A MULTI-DIRECTIONAL MODIFIED *PHYSARUM* ALGORITHM FOR OPTIMAL MULTI-OBJECTIVE DISCRETE DECISION MAKING

L. Masi[a] and **M. Vasile**[b]

[a]Department of Mechanical & Aerospace Engineering
University of Strathclyde
75 Montrose Street, G1 1XJ, Glasgow, UK
luca.masi@strath.ac.uk

[b]Department of Mechanical & Aerospace Engineering
University of Strathclyde
75 Montrose Street, G1 1XJ, Glasgow, UK
massimiliano.vasile@strath.ac.uk

Abstract. This paper will address an innovative bio-inspired algorithm able to incrementally grow decision graphs in multiple directions for discrete multi-objective optimisation. The algorithm takes inspiration from the slime mould *Physarum Polycephalum*, an amoeboid organism that in its plasmodium state extends and optimizes a net of veins looking for food. The algorithm is here used to solve multi-objective Traveling Salesman and Vehicle Routing Problems selected as representative examples of multi-objective discrete decision making problems. Simulations on selected test cases showed that building decision sequences in two directions and adding a matching ability (multi-directional approach) is an advantageous choice if compared with the choice of building decision sequences in only one direction (unidirectional approach). The ability to evaluate decisions from multiple directions enhances the performance of the solver in the construction and selection of optimal decision sequences.

Keywords: Physarum, Multi-Objective, Multi-Directional

1 Introduction

The idea that nature can inspire humans to solve complex decision making problems was widely used over the past two decades. A number of bio-inspired algorithms designed to solve decision making problems was and still are studied and developed. An example is the ACO (Ant Colony Optimisation) alghorithm [1] that takes inspiration from the social behaviour of ants looking for food. Following this concept, the behaviour of other social animals was successfully used: examples are bee colonies [2], fireflies [3], birds [4].

The method proposed in this paper takes inspiration from *Physarum Polycephalum*, see Fig. 1, known as many-headed slime mould, a simple organism

Fig. 1. Physarum Polycephalum. Image courtesy of *Howard County Bird Club* at www.howardbirds.org.

inahabiting moist areas that was endowed by nature with heuristics that can be used to solve single-objective and multi-objective discrete decision making problems. In [5] it has been shown that *Physarum Polycephalum* is able to solve a maze finding the shortest path that connects the maze's entrance and exit by changing its shape. It has been shown also that a living *Physarum* is able to recreate the Japan rail network [6] and the Mexican highway network [7], both using an experimental arena with food sources at each of the major cities in the regions. *Physarum* based algorithms have been developed recently to solve multi-source problems with a simple geometry [8,9], mazes [10] and transport network problems [6,10].

In this paper a multi-directional modified *Physarum Polycephalum* algorithm able to solve NP-hard multi-objective classical problems in operations research is proposed. In [11,12,13] multi-objective bio-inspired algorithms, i.e. ant colony algorithms, have been proposed and studied. The algorithm presented in this work is a multi-objective generalization of the single-objective multi-directional modified *Physarum* solver previously presented in [14] for discrete decision making.

In Sect. 2 the physiology of *Physarum* is introduced: discrete decision making problems are modeled with decision graphs where nodes represent the possible decisions while arcs represent the cost vector associated with decisions. Each arc has a scalar dominance index associated which is calculated comparing all the arcs leaving a node, as explained in Sect. 2. Decision graphs are incrementally grown and explored in multiple directions using the *Physarum*-based heuristic. This paper aims at proving that a multi-directional incremental *Physarum* solver is more efficient, in terms of success indexes (see Sect. 3.1), than a unidirectional incremental *Physarum* solver when applied to the solution of multi-objective 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. In [14] it has been already shown that the single-objective multi-directional modified *Physarum* algorithm is more efficient than the unidirectional algorithm when applied to small scale single-objective discrete decision making problems. This thesis will be demonstrated for the multi-objective algorithm in Sect. 4 solving some test cases. Bi-objective Traveling Salesman and Vehicle Routing Problems (TSP and VRP), introduced in Sect. 3, with a number of nodes between 10 and 100, 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 [9]. *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. The longest is the path connecting the centre with the source of food, the smallest is the flux and viceversa: best veins in terms of length that connect its centre with the food tend to increase their radius and the flux of nutrients inside, while longer veins tend to decrement the flux and close with time. This behaviour can be interpreted as a natural attitude in optimising the energy required to feed the organism by shape variation.

2.1 Problem Formulation: Multi-Objective Discrete Decision Making

Given a solution j to a discrete multi-objective decision making problem P, with cost vector $\mathbf{s}^j = [s^j_1, s^j_2, ..., s^j_n]$ and a solution i with cost vector $\mathbf{s}^i = [s^i_1, s^i_2, ..., s^i_n]$, the solution j dominates i if $s^j_k \leq s^i_k$ for all the $k = 1, 2, ..., n$ and $s^j_k < s^i_k$ for at least one k. The relation $\mathbf{s}^j \prec \mathbf{s}^i$ states that $\mathbf{s}^j$ dominates $\mathbf{s}^i$. The dimension of the vector $\mathbf{s}^j$ expresses the number of evaluating criteria for a solution j. The cost vector represents the cost associated with a decision. A general problem in discrete multi-objective optimisation is to find the feasible non dominated solutions to the given discrete multi-objective decision making problem P. Following the theory developed in [18], it is possible to associate a scalar dominance index $I(\mathbf{s})$ to each solution. The lower is the index, the better is the solution: if one considers the set of solutions $S = \{\mathbf{s}^j, \mathbf{s}^i, \mathbf{s}^k\}$ where $\mathbf{s}^j \prec \mathbf{s}^i \prec \mathbf{s}^k$, the set of

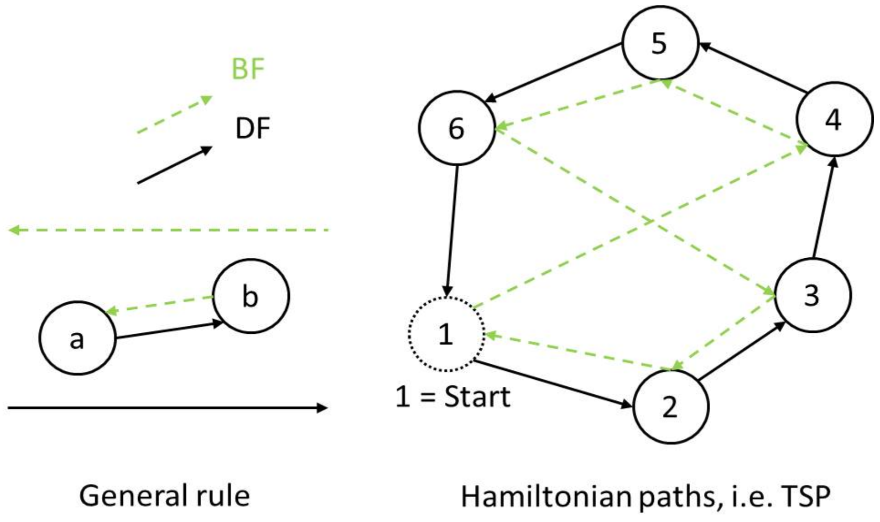


Fig. 2. Generally a *Physarum* working in direct flow (DF) would build the decision that brings from a to b, while the *Physarum* working in back flow (BF) would build the symmetric decision bringing from b to a (left). In a traveling salesman problem (TSP, right) nodes are fixed while arcs are built with time and a decision that brings from a to b can be built from both DF and BF *Physarum*, as for the arc that connect 4 to 5.

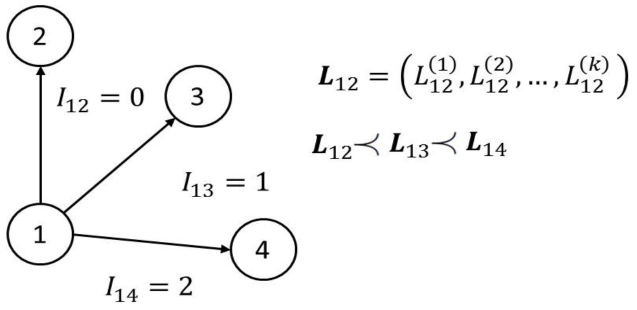
associated scalar indexes will be $I = \{I(\mathbf{s}^j) = 0, I(\mathbf{s}^i) = 1, I(\mathbf{s}^k) = 2\}$. All the non-dominated solutions in a general set S form the set:

$$PF = \{\mathbf{s} | I(\mathbf{s}) = 0\} \tag{1}$$

which is called Pareto front. Therefore, the solution of the problem P translates into finding the elements of PF.

2.2 Multi-Objective multi-directional *Physarum* Algorithmic

A reversible discrete decision problem can be modeled using a symmetric directed graph. The reversibility of a decision that induces a change from a state a to state b indicates here that the decision that brings back from b to a exists and can be evaluated. Not necessarily these two decisions have the same cost. 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



2
3
1
4
$I_{12} = 0$
$I_{13} = 1$
$I_{14} = 2$
$\boldsymbol{L}_{12} = \left(L_{12}^{(1)}, L_{12}^{(2)}, \ldots, L_{12}^{(k)}\right)$
$\boldsymbol{L}_{12} \prec \boldsymbol{L}_{13} \prec \boldsymbol{L}_{14}$

Fig. 3. Example of acrs' values assignment. $\boldsymbol{L}_{ij}$ and I_{ij} are respectively the cost vector and dominance index associated with the general decision from node i to node j.

by extending their net of veins. A multiple direction growing decision *Physarum* graph is obtained. In the example mentioned before, the *Physarum* working in DF would build its graph by creating arcs that move from a to b. If the graph was traversed by a virtual agent, the agent would walk along an arc from a to b. The other *Physarum* would build its graph in the opposite direction then walking along each arc from b to a. The result is a graph where both nodes and links are incrementally built by two expanding *Physarum*.

In this work the decisional problems analyzed are the TSP and the VRP, see Sect. 3; being Hamiltonian paths, all the nodes are built at the beginning, and only the arcs are built with time. This means that both the *Physarum* working in *DF* and the one working in *BF* can build arcs connecting two nodes in both directions. If the graphs built by the two *Physarum* are fully connected after a transient of growth, which is the case in this paper, two superposed symmetric directed graph are obtained. Fig. 2 shows a simple example for a TSP problem.

Exploration Using the Hagen-Poiseuille law, the flux through the net of *Physarum* veins is [6,8,9,10]:

$$Q_{ij} = \frac{\pi r_{ij}^4}{8\mu} \frac{\Delta p_{ij}}{L_{ij}} \tag{2}$$

where Q_{ij} is the flux between i and j, μ is the dynamic viscosity, r_{ij} the radius, L_{ij} the length and Δp_{ij} the pressure gradient.

In a multi-objective algorithm the length L_{ij}, representing the cost of the decision that brings from i to j, is a vector $\boldsymbol{L}_{ij}$. Its value can be substituted with the scalar dominance index $I_{ij} \in \mathbb{N}^0$, see Fig. 3: the cost vectors associated with each veins that connect a node N_i with other nodes N_j^k can be compared and the dominance indexes can be evaluated. Eq. (2) becomes:

$$Q_{ij} = \frac{\pi r_{ij}^4}{8\mu} \frac{\Delta p_{ij}}{(I_{ij} + 1)} \tag{3}$$

where plus 1 was added to avoid a singularity for $I_{ij} = 0$.

Table 1. Input parameters for the modified *Physarum* solver.

m	Linear dilation coefficient, see Eq. (9).
ρ	Evaporation coefficient, see Eq. (5).
GF_{ini}	Initial growth factor, see Eq. (7)
N_{agents}	Number of virtual agents.
p_{ram}	Probability of ramification, see Sect. 2.2.
α	Weights on ramification, see Eq. (8).
$k_{explosion}$	radii upper limit, see Eq. (10)

The strategy of using a single structure (in this case the flux) has been previously examined in [11] for Ant Colony Optimisation applied to multi-objective problems (where a weighted sum of all the objectives is used instead of the index I_{ij}). Another strategy, as reported in [11], could be the use of several fluxes structures ([11] refers to pheromones), one for each objective. The first strategy was chosen because it has the advantage of being easy to implement and usable when the number of objectives is high, as in many real-world problems. This considerations will be further discussed in Sect. 4.1.

Diameter variations then cause 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:

$$\left.\frac{d}{dt} r_{ij}\right|_{dilation} = f\left(Q_{ij}\right) \tag{4}$$

where $f(0) = 0$, i.e. linear, sigmoidal, etc. Veins' contraction, similarly to the evaporative effect in ACO [1], can be assumed to be linear with radius:

$$\left.\frac{d}{dt} r_{ij}\right|_{contraction} = -\rho r_{ij} \tag{5}$$

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:

$$P_{ij} = \begin{cases} \frac{Q_{ij}}{\sum_{j \in N_i} Q_{ij}} & if\, j \in N_i \\ 0 & if\, j \notin N_i \end{cases} \tag{6}$$

where N_i is the set of neighbour for i.

An additive term in the veins' dilation process, whose first main term is expressed in Eq. (4) was added in the algorithm and takes inspiration from the behaviour of the amoeba *Dictyostelium discoideum* [16]. This dilation is:

$$\left.\frac{d}{dt} r_{ij_{best}}\right|_{elasticity} = GF r_{ij_{best}} \tag{7}$$

where GF is the growth factor and $r_{ij_{best}}$ the veins' radius of the best chains of veins, i.e. the veins that form the paths in the decision graph that are in the current calculated Pareto front. This dilation, as explained in [14], simulates the tendency of best veins to further increase their radius for the effect of the flux.

Growth in multiple directions and matching 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.(6). Once a node is selected, there is a probability p_{ram} of ramification towards new nodes that are not yet connected with the actual node. The value of p_{ram} can be chosen *a priori* or a law can be defined, i.e. $p^c_{ram} = p^c_{ram}(A_c)$, where c is the current ramifying node and A_c the number of arcs that leave node c. In this work *a priori* values were chosen before the simulations.

If ramification is the choice, a weighted roulette, based on objective function evaluations, helps the *Physarum* with the selection and construction of a new link. 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 I_{cn_i} of the decision between c and n_i, i.e. the dominance scalar index associated with the decision:

$$p_{cn_i} \propto \frac{1}{(I_{cn_i} + 1)^\alpha} \tag{8}$$

where α is a weight. Once a new link is built, a complete decision path is constructed (creating other links if necessary).

Assuming then the presence of two counter expanding *Physarum*, one in direct-flow *DF* and one superposed in back-flow *BF*, as explained in the previous paragraph, a matching condition can be then defined. If an arc connecting two nodes that belongs 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*. Some matching strategies were compared in [14] for single-objective problems. In this paper, two matching strategies were implemented in the multi-objective modified *Physarum* solver.

The first one, called *selective-matching*, follows an elitist criterion where at each generation a joint path is selected if and only if its total cost vector is not dominated by the previous joint paths selected during the same generation, as in [14]. It could be noted that if a high number of exploring agents is chosen, a high number of paths are matched. This could lead to a slowdown of the code speed, especially if complete decision sequences are long, as in more complex multi-objective VRPs and TSPs (more than 20 cities). For this reason, an other strategy, called *mix-matching*, was designed for larger scale problems. Selective matching is done only considering the best n solutions in DF and BF during a generation, so that worst routes are excluded *a priori*. A value $n = \frac{dim}{5}$, where *dim* is the problem dimension, i.e. the number of cities for TSP and VRP, was used in the simulations presented in this paper. Furthermore the n best decisions in DF and BF are matched with the Pareto front found by the algorithm at the time of matching.

Algorithm 1 multi-directional incremental modified Physarum solver

```
initialize $m$, $\rho$, $GF_{ini}$, $N_{agents}$, $p_{ram}$, $\alpha$
generate a random route from start to destination both 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. (8) create a new link to a node not yet connected
            update scalar dominance indexes for current node, see Sect. 2.1
         else
            move on existing graph using Eq. (6)
         end if
      end if
   end for
   look for possible matchings among decision sequences in $DF$ and $BF$
   update Pareto front
   contract and dilate veins using Eqs. (4), (5), (7)
   if $r_{ij}$ exceeds upper radius limit, see Eq. (10) then
      block radius increment
   end if
   update fluxes and probabilities using Eqs. (3), (6)
end for
```

These matching conditions can be interpreted as a communication ability between the two *Physarum*: they move according to their nature and the knowledge acquired exploring the decision space, which contains both personal experience and shared information.

Restart procedure Simulations on selected test cases (see Sect. 4) were carried out adding in the *Physarum* algorithm two restart procedures to avoid stagnation on local minima. The first restart procedure, called *restart1*, is 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 when used on small scale problems (number of cities less than 16 in this paper). Given an initial value for the growth factor GF_{ini}, GF is incremented by a fixed percentage σ after every generation. If the highest probability p_{best}^{PF} associated with the paths in the calculated Pareto front so far is higher than a fixed value p_{lim}^{low}, the increment is set to zero. Then, if p_{best}^{PF} exceeds a value p_{lim}^{high}, GF is set equal to GF_{ini} and veins are dilated and contracted to their initial value. In the present paper is assumed $\sigma = 0.01$, $p_{lim}^{low} = 10^{-4}$, $p_{lim}^{high} = 0.85$ for the bi-objective Vehicle Routing Problem test case *Tuscany10* and $p_{lim}^{high} = 0.95$ for the bi-objective Traveling Salesman Problem test case *Ulysses16*. A second restart procedure, called *restart2*, was designed for larger scale problem, i.e. the bi-objective traveling salesman instance *KroAB100*. It is based on minimum

Table 2. Values used as input parameters - TSP test case (first row) and VRP test case (second row).

Instance	m	ρ	GF_{ini}	N_{agents}	p_{ram}	α	$k_{explosion}$
Ulysses16	5×10^{-5}	$\frac{1 \cdot 10^{-5}}{N_{agents}}$	$5 \cdot 10^{-3}$	100	0.8	0	10^8
Tuscany10	5×10^{-5}	$\frac{1 \cdot 10^{-5}}{N_{agents}}$	$1 \cdot 10^{-3}$	150	0.8	0	10^8
KroAB100	5×10^{-5}	$\frac{5 \cdot 10^{-6}}{N_{agents}}$	$5 \cdot 10^{-3}$	50	1	0	5

nodes in common among decision sequences in a generation and among decision sequences and Pareto front; the algorithm is restarted if one of the following conditions is achieved:
I) the minimum number of nodes in common n_{min}^{com}, obtained comparing all decision sequences among each other in a generation, exceeds a threshold n_{com}.
II) a fraction β of the decision sequences built during a generation belong to the calculated Pareto front at same generation.
In this paper a value $n_{com} = \frac{dim}{2}$, where dim is dimension of the problem, i.e. the number of cities, and a value $\beta = \frac{2}{3}$ were used. The goal of this restart procedure is to avoid both a stagnation to local single minima and to the calculated Pareto front itself.

Considerations on the algorithm The set of Eqs. (2)-(6) can be implemented as in the following. In accordance to Eq. (2), flux in each vein is proportional to the radius and inversely proportional to the length (the scalar dominance index in a multi-objective problem, Eq. (3)). These two main parameters are taken into account in the algorithm. Once a vein is selected by a virtual agent in a generation, its radius is incremented using Eq. (4). In the present work, a function linear with respect to the product between the radius $r_{ij}^{(n)}$ of the veins traversed by agent n, and the inverse of the sum of dominance indexes ($I_{tot}^{(n)}$, see Sect. 2.1), associated with each arc of the decision taken by agent n, will be used for the veins' dilation:

$$\left.\frac{d}{dt} r_{ij}^{(n)} \right|_{dilation} = m \frac{r_{ij}^{(n)}}{I_{tot}^{(n)}} \tag{9}$$

where the coefficient m is the linear dilation coefficient. Evaporation is taken into account using Eq. (5) for each agent. Fluxes are then calculated using Eq. (3) and probabilities are updated in accordance with Eq. (6). Due to the mathematical nature of the algorithm (the flux is related to the fourth power of the radius), an upper limit on the maximum vein radius was introduced in order to avoid veins' flux explosion. If the radius r_{ij} exceeds a maximum value r_{max}, the vein dilation is blocked up until the radius is again below r_{max} for the effect of evaporation. This upper limit, called $k_{explosion}$, is given as ratio between r_{ij} and r_{ini}:

$$k_{explosion} = \frac{r_{ij}}{r_{ini}} \tag{10}$$

where r_{ini} is the initial radius of the veins. The main parameters of the modified *Physarum* solver are listed in Table 1. The initial radius of the veins r_{ini} is always set equal to 1 in the simulations presented in this paper. The pseudocode of the multi-directional incremental modified *Physarum* solver is provided in Algorithm 1.

3 Application to Multi-Objective Traveling Salesman and Vehicle Routing Problems and Benchmark

In single-objective optimisation the Traveling Salesman problem, TSP, is the problem of finding the shortest tour that visit each city of a given set S of n cities.

In the multi-objective optimisation case considered in this paper the cost function to be minimized is a vector of two values: the total length $L_{tot} = \sum_j L_j$ and the total road traffic $T_{tot} = \sum_j T_j$ of each tours, where $j = 1, ..., n$ is the index that identifies each part of the tour. The road traffic is here assumed to be inversely proportional to the length $T_j = 1/L_j$. The shorter is the tour, the higher is the probability that the tour is chosen by drivers, increasing the road traffic. The total road traffic $T_{tot} = \sum_j T_j$ will be called Road Traffic Index in the following. Although conflicting criteria, both length and road traffic in a tour have to be minimised.

TSPLIB [17] 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 multi-objective multi-directional *Physarum* solver to test case *Ulysses16* that was modified adding the road traffic to the cost function and to the test case *KroAB100*, obtained from the single-objective intances *KroA100* and *KroB100*. For the *KroAB100* the consideration above on the road traffic index does not apply: it is a bi-objective problem itself and two objectives to be minimised are included in the instance.

The multi-objective Vehicle Routing Problem, VRP, considered in this paper is a similar problem. Given a set of n cities with a demand k, whose reciprocal distance L_j and road traffic $T_j = 1/L_j$ are known, v vehicles of capacity c, d depots located in fixed cities, the VRP is the problem of delivering goods located in the depots using a defined amount of vehicles with finite capacity. The goal is to satisfy the demand of each city minimizing the cost functions, i.e. the distance and road traffic. VRP reduces to a TSP if there is only one vehicle with infinite capacity. When the modified *Physarum* algorithm is applied to VRP, a probability skew factor ψ 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 $(1 - \psi)$. Other probabilities are then risen of a same value in order to have the sum of probabilities equals to 1. The skew factor ψ is introduced in the model to avoid frequent returns to depot in the decision sequences and is here set equals to 0.5. 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

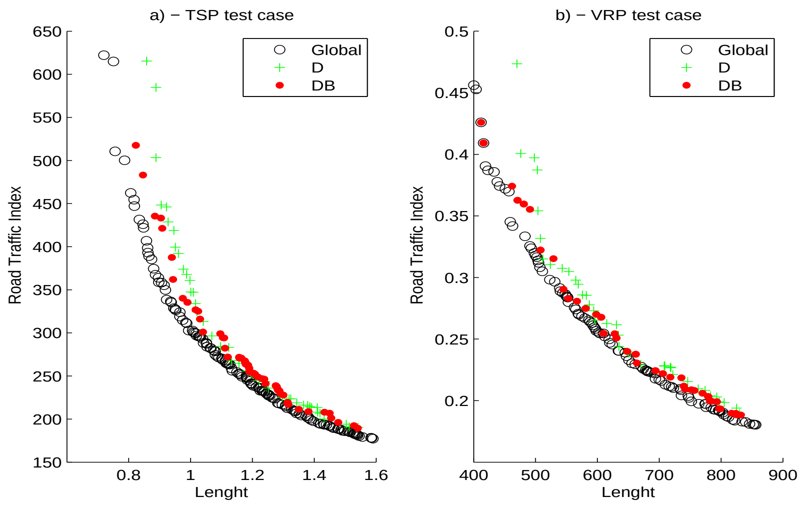


Fig. 4. Pareto front - TSP test case *Ulysses16* at $6.5 \cdot 10^5$ function evaluations, (a) - VRP test case *Tuscany10* at $3 \cdot 10^5$ function evaluations, (b). In the legend, *Global* indicates the global Pareto front obtained from all the runs of the D and D&B algorithms (1600 for the VRP test case and 2800 for the TSP test case), while *D* and *D&B* indicate an example of a Pareto front found during a run of the D and D&B algorithms respectively.

a city considered the depot (Ponsacco). The Euclidean distance in kilometers was used. VRP parameters were set to $n = 9, k = cost = 1, v = 1, c = 4, d = 1$, i.e. one vehicle with capacity equals to 4, one depot and a constant demand equals to 1.

3.1 Testing Procedure

The testing procedure proposed in [18] was used in this paper. Two metrics are defined:

$$M_{spr} = \frac{1}{M_p} \sum_{i=1}^{Mp} \min_{j \in N_p} \| \frac{\mathbf{f}_j - \mathbf{g}_i}{\mathbf{g}_i} \| \tag{11}$$

$$M_{conv} = \frac{1}{N_p} \sum_{i=1}^{Np} \min_{j \in M_p} \| \frac{\mathbf{g}_j - \mathbf{f}_i}{\mathbf{g}_j} \| \tag{12}$$

where M_p is the number of elements, with objective cost function $\mathbf{g}$, in the true global Pareto front and N_p is the number of elements, with objective cost function $\mathbf{f}$, in the Pareto front that a given algorithm is producing. Although similar, the two metrics are measuring two different things: M_{spr} is the sum,

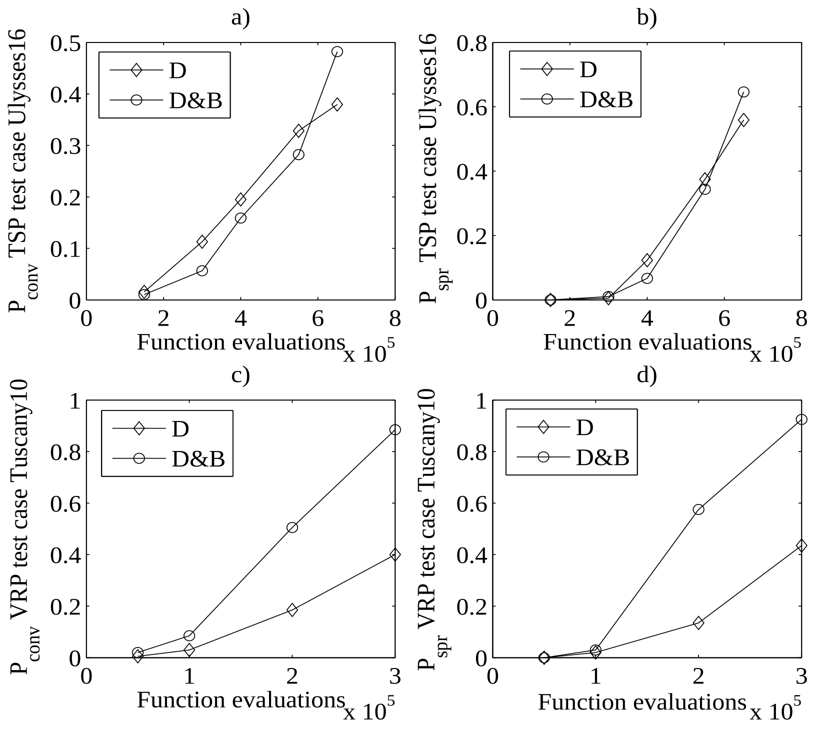


Fig. 5. Variation of the indexes of performance p_{conv} and p_{spr} with the number of function evaluations - TSP test case *Ulysses16*, (a) and (b) - VRP test case *Tuscany10*, (c) and (d).

over all the elements in the global Pareto front, of the minimum distance of all the elements in the Pareto front N_p from the i^{th} element in the global Pareto front: this metric would be high if N_p was only a partial representation of the global Pareto front. M_{conv}, instead, is the sum, over all the elements in the Pareto front N_p, of the minimum distance of the elements in the global Pareto front from the i^{th} element in the Pareto front N_p: this metric would give a low value if N_p was an accurate, although partial, representation of the global Pareto front.

From the considerations above, both the metrics M_{spr} and M_{conv} should be low for a good estimate of the global calculated Pareto front. The indexes of performance $p_{conv} = P(M_{conv} < tol_{conv})$ and $p_{spr} = P(M_{spr} < tol_{spr})$ will be used to explore the efficiency of the algorithm and to compare the multi-directional and the unidirectional versions. Given n repeated runs, p_{conv} is the probability that M_{conv} achieves a value less than tol_{conv}, while p_{spr} is the probability that M_{spr} achieves a value less than tol_{spr}. 200 runs are sufficient in order to obtain

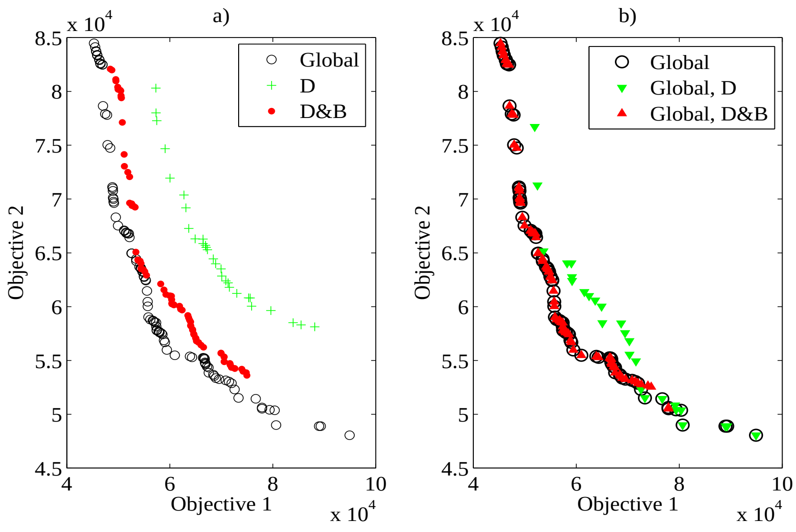


Fig. 6. Pareto fronts for TSP test case *KroAB100* at $4 \cdot 10^7$ function evaluations. In the legend of figure a), *Global* indicates the global Pareto front obtained from all the runs of the D and D&B algorithms (40), while *D* and *D&B* indicate an example of a Pareto front found during a run of the D and D&B algorithms respectively. In the legend of figure b), *Global* indicates the global Pareto front as in a), while *Global, D* and *Global, D&B* indicate the global Pareto fronts obtained from all the runs of the D and D&B algorithms respectively (20).

an error $\leq 5\%$ with a 95% of confidence [18]. For the TSP test case *Ulysses16* the tolerances tol_{conv} and tol_{spr} are set equal to 0.0465 and 0.045 respectively, for the VRP test case *Tuscany10* to 0.030 and 0.035, and for the TSP test case *KroAB100* to 0.048 and 0.058.

4 Results

The multi-objective multi-directional modified *Physarum* solver, named D&B in the following, was compared against a multi-objective unidirectional modified *Physarum* solver, named D. The D algorithm is obtained by freezing the backflow BF. The two algorithms were applied to the modified symmetric traveling salesman problem test case *Ulysses16*, to the symmetric traveling salesman problem test case *KroAB100* and to the vehicle routing problem test case *Tuscany10*, described in Sect. 3. The values used as input parameters in the simulations, chosen after a series of trials, are listed in Table 2. Selected ones showed best performance. The restart procedure *restart1* and the matching strategy *selective-matching* were used for *Ulysses16* and *Tuscany10*, while *restart2* and *mix-matching* were used for *KroAB100*. Simulations were carried out on a 64-bit

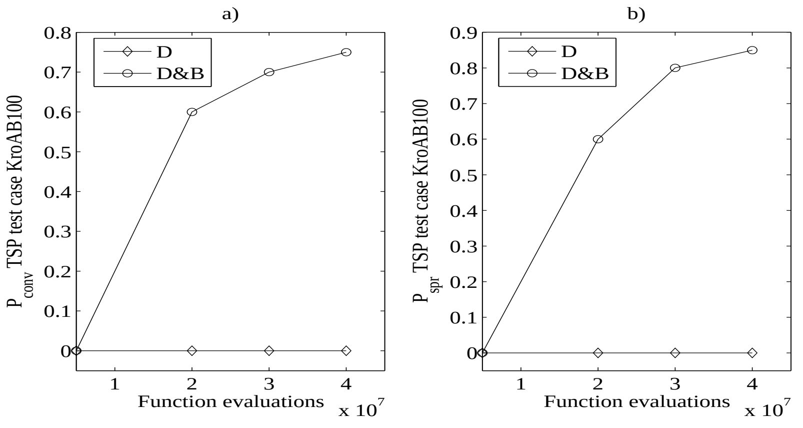


Fig. 7. Variation of the indexes of performance p_{conv} and p_{spr} with the number of function evaluations - TSP test case *KroAB100.*

OS Windows 7 Intel® Core^{TM}2 Duo CPU E8500 3.16GHz 3.17GHz.

Ulysses16 & Tuscany10. In both the test cases the true global Pareto front was unknown. In order to obtain a global Pareto front all the runs (1600 for the VRP test case *Tuscany10* and 2800 for the TSP test case *Ulysses16*) of the multi-directional and unidirectional algorithms were used: two global Pareto fronts for both the VRP and TSP test cases were built using all the solutions found by the algorithms. In Fig. 4 the global Pareto fronts are shown. The figure reports also an example of Pareto front found by unidirectional (D) and multi-directional (DB) algorithms, for both*Ulysses16* and *Tuscany10.* Fig. 5 shows the variation of the indexes of performance p_{conv} and p_{spr} with the number of function evaluations for the TSP test case ((a) and (b)) and for the VRP test case ((c) and (d)). A function evaluation is defined as the call to the objective function, i.e. each arc selected by the virtual exploring agents (see Sect. 2.2) is considered a function evaluation. Results for the VRP test case *Tuscany10*, as shown in Fig. 5 (c) and (d), demonstrate that the multi-objective multi-directional modified *Physarum* algorithm with matching ability (D&B) provides higher indexes of performance p_{conv} and p_{spr}, than the multi-objective unidirectional modified *Physarum* algorithm (D), at all the function evaluations limit. This gain is up to approximately 50% for the p_{spr} and p_{conv} at $3 \cdot 10^5$. The results for TSP test case *Ulysses16*, reported in Fig. 5 (a) and (b), show that the multi-directional algorithm provides better performance after $6 \cdot 10^5$ function evaluations and the gain is up to 10% for both p_{spr} and p_{conv} at $6.5 \cdot 10^5$. The behaviour of the indexes of performance for this multi-objective instance are similar to the behaviour of the index of performance in [14] for the same TSP test case with single-objective: the unidirectional algorithm tends to have a better performance during the early stage of the simulation, then the performance of the multi-directional algorithm

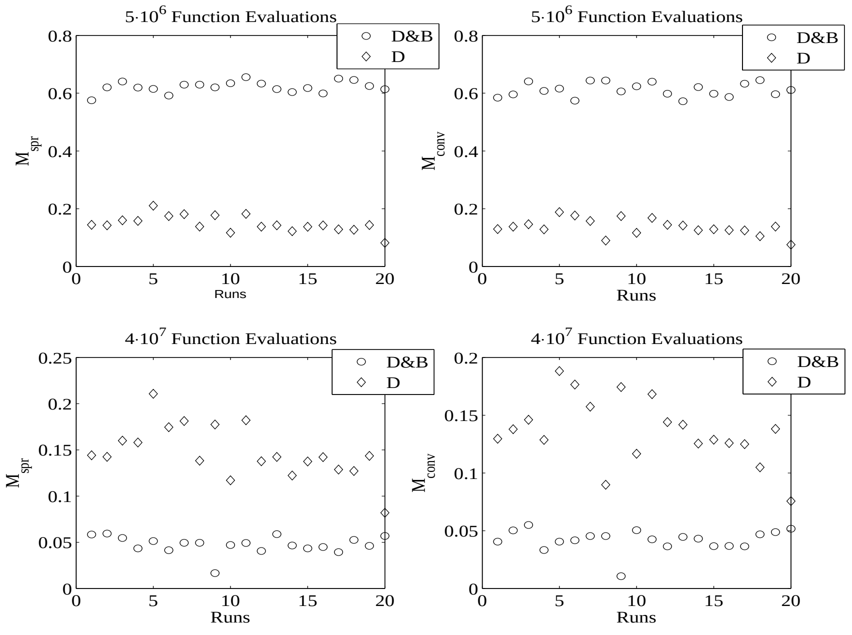


Fig. 8. Value of the metrics in Eqs. (11)-(12) for the D and D&B algorithms at $5 \cdot 10^6$ and $4 \cdot 10^7$ function evaluations.

exceeds the performance of the unidirectional.

KroAB100. As for the test cases above, for *KroAB100* the true global Pareto front was unknown. In order to obtain a global Pareto front all the runs (40) of the multi-directional D&B and unidirectional D *Physarum* algorithms were used. For this test case, only 20 runs were performed for each algorithm instead of 200. However, this number of runs doubles the one used to compare various algorithms on the same instance in [13]. In the following, *Global* is used to indicate the Pareto front found by all the runs of the multi-directional and unidirectional *Physarum* algorithms as explained above, while *Global D* and *Global D&B* are used to indicate the global Pareto fronts obtained from all the runs of the D and D&B algorithms respectively (20). Fig. 6 a) shows two examples of Pareto fronts found after one single run of the D and D&B algorithms. Fig. 6 b) shows a comparison of the *Global D* and *Global D&B*. Both figures let the reader see that the introduction of multi-directionality in the algorithm is an optimal choice. This is confirmed analyzing Fig. 7: while the indexes of performance p_{spr} and p_{conv} of the multi-directional algorithm reach respectively 88% and 75%, the ones of the unidirectional algorithm are still under 1%. However there is a small transient at a low number of function evaluations (less than $1 \cdot 10^7$), not visible

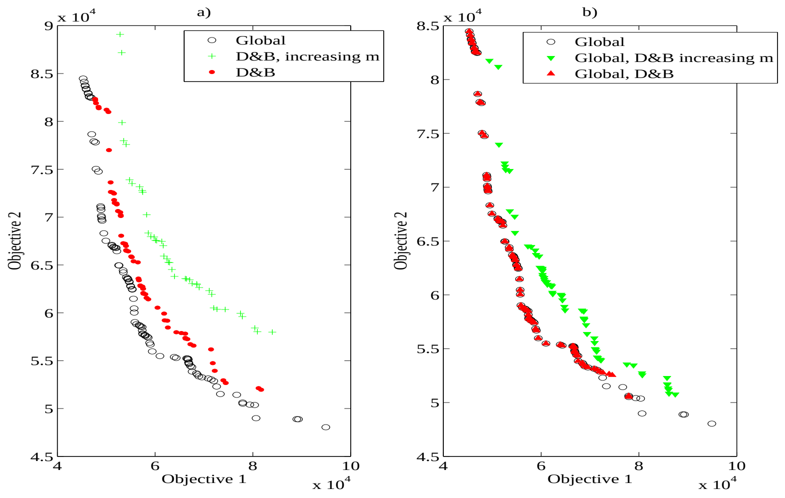


Fig. 9. Effect of the variation of the linear growth coefficient m and evaporation ρ at $4 \cdot 10^7$ function evaluations. In a) *Global* is the global Pareto front obtained from all the runs of the D and D&B algorithms (40) as in Fig. 4, *D&B* indicates an example of a Pareto front found during a run of the D&B algorithm with classical settings, as in Tab. 2, and *D&B increasing m* indicates an example of a Pareto front found during a run of the D&B algorithm where $m = 5 \cdot 10^{-3}$ and $\rho = 10^{-5}$. In b), *Global* is the global Pareto front as above, while *Global D&B increasing m* indicate the global Pareto fronts obtained from all the runs (20) of the D&B algorithms with $m = 5 \cdot 10^{-3}$ and $\rho = 10^{-4}$ and *Global, D&B* indicate the global Pareto fronts obtained from all the runs (20) of the D&B algorithms with classical settings.

in Fig. 7, where the unidirectional algorithm performs better than the multi-directional (although the performance is still not significant). This transient can be appreciated in Fig. 8: it reports the value of the metrics in Eqs. (11)-(12) for the D and D&B algorithms at $5 \cdot 10^6$ and $4 \cdot 10^7$ function evaluations. From the figure it is evident that D performs better at low function evaluations, but their increment is not able to improve its performance significantly, as for D&B. This behaviour is similar to that found in [14].
Fig. 9 shows the effect of increasing the linear growth coefficient to $m = 5 \cdot 10^{-3}$ and evaporation parameter to $\rho = 10^{-5}$. This variation should increase the rate of veins' expansion with time and limit the effect of contraction. From Fig. 9 one can argue that increasing the rate of veins' expansion is a bad choice: the algorithm tends to converge very rapidly to a set of solutions and the exploration is not efficient as with lower rate of expansion.

The results obtained by applying the Physarum algorithm to the three aforementioned test cases are quite interesting and prove the initial assumption that building decision sequences in two directions and adding a matching ability is an advantageous choice if compared with the choice of building decision sequences in only one direction in the solution of multi-objective discrete decision making problems. The two *Physarum* can evaluate each step of the decision sequence from two directions and create joint paths: this forward and backward decision making process improves the performance of the algorithm.

A comparison among the proposed algorithm and other bio-inspired ACO-style algorithms is not reported in this paper and will be the subject of the future work. Recently [13] provided an excellent comparison of the performance of multi-objective ACO algorithms and of SPEA2 and NSGA-II, applied to a benchmark of TSP problems, including KroAB100. In [13] the higher performance of the ACO algorithms if compared to NSGA-II and SPEA2 is shown. A first visual analysis of the Pareto Front obtained by the Physarum algorithm and the best multi-objective ACO algorithms in [13] indicates that the Physarum is able to find very quickly the centre of the front while there is a difficulty in finding the tails. This can be explained by the fact that a single structure is used (see Sect. 4.1).

4.1 Conclusion

This paper proposed an innovative multi-objective multi-directional incremental modified *Physarum* solver for multi-objective discrete decision making problems. The algorithm showed the ability to solve multi-objective problems in combinatorial optimisation, i.e. symmetric traveling salesman and vehicle routing problems, that were selected as representative examples of multi-objective reversible decision making problems. Simulations on selected test cases proved that a multi-directional approach with matching ability performs better than a unidirectional one when applied to small scale multi-objective reversible discrete decision making problems. This result is in line with the results showed in [14] for single-objective discrete decision making using a multi-directional modified Physarum algorithm. The multi-directional decision making process enhances the performance of the multi-objective solver: this gain is up to 50% (based on the indexes of performance proposed in Sect. 3.1) for the VRP test case.
It should be noted that, as introduced in Sect. 2.2, the strategy of using a single structure (in this case the flux), where the index of dominance I_{ij} is the parameter from which the *Physarum* draws knowledge on the decision space, is new. It has the advantage of being very easy to implement and it can be used when the number of objectives is high, as in many real-world problems. The disadvantage is that the proposed approach tends to concentrate the virtual exploring agents in the centre of the Pareto front, excluding the tails. On the other hand, the use of multiple structures, one for each objective, was well described and studied in [13]: it has the advantage of being able to expand the tails of the Pareto front, but an increase in the number of objectives would lead to the introduction of more structures, resulting in computational cost and complexity. However, the

disadvantage of using a single structure as proposed in this paper, could be overcome by adding sub-populations of agents that consider only one objective. This will be further studied in the future, although first results are encouraging.

References

1. Dorigo, M., Maniezzo, V., Colorni, A.: The Ant System: optimisation by a colony of cooperating agents. Systems, Man, and Cybernetics-Part B IEEE Transactions, 26(1), 29-41 (1996)
2. Chong, Chin Soon, Malcolm Yoke Hean Low, Appa Iyer Sivakumar, and Kheng Leng Gay: A bee colony optimisation algorithm to job shop scheduling. IEEE Simulation Conference. WSC 06. Proceedings of the Winter, 1954-1961 (2006)
3. Sayadi, M. K., Ramezanian, R., Ghaffari-Nasab, N.: A discrete firefly meta-heuristic with local search for makespan minimisation in permutation flow shop scheduling problems. International Journal of Industrial Engineering Computations, 1(1), 1-10 (2010)
4. Yang, Xin-She, Suash Deb: Cuckoo search via Lévy flights. Nature & Biologically Inspired Computing. NaBIC 2009. World Congress on. IEEE, 210-214 (2009)
5. Nakagaki, T., Yamada, H., Toth, A.: Maze-Solving by an Amoeboid Organism. Nature, 407, 470 (2000)
6. Tero, A., Takagi, S., Saigusa, T., Ito, K., Bebber, D.P., Fricker, M.D., Yumiki, K., Kobayashi, R., Nakagaki, T.: Rules for Biologically Inspired Adaptive Network Design. Science, 439, 327 (2010)
7. Adamatzky, A., Martínez, G. J., Chapa-Vergara, S. V., Asomoza-Palacio, R., Stephens, C. R.: Approximating Mexican highways with slime mould. Natural Computing, 10(3), 1195-1214 (2011)
8. Hickey, D. S., Noriega, L. A.: Insights into Information Processing by the Single Cell Slime Mold Physarum Polycephalum. UKACC Control Conference, 2008
9. Tero,A. , Yumiki, K., Kobayashi, R., Saigusa, T., Nakagaki, T.: Flow-Network Adaptation in Physarum Amoebae. Theory in Biosciences, 127(2), 89-94 (2008)
10. Tero, A., Kobayashi, R., Nakagaki, T.: Physarum Solver: a Biologically Inspired Method of Road-Network Navigation. Physica: A Statistical Mechanics and its Applications, 363(1), 115-119 (2006)
11. Alaya, I., Solnon, C., Ghedira, K.: Ant colony optimisation for multi-objective optimisation problems. Tools with Artificial Intelligence. ICTAI 2007. 19th IEEE International Conference, 1, 450-457 (2007)
12. Lopez-Ibanez,M.: Multi-objective Ant Colony optimisation. Diploma thesis, Intellectics Group, Computer Science Department, Technische Universitat Darmstadt, Germany, 2004
13. Garcia-Martinez, C., Cordon, O., Herrera, F.: A Taxonomy and an Empirical Analysis of Multiple Objective Ant Colony optimisation Algorithms for the Bi-Criteria TSP. European Journal of Operational Research, 180, 116-148 (2007).
14. Masi,L., Vasile,M.: A multi-directional Modified Physarum Solver for Optimal Discrete Decision Making. Proceedings of International Conference on Bio-Inspired optimisation Methods and their Applications, BIOMA, Bohinj, Slovenia, 2012
15. Dorigo, M., Gambardella, L.M.: Ant Colonies for the Traveling Salesman Problem. BioSystems, 43, 73-81 (1997)
16. Monismith Jr.,D.R. , Mayfield, B.E.: Slime Mold as a Model for Numerical optimisation. IEEE Swarm Intelligence Symposium, St. Louis MO, USA, 2008

17. TSPLIB, library of instances for Traveling Salesman and Vehicle Routing Problems, Ruprecht Karls Universitaet Heidelberg, URL: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.
18. Vasile M., Zuiani F. MACS: An Agent-Based Memetic Multiobjective optimisation Algorithm Applied to Space Trajectory Design. Institution of Mechanical Engineers, Part G, Journal of Aerospace Engineering, September, 2011