A Bio-Inspired Deployment Method for Data Collection Networks in Wide White Areas

Djibrilla Incha Adamou, Alexandre Mouradian and Véronique Vèque
Laboratoire des Signaux et Systèmes (L2S, UMR8506),
Université Paris Sud-CNRS-CentraleSupélec, Université Paris-Saclay
F-91192 Gif-sur-Yvette
Email: {djibrilla.adamou, alexandre.mouradian, veronique.veque}@u-psud.fr

***Abstract*—Wide white areas are defined as large regions with very little to no infrastructure. For example, deserts and large forest areas fall in this category. Many strategic phenomena and activities take place in these areas (e.g. mining, environmental monitoring) which necessitate data collection and analysis. In this context, we propose a network deployment scheme which aims at efficiently linking sparse points of interest in a very wide white area. The goal of the method is to minimize the cost of the deployment while providing a fault tolerant network. The proposed method is based on an algorithm which mimics the evolution of a type of mold called physarum. Our deployment problem is close to a Minimum Steiner Tree (MST) problem known to be NP-hard, we thus compare our results to a heuristic of MST.**

I. Introduction

The lack of telecommunication infrastructures in sparsely populated regions results in large unconnected zones called white areas. Whites areas are often wide regions with harsh climate, difficult access and little to no communication infrastructure. However, in these areas, strategic human activities are being carried out. For instance, we can cite sandstorm monitoring, bushfire alert, mining area monitoring, highway and pipeline monitoring. All these applications have in common the need for data collection and analysis.

In this work, we are interested in the deployment of data collection networks, such as Wireless Sensor Networks (WSN) [1] and Low Power WAN [2] in wide white areas. The goal is to link points or areas of interest to a data collection center. We aim at efficiently deploying sensor and relay nodes such that all areas of interest are covered, the network is connected and properly dimensioned to handle the data traffic.

As in [3], we consider a set of potential relay nodes distributed on the map of the considered region. The relays can be randomly placed on the map [4], [5] or form a grid [6]. Our goal is to chose a subset of relays to connect the areas of interest with the data collection center. These chosen relays are then to be deployed on the ground to form the network. We thus aim at minimizing the number of used relays and links. This problem is equivalent to finding the Minimum Steiner Tree (MST) covering all the points of interest and using a subset of the relays [3]. The MST problem is proven to be NP-hard [7], [8]. It is thus not applicable to scenarios with a large number of potential relay nodes. Moreover, it provides no fault tolerance due to its minimal property: any node or link failure results in a disconnected network. This aspect is problematic because WSNs and LPWANs are known to be error-prone [1]. Moreover, fault-tolerance is an essential characteristic in harsh environments such as wide white areas where network maintenance is either very difficult or impossible.

In this paper, we study the application of a heuristic to deploy a network in wide white areas by selecting nodes among a potential relay set. This heuristic mimics the behavior of physarum mold [9], [10]. As detailed in section III, this mold is able to construct a material efficient, yet resilient, network linking food sources.

The main contributions of this work are:

- the adaptation of the physarum algorithm [9] to data collection network deployment;
- the application of the algorithm to wide white area scenarios;
- the tuning of the algorithm to obtain a good trade off between deployment cost and fault-tolerance.

The remainder of the paper is organized as follows. Section II gives a survey of literature. Section III introduces the physarum algorithm, and presents how to adapt it to network deployment. In Section IV, we present the algorithm evaluation setup and results. Finally, conclusions and discussions are provided in Section V.

II. Related work

In this section we review data collection network deployment schemes of the literature. They fall into two main categories: random and deterministic deployment methods.

Many random deployment schemes have been proposed in the literature. In [4], a survey is provided. Most of the presented schemes aim at covering completely and reliably a given area [11]. Different node probability distributions are studied, they are non-homogeneous point processes in most cases [4], [12]. In [12] a random

deployment scheme is proposed for covering an area while taking into account the dynamic of the sensed physical parameter. Nevertheless, these works focus on the coverage of the full area instead of selected points of interest in a wide area. Moreover, whereas random deployment where nodes are scattered from the air can seem practical for very wide areas, deterministic methods such as grid based placement requires $O(log\ N)$ times fewer sensors than random deployments [13] where N is the number of sensors deployed in the random scheme.

Deterministic deployments have been increasingly applied in WSN applications [14]. Examples include Pipenet [15] or SimpliMote [16] for pipelines monitoring; A Line in the Sand [17]: a dense and distributed WSN for intrusion detection and targets tracking; ATMS [18]: an Authenticated Tracking and Monitoring System where a WSN combines both terrestrial and satellite links to secure uranium shipments from Australian uranium mines to Europe. In [19] and [20], studies of deployment patterns to achieve full coverage and k-connectivity (with $k \leq 6$) are presented. Nevertheless, in this case as well, these works focus on the coverage of the full area instead of selected points of interest in a wide area. More recent works such as [21] focus on deployments for a specific application but still aim at covering the whole area.

In [3], the authors treat the problem of optimal placement of relays on candidate locations in order to connect sensors to the sink in WSNs. They show that this problem is similar to the MST and thus NP-hard. They propose to use known heuristics to solve this MST problem. The problem treated in [3] is similar to our problem, but we use a different approach which is suited for wide white areas. We compare our results to a heuristic which solves MST.

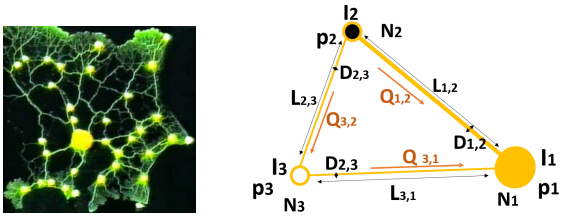
Most of the works presented in this section aim at deploying a network for covering a target area, which is often rather small. In our work, we consider the case where there are few points of interest sparsely distributed in a very large area. The number of potential relay is thus very high. We present a new approach of deployment based on the growth of the physarum mold which we describe in the next section.

III. Bio-inspired network deployment

In this section, we first present the background on physarum, then we present the physarum algorithm and its adaptation to network deployment.

A. Physiological concept of physarum growth

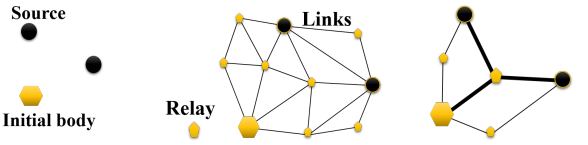
Physarum is a large, single celled amoeboid organism that forages to join irregularly distributed food sources [22] and transforms them into nutrients. Fig. 1a shows an example of physarum connecting 36 food sources.



(a) Network on 36 food sources [22]

(b) Nutrient dynamic

Fig. 1: Physarum network.



(a) Initial state

(b) Connected state

(c) Optimized state

Fig. 2: Physarum network.

To explain this natural network creation and optimization process, let us consider an initial physarum body placed in the space with 2 food sources (Fig. 2a). It growths toward all foods, once found it exploits them into nutrient which are conveyed through links. This leads to the creation of a fully connected network with lot of tube cross-connect points (pentagons in Fig. 2b). Finally, when there is no food anymore, it is able to optimize its body resources by deleting some unnecessary links and cross-connect points (Fig. 2c).

We are interested in physarum resource optimization mechanism, which we explain in the next subsection.

B. Physarum network optimization model

Mathematically, the network of tubes of the physarum is described as a graph [9]. The vertices represent tubes connexion points (food sources or simple joining points) while edges represent the tubes themselves which allow fluid propagation in the physarum body [22]. The physarum algorithm described in this section mimics the optimal resource exploitation part of the physarum behavior. The goal of the algorithm is thus to reproduce the capacity of physarum to prune edges and vertices of the network in order to keep only the most efficient (in terms of resources and resilience) connexion graph between food sources. The initial graph might thus contain many vertices which are not food sources. The algorithm convergence properties are investigated in [23]. It converges with an exponential rate. Moreover, it perfoms an output graph computation with a complexity in $O(N^3)$ [24], N being the size of the input graph.

As shown in Fig. 1b, to each node is associated an initial flow I_i and a pressure P_i ($i = 1, 2, 3$ in this example). The initial flow is the amount of nutrient

obtained by food transformation. For food sources we thus have $I_i \neq 0$ and for other connexion points $I_i = 0$. In the algorithm, food sources are actually alternatively considered as sources or sinks of a nutrient flow as will be clarified further. The pressure P_i is a variable for which we solve at each iteration of the algorithm. An edge connecting node N_i to N_j has a distance L_{ij} and a width D_{ij} which can change during the network evolution process. D_{ij} is initialized to D_0. Finally, Q_{ij} is the flow [9] traveling through the edge (i, j). It is defined as:

$$Q_{ij} = \frac{D_{ij}}{L_{ij}}(P_i - P_j). \tag{1}$$

The algorithm is iterative. At each iteration, two steps are performed: (1) solving a set of equations for the variables P_i, (2) update the edges parameters.

The first step consists of solving the following equations (equivalent to Kirchhoff's circuit law):

$$\begin{aligned} &\sum_i Q_{ij} = 0 \quad j \neq s, t \\ &\sum_i Q_{is} + I_0 = 0, \qquad \sum_i Q_{it} - I_0 = 0, \end{aligned} \tag{2}$$

with $j = s, t$ for source and sink nodes respectively, and I_0 the absolute value of the initial flow for source and sink nodes. Equations (2) form a linear system whose associated matrix is sparse. It can be solved by Incomplete Cholesky Conjugate Gradient (standard ICCG) method [25]. In the original algorithm [9], only one sink and one source are considered. In subsequent works [22], [10], more sources and sinks are considered by sequentially selecting every possible pairs of nodes with $I_0 \neq 0$. At each iteration a sink and a source are chosen among the food sources. The sink is chosen according to a uniform law. The source selection follows a specific probability mass function defined as:

$$p(S = j) = \frac{d_{ij}^{\gamma}}{\sum_{k \neq j} d_{kj}^{\gamma}}, \tag{3}$$

with d_{ij} the geographical distance between N_i the sink and N_j, and γ a parameter. After the resolution of (2), the Q_{ij} are recomputed with the obtained values of P_i according to Equation (1).

The second step is to update the edges to adapt them to the newly computed flow. Indeed, the link width D_{ij} changes over time according to a flow adaptation function [26]. The algorithm uses a discretized flow adaptation equation:

$$\frac{D_{ij}^{n+1} - D_{ij}^{n}}{\delta_t} = f(|Q_{ij}^{n}|) - D_{ij}^{n+1}, \tag{4}$$

where f is an increasing function with $f(0) = 0$, (D_{ij} tends to decline if there is no flow in the edge but is augmented when the flow increases), n is the iteration index, and δ_t is the time step. In [27], the following function f is proposed:

$$f(Q) = \frac{a|Q|^{\mu}}{1 + a|Q|^{\mu}}, \tag{5}$$

with μ and a positive parameters [28]. The algorithm stops when the difference $D_{ij}^{n+1} - D_{ij}^{n}$ is less than a predefined threshold. The output of the algorithm consists in the D_{ij} for each link. Then, to obtain the optimized network, it suffice to remove all links with $D_{ij} = 0$ (or close to 0) and remove all the disconnected nodes. Table I summarizes all the notations.

Names	Definitions
i	Node index
N	Number of deployed Potential Nodes
N_s	Source Node
N_t	Sink Node
P_i	Pressure in node i
I_i	Initial flow in node i
D_{ij}	Width of link between node i and j
Q_{ij}	Flow in link separating node i and j
L_{ij}	Distance of the link between node i and j
δ_t	Time step size

TABLE I: Model parameters

In the next subsection, we show how to adapt this algorithm to solve the data collection network deployment problem in wide white areas.

C. Deploying wireless networks using the physarum algorithm

Previous works [22], [29], [26] have considered the physarum algorithm to model transportation networks. To the best of our knowledge this is the first work which considers applying the physarum algorithm to wireless network deployment.

We assume that the edges and vertices of the physarum graph respectively model the wireless data links and the network nodes. The input graph consists in all the potential relay nodes, all the points of interest and the data collection center. To build this graph, we consider a map of the area with all the points of interest. Then we add the potential relay nodes. This can be done either deterministically using a grid pattern or randomly. In this work, we chose to place potential relays randomly (uniform distribution) as detailed in Section IV-A. The next step is to add links to this initial graph. In order to do that, we consider the radio range of the nodes. We link every pair of nodes (relay or point of interest) which are in communication range.

We then initialize the node and edge variables of the algorithm. We consider that D_{ij} is proportional to the rate capacity of a link (as it is the case for the physarum where the flow Q_{ij} is proportional to D_{ij}). D_{ij} is the output of the algorithm. The initial value of D_{ij}, D_0 has a very small impact on the output, actually, it is even chosen randomly in [9]. L_{ij} is the geographical distance

between the nodes N_i and N_j. In our case, each point of interest i is modeled as a food source with $I_i = I_0$, if it is set as a source, or $I_i = -I_0$, if it is set as a sink. The impact of the value of I_0 and μ (Equation (5)) on the characteristics of the deployed network is investigated in Section IV-C.

During step (1) of each iteration, we define the pmf of the random variable S' representing the sink to be chosen:

$$p(S' = k) = \begin{cases} \frac{1}{K+W} & \text{if } k \text{ is a point of interest,} \\ \frac{W}{K+W} & \text{if } k \text{ is a the destination,} \end{cases} \quad (6)$$

with K the number of points of interest and W the weight of the data center (destination). The source is then chosen according to Equation (3). The algorithm not only outputs the topology of the network to be deployed, but also the D_{ij} which gives an indication on how to dimension each link.

In the next section, we evaluate the algorithm and we show that it is able to compute efficient and fault-tolerant data collection network deployments in wide white areas with a huge number of potential relays.

IV. Algorithm Evaluation

A. Considered scenarios

In this section, we define the scenarios we consider to evaluate the network deployment physarum algorithm.

We use a 180x180 km^2 zone where 16 points of interest are placed arbitrarily (they are chosen randomly and kept for all the experimentations). The data collection center is positioned at the center of the area. The potential nodes are placed randomly (uniformly). We make their number N varying from 500 to 5000. We consider a radio range R_c varying from 5 to 30 km. This corresponds to the realistic ranges of LPWAN radios such as LoRa [30]. The larger the R_c is, the more the initial graph is connected. Fig. 3a shows an example of an initial graph with 1500 potential relay nodes and their communication range R_c equals to 15 km.

Initial values of D_{ij} are set to 1. We then make the values of the physarum algorithm parameters vary in order to tune them. We use the algorithm described in Section III, parameters and their ranges are summarized in Table II. The convergence criteria is set as the time where D_{ij} stops changing for all links: $D_{ij}^{n+1} - D_{ij}^{n} = 0$. D_{ij}^{+} and D_{ij}^{-} are respectively present and previous values of D_{ij} and n the iteration index. The scenarios are generated on Matlab 2016b.

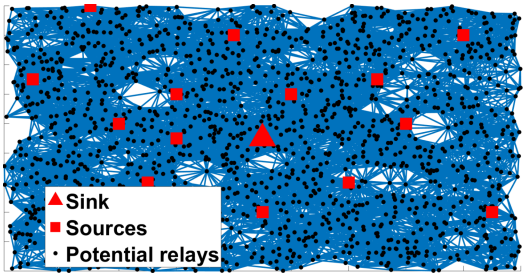
B. Performance metrics

The performance metrics we analyze on the output network deployment are the connectivity, the total length of the links, the number of relays, and the fault tolerance.

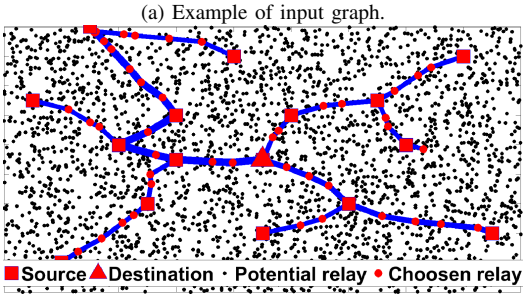
The connectivity is a boolean variable C equals to 1 if there exists at least one path between the data collection center and all the points of interest, otherwise it equals to 0. The network total length (TL) is measured in kilometers. TL is the sum of all the links length in the output network. The number of relay nodes (RN) is the number of nodes used in order to connect all the points of interest to the data collection center. Finally, the fault tolerance (FT_n) of the network is defined as the probability that the network remains connected even if an accidental failure occurs at n random links (or node) of the network. It is expressed as:

$$FT_n = \frac{\xi - \xi^*}{\xi}, \quad (7)$$

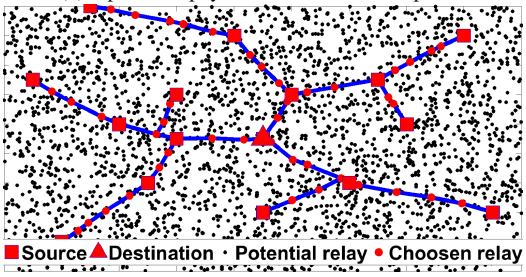
where $\xi = \binom{|E|}{n}$ is the total number of ways to choose



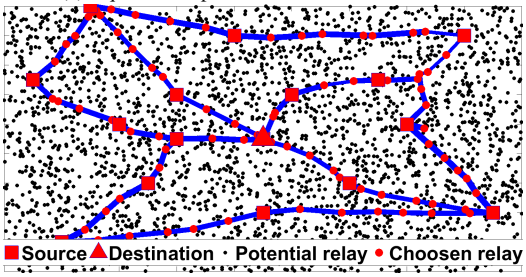
(a) Example of input graph.



(b) Optimized physarum result on 3000 points



(c) Equivalent optimal tree with MST-CHINS.



(d) Resilient network with phasarum method

Fig. 3: Structure depending of the parameters.

Parameter	Value
Evaluation area	180x180 km^2
N	500.i with $i \in 1, .., 10$
$N_{Interest}$	16
R_c	{5, 10, 15, 20, 25,30}
D_0	1
δ_t	{0.001,0.5}
I_0	[1,7]
μ	[1,9]
γ	{1,3,5,20}

TABLE II: Evaluation parameters

n links among the $|E|$ of the output network, and ξ^* is the number of configurations where we remove n links and the network ends up disconnected.

C. Reference algorithm for comparison

As mentioned in Section I, our deployment problem is very similar to a MST problem. MST is NP-hard, we thus compare our heuristic to another heuristic of MST called CHINS [8] for CHeapest INSertion. For comparison aim, we define two cost functions as TL/TL_{MST} and RN/RN_{MST} respectively for TL and RN cost index, TL_{MST} and RN_{MST} are the obtained MST total length and relay nodes respectively.

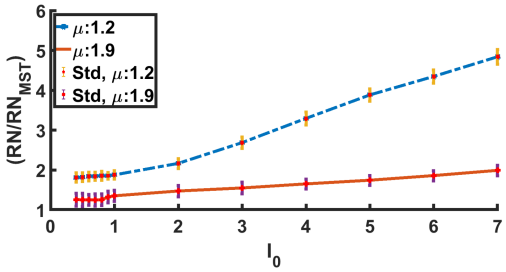
In the next subsection, we comment our results.

D. Results

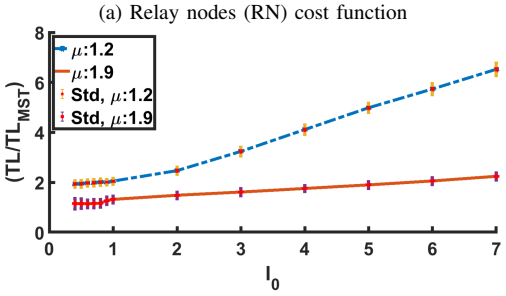
In this section, we investigate the impact of the values of the initial parameters on the results of the algorithm. We especially focus on the trade-off between fault-tolerance and deployment cost (total length and number of nodes).

Fig. 3b depicts an example of an output for the input graph of 3000 points. The following parameters $I_0 = 1$, $\delta_t = 0.5$, $\mu = 2$ are used. The width of the edges represents the final D_{ij}. The output network has 62 relay nodes (red points) and its total length is 611.26 km. It is close to the optimal MST (Fig. 3c) which contains 60 relays. For this example, the algorithm converges in less than 4 minutes on a modern computer (Intel i7 with 8G of RAM).

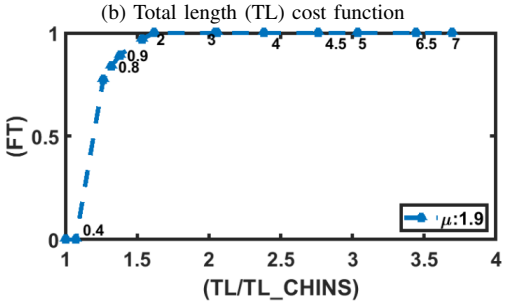
I_0 and μ have an important impact on the output graph. Some values (I_0, μ) = {(1.9, 0.7), (1.8, 0.6)} can even produce disconnected output networks. In our scenarios, we observe that for $\mu > 2$ and $I_0 \geq 1$ the algorithm always converges and the output is close to a steiner tree. Therefore, if the goal is to dimension a very optimized resource network such as in Fig. 3b, $I_0 = 1$ and $\mu = \{2, 3, 4, 5\}$ are appropriate values. I_0 is the parameter that has the most impact on the fault-tolerance of the network. For example, in Fig. 3d we obtain a resilient network using the following parameters $I_0 = 5$, $\mu = 1.8$. In the remainder of this section, we investigate the trade off between cost and fault tolerance for different values of I_0 and μ.



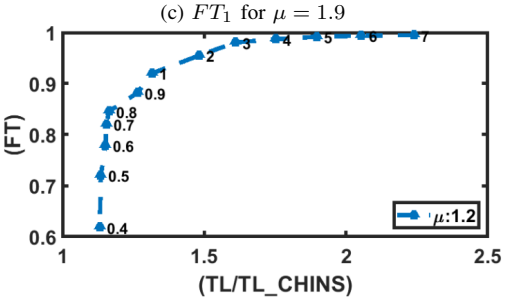
(a) Relay nodes (RN) cost function



(b) Total length (TL) cost function



(c) FT_1 for $\mu = 1.9$



(d) FT_1 for $\mu = 1.2$

Fig. 4: Performances analysis.

We compare the cost of the physarum algorithm with the CHINS-MST algorithm. As can be observed in Fig. 3c, the CHINS-MST does not provide the width of the edges. We consider the cost functions (RN/RN_{MST}) and (TL/TL_{MST}). In Fig. 4a, the cost function is plotted against I_0 for $\mu = 1.2, 1.9$. We observe that the cost increases with I_0. In fact, we observe in the obtained graph (for instance Fig. 3d), that the physarum algorithm creates more alternative routes when I_0 increases.

In Figs. 4c and 4d, we observe the trade off between cost and fault tolerance FT_1 for different μ and I_0 In these curves, each point of the plot represents the mean of 100 executions of the algorithm for a given value of I_0 which is written as a label next to the

point. 16 points of interest and $N = 3000$ relays with $R_c = 15$ km are considered (similar results are obtained when N and R_c vary). Only connected ($C = 1$) output graphs are kept. These graphs allow to tune μ and I_0 in order to obtain a target trade off between cost and fault tolerance. Networks with a higher fault tolerance are more expensive to deploy. The network designer can choose according to his budget and the reliability requirements of the application.

V. Conclusion

In this work we consider the problem of data collection network deployment in wide white areas. The goal is to deploy a network linking sparse points of interest which minimizes the amount of resources while providing a level of fault tolerance. The problem of resource minimization corresponds to the MST problem which is NP-hard. We propose to use a heuristic inspired from the behavior of the physarum mold. We present the model and detail how it can be applied to network deployment in wide white areas. We then study the performance of the algorithm on different scenarios in order to tune its parameters. Finally, we compare the performance of the physarum algorithm with an heuristic solving MST. We find that the physarum algorithm is able to find trade offs between cost and fault tolerance on very large considered areas while converging within minutes on a recent computer. Moreover, the physarum algorithms gives an indication on the capacity that should be allocated to each link by providing the width of the selected edges. In the future, we plan to use this indication in order to route the data flows in the network.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, 2002.

[2] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A study of lora: Long range & low power networks for the internet of things," *Sensors*, 2016.

[3] S. Misra, S. D. Hong, G. Xue, and J. Tang, "Constrained relay node placement in wireless sensor networks to meet connectivity and survivability requirements," *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, 2008.

[4] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Ad Hoc Networks*, 2008.

[5] LiDing and Zhi-HongGuan, "Modeling wireless sensor networks using random graph theory," *Physica A: Statistical Mechanics and its Applications*, 2008.

[6] W. Zhu, S. Xianhe, L. Cuicui, and C. Jianhui, "Relay node placement algorithm based on grid in wireless sensor network," *Third International Conference on Instrumentation, Measurement, Computer, Communication and Control (IMCCC)*, 2013.

[7] J. Suomela, "Computational complexity of relay placement in sensor networks," *International Conference on Current Trends in Theory and Practice of Computer Science*, 2006.

[8] S. VoB, "Steiner's problem in graphs: heuristic methods," *Discrete Applied Mathematics 40*, 1992.

[9] A. Tero, R. Kobayashia, and T. Nakagaki, "A mathematical model for adaptive transport network in path finding by true slime mold," *Journal of Theoretical Biology*, 2007.

[10] T. Nakagaki, H. Yamada, and M. Hara, "Smart network solutions in an amoeboid organism," *Biophysical Chemistry*, 2004.

[11] M. Ishizuka and M. Aida, "Performance study of node placement in sensor networks," *International Conference on Distributed Computing Systems Workshops*, 2004.

[12] T.-C. Hsu, Y.-W. P. Hong, and T.-Y. Wang, "Optimized random deployment of energy harvesting sensors for field reconstruction in analog and digital forwarding systems," *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, 2015.

[13] S. Kumar, T. H. Lai, and J. Balogh, "On k-coverage in a mostly sleeping sensor network," *MobiCom*, 2004.

[14] L. P. Damuut and D. Gu, "A survey of deterministic vs. non-deterministic node placement schemes in wsns," in *SENSORCOMM 2012 : The Sixth International Conference on Sensor Technologies and Applications*, 2012.

[15] I. Stoianov, L. Nachman, S. Madden, T. Tokmouline, and M. Csail, "Pipenet: A wireless sensor network for pipeline monitoring," *International Symposium on Information Processing in Sensor Networks*, 2007.

[16] S. Ali, A. Ashraf, S. B. Qaisar, M. K. Afridi, H. Saeed, S. Rashid, E. A. Felemban, and A. A. Sheikh, "Simplimote: A wireless sensor network monitoring platform for oil and gas pipelines," *IEEE Systems Journal*, 2016.

[17] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, and H. Zhang, "A line in the sand: a wireless sensor network for target detection, classification, and tracking," *Computer Networks*, 2004.

[18] J. Schoeneman and D. Sorokowski, "Authenticated tracking and monitoring system (atms) tracking shipments from an australian uranium mine," *International Carnahan Conference on Security Technology*, 1997.

[19] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. H. Lai, "Deploying wireless sensors to achieve both coverage and connectivity," *International symposium on Mobile ad hoc networking and computing*, 2006.

[20] Z. Yun, X. Bai, D. Xuan, T. H. Lai, and W. Jia, "Optimal deployment patterns for full coverage and k-connectivity ($k_¡$=6) wireless sensor networks," *IEEE/ACM Transactions on Networking, VOL. 18, NO. 3, JUNE 2010*, 2010.

[21] A. Boubrima, W. Bechkit, and H. Rivano, "Optimal wsn deployment models for air pollution monitoring," *IEEE Transactions on Wireless Communications*, 2017.

[22] A. Tero, S. Takagio, T. Saigusa, K. Ito, D. P.Bebber, M. D.Fricker, K. Yumiki, R. Kobayashi, and T. Nakagaki, "Rules for biologically inspired adaptive network design," *Science*, 2010.

[23] I. Kentaro, J. Anders, N. Toshiyuki, and T. Atsushi, "Convergence properties for the physarum solver," Tech. Rep., 2011.

[24] Y. D. Xiaoge Zhang, Cai Gao and Z. Zhang, *Advances in Physarum Machines: Sensing and Computing with Slime Mould*, G. C. Ivan Zelinka, Andrew Adamatzky, Ed. Springer Nature, 2016, no. 519-558.

[25] D. S. Kershaw, "The incomplete cholesky-conjugate gradient method for the iterative solution of systems of linear equations," *Journal of Computational Physics*, 1978.

[26] A. Tero, R. Kobayashia, and T. Nakagaki, "Physarum solver: A biologically inspired method of road-network navigation," *Physica A: Statistical Mechanics and its Applications*, 2006.

[27] A. Tero, K. Yumiki, R. Kobayashi, T. Saigusa, and T. Nakagaki, "Flow-network adaptation in physarum amoebae," *Theory in Biosciences*, 2008.

[28] C. Brummitt, I. Laureyns, T. Lin, D. Martin, D. Parry, D. Timmers, A. Volfson, T. Yang, and H. Yaple, "A mathematical study of physarum polycephalum," American Mathematical Society, Tech. Rep., 2010.

[29] X. Zhang, Z. Zhang, Y. Zhang, D. Wei, and Y. Deng, "Route selection for emergency logistics management: A bio-inspired algorithm," *Safety Science*, 2013.

[30] C. Goursaud and J.-M. Gorce, "Dedicated networks for iot : Phy / mac state of the art and challenges," *EAI Endorsed Transactions on the Internet of Things*, 2015.