A MULTIDIRECTIONAL MODIFIED PHYSARUM SOLVER FOR DISCRETE DECISION MAKING

Luca Masi, Massimiliano Vasile

Department of Mechanical & Aerospace Engineering, University of Strathclyde, 75 Montrose Street G1 1XJ, Glasgow, UK

Tel.: 0141 548 2083

Fax: 0141 552 5105

luca.masi@strath.ac.uk, massimiliano.vasile@strath.ac.uk

Abstract In this paper, a bio-inspired algorithm able to incrementally grow decision graphs in multiple directions is presented. The heuristic draws inspiration from the behaviour of the slime mould *Physarum Polycephalum.* In its main vegetative state, the *plasmodium*, this large single-celled amoeboid organism extends and optimizes a net of veins looking for food. The algorithm is here used to solve classical problems in operations research (symmetric Traveling Salesman and Vehicle Routing Problems). Simulations on selected test cases demonstrate that a multidirectional modified *Physarum* solver performs better than a unidirectional one. The ability to evaluate decisions from multiple directions enhances the performance of the solver in the construction and selection of optimal decision sequences.

Keywords: Discrete Optimization, *Physarum*, Multidirectional

1. Introduction

Over the past two decades, bio-inspired computation has become an appealing topic to solve NP-hard problems in combinatorial optimization using a reasonable computational time [3]. Ant Colony Systems, Genetic Algorithms and Particle Swarms are examples [6]. The method proposed in this paper takes inspiration from *Physarum Polycephalum*, an organism that was endowed by nature with heuristics that can be used to solve discrete decision making problems. It has been shown that *Physarum Polycephalum* is able to connect two food sources by solving a maze, using morphing properties [7]. In [9] and [1] it has been shown that a living *Physarum* is able to reproduce man-made transportation networks, i.e. Japan rail network and Mexican highway network. *Physarum* based algorithms have been developed recently to solve multi-source problems with a simple geometry [5, 11], mazes [8] and transport network problems [9, 8]. In this paper a multidirectional modified *Physarum Polycephalum* algorithm able to solve NP-hard discrete decision making problems is presented. In the mathematical model proposed in Sect. 2, discrete decision making problems are modeled with decision graphs where nodes represent the possible decisions while arcs represent the cost associated with decisions. Decision graphs are incrementally grown and explored in multiple directions using the *Physarum*-based heuristic. This paper aims at proving that a multidirectional incremental *Physarum* solver is more efficient, in terms of success rate (see Sect. 3.2), than a unidirectional incremental *Physarum* solver when applied to the solution of decision problems that can be represented with directed symmetric decision graphs, i.e. graphs where the contribution of an arc to a complete path can be evaluated moving forward or backward along the graph. This thesis will be demonstrated in Sect. 4 solving a series of test cases. Symmetric Traveling Salesman and Vehicle Routing Problems (known with the acronyms TSP and VRP), introduced in Sect. 3, were chosen as representative examples of the above type of decision making problems, here called reversible decision-making problems, i.e. problems in which a decision can be taken either moving forward or backward along the graph, as explained in Sect. 2.

2. Biology and mathematical modeling

Physarum Polycephalum is a large, single-celled amoeboid organism that exhibits intelligent plant-like and animal-like characteristics. Its main vegetative state, the *plasmodium*, is formed of a network of veins (*pseudopodia*). The stream in these tubes is both a carrier of chemical and physical signals, and a supply network of nutrients throughout the organism [11]. *Physarum* searches for food by extending this net of veins, whose flux is incremented or decremented depending on the food position with reference to its centre.

2.1 Mathematical modeling and multiple direction growing decision *Physarum* graphs

Reversible decision making problems. Assuming that a discrete decision problem is reversible, i.e. for each decision that induces a change from state a to state b it is possible to evaluate a symmetric inverted decision that brings back from b to a, it can be modeled with a symmetric directed graph. Here the interest is for general graphs in which the cost $C(a,b)$ associated with the decision a to b may not be identical to the cost *C(b,a)*, but the decision that brings from b to a exists and can be evaluated. The symmetric directed graph can be seen as the superposition of two directed graphs (direct-flow, *DF*, and back-flow, *BF*, graphs) whose nodes are coincident and edges have opposite orientation. In so doing, the decision between state a and b has a forward link a to b and a superposed backward link b to a. It is assumed that the first decision node is the heart of a growing *Physarum* in *DF*, and the end decision node the heart of a growing *Physarum* in *BF*. The two *Physarum* are supposed able to incrementally grow the decision graph in the two directions by extending their net of veins. The result is a graph where both nodes and links are incrementally built by two expanding *Physarum*.

Multidirectional incremental modified Physarum algorithmic. The flux through the net of *Physarum* veins can be modeled as a classical Hagen-Poiseuille flow in cylindrical ducts with diameter variable with time [9, 5, 11, 8]:

$$Q_{ij} = \frac{\pi r_{ij}^4}{8\mu}\frac{\Delta p_{ij}}{L_{ij}} \tag{1}$$

where Q_{ij} is the flux between i and j, μ is the dynamic viscosity, Δp_{ij} the pressure gradient, L_{ij} the length and r_{ij} the radius. Following [5], these two last main parameters are taken into account in the algorithm. Diameter variations allow a change in the flux. Veins' dilation due to an increasing number of nutrients flowing can be modeled using a monotonic function of the flux. In the present work, a function linear with respect to the product between the radius r_{ij} of the veins traversed by nutrients, and the inverse of the total length L_{tot} traveled, will be used for the veins' dilation:

$$\left.\frac{d}{dt}r_{ij}\right|_{dilation} = m\frac{r_{ij}}{L_{tot}} \tag{2}$$

where the coefficient m is here called linear dilation coefficient. Veins' contraction due to evaporative effect can be assumed to be linear with radius:

$$\left.\frac{d}{dt}r_{ij}\right|_{contraction} = -\rho r_{ij} \tag{3}$$

where $\rho \in [0, 1]$ is defined evaporation coefficient. The probability associated with each vein connecting i and j is then computed using a simple adjacency probability matrix based on fluxes:

$$P_{ij} = \begin{cases} \frac{Q_{ij}}{\sum_{j\in N_i} Q_{ij}} & if\ j \in N_i \\ 0 & if\ j \notin N_i \end{cases} \tag{4}$$

Algorithm 1 Multidirectional incremental modified Physarum solver

```
initialize $m$, $\rho$, $GF_{ini}$, $N_{agents}$, $p_{ram}$, $\alpha$, $r_{ini}$
generate a random route from start to destination in $DF$ and $BF$
for each generation do
  for each virtual agent in all directions ($DF$ and $BF$) do
    if current node $\neq$ end node then
      if $rand \leq p_{ram}$ then
        using Eq. (6) create a new path, building missing links and
        nodes
      else
        move on existing graph using Eq. (4).
      end if
    end if
  end for
  look for possible matchings
  contract and dilate veins using Eqs. (2), (3) for each agent, (5)
  update fluxes and probabilities using Eqs. (1), (4)
end for
```

where N_i is the set of neighbour for i.

A further social term in the dilation process was added in the algorithm and takes inspiration from the behavior of the *Dictyostelium Discoideum*'s pacemaker amoeba [6]:

$$\left.\frac{d}{dt}r_{ij_{best}}\right|_{elasticity} = GF r_{ij_{best}} \tag{5}$$

where GF is the growth factor of the best chain of veins and $r_{ij_{best}}$ the veins' radii. This is an additive term in the veins' dilation process, whose first main term is expressed in Eq. (2). The incremental growth of decision network in multiple directions is then based on a weighted roulette. Nutrients inside veins are interpreted as virtual agents that move in accord with adjacency probability matrix in Eq. (4). Once a node is selected, there is an *a priori* probability p_{ram} of ramification towards new nodes that are not yet connected with the actual node. The probability of a new link construction from the current node c to a new possible node $n_i \in N$, where N is the set of new possible decisions, is here assumed to be inversely proportional to the cost L_{cn_i} of the decision between c and n_i, i.e. the length:

$$p_{cn_i} \propto \frac{1}{L_{cn_i}^{\alpha}} \tag{6}$$

where α is a weight. Once a new link is built, a complete decision path is constructed (creating other links if necessary). Assuming then one *DF* and one superposed *BF Physarum*, a matching condition can be defined. If an arc connecting two nodes that belong to *DF* and *BF Physarum* respectively, exists or can be created, it is traversed by the agent and becomes part of both the *DF* and the *BF*. Several types of matching strategies are possible: *all-matching*, in which all joint paths are saved and evaluated, *random-matching*, in which some joint paths are selected among all the possible joint paths with a probabilistic rule and *selective-matching*, in which some joint paths are selected among all the possible joint paths according to an elitist criterion. At each generation, the elitist criterion selects a joint path if and only if it is better in terms of total cost than the previous joint paths selected during the same generation. The pseudocode of the multidirectional incremental modified *Physarum* solver is provided in Algorithm 1, where r_{ini} is the initial radius of the veins, always sets equals to 1 in this paper, and N_{agents} the number of virtual agents. A unidirectional algorithm is a special case of multidirectional algorithm, obtained by freezing the *BF*: flux and graph growth are allowed in only one direction.

Simulations on selected test cases (see Sect. 4) were carried out adding in the *Physarum* software a routine for the adaptive control of the growth factor GF. This control was introduced in order to incrementally boost the effect of GF during a simulation, driving exploring agents towards best veins. Simulations showed that the adaptive control of GF helps the convergence of the algorithm towards optimal solution. Given an initial value for the growth factor GF_{ini}, GF is incremented by a fixed percentage $\sigma = 0.01$ after every generation. If the probability p_{best} associated with the best path so far (see adjacency probability matrix, Eq. (4)) is higher than a fixed value $p_{lim}^{low} = 10^{-4}$, the increment is set to zero. Then, if p_{best} exceeds a value $p_{lim}^{high} = 0.85$, GF is set equal to GF_{ini} and veins are dilated and contracted to their initial value.

3. Application to Traveling Salesman and Vehicle Routing Problems and benchmark

Traveling Salesman Problem (TSP) and Vehicle Routing Problem (VRP) are classical problems in combinatorial optimization. When the modified *Physarum* algorithm is applied to VRP, a probability skew factor $\psi = 0.5$ is included in the algorithm. If an agent is not obliged to go to depot, the probability to reach the depot is lowered of a factor $(1 - \psi)$. The skew factor is introduced in the model to avoid frequent returns to depot.

Algorithm 2 Testing procedure

```
set to $N$ the max number of function evaluations for the algorithm $P$
apply $P$ to $p$ for $n$ times and set $j_s = 0$
for all $i \in [1, ..., n]$ do
  if $f_p^{best} + tol \leq f_i^{best} \leq f_p^{best} - tol$ then
    $j_s = j_s + 1$
  end if
end for
evaluate $p_s = j_s/n$
```

3.1 Benchmark

TSPLIB [12] was used to benchmark the proposed *Physarum* algorithm, developed in Matlab® R2010b, on the TSP problem. In Sect. 4 are reported the results obtained by applying the multidirectional solver to test case *Ulysses16*, normalised with a factor equals to 10000, and to test case *Eil51*. The reference optimal solutions for this particular TSP instance can be found in the TSPLIB. The *Physarum* solver applied to VRP was tested on a map of 9 cities plus one depot. The map is built using 9 Italian cities (Firenze, Livorno, Montecatini, Pistoia, Prato, Montevarchi, Arezzo, Siena, San Gimignano), with a city considered the depot (Ponsacco). The Euclidean distance in kilometers was used. Optimal tour with constant demand for each city equals to 1 and one vehicle with capacity equals to 4, was found using an exhaustive search.

3.2 Testing procedure

The testing procedure proposed in [13] was used in this paper and is reported in Algorithm 2. The success rate $p_s = j_s/n$, expressing the percentage of success after N function evaluations of algorithm P over a group of n simulations, will be used for the comparative assessment of the algorithm's performance. A function evaluation is defined as the call to the objective function, i.e. each arc selected by the virtual exploring agents (see Sect. 2.1) is considered a function evaluation. For example, each exploring agents during a generation would call the objective function a number of time proportional to the dimension of the problem, i.e. 17 times for *Ulisses16* test case and 52 times for *Eil51* test case (see Sect. 3.1). The number of calls for an agent during a generation have to be then multiplied by the total number of exploring agents and doubled if agents move in two directions, in order to have an estimate of total calls during a complete generation. In [13] one can find a full explanation on the setting of the value of n in order to have a reliable estimate of algorithm's success probability. Using $n = 175$ one would obtain an error ≤ 0.05 with a 95% of confidence. A value equals to 200 for n is used in the simulations presented in this paper. Tolerance is set to 10^{-8}.

4. Results

The multidirectional modified *Physarum* solver, named D&B in the following, was compared against a unidirectional modified *Physarum* solver, named D. Simulations were carried out on a 64-bit OS Windows 7 Intel® Core$^{\text{TM}}$2 Duo CPU E8500 3.16GHz 3.17GHz. D&B was tested without and with matching ability (see Sect. 2.1), named D&B with xM. The prefix x indicates the type of matching selected. nM means *no-matching*, rM *random-matching*, aM *all-matching* and sM indicates *selective-matching.*

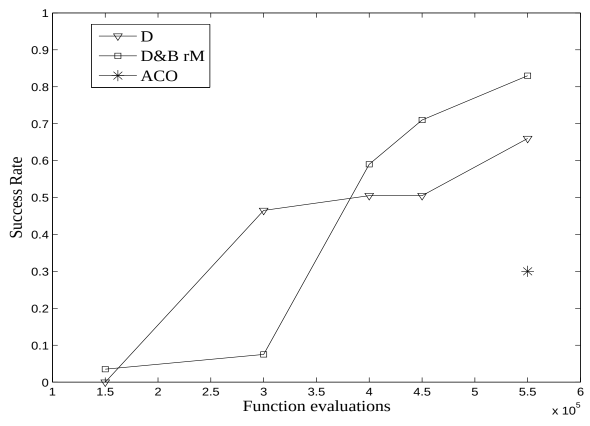

Figure 1. Variation of the success rate with the number of function evaluations - TSP test case *Ulysses16*

The multidirectional and unidirectional modified *Physarum* algorithms were applied to the symmetric TSP test case *Ulysses16*, described in Sect. 3.1. The values used as input parameters in the simulations are $m = 5\times10^{-5}$, $\rho = 1\times10^{-5}$, $GF_{ini} = 5\times10^{-3}$, $N_{agents} = 100$, $p_{ram} = 0.8$, $\alpha = 0$, $r_{ini} = 1$, and were chosen after a series of trials. Results, as shown in Fig. 1, demonstrate that the multidirectional modified *Physarum* algorithm with matching ability (D&B with rM) provides higher success rate than the unidirectional modified *Physarum* algorithm (D). Although D performs better at 300000 function evaluations (success rate is near 0.5

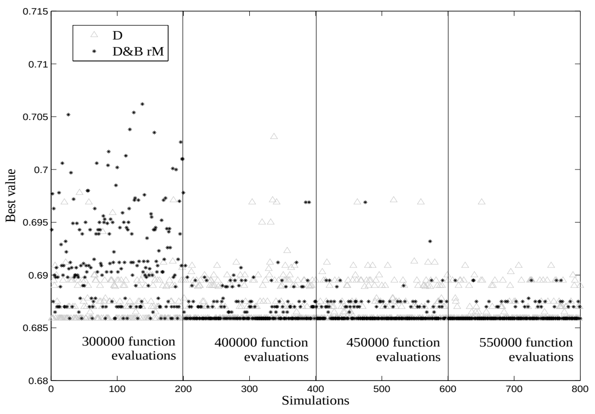

Figure 2. Best solution found by D and D&B with rM on groups of 200 runs for different function evaluation limits - TSP test case *Ulysses16* - best value = 0.6859.

whilst success rate of D&B with rM is under 0.1), increasing the number of function evaluations the performance of multidirectional exceeds the performance of direct only solver. At 550000 function evaluations the success rate of D&B with rM reaches 0.83, against the 0.66 of D. Fig. 1 shows also the performance of a simple Ant Colony Optimization solver (ACO [10]) at 550000 function evaluations. This solver is the implementation of the algorithm provided in [4]. Results show that D&B and D&B with rM have a good performance (success rate are respectively 0.66 and 0.83) if compared with ACO solver (success rate is 0.30). This comparison was made in order to check if the required function evaluations for the *Physarum* solver were comparable, in terms of order of magnitude, with a simple ACO solver. This comparison did not absolutely aims at demonstrating that the modified *Physarum* solver performs better than a general ACO solver.

Fig. 2 shows a comparison among the best solutions for each of the 200 runs at 300000, 400000, 450000, 550000 function evaluations. The best solution is known to be 0.6859 (TSPLIB [12]). Analyzing the figure, the D&B with rM algorithm (black dots) has an output comprised in a narrower band of values if compared with D algorithm (grey dots) increasing the number of function evaluations. At 300000 function evaluations there is a 9% probability to reach the exact value, a 71% probability to reach a value in the band $(0.6859, 0.6978]$ and 20% probability to reach a value in the band $(0.6978, 0.7070]$ using D&B with rM, while, using D, there is a 48% probability to reach the exact value, a 52% to reach a value

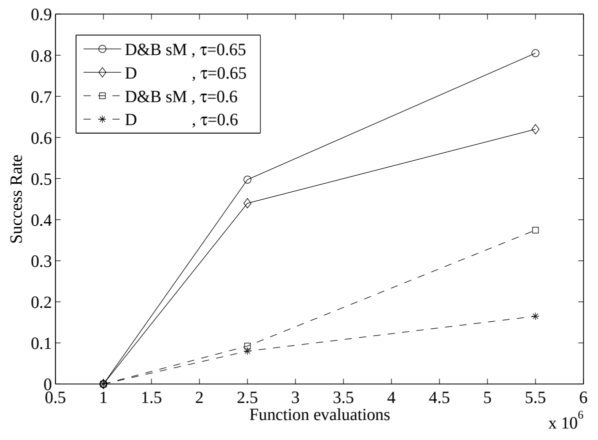

Figure 3. Variation of the success rate with the number of function evaluations - TSP test case *Eil51*.

between $(0.6859, 0.6978]$ and a 0% probability to reach a value in the band $(0.6978, 0.7070]$. At 550000 function evaluations the probability of exact value outcome using D&B with rM rise up to 83% with a 17% probability to reach a value in the band $(0.6859, 0.6950]$, whilst there is only a 66% probability using D, with a 27% probability to reach a value in the band $(0.6859, 0.6950]$ and a 7% in the band $(0.6950, 0.6978]$. It is evident the fastest convergence towards exact solution of black dots increasing the number of function evaluations.

The multidirectional and unidirectional modified *Physarum* algorithms were also applied to the symmetric TSP test case *Eil51*, described in Sect. 3.1. The values used as input parameters in the simulations are $m = 5 \times 10^{-5}$, $\rho = 1 \times 10^{-5}$, $GF_{ini} = 5 \times 10^{-3}$, $N_{agents} = 100$, $p_{ram} = 0.8$, $\alpha = 1$, $r_{ini} = 1$, and were chosen after a series of trials. A candidate list of dimension 10 was introduced. Results, as shown in Fig. 3 where $\tau = f_i^{best}$ (see Algorithm 2) represents the value of the threshold chosen for the success rate evaluation, demonstrate again that the multidirectional modified *Physarum* algorithm with matching ability (D&B with sM) provides higher success rate than the unidirectional modified *Physarum* algorithm (D). With both $\tau = 0.65$ and $\tau = 0.6$, the gain in success rate using a multidirectional approach reaches approximately 20% at 5500000 function evaluations.

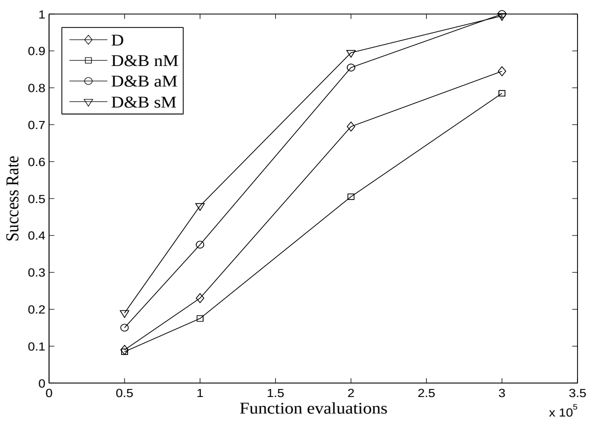

Figure 4. Variation of the success rate with the number of function evaluations - VRP test case.

Finally, the multidirectional and unidirectional modified *Physarum* algorithms were applied to the VRP test case described in Sect. 3.1. The values used as input parameters in the simulations are $m = 1 \times 10^{-5}$, $\rho = 1 \times 10^{-5}$, $GF_{ini} = 5 \times 10^{-3}$, $N_{agents} = 150$, $p_{ram} = 0.8$, $\alpha = 0$, $r_{ini} = 1$, and were chosen after a series of trials. The success rate is reported in Fig. 4. Results demonstrate that the multidirectional algorithm with matching ability (D&B with sM and D&B with aM) performs better than the unidirectional algorithm (D). After analysing more in depth the results and considering the mathematical modeling in Sect. 2.1, one could argue that:

I. At 300000 function evaluations the gain in success rate using D&B with sM or aM instead of D is around 20%.

II. The performance of D&B with sM and D&B with aM are almost comparable. The real gain using D&B with sM is the saving in computational time. The mean computational time in the simulations (200 runs) is reduced considerably (between 33% and 37%) using a selective matching instead of a non selective matching. This is due to the selection of best joint decision sequence at each generation: discarding worst joint decision sequences the updating of veins is done only for the best sequences.

III. The multidirectional algorithm without matching ability (D&B with nM) has the worst performance compared to others in all the cases. This is due to lack of communications between the two *Physarum*.

Simulations on bigger TSP and VRP problems (50 to more than 100 cities) are currently under way. Due to the fact that an higher number of generations are necessary to solve higher dimensional problems, the adaptive control of GF (see Sect. 2.1) may cause an uncontrolled growth of the flux inside few veins (it should be noted that flux is related to the fourth power of the radius, see Eq. (1)). A good solution is to set $\sigma = 0$ when dealing with high dimensional problems.

4.1 Conclusion

This paper proposed a multidirectional incremental modified *Physarum* solver for discrete decision making problems. The algorithm showed the ability to solve simple symmetric TSP and VRP problems, selected as representative examples of reversible decision making problems. Simulations on selected test cases proved that a multidirectional approach with matching ability performs better than a unidirectional one when applied to reversible discrete decision making problems. The possibility for the two *Physarum* to evaluate each step of the decision sequence from multiple directions and create joint paths enhances the performance of the solver: this forward and backward decision making process increased the gain in success rate up to 20% during selected simulations. Furthermore simulations on TSP selected test cases showed a good performance of multidirectional incremental modified *Physarum* solver if compared to a simple ACO solver. Simulations on bigger TSP and VRP problems (50 to more than 100 cities) are currently under way. After these last series of tests, the multidirectional incremental *Physarum* solver will be applied to other challenging decisional problems with a special focus on aerospace engineering.

References

[1] A. Adamatzky, G.J. Martinez, S.V. Chapa-Vergara, R. Asomoza-Palacio and C.R. Stephens. Approximating Mexican Highways with Slime Mould. *Natural Computation*, 2011, in press.

[2] N.R. Burns, M.D. Lee, D. Vickers. Are Individual Differences in Performance on Perceptual and Cognitive Optimization Problems Determined by General Intelligence?. *The Journal of Problem Solving* 1(1):5-19, 2006.

[3] M. Dorigo and L.M. Gambardella. Ant Colonies for the Travelling Salesman Problem. *BioSystems*, 43:73-81, 1997.

[4] M. Dorigo, V. Maniezzo, A. Colorni. The Ant System: Optimization by a colony of cooperating agents. In *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26(1):29-41,1996.

[5] D.S. Hickey and L.A. Noriega. Insights into Information Processing by the Single Cell Slime Mold Physarum Polycephalum. In *UKACC Control Conference*,

Manchester University, September 2-4, 2008

[6] D.R. Monismith Jr. and B.E.Mayfield. Slime Mold as a Model for Numerical Optimization. In *2008 IEEE Swarm Intelligence Symposium*, St. Louis MO USA, September 21-23, 2008.

[7] T. Nakagaki, H. Yamada and A. Toth. Maze-Solving by an Amoeboid Organism. *Nature*, 407: 470, 2000.

[8] A. Tero, R. Kobayashi and T. Nakagaki. Physarum Solver: a Biologically Inspired Method of Road-Network Navigation. *Physica: A Statistical Mechanics and its Applications*, 363(1):115-119, 2006.

[9] A. Tero, S. Takagi, 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*, 439:327, 2010.

[10] H. Wang. Solving Symmetrical and DisSymmetrical TSP based on Ant Colony Algorithm. Retrieved from MATLAB® CENTRAL, URL: http://www.mathworks.com/matlabcentral/fileexchange/14543.

[11] A. Tero, K. Yumiki, R. Kobayashi, T. Saigusa and T. Nakagaki. Flow-Network Adaptation in Physarum Amoebae. *Theory in Biosciences*, 127(2): 89-94, 2008.

[12] TSPLIB, library of instances for travelling salesman and vehicle routing problems, Ruprecht Karls Universitaet Heidelberg, URL: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.

[13] M. Vasile, E. Minisci, M. Locatelli. An Inflationary Differential Evolution Algorithm for Space Trajectory Optimization. In *IEEE Transaction on Evolutionary Computation*, 2(2):267-281, 2011.