Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 271280, 11 pages
http://dx.doi.org/10.1155/2014/271280

Research Article

A Bio-Inspired Method for the Constrained Shortest Path Problem

Hongping Wang,[1] Xi Lu,[1] Xiaoge Zhang,[1] Qing Wang,[1] and Yong Deng[1,2]

[1] *School of Computer and Information Science, Southwest University, Chongqing 400715, China*
[2] *School of Engineering, Vanderbilt University, Nashiville, TN 37235, USA*

Correspondence should be addressed to Yong Deng; ydeng@swu.edu.cn

Received 2 March 2014; Revised 13 April 2014; Accepted 23 April 2014; Published 14 May 2014

Academic Editor: Wen-Chiung Lee

The constrained shortest path (CSP) problem has been widely used in transportation optimization, crew scheduling, network routing and so on. It is an open issue since it is a NP-hard problem. In this paper, we propose an innovative method which is based on the internal mechanism of the adaptive amoeba algorithm. The proposed method is divided into two parts. In the first part, we employ the original amoeba algorithm to solve the shortest path problem in directed networks. In the second part, we combine the *Physarum* algorithm with a bio-inspired rule to deal with the CSP. Finally, by comparing the results with other method using an examples in DCLC problem, we demonstrate the accuracy of the proposed method.

1. Introduction

The shortest path problem (SPP) is one of the most classic problems, which is widely used in many fields, like network optimization [1, 2], road navigation [3, 4], and so on [5–7]. The constrained shortest path (CSP) problem aims at finding the shortest path from a predetermined source node to a predetermined destination node in a directed network under the constraint of being less than or equal to a given upper limit. Such constraints on the classical shortest path problem generally result in the problem becoming NP-hard. CSP is often considered as a variant of SPP and is frequently put into practice, like tail assignment problem in aircraft scheduling [8–10], the quality of service routing in communications networks [11, 12], and so forth [13].

Strategies of solving the CSP can be classified into three main categories: Lagrangian relaxation methods, dynamic programming (DP), and path ranking methods. The first method of CSP relies on the solution to a Lagrangian dual problem and uses different approaches to close the duality gap. Handler and Zang [14] proposed a method to solve the Lagrangian dual problem. They closed the duality gap by using the k shortest paths algorithm, terminating with the first path satisfying the constraint. They also proposed a method which searched in a direction. The method was based on combination of the original objective and the constraint. Carlyle et al. [15] presented a new algorithm for CSP. The improvement of this method is that the generated Lagrangian function was optimized and the optimality gap was closed by enumerating near-shortest paths. Recently, Zhang et al. [16] used Lagrangian relaxation combined with an adaptive amoeba algorithm to solve the CSP. In fact, if the constraint was relaxed, it made the CSP as the classical SPP, which could be solved repeatedly by plentiful computational efforts. What is more, the efficiency of this approach relied on the effectiveness of the underlying unconstrained shortest path algorithms.

A number of previous works were based on DP. All of these methods were based on label-setting or label-correcting algorithm. Starting from the the work of Joksch [17], the approaches of node labeling were deeply investigated. Dumitrescu and Boland [18] extended this idea and solved this problem by using preprocessing techniques. They [18] showed how the performance of the label-setting method could be further improved by making use of all Lagrange multiplier information collected in a Lagrangian relaxation's first step. Recently, Likhyani and Bedathur [19] developed SkIt index structure, which supported a wide range of label constraints on paths, and returned an estimation of the shortest path that satisfied the constraints. However, all of these methods mentioned above had a disadvantage: they might fail to well-estimate very large networks due to the "curse of dimensionality." As a result, it could not apply to real world with a large size of nodes.

A path ranking method was presented by Handler and Zang [14] by k shortest paths algorithm. Santos et al. [20] extended this idea by improving the way of sorting the k shortest paths. Its direction was based on the relative tightness of the constraint. Jia and Varaiya [21] presented two new methods based on the k-shortest-path approach. These heuristic methods, where one is centralized and the other is distributed, were both polynomial. In numerical experiments the proposed algorithms almost found the optimal paths. The main drawback of this method is that large value for k is required before a feasible solution of CSP is found.

Recently, a slime mold called *Physarum polycephalum* has attracted many investigators [22–25]. The body of the plasmodium of *Physarum polycephalum* contained a network of tubular elements by means of which nutrients and chemical signals circulated through the organism in an effective manner [26, 27]. Since the tubes disassembled and reassembled within a period of a few hours in response to external conditions, this organism was very useful for studying the function and dynamics of natural adaptive networks [28–30]. It was used to find the shortest path [31–33], solve the amaze [34, 35], optimize the networks [36, 37], and structure the road networks [38–40] and other fields [41–45]. Due to the uncertainty in real application [46–53], this algorithm is also applied to solve shortest path under uncertain environment [54, 55].

Inspired by the intelligence of the amoeba, we combine the *Physarum* algorithm with a bioinspired rule to deal with the CSP. The main idea behind the approach is to employ *Physarum* algorithm to find the shortest path. Then a check procedure is processed to ensure it satisfies the constraint. Once it does not meet the constraint, a penalty function is conducted. This procedure will go on executing until the path satisfying the constraint is found.

The paper is organized as follows. In Section 2, the basic theories, including CSP and the *Physarum* model, are introduced. In Section 3, the bioinspired method which is based on amoeba algorithm for CSP is proposed. In Section 4, an example is used to illustrate the proposed method. In comparison with other methods, the validity of the proposed method is demonstrated. In Section 5, the conclusion is given.

2. Basic Theories

In this section, we will introduce some basic theories, including the constrained shortest path problem and *Physarum polycephalum* model.

2.1. The Constrained Shortest Path Problem. Let $G = (V, E)$ be a directed network, where $V = 1, \ldots, n$ is the set of nodes and $E = (i, j) : i, j \in N, i \neq j$, is the set of directed edges. Each edge has two nonnegative weights c_{ij} and t_{ij}. c_{ij} and t_{ij} represent the generalized cost and the constrained variable, respectively. The CSP can be expressed by the following mathematical formulation:

$$\min \sum_{ij} c_{ij} x_{ij}, \tag{1}$$

subject to

$$\sum_{ij} x_{ij} - \sum_{j,i} x_{ji} = \begin{cases} 1, & \text{for } j = 1, \\ -1, & \text{for } j = 2, \\ 0, & \text{for otherwise}, \end{cases} \tag{2}$$

$$\sum_{ij} t_{ij} x_{ij} \leqslant T, \tag{3}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j, \tag{4}$$

where x_{ij} is binary variable, which is defined as follows:

$$x_{ij} = \begin{cases} 1, & \text{if } x_{ij} \text{ is in optimal path}, \\ 0, & \text{others}. \end{cases} \tag{5}$$

The parameter T indicates the maximum value allowed for the sum of t_{ij}. Equations (1), (2), and (4) define the shortest problem. The constraint (3) causes the CSP to be an NP-hard problem.

2.2. Physarum polycephalum Model. *Physarum polycephalum* is a single-celled amoeboid organism that can form a dynamic tubular network to link the discovered food sources during foraging. The physiological mechanism behind the phenomenon is that tubes thicken in a given direction when the flux through it persists in that direction for a certain time. By this mechanism, a mathematical model for path finding has been constructed to describe the adaptive dynamics of tubular network. In what follows, we will give a brief introduction to this path finding model [56].

Suppose the shape of the network formed by the *Physarum* is represented by a graph; the edge and the junction in the graph can be treated as a plasmodial tube and the node, respectively. Two special nodes labeled as N_1 and N_2 act as the source node and sink node, respectively. The other nodes are labeled as N_3, N_4, N_5, N_6, and so forth. The edge between nodes N_i and N_j is expressed as M_{ij}. The parameter Q_{ij} denotes the flux through tube M_{ij} from node N_i to N_j. Assuming the flow along the tube is an approximate Poiseuille flow, the flux Q_{ij} can be expressed as

$$Q_{ij} = \frac{D_{ij}}{L_{ij}} \left(p_i - p_j \right), \tag{6}$$

where p_i is the pressure at the node N_i, D_{ij} is the conductivity of the tube M_{ij}, and L_{ij} is its length.

By considering that the inflow and outflow must be balanced, we have

$$\sum Q_{ij} = 0 \quad (j \neq 1, 2). \tag{7}$$

As for the source node N_1 and the sink node N_2 the following two equations hold:

$$\sum_i Q_{i1} + I_0 = 0, \qquad \sum_i Q_{i2} - I_0 = 0, \tag{8}$$

where I_0 is the flux flowing from the source node, which is set as a constant in the model.

Then the network Poisson equation for the pressure can be derived from (6)–(8) as follows:

$$\sum_i \frac{D_{ij}}{L_{ij}} \left(p_i - p_j\right) = \begin{cases} -1, & \text{for } j = 1, \\ +1, & \text{for } j = 2, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

By setting $p_2 = 0$ as a basic pressure level, all p_i can be determined by solving (9) and Q_{ij} can also be obtained according to (6).

With the mechanism between the flux and tube thickness (described as the conductivity of the tube), the adaptation equation for conductivity is constructed as follows:

$$\frac{d}{dt} D_{ij} = f\left(\left|Q_{ij}\right|\right) - rD_{ij}, \tag{10}$$

where r is a decay rate of the tube. It can be obtained from the equation that the conductivity will finally vanish if there is no flux along the edge, while it is enhanced by the flux. The f is monotonically increasing continuous function satisfying $f(0) = 0$.

3. Proposed Method

The original amoeba algorithm can only find the shortest path in undirected networks. In order to solve the CSP in directed networks, we have to solve two problems. The first one is how to find the shortest path in directed networks. The second one is how to efficiently solve CSP by modified amoeba algorithm.

3.1. Modified Amoeba Model for the Shortest Path Problem in Directed Networks. Let $G = (N, E, W)$ be a directed network, where N represents the set of n nodes, E represents the set of directed edges, and W represents the weight set of E. We suppose that there is only one direction between node i and node j in the network G. Assume node s and node t are source node and sink node, respectively. The shortest path problem in the network can be defined as how to find a path from node s to node t, which only consists of directed edges of E, with the minimum sum of weights on the edges.

In the original amoeba model, every arc is bidirectional. It means that the flux can flow from node i to node j. They can also flow in the opposite direction. It is no different for original amoeba model, while the direction should be strictly distinguished in the directed networks. In particular, there is only one direction between two nodes in network G. It means that if the flux can flow from node i to node j, they cannot flow in the opposite direction. In order to take the factor of direction into account, each edge is regarded as two tubes with opposite direction and equal weight. Here, we assume the weight represents the length between two nodes represented by L. If there is a link between nodes i and j, then there are two tubes with equal length L_{ij} and L_{ji}, respectively, denoting two opposite directions from node i to node j and from node j to node i. The initial value of conductivity along with each tube is used to imply whether the tube is accessible or not. If arbitrary tube satisfies $L_{ij} \in E$, then the corresponding initial value of conductivity D_{ij} is assigned an arbitrary value among $[0.5, 1]$. Besides, D_{ji} is assigned as 0. The matrix D implies not only the conductivity but also the direction of each edge. By this way, the factor of direction is considered. In order to make original amoeba model that can adapt the directed networks, (9) is improved as follows:

$$\sum_i \left(\frac{D_{ij}}{L_{ij}} + \frac{D_{ji}}{L_{ji}}\right) \left(p_i - p_j\right) = \begin{cases} -1, & \text{for } j = s, \\ +1, & \text{for } j = t, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

With the purpose of keeping the validity of conductivity, the flux along with each tube needs to be adjusted. Equation (10) is modified as follows:

$$\frac{D_{ij}^{n+1} - D_{ij}^{n}}{\delta t} = \begin{cases} Q_{ij} - D_{ij}^{n+1}, & Q_{ij} > 0, \\ -D_{ij}^{n+1}, & \text{otherwise.} \end{cases} \tag{12}$$

Amoeba model has a positive feedback mechanism: the higher the pressure difference is, the more the flux will be. The more the flux is, the higher the pressure difference will be. When the iteration continues, the shortest path gradually appears through this mechanism. In directed networks, the positive feedback mechanism also exists. The method for finding the shortest path in directed network is described as follows.

(1) Initialize values of each parameter. According to the weight of the edge in G, the length of L_{ij} is initialized. According to the direction of each edge, the conductivity is initialized. Here, the initial value is assigned equally as 0.5. If there is an edge from node i to node j, $D_{ij} = 0.5$. Otherwise, $D_{ij} = 0$. The pressure of the node i is represented by p_i. The basic information of this directed network is initialized.

(2) According to current conductivity, the pressure of each node can be calculated using (11).

(3) According to (6), we can obtain the flux of each edge.

(4) According to (12), we can calculate the conductivity of the next moment.

(5) If the conductivity D_{ij} of each edge is no longer changed, the termination criterion is met. The directed shortest path consists of the edges with conductivity approximately equaling 1. Otherwise, go back to Step 2 and repeat until convergence.

3.2. Bioinspired Method. In this section, we will introduce some nomenclatures and phenomena about amoeba. Then we will introduce the proposed method.

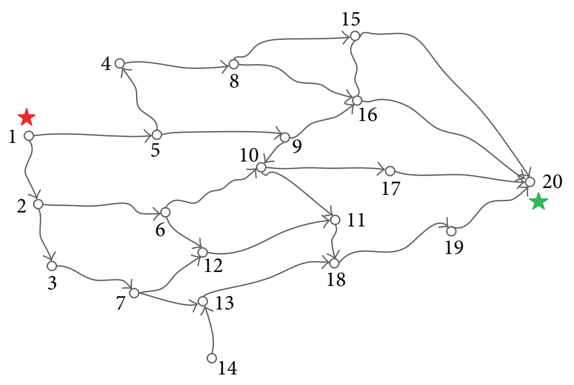

Figure 1: A directed transportation network with 20 nodes used in [16].

3.2.1. Nomenclatures and Phenomena

Segment: the tubular pseudopodia of amoeba linking node i to node j.

Growing segment: the segments with continuous thickening.

Potential segment: the growing segments with growing times exceeding given threshold.

Threshold: the critical value of judging whether growing segment is potential segment.

Important segment: the potential segments which can make up an unbroken path.

Important path: the path which is made up by potential segments.

Conductivity: indicating the thickness of the segment.

As the iteration continues, some segments become thicker while others degenerate. Simultaneously, the corresponding conductivity tends to increase or decrease. When the conductivity value in the current iteration is greater than the previous one, it is called thickening segment. When a segment is thickening, it means that the segment will continue to thicken. The segment becomes a growing segment. If the growing times of the growing segment exceed the threshold, it becomes a potential segment. If the potential segment belongs to the important path, it becomes an important segment. Through many trials, we find that the segment which belongs to the shortest paths continuously thickens from the beginning to the end. At the beginning, growing segments may be dispersed in the network. Then, potential segments and important segments occur. At the end, there are only important segments left, which make up a shortest path. In other words, we think some of those important segments are the final part of the shortest path and the important path is the shortest path on the great probability.

3.2.2. Proposed Method. As shown in Figure 1, an example employed in [16] is used to illustrate the general flow of the proposed method. Each arc has two attributes: the length L_{ij} and the cost C_{ij}. The specific values can be obtained according to Table 1. For instance, the numbers (120 and 60) along the arc between node 1 and node 2 mean the length along the arc is 120, and the cost along the arc is 60. We want to find a length shortest path from source node 1 to sink node 20 with the cost no more than 200.

Table 1: Attribute value of each edge employed in [16].

Edge	Length	Cost
$1 \rightarrow 2$	120	60
$1 \rightarrow 5$	90	50
$2 \rightarrow 3$	100	40
$2 \rightarrow 6$	90	60
$3 \rightarrow 7$	90	50
$4 \rightarrow 8$	70	40
$5 \rightarrow 4$	120	60
$5 \rightarrow 9$	80	50
$6 \rightarrow 10$	40	20
$6 \rightarrow 12$	60	40
$7 \rightarrow 12$	80	60
$7 \rightarrow 13$	60	50
$8 \rightarrow 15$	70	40
$8 \rightarrow 16$	100	40
$9 \rightarrow 10$	20	60
$9 \rightarrow 16$	90	40
$10 \rightarrow 11$	90	50
$10 \rightarrow 17$	70	40
$11 \rightarrow 18$	70	60
$12 \rightarrow 11$	50	50
$13 \rightarrow 18$	140	40
$14 \rightarrow 13$	100	60
$15 \rightarrow 20$	150	50
$15 \rightarrow 16$	40	40
$16 \rightarrow 20$	80	60
$17 \rightarrow 20$	60	50
$18 \rightarrow 19$	40	40
$19 \rightarrow 20$	50	40

Inspired by the important segment, we propose a bioinspired method based on the internal mechanism of the amoeba model. Steps are introduced in detail as follows.

(1) On the basis of Step 1 in Section 3.1, we need to do some extra operations as follows.

According to the information of the given CSP, we initialize the constraint θ. κ is a parameter which is used to judge whether the growing segment is the potential segment or not. γ is the divisor of the punishment. When the important path does not satisfy the constraint conditions, γ is used to decrease the conductivity of those important segments so that the amoeba can converge to the path satisfying the constraint. Besides, we set a variable *flag* representing the end mark and assign 0, which means the end mark does not satisfy the termination criterion.

(2) Step 2 to Step 5 are the same as in Section 3.1.

(3) According to the current and the previous conductivity, the growing segments can be found and marked. The growing times of each growing segment are also saved.

(4) According to the saved times and the threshold κ, we can judge whether the growing segment exceeds the threshold. If it exceeds it, it is the potential segment. Save these potential segments.

(5) Judge whether these potential segments can make up a complete path from the source node to the sink node. If they can, we mark these important segments and then go to Step 6. Otherwise, go to Step 7.

(6) According to these important segments, the important path can be obtained. Judge whether the important path satisfies the constraint θ. If it can, the original data such as D_{ij}, Q_{ij} is kept and makes the flag equal 1, which means the flag satisfies the termination criterion. In other words, the constrained shortest path is found. Then go to Step 7. Otherwise, do the following. The conductivities of each important segment (the segment who belongs to this important path) are decreased. The updated conductivity is equal to the original value divided by the divisor of the punishment γ. The marks including times and important segments are initialized.

(7) Check whether the termination criterion is met or not. The termination criterion is that the value of flag is 1. If it is satisfied, tubes with maximal conductivity make up the constrained shortest path. Otherwise, go to Step 2 and repeat until termination is satisfied.

More detailed procedures are showed in Algorithm 1. The basic idea behind this method is that we take full advantage of *Physarum* algorithm to find the path. After finding the path successfully, the check procedure is conducted to insure the path satisfies the constraint. If it does not meet the constraint, the penalty mechanism will be processed. By this way, the successive path will fade out. The program will continue to execute until the path satisfying the constraint is found. As can be seen in Algorithm 1, it is necessary for us to solve the linear equations shown in (9). The time complexity of solving the linear equations is $O(N^3)$, where the $|N|$ is the number of equations. In the original amoeba model, the number of equations and nodes in the network are equal. Hence, the time complexity of the bioinspired method is $O(N^3)$, where the $|N|$ is the number of nodes in the network. According to Figure 1, we let s be 1, t 20, and θ 200. Parameters κ and γ are experimental values. Here, we let them be 3 and 10, respectively. As shown in Figure 2, the solution 1-5-9-16-20 to this problem is obtained easily. The length of the path is 340 and the cost of the path is 200.

It can be noted that the solution obtained by the proposed method is the same as the result presented by Zhang et al. [16]. Both of them can find the optimal solution. Therefore, the accuracy of the presented method is demonstrated. Different from the method proposed by Zhang et al. [16], the bioinspired method combines the *Physarum* algorithm and the penalty mechanism by modifying the internal mechanism of the amoeba model while Zhang et al. [16] combine the *Physarum* algorithm with the *Lagrangian relaxation* method.

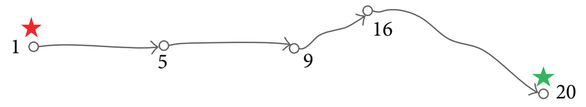

Figure 2: The solution.

Table 2: The attributes' value associated with each edge.

Link	Cost	Delay	Edge	Cost	Delay
1 → 2	15.4922	4.3346	12 → 15	18.0059	7.1883
1 → 3	10.3278	6.9174	19 → 15	18.4855	9.0751
1 → 4	6.9264	9.2813	9 → 16	6.0788	13.7941
1 → 5	14.1113	11.0636	10 → 16	4.7168	4.0935
2 → 6	20.4833	17.6400	10 → 17	8.9175	10.1357
2 → 7	21.4486	7.7884	11 → 17	10.7635	11.7094
4 → 7	12.5672	9.6045	15 → 18	10.0545	12.9549
3 → 8	18.7087	6.8806	13 → 19	7.1035	6.8859
5 → 8	12.0780	7.2797	15 → 19	9.5816	7.9918
6 → 9	16.4558	6.3616	16 → 20	14.1424	9.7289
6 → 10	10.9584	8.1689	17 → 20	11.3280	12.6545
7 → 10	12.5739	11.7066	14 → 21	15.2885	11.3513
4 → 11	8.9370	12.9112	17 → 21	15.9805	11.7076
5 → 11	19.0253	11.5669	18 → 21	11.5145	5.7829
7 → 11	10.8697	10.4742	18 → 22	10.0890	12.2744
5 → 12	19.1114	7.4606	19 → 22	14.6551	11.5192
8 → 13	11.2672	2.6939	18 → 23	2.9196	10.9407
11 → 14	16.8435	12.3477	20 → 23	10.8112	7.6235
12 → 14	8.3291	9.7940	21 → 23	13.0190	12.2323
22 → 23	12.5263	9.1583			

4. Examples

In this section, an example is introduced in order to prove the validity of the proposed method. The results are compared with the algorithm presented by Handler and Zang [14].

It is well known that the delay constrained least cost problem (DCLC) is a typical problem of CSP, which is widely used in network routing [57, 58]. The DCLC problem is to find the least cost path in a graph while keeping the path delay below a specified value. Let $G = (V, E)$ be a directed network, where $V = 1, \ldots, n$ is the set of nodes and $E = (i, j) : i, j \in n, i \neq j$, is the set of edges. The edge (i, j) represents the direction from node i to node j. Each edge has two nonnegative attributes: c_{ij} and d_{ij} represent cost and transmission delay from node i to node j, respectively.

Given a source node $s \in V$, a sink node $t \in V$. Δ_{delay} is the given delay constraint. Functions $D(p)$ and $C(p)$ represent delay and cost of the path p. The DCLC problem can be formulated as follows:

$$\min_{p \in P'(s,t)} C(p). \tag{13}$$

```
// $L$ is the distance matrix, $L_{ij}$ represents the edge length from node $i$ to node $j$.
// $C$ is the cost matrix, $C_{ij}$ represents the edge cost from node $i$ to node $j$.
// $s$ is the source node, $t$ is the sink node
// $M : M_{ij}$ represents the increasing times for the segment with flux continuing to grow in the network (growing segment).
// $U : U_{ij}$ represents the $M_{ij}$ whose value is beyond the threshold (potential segment).
// $\theta$ is the constraint, $\kappa$ is the threshold, $\gamma$ is the divisor of punishment
$D_{ij} \leftarrow [0.5, 1] \quad (\forall i, j = 1, 2, \ldots, N \wedge L_{ij} \neq 0)$
$Q_{ij} \leftarrow 0 \quad (\forall i, j = 1, 2, \ldots, N)$
$p_i \leftarrow 0 \quad (\forall i = 1, 2, \ldots, N)$
$flag = 0$
while $flag \neq 0$ do
    $p_t \leftarrow 0$ // The pressure at the ending node $t$
    Calculate the pressure of every node using (11)
        $\sum_{j\in V}\left(\frac{D_{ij}}{L_{ij}} + \frac{D_{ji}}{L_{ji}}\right)(p_i - p_j) = \begin{cases} +1 & \text{for } i = s \\ -1 & \text{for } i = t \\ 0 & \text{otherwise} \end{cases}$
    $Q_{ij} \leftarrow D_{ij} \times (p_i - p_j) / L_{ij}$ // Using (6)
    if $Q_{ij} < 0$ then
        $Q_{ij} = 0$; //Using (12)
        $D_{ij}^{n+1} = D_{ij}^{n}/2$;
    else
        $D_{ij}^{n+1} = (D_{ij}^{n} + Q_{ij})/2$;
    end if
    if $D_{ij}^{n+1} - D_{ij}^{n} > \varepsilon\,(\varepsilon \to 0^{+})$ then
        $M_{ij} = M_{ij} + 1$
    else
        $M_{ij} = 0$
    end if
    if $M_{ij} > \kappa$ then
        $U_{ij} = 1$
    else
        $U_{ij} = 0$
    end if
    if judgePath($U, s, t$) //To judge wether there is a path, the result is boolean then
        $path$ = findPath($U, s, t$) //Find the path and return. The nodes (important segment) making up the important path are saved.
        if calculateCost($path, C, s, t$) $> \theta$ //To judge wether the cost of the path is beyond the constraint then
            for $i$ = 1 to (length(paths) − 1) do
                $M(paths(i), paths(i + 1)) = 0$
                $M(paths(i + 1), paths(i)) = 0$
                $D(paths(i), paths(i + 1)) = \max(D(paths(i), :))/\gamma$
                $D(paths(i + 1), paths(i)) = D(paths(i), paths(i + 1))$
            end for
        else
            $flag = 1$
        end if
    end if
end while
```

Algorithm 1: A bioinspired method for the constrained shortest path problem.

Here, $P'(s,t)$ meets the following constraint:

$$\Delta_{\text{delay}} \geqslant D(p), \quad p \in P(s,t), \tag{14}$$

where $P(s,t)$ represents the set of all paths from node s to node t.

As shown in Figure 3, it is a real network topological structure with source node 1 and sink node 23. It was frequently an example in fuzzy shortest path problem [54, 59, 60]. The delay and cost of each link in the network are randomly generated. The costs of links are randomly generated by normal distribution with mean value equaling 10 and standard deviation equaling 3. The delays of links are randomly generated using normal distribution with mean value equaling 10 and standard deviation equaling 5. The cost and delay of each link randomly generated by normal distribution are shown in Table 2.

Table 3: Paths and their attributes.

Order	Handler and Zang's method			Proposed method		
	k shortest path	Delay	Cost	Important path	Delay	Cost
1	1-4-11-17-20-23	54.1798	48.7660	1-4-11-17-20-23	54.1798	48.7660
2	1-3-8-13-19-22-23	54.1798	48.7660	1-3-8-13-19-15-18-23	56.3484	78.8668
3	1-4-11-17-21-23	57.8418	55.6264	1-4-7-10-17-21-23	64.6681	69.9845
4	1-4-11-14-21-23	58.1237	61.0144	1-2-6-9-16-20-23	64.6681	69.9845
5	1-4-7-10-16-20-23	52.0383	61.7379	1-3-8-13-19-22-23	44.0553	74.5885
6	1-4-7-10-17-20-23	61.0060	63.1241			
7	1-4-7-11-17-20-23	61.3472	63.2659			
8	1-5-12-15-18-23	49.6080	64.2027			
9	1-5-11-17-20-23	54.6178	66.0392			
10	1-5-12-14-21-23	51.9017	69.8593			
11	1-5-8-13-19-22-23	48.6006	71.7413			
12	1-3-8-13-19-22-23	44.0553	74.5885			

Table 4: Setting parameter values.

BA model			Constraint	Cost		Delay		Proposed method	
N_0	E_0	m	p	Mean	std	Mean	std	κ	γ
3	3	3	0.1	10	3	10	5	2	30

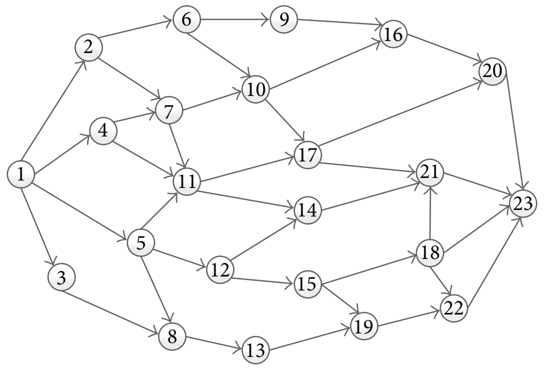

Figure 3: A real directed network with 23 nodes used in [59].

In order to get significative constraint, the delay constraint is got as follows [20]:

$$\Delta_{\text{delay}} = p * (p_{dc} - p_{ld}) + p_{ld}, \tag{15}$$

where p_{ld} is the least delay and p_{dc} is the delay of the path which has the least cost. They can be obtained by amoeba algorithm. The parameter p is a problem specific ratio that measures the "tightness" of constraint (3) such that $0 < p < 1$. The more constraining (or "tight") the value of T in constraint (3) is, the lower the value of p will be. For simplicity, here, we let $p = 0.1$.

Here, the least delay path is 1-3-8-13-19-22-23 and its delay is 44.0553. The least cost path is 1-4-11-17-20-23 and its delay is 54.1798 and its cost is 48.7660. The constraint 45.0680 can be obtained by using (15): $\Delta_{\text{delay}} = 0.1 * (54.1798 - 44.0553) + 44.0553 = 45.0680$.

Let θ be 45.0680, κ 2, and γ 30. Finally, we can get the solution: 1-3-8-13-19-22-23. The cost of the solution is 74.5885. The solution is exactly the same as the method [14] as well. Table 3 lists these k shortest paths [14] and important paths.

As can be seen in Table 3, our searching order is different from Hanlder and Zangs. The reason is that the order of important path depends on the specific values of the parameters κ and γ. Generally speaking, smaller κ and larger γ lead to a bigger step when it searches for the following path in our method. Otherwise, the step will be small. It will search the alternative paths one by one. The parameters setting is in association with the scale of network and the parameter p. In the case of a large scale of network, different parameters result in a big difference in executing time. The problem of how to set specific parameters will be further studied in the future work.

5. Computational Comparisons of the Solution Algorithms

In order to demonstrate the accuracy of proposed method, a number of experiments have been conducted on different networks. In addition, the CPU run-time has been compared between our method and the method proposed by Handler and Zang [14]. The main idea of the algorithm proposed by Handler and Zang is to use the k shortest paths algorithm to find the next shortest path terminating with the first path satisfying the constraint. This algorithm requires solving the k shortest path problem. The method finding the k shortest loopless paths in a network proposed by Yen [61] and the Dijkstra algorithm for finding the shortest path are used to solve the k shortest path problem.

Sixty-five networks were randomly generated. There are 5 different networks sizes based on the number of nodes in the networks. For each network in the same size, four networks are randomly generated. For each of the four random networks, three different attributes, but the same topologies networks, are generated. These attributes are randomly generated by normal distribution.

As for the random networks, we use the BA model [62] to generate a randomly directed network. In BA model, there are two assumptions: firstly, the network is formed by the continuous addition of new vertices to the system. Therefore, the number of vertices N increases throughout the lifetime of the network. Secondly, when there is a new node added in the network, it tends to be connected with the vertex who has more connections.

Under this assumption, we use the following steps to construct the network.

(i) Beginning with a small network G_0, it has N_0 nodes and E_0 edges. At each time step, we add only one new node.

(ii) Assume there are n nodes $(s_1, s_2, \ldots, s_n)$. When adding a new node s_{n+1}, it has $m\ (\leqslant N_0)$ edges linking m different already existing nodes in the network.

(iii) d_i indicates the degree of node s_i $(1 \leqslant i \leqslant n)$. The linking probability P_i between nodes s_i and s_{n+1} is defined as follows:

$$P_i = \frac{d_i}{\sum_{j=1}^{n} d_j}. \qquad (16)$$

Because the network is generated by this model, its degree distribution follows the power-law distribution with $\gamma_{\text{model}} = 2.9 \pm 0.1$. In this paper, we let N_0, E_0, and m be equal to 3.

All parameter values are listed in Table 4.

The tests are executed on Intel(R) Core(TM) i3-2120 processor (3.30 GHz) with 2 GB of RAM under Windows Seven. Each instance is run for 3 times, and we compute the average executing time and average accuracy. Results of 65 test problems are summarized in Table 5.

After experiment, the accuracy is 95.38% in the proposed method. In all the 65 networks, there are only 3 networks we cannot get optimal solution when $\kappa = 2$, $\gamma = 30$. However, we have verified that once these two parameters are adjusted, the optimal solution can be obtained. As can be seen in Table 5, the method proposed by Handler and Zang [14] is faster than the approach proposed by this paper. However, the method combining amoeba model and penalty mechanism has three potential advantages. First of all, the CSP problem is an NP-hard problem and it is difficult for us to solve this problem. The method proposed by this paper is to contribute an innovative idea and to enrich the solution strategies for solving this NP-hard problem. Secondly, the reason why the proposed method is relatively slow is that the amoeba model in this paper is implemented in a tradition way. In fact, there are already many investigations on parallel amoeba algorithm which have dramatically decreased the computational time. For example, worthwhile works [63, 64] have been achieved. Amoeba algorithm is constantly improving and it has a great potential for the development of the bioinspired artificial intelligence. It is certain that the computational time of the proposed method will be greatly reduced if there was a sophisticatedly parallel amoeba algorithm. At the current stage, we just aim at its methodology. Finally, the proposed method is easy to combine with other algorithms. By combining with other approaches, we can take the advantages of both of them. To sum up, although the executing time of the proposed method is relatively slow, it can be improved with the implementation of the parallel amoeba algorithm. What is more, the proposed method has many advantages over the existing algorithms.

Table 5: Computational results.

Nodes	Handler and Zang's method [14]	Proposed method
	Av. computer time (s)	Av. computer time (s)
100		
1	0.0147	0.236
2	0.0513	0.048
3	0.00559	0.0363
4	0.00575	0.0259
Av. time for 100 nodes	**0.0193**	**0.0866**
300		
1	0.0163	0.353
2	0.0189	1.009
3	0.0229	1.171
4	0.0148	0.231
Av. time for 300 nodes	**0.0182**	**0.691**
500		
1	0.038	1.278
2	0.037	1.14
3	0.039	1.298
4	0.039	1.346
Av. time for 500 nodes	**0.0383**	**1.266**
700		
1	0.733	3.989
2	0.0689	4.107
3	0.0791	5.563
4	0.069	3.775
Av. time for 700 nodes	**0.238**	**4.359**
1000		
1	0.121	17.34
2	0.126	24.531
3	0.143	20.135
4	0.135	42.846
Av. time for 1000 nodes	**0.13125**	**26.213**
Global Av.	0.0890	6.523

6. Conclusion

In this paper, we propose an innovative method to solve the CSP, which employs the internal mechanism of the amoeba model. The solution to the CSP consists of two parts. Firstly, we extend *Physarum* algorithm into the shortest path problem in directed networks. Secondly, we combine the *Physarum* algorithm with a bioinspired rule to deal with the CSP. In addition, we have shown the validity of the proposed method by comparing with other existing approaches using an example in the DCLC problem.

In the future, on the one hand, we will further investigate how to determine the associated parameters, such as κ and γ. Currently, these parameters are given according to the many experimental results. In the near future, we will try to provide theoretical analysis to these parameters and to provide an interval to these parameters. On the other hand, we will explore how to implement the proposed method in parallel environment, which will greatly reduce the executing time of the proposed method.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Authors' Contribution

Hongping Wang and Xiaoge Zhang contributed equally.

Acknowledgments

The work is partially supported by National Natural Science Foundation of China (Grant no. 61174022), Specialized Research Fund for the Doctoral Program of Higher Education (Grant no. 20131102130002), R&D Program of China (2012BAH07B01), National High Technology Research and Development Program of China (863 Program) (Grant no. 2013AA013801), the open funding project of State Key Laboratory of Virtual Reality Technology and Systems, Beihang University (Grant no. BUAA-VR-14KF-02), the Fundamental Research Funds for the Central Universities, Grant no. XDJK2013D010, and Southwest University Research Funds for Undergraduate Students in School of Computer and Information Science, Grant no. SWUJX2013-18.

References

[1] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[2] R. W. Floyd, "Algorithm 97: shortest path," *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962.

[3] K. Schüpbach and R. Zenklusen, "An adaptive routing approach for personal rapid transit," *Mathematical Methods of Operations Research*, vol. 77, no. 3, pp. 371–380, 2013.

[4] W. Dai, H. Hu, T. Wu, and Y. Dai, "Information spread of emergency events: path searching on social networks," *The Scientific World Journal*, vol. 2014, Article ID 179620, 7 pages, 2014.

[5] X. Deng, Q. Liu, Y. Hu, Y. Deng, and Topper:, "Topology prediction of transmembrane protein based on evidential reasoning," *The Scientific World Journal*, vol. 2013, Article ID 123731, 8 pages, 2013.

[6] X. Zhang, Q. Wang, A. Adamatzky, F. T. Chan, S. Mahadevan, and Y. Deng, "An improved physarum polycephalum algorithm for the shortest path problem," *The Scientific World Journal*, vol. 2014, Article ID 487069, 9 pages, 2014.

[7] E. Delgado-Eckert, "Boolean monomial control systems," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 15, no. 2, pp. 107–137, 2009.

[8] M. Grönkvist, "Accelerating column generation for aircraft scheduling using constraint propagation," *Computers and Operations Research*, vol. 33, no. 10, pp. 2918–2934, 2006.

[9] O. Weide, D. Ryan, and M. Ehrgott, "An iterative approach to robust and integrated aircraft routing and crew scheduling," *Computers and Operations Research*, vol. 37, no. 5, pp. 833–844, 2010.

[10] M. H. Almasi, S. Mirzapour Mounes, S. Koting, and M. R. Karim, "Analysis of feeder bus network design and scheduling problems," *The Scientific World Journal*, vol. 2014, Article ID 408473, 10 pages, 2014.

[11] G. Xue, "Minimum-cost QoS multicast and unicast routing in communication networks," *IEEE Transactions on Communications*, vol. 51, no. 5, pp. 817–824, 2003.

[12] B. Sivakumar, N. Bhalaji, and D. Sivakumar, "A survey on investigating the need for intelligent power-aware load balanced routing protocols for handling critical links in manets," *The Scientific World Journal*, vol. 2014, Article ID 138972, 12 pages, 2014.

[13] L. D. P. Pugliese and F. Guerriero, "A reference point approach for the resource constrained shortest path problems," *Transportation Science*, vol. 47, no. 2, pp. 247–265, 2013.

[14] G. Y. Handler and I. Zang, "A dual algorithm for the constrained shortest path problem," *Networks*, vol. 10, no. 4, pp. 293–309, 1980.

[15] W. M. Carlyle, J. O. Royset, and R. K. Wood, "Lagrangian relaxation and enumeration for solving constrained shortest-path problems," *Networks*, vol. 52, no. 4, pp. 256–270, 2008.

[16] X. Zhang, Y. Zhang, Y. Hu, Y. Deng, and S. Mahadevan, "An adaptive amoeba algorithm for constrained shortest paths," *Expert Systems with Applications*, vol. 40, no. 18, pp. 7607–7616, 2013.

[17] H. C. Joksch, "The shortest route problem with constraints," *Journal of Mathematical Analysis and Applications*, vol. 14, no. 2, pp. 191–197, 1966.

[18] I. Dumitrescu and N. Boland, "Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem," *Networks*, vol. 42, no. 3, pp. 135–153, 2003.

[19] A. Likhyani and S. Bedathur, "Label constrained shortest path estimation," in *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pp. 1177–1180, ACM, 2013.

[20] L. Santos, J. Coutinho-Rodrigues, and J. R. Current, "An improved solution algorithm for the constrained shortest path problem," *Transportation Research Part B: Methodological*, vol. 40, pp. 7607–7616, 2013.

[21] Z. Jia and P. Varaiya, "Heuristic methods for delay constrained least cost routing using k-shortest-paths," *IEEE Transactions on Automatic Control*, vol. 51, no. 4, pp. 707–712, 2006.

[22] A. Tero, S. Takagi, T. Saigusa et al., "Rules for biologically inspired adaptive network design," *Science*, vol. 327, no. 5964, pp. 439–442, 2010.

[23] A. Adamatzky, M. Lees, and P. Sloot, "Bio-development of motorway network in the netherlands: a slime mould approach," *Advances in Complex Systems*, vol. 16, Article ID 1250034, 28 pages, 2013.

[24] A. Adamatzky and M. Prokopenko, "Slime mould evaluation of australian motorways," *International Journal of Parallel, Emergent and Distributed Systems*, vol. 27, no. 4, pp. 275–295, 2012.

[25] G. Gruenert, J. Szymanski, J. Holley et al., "Multi-scale modelling of computers made from excitable chemical droplets," *International Journal of Unconventional Computing*, vol. 9, no. 3-4, pp. 203–236, 2013.

[26] T. Nakagaki, H. Yamada, and Á. Tóth, "Maze-solving by an amoeboid organism," *Nature*, vol. 407, no. 6803, p. 470, 2000.

[27] T. Nakagaki, H. Yamada, and M. Hara, "Smart network solutions in an amoeboid organism," *Biophysical Chemistry*, vol. 107, no. 1, pp. 1–5, 2004.

[28] X. Zhang, Z. Zhang, Y. Zhang, D. Wei, and Y. Deng, "Route selection for emergency logistics management: a bio-inspired algorithm," *Safety Science*, vol. 54, pp. 87–91, 2013.

[29] X. Zhang, S. Huang, Y. Hu, Y. Zhang, S. Mahadevan, and Y. Deng, "Solving 0-1 knapsack problems based on amoeboid organism algorithm," *Applied Mathematics and Computation*, vol. 219, no. 19, pp. 9959–9970, 2013.

[30] A. Adamatzky, "From reaction-diffusion to physarum computing," *Natural Computing*, vol. 8, no. 3, pp. 431–447, 2009.

[31] A. Adamatzky, "Physarum machines: encapsulating reaction-diffusion to compute spanning tree," *Naturwissenschaften*, vol. 94, no. 12, pp. 975–980, 2007.

[32] A. Tero, R. Kobayashi, and T. Nakagaki, "Physarum solver: a biologically inspired method of road-network navigation," *Physica A: Statistical Mechanics and Its Applications*, vol. 363, no. 1, pp. 115–119, 2006.

[33] X. Zhang, Q. Liu, Y. Hu et al., "An adaptive amoeba algorithmfor shortest path tree computation in dynamic graphs," http://arxiv.org/abs/1311.0460 .

[34] A. Adamatzky, "Slime mold solves maze in one pass, assisted by gradient of chemo-attractants," *IEEE Transactions on NanoBioscience*, vol. 11, no. 2, pp. 131–134, 2012.

[35] T. Nakagaki, "Smart behavior of true slime mold in a labyrinth," *Research in Microbiology*, vol. 152, no. 9, pp. 767–770, 2001.

[36] S. Watanabe, A. Tero, A. Takamatsu, and T. Nakagaki, "Traffic optimization in railroad networks using an algorithm mimicking an amoeba-like organism, Physarum plasmodium," *BioSystems*, vol. 105, no. 3, pp. 225–232, 2011.

[37] A. Tero, K. Yumiki, R. Kobayashi, T. Saigusa, and T. Nakagaki, "Flow-network adaptation in Physarum amoebae," *Theory in Biosciences*, vol. 127, no. 2, pp. 89–94, 2008.

[38] A. Adamatzky, G. J. Martínez, S. V. Chapa-Vergara, R. Asomoza-Palacio, and C. R. Stephens, "Approximating Mexican highways with slime mould," *Natural Computing*, vol. 10, no. 3, pp. 1195–1214, 2011.

[39] A. Adamatzky and R. Alonso-Sanz, "Rebuilding Iberian motorways with slime mould," *BioSystems*, vol. 105, no. 1, pp. 89–100, 2011.

[40] A. Adamatzky and P. P. B. de Oliveira, "Brazilian highways from slime mold's point of view," *Kybernetes*, vol. 40, no. 9, pp. 1373–1394, 2011.

[41] H. Zhang, Y. Deng, F. T. Chan, and X. Zhang, "A modified multi-criterion optimization genetic algorithm for order distribution in collaborative supply chain," *Applied Mathematical Modelling*, vol. 37, no. 14, pp. 7855–7864, 2013.

[42] E. R. Miranda, "Harnessing the intelligence of physarum polycephalum for unconventional computing-aided musical composition," *International Journal of Unconventional Computing*, vol. 10, no. 3, pp. 251–268, 2014.

[43] C. Gao, X. Lan, X. Zhang, and Y. Deng, "A bio-inspired methodology of identifying influential nodes in complex networks," *PLoS ONE*, vol. 8, no. 6, Article ID e66732, 2013.

[44] X. Zhang, Q. Wang, F. T. Chan, S. Mahadevan, and Y. Deng, "A physarum polycephalum optimization algorithm for the bi-objective shortest path problem," *International Journal of Unconventional Computing*, vol. 10, no. 1-2, pp. 143–162, 2014.

[45] C. Gao, C. Yan, Z. Zhang, Y. Hu, S. Mahadevan, and Y. Deng, "An amoeboid algorithm for solving linear transportation problem," *Physica A: Statistical Mechanics and Its Applications*, vol. 398, pp. 179–186, 2014.

[46] X. Zhang, Y. Deng, F. T. Chan et al., "A novel methodology for the job-shop scheduling problem basedon intuitionistic fuzzy sets," *International Journal of Production Research*, vol. 51, no. 17, pp. 5100–5119, 2013.

[47] G. Kumar and K. Kumar, "Design of an evolutionary approach for intrusion detection," *The Scientific World Journal*, vol. 2013, Article ID 962185, 14 pages, 2013.

[48] D. Wei, X. Deng, X. Zhang, Y. Deng, and S. Mahadevan, "Identifying influential nodes in weighted networks based on evidence theory," *Physica A: Statistical Mechanics and Its Applications*, vol. 392, no. 10, pp. 2564–2575, 2013.

[49] M. Whaiduzzaman, A. Gani, N. B. Anuar, M. Shiraz, M. N. Haque, and I. T. Haque, "Cloud service selection using multi-criteria decision analysis," *The Scientific World Journal*, vol. 2014, Article ID 459375, 10 pages, 2014.

[50] B. Kang, Y. Deng, R. Sadiq, and S. Mahadevan, "Evidential cognitive maps," *Knowledge-Based Systems*, vol. 35, pp. 77–86, 2012.

[51] N. Mohd Khairudin, A. Mustapha, and M. H. Ahmad, "Effect of temporal relationships in associative rule mining for web log data," *The Scientific World Journal*, vol. 2014, Article ID 813983, 10 pages, 2014.

[52] J. Liu, F. T. Chan, Y. Li, Y. Zhang, and Y. Deng, "A new optimal consensus method with minimum cost in fuzzy group decision," *Knowledge-Based Systems*, vol. 35, pp. 357–360, 2012.

[53] Y. Deng, Y. Chen, Y. Zhang, and S. Mahadevan, "Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment," *Applied Soft Computing Journal*, vol. 12, no. 3, pp. 1231–1237, 2012.

[54] Y. Zhang, Z. Zhang, Y. Deng, and S. Mahadevan, "A biologically inspired solution for fuzzy shortest path problems," *Applied Soft Computing*, vol. 13, no. 5, pp. 2356–2363, 2013.

[55] X. Zhang, Q. Wang, A. Adamatzky, F. T. Chan, S. Mahadevan, and Y. Deng, "A biologically inspired optimization algorithm for solving fuzzy shortest path problems with mixed fuzzy arc lengths," *Journal of Optimization Theory and Applications*, 2014.

[56] A. Tero, R. Kobayashi, and T. Nakagaki, "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, 2007.

[57] R. Sriram, G. Manimaran, and C. Siva Ram Murthy, "Preferred link based delay-constrained least-cost routing in wide area

networks," *Computer Communications*, vol. 21, no. 18, pp. 1655–1669, 1998.

[58] A. W. Mohemmed, N. C. Sahoo, and T. K. Geok, "Hybrid co-evolutionary particle swarm optimization and noising meta-heuristics for the delay constrained least cost path problem," *Journal of Heuristics*, vol. 16, no. 4, pp. 593–616, 2010.

[59] A. Tajdin, I. Mahdavi, N. Mahdavi-Amiri, and B. Sadeghpour-Gildeh, "Computing a fuzzy shortest path in a network with mixed fuzzy arc lengths using α-cuts," *Computers and Mathematics with Applications*, vol. 60, no. 4, pp. 989–1002, 2010.

[60] R. Hassanzadeh, I. Mahdavi, N. Mahdavi-Amiri, and A. Tajdin, "A genetic algorithm for solving fuzzy shortest path problems with mixed fuzzy arc lengths," *Mathematical and Computer Modelling*, vol. 57, no. 1, pp. 84–99, 2013.

[61] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[62] A. L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[63] A. Adamatzky and J. Jones, "Towards physarum robots: computing and manipulating on water surface," *Journal of Bionic Engineering*, vol. 5, no. 4, pp. 348–357, 2008.

[64] Y. Song, L. Liu, H. Ma, and A. V. Vasilakos, "Physarum optimization: a new heuristic algorithm to minimal exposure problem," in *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, pp. 419–422, ACM, 2012.