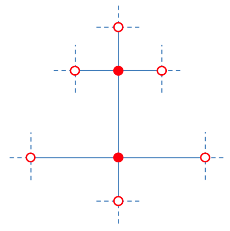

Figure 1. Slime mold evolution.

2.4. Phase C: Tube Dynamics

When the stopping time t_{max} is reached and all sources are connected to the sinks, the nutrient flow from sources to sinks is started leading to a dynamic evolution of the tubular network. Here, we state the standard Physarum model which describes the transport inside the tube networks. The slime mold tubular network $SM^t = (V^t_{SM}, E^t_{SM})$, $t \in \mathbb{N}_0$, consists of edges (tubes) and nodes (the connection points). The initial slime mold network is given by $SM^t = \left(V^{t_{max}}_{MG}, E^{t_{max}}_{MG}\right)$.
The model assumes that there are at least two special nodes, the food source and a sink acting as a receptor for the flow driven by different pressure values p_i at each node $i \in V^t_s$. For two nodes $i, j \in V^t_s$ connected by a bi-directional tube the flux or (Poiseuille) flow can be given by

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

with L_{ij} the length of the tube and D_{ij} the conductance (or "diameter") of the tube which is assumed to be symmetric (i.e., $D_{ij} = D_{ji}$). The conductance is connected to the tube thickness r_{ij} by $D_{ij} = \pi r^4_{ij}/(8\eta)$ with η the viscosity of the fluid and was shown to change with

$$\frac{\mathrm{d}}{\mathrm{d}t} D_{ij} = g(|Q_{ij}|) - d_{ij} D_{ij}, \tag{2}$$

where $d_{ij} = r$ is a constant decay rate. The function g is usually a monotonically increasing function and represents the change of the conductance with the flow rate. A common choice is a function of the form

$$g(x) = D_{\max} \frac{x^\alpha}{1 + x^\alpha} \tag{3}$$

which approaches the maximal conductance $D_{\max}$ for $x \to \infty$. In this paper, we modify this function and apply the sigmoidal function

$$g(Q_{ij}) = \mu_{ij} r D_{\max} \frac{a|Q_{ij}|^\alpha}{1 + a|Q_{ij}|^\alpha}, \tag{4}$$

where $\mu_{ij} = \min\{\mu(i), \mu(j)\}$ and a is a shape parameter. This function integrates the uncertainty layer μ. In particular, we obtain $\mu_{ij} = 0$ if there is no flow at one of the nodes, i.e., $\mu(i) = 0$ or $\mu(j) = 0$. The exponents α play an important role in the feedback system as they adjust the route selection process between efficient single paths ($\alpha > 1$) and robust multiple paths ($0 < \mu < 1$).

The flow at the nodes must follow Kirchhoff's laws. That is, there must be an equilibrium of in- and outflow of the form

$$\sum_{j \in \mathcal{N}_i} Q_{ij} = m_i, \tag{5}$$

where $m_i = 1$ at the sources, $m_i = -1$ at the sinks and $m_i = 0$ in the case of intermediate nodes.
Since the plasmodium connects at least a sink and a source, their inflows and outflows have to be balanced. At the sinks we specify an invariant pressure value of zero as baseline value. The initial pressure value of all other nodes is set to $p^{(0)}_i = 1$. Substitution of (5) in (4) leads to the pressure update formula

$$p^{(n+1)}_i = \frac{m_i + \sum_{j \in \mathcal{N}_i} \frac{D^{(n)}_{ij}}{L_{ij}} p^{(n)}_j}{\sum_{j \in \mathcal{N}_i} \frac{D^{(n)}_{ij}}{L_{ij}}}. \tag{6}$$

Next, D_{ij} is advanced in time for a discrete time step Δt. Here, we have to solve the differential equation (2). For a numerical solution, the following first order scheme was proposed in [14]:

$$D^{(n+1)}_{ij} = \frac{D^{(n)}_{ij} + g\big(|Q^{(n)}_i j|\big)}{1 + r\Delta t}. \tag{7}$$

The fluxes $Q^{(n)}_{ij}$ are determined by (1) using the pressure values $p^{(n+1)}_i$. If the $D^{(n)}_{ij}$ is below a certain threshold $\vartheta_{elim} > 0$, we can eliminate the corresponding branch from the slime mold network.

Algorithm 1: SLIMO

Phase A:	Preprocessing.
Phase B:	Slime mold evolution.
Phase C:	Initialization. Apply the pressure update formula (6). Determine fluxes $Q^{(n)}_{ij}$ by (1). Solve first order scheme (7) for the update of the conductance $D^{(n)}_{ij}$. Delete edges where $D^{(n)}_{ij}$ is below the threshold $\vartheta_{elim} > 0$.

2.5. Phase D: Network Adaptation

In this section, we investigate how the slime mold algorithm could be adapted to a dynamic situation where a new information layer μ^{new} is presented to the slime mold during the tube elimination Phase C.
The tube model discussed in section 2.4 (Phase C) dynamically updates the pressure values at the nodes as well as the size of the tubes. Edges are removed from the slime mold, if the corresponding conductance is below a specific threshold.

As the uncertainty function μ directly takes influence on the conductance D, the behaviour of the slime mold depends on its ability to adapt to a new situation μ^{new}.
Sudden changes of the information layer can affect the slime mold network to certain degrees. In particular, severe changes of the situation (e.g., parts of a road network are blocked) may lead to a new information layer μ^{new} that destroys the connectivity of the slime mold network. Sources and sinks are no longer connected by a path through the slime mold. In other cases, the connectivity can still be maintained, but the new information layer deviates significantly from the previous layer at least in some regions.

Connectivity Destroyed
If a new information layer destroys the connectivity of the slime mold graph in Phase C, the tube dynamics algorithm is stopped and a re-growth of the network is initialized in order to connect the sources to the network again. The re-wiring reflects this new information as new values of $\mathcal{I}$ are included. We propose the following scheme:

Algorithm 2: Connectivity Destroyed	
Event:	In Phase C, a new information layer μ^{new} is available at time $\widehat{t} \in \mathbb{N}$ such that at least one of the sources in the slime mold graph $SM^{\widehat{t}+1}$ is no longer connected to any of the sinks.
Initialize re-growth of the network	
Step 1:	Calculate $\mathcal{I}^{new}$ with regard to μ^{new} (Phase A).
Step 2:	Enter Phase B again with the new initial values $V^0_{MG} = V^{\widehat{t}}_{SM}$, $E^0_{MG} = E^{\widehat{t}}_{SM}$.
Step 3:	Start Phase C again.

Significant changes (connectivity maintained)
When severe changes of the information layer occur, an adaptation of the slime mold is necessary, even if the connectivity is not lost. In order to compare the new information layer μ^{new} to the previous layer μ^0, we introduce the measure

$$\Delta\mu = \int_{MG} |\mu^{new}(x) - \mu^0(x)|\, dx.$$

If this measure exceeds a specific threshold, a recalculation is necessary (the threshold depends on the application under consideration). However, a time-consuming full re-growth of the slime mold can be avoided if the re-growth of the slime mold is focussed on regions with significant changes (e.g., re-growth around blocked parts of the network). These can be identified with

$$R_{diff} = \{v \in V \mid \mu_{diff}(v) > \vartheta > 0\}$$

where

$$\mu_{diff}(v) = |\mu^{new}(v) - \mu^0(v)|,\ v \in V.$$

In regions with no significant changes, the network can be considered as quasi-optimal after the preceding steps of Phase C.

The following scheme can be applied in case of significant changes of the information layer:

Algorithm 3: Significant Changes	
Event:	In Phase C, a new information layer μ^{new} is available at time $\widehat{t} \in \mathbb{N}$. Sources and sinks of the slime mold graph $SM^{\widehat{t}+1}$ are still connected, but the measure $\Delta\mu$ exceeds a specific threshold.
Initialize re-growth of the network	
Step 1:	Calculate $\mathcal{I}^{new}$ with regard to μ_{diff} (Phase A).
Step 2:	Enter Phase B again with the new initial values $V^0_{MG} = V^{\widehat{t}}_{SM}$, $E^0_{MG} = E^{\widehat{t}}_{SM}$.
Step 3:	Start Phase C again.

In our studies, we investigated several strategies to deal with (sudden) changes of the information layer μ_t and its implications for slime mold growth and the tube dynamics. Due to space limitations, we restrict ourselves to an adaptive model dealing with connectivity issues. Further results with regard to local adaptation in case of connected networks are left to future studies.

3. Experiments on Grids

In this section, we present some numerical examples for the slime mold-based solution of shortest problems on a regular grid. We address the *single path solution* as well as the *multiple-path solution* for the undisturbed and the disturbed information layer, respectively. In addition, we provide an example of a complete rewiring after the new information layer has destroyed the network connectivity.

3.1. Single Path Problems

Firstly, we discuss two examples of a single path solution of the shortest path problem based on SLIMO. We consider a regular grid in the domain $\Omega = [1, 20]^2$ where the length of each branch is equal to one. Three sources are located at $q_1 = (3, 15)$, $q_2 = (10, 17)$, and $q_3 = (17, 15)$ and we have three sinks at $s_1 = (5, 5)$, $s_2 = (12, 3)$, and $s_3 = (18, 5)$. With the sources we assign the demand $Q_1 = 70$, $Q_2 = 120$, $Q_3 = 70$ and the supply at the sinks is given by $S_1 = 70$, $S_2 = 120$, $S_3 = 70$. In addition, we introduce the parameters $\Delta t = 0.1$, $r = 0.2$, $DMAX = 10$, $\alpha = 2$, and $\vartheta_{elim} = 0.1$. In order to solve the single path problem, we choose the parameter $\alpha = 2 > 1$ in the SLIMO-algorithm.

Single path problem - undisturbed

We consider an undisturbed problem with $\mu \equiv 1$. Figure 2 shows the values $\mathcal{I}$ obtained in the preprocessing part and Figure 3 illustrates the slime mold evolution obtained with a greedy strategy in phase B, that is based on $\mathcal{I}$. When all sources are connected to the slime mold, the slime mold evolution terminates (Figure 4) and the tube dynamics starts. The final result shows the slime mold based single path solution of the shortest path problem obtained with SLIMO (Figure 5) .

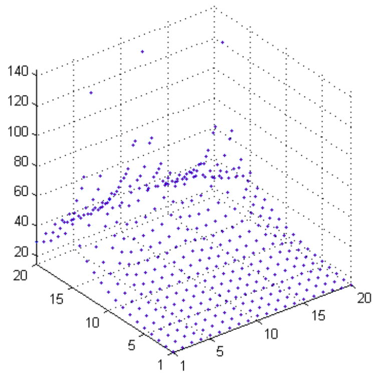

Figure 2. Phase A: Preprocessing (strategy A).

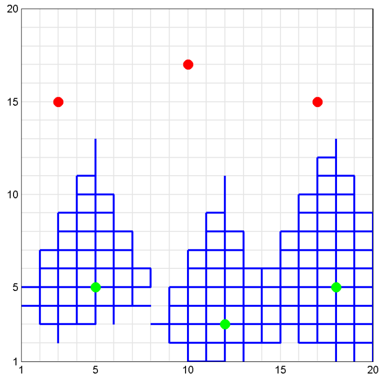

Figure 3. Phase B: Slime mold evolution (strategy A - greedy).

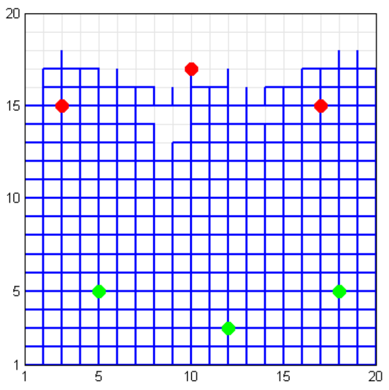

Figure 4. Phase B: Slime mold after phase B.

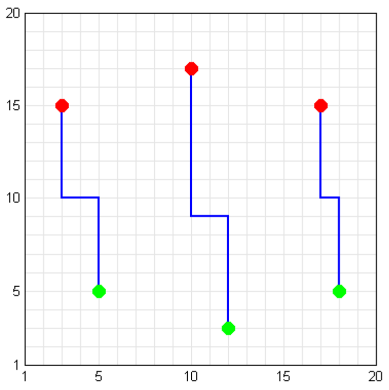

Figure 5. Phase B: Single path solution of the shortest path problem.

Single path problem - disturbed

In this example, we consider the same situation as before, but now the information layer μ has changed. For the patches $A_1 = [3,6] \times [8,12]$, $A_2 = [8,13] \times [7,12]$, and $A_3 = [16,18] \times [8,12]$, the information layer takes the values $\mu = 0$, i.e., several parts of the full domain Ω are blocked (Figure 6). The function $\mathcal{I}$ is equal to zero in these parts (Figure 7) and the slime evolution shows a different behaviour (Figure 8). Figure 9 displays the solution of the single path problem on the grid with respect to the new information layer μ.

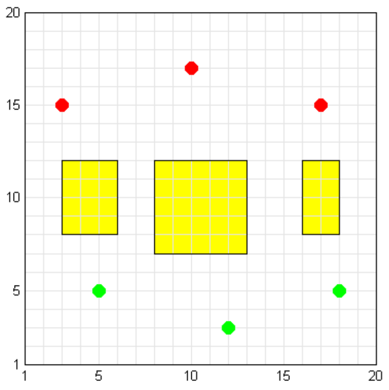

Figure 6. Perturbation of the information layer.

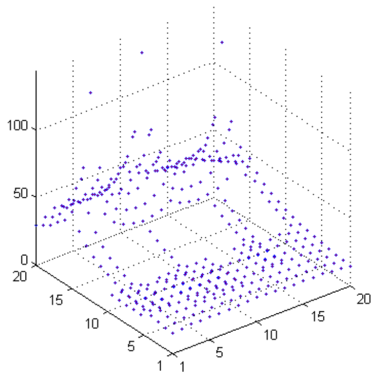

Figure 7. Phase A: Preprocessing (strategy A).

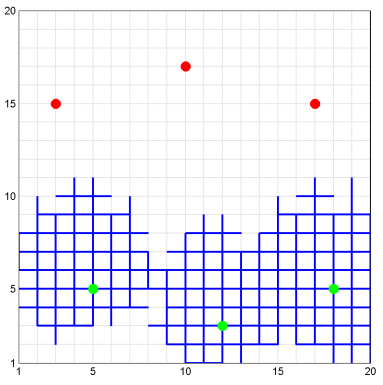

Figure 8. Phase B: Slime mold evolution (strategy A - greedy).

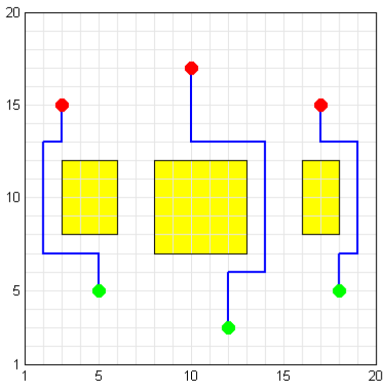

Figure 9. Phase B: Single path solution of the disturbed shortest path problem.

3.2. Multiple Path Problems

The SLIMO-algorithm can also provide multiple path solutions of the shortest path problem. We choose the parameter $\alpha = 0.9$ and consider again an undisturbed and a disturbed situation.

Multiple path problem - undisturbed
Figure 10 shows the multiple-path SLIMO-solution of the undisturbed problem with $\mu \equiv 1$. Multiple path solutions are considered as more robust against disturbances. In particular, disturbances usually affect the network in local regions and can be addressed with the algorithm 1 ("significant changes").

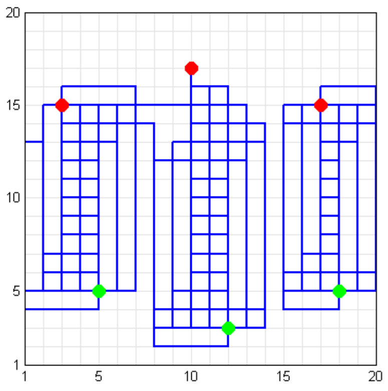

Figure 10. Phase B: Multiple path solution of the shortest path problem.

Multiple path problem - disturbed

Now, we disturb the indicator field and set $\mu = 0$ on the patch $A = [6, 14] \times [8, 12]$. Figure 11 shows the multiple path solution obtained with SLIMO, that avoids all blocked parts of the domain Ω.

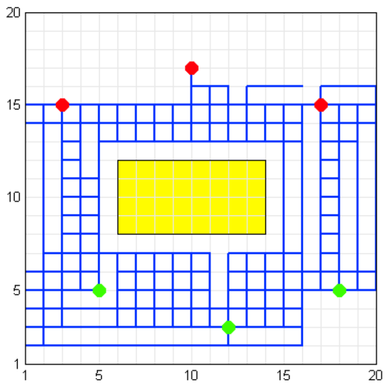

Figure 11. Phase C: Multiple path solution of the shortest path problem.

3.3. Network Adaptation

As an example of the network adaption approach of SLIMO we consider the case of slime mold growth where the connectivity of the underlying network is destroyed and a complete recalculation is required (algorithm 2: "connectivity destroyed").

Figure 12 shows the slime mold network for the information layer μ that has blocked the source node in the middle. SLIMO provides a single path solution of the shortest path problem (Figure 13).

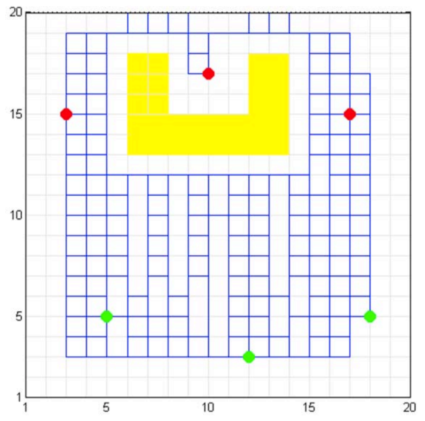

Figure 12. Phase B: Single path solution of the shortest path problem.

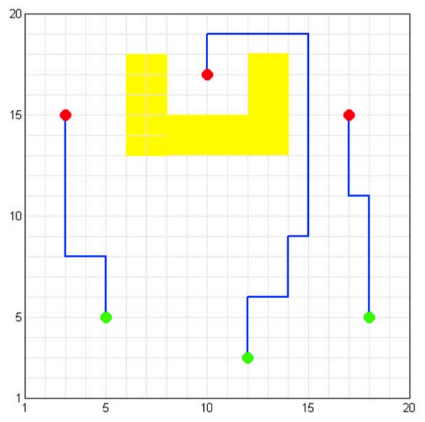

Figure 13. Phase B: Single path solution of the shortest path problem.

As the information layer changes and additional patches $A_1 = [3, 6] \times [8, 12]$, $A_2 = [8, 13] \times [7, 12]$, $A_3 = [16, 18] \times [8, 12]$ are removed, the situation changes completely as now the network connectivity is destroyed. A recalculation (algorithm 2: connectivity destroyed) leads to a rewiring shown in Figure 14 and finally to a new solution that is adapted to the new information layer μ^{new} (15).

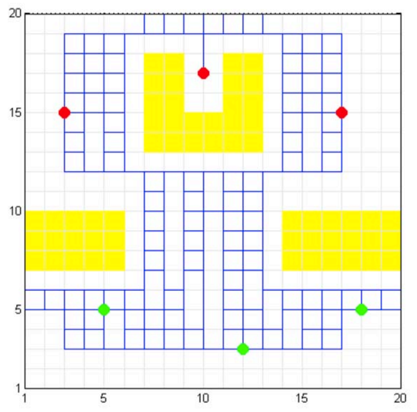

Figure 14. Slime mold evolution adapted to the new information layer μ^{new}.

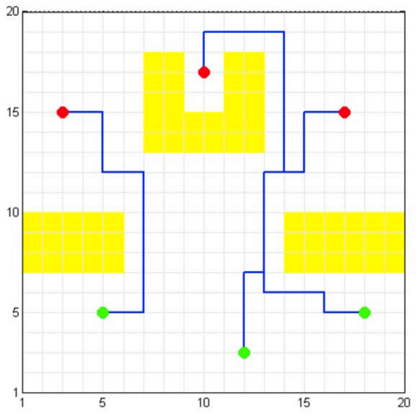

Figure 15. SLIMO multiple path solution for the new information layer μ^{new}.

4. Conclusions and Outlook

In this paper we presented a slime-mold based approach (SLIMO) for computing shortest paths under uncertainty. The capabilities of the SLIMO algorithm were investigated by considering several schematic representative situations that may occur during disaster and emergency relief operations but do also play a role for other dynamically changing and uncertain network problems. In contrast to most of the previously introduced methods it explicitly changes the slime mold's tube network by considering a growth phase. Furthermore, it is possible to integrate uncertainty or risk measures into the process. This gives the slime mold the ability to evolve its network according to situational changes.
The presented algorithm is a first step of a more comprehensive approach. It is the aim to extend the algorithm focussing on transportation problems and supply chains under uncertainty. The SLIMO algorithm is a completely deterministic algorithm. Usually, stochastic approaches are regarded as a useful means enabling the potential escape of local optima and increasing the exploration tendency of the algorithm. In future studies, we introduce stochastic elements into the SLIMO approach. In addition, a comparison with ant colony optimization would be interesting. This even more so, since some of the more powerful approaches always provide for alternative paths. This concept could be further developed and combined with the multiple path approach of SLIMO.

References

[1] Adamatzky, A. (2012). Slime mold solves maze in one pass, assisted by gradient of chemo-attractants. *IEEE Transactions on NanoBioscience 11*(2), pp. 131-134.

[2] Adamatzky, A. and J. Jones (2010). Road planning with slime mould: if physarum built motorways it would route m6/m74 through newcastle. *International Journal of Bifurcation and Chaos 20*(10), pp. 3065-3084.

[3] Becker, M. (2011). Design of fault tolerant networks with agent-based simulation of physarum polycephalum. In *2011 IEEE Congress on Evolutionary Computation (CEC)*, pp. 285-291.

[4] Bonifaci, V., K. Mehlhorn, and G. Varma (2012). Physarum can compute shortest paths. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pp. 233-240.

[5] Houbraken, M., S. Demeyer, D. Staessens, P. Audenaert, D. Colle, and M. Pickavet (2013). Fault tolerant network design inspired by physarum polycephalum. *Natural Computing 12*(2), pp. 277-289.

[6] Ito, K., D. Sumpter, and T. Nakagaki (2010). Risk management in spatio-temporally varying field by true slime mold. *Nonlinear Theory and Its Applications1*(1), pp. 26-36.

[7] Johannson, A. and J. Zou (2012). A slime mold solver for linear programming problems. In S. Cooper, A. Dawar, and B. Löwe (Eds.), *How the World Computes*, Volume 7318 of *Lecture Notes in Computer Science*, pp. 344-354. Springer Berlin, Heidelberg.

[8] Li, K., K. Thomas, C. Torres, L. Rossi, and C.-C. Shen (2009). Naturally adaptive protocol for wireless sensor networks based on slime mold. In *Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems, 2009. SASO '09*, pp. 280-281.

[9] Li, K., K. Thomas, C. Torres, L. Rossi, and C.-C. Shen (2010). Slime mold inspired path formation protocol for wireless sensor networks. In *Proceedings of the 7th International Conference on Swarm Intelligence*, ANTS'10, Berlin, Heidelberg, pp. 299-311. Springer-Verlag.

[10] Nakagaki, T., H. Yamada, M. Hara, et al. (2004). Smart network solutions in an amoeboid organism. *Biophysical chemistry 107*(1), pp. 1-6.

[11] Nakagaki, T., H. Yamada, and A. Toth (2000). Intelligence: Maze-solving by an amoeboid organism. *Nature 407*(6803), p. 470.

[12] Siriwardana, J. and S. Halgamuge (2012). Fast shortest path optimization inspired by shuttle streaming of physarum polycephalum. In *2012 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1-8.

[13] Song, Y., L. Liu, and H. Ma (2012). A physarum-inspired algorithm for minimal exposure problem in wireless sensor networks. In *2012 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 2151-2156.

[14] Tero, A., R. Kobayashi, and T. Nakagaki (2007). A mathematical model for adaptive transport network in path finding by true slime mold. *Journal of Theoretical Biology 244*(4), pp. 553-564.

[15] Watanabe, S., A. Tero, A. Takamatsu, and T. Nakagaki (2011). Traffic optimization in railroad networks using an algorithm mimicking an amoeba-like organism, physarum plasmodium. *Biosystems 105*(3), pp. 225-232.

[16] Yokoi, H., T. Mizuno, M. Takita, and Y. Kakazu (1995). Euclidean tsp using characteristics of slime mold. In *IEEE International Conference on Evolutionary Computation, 1995*, Volume 2, pp. 689-694.