Utrecht University

Transport network creation of target-oriented particle-based models of Physarum Polycephalum

Author: F.S. Slijkhuis 2nd Reviewer: Dr. B.G. Rin

Supervisor: Dr. G.A.W. Vreeswijk

> A thesis of 7.5 ECTS submitted in partial fulfilment for the degree of Bachelor of Science

> > carried out for the

Department of Philosophy and Religious Studies, Faculty of Humanities (UU)

December 8, 2018



Universiteit Utrecht

Abstract

'Physarum Polycephalum' is a slime mold capable of solving the shortest path-problem. It tries to maximize its food intake while preserving minimal length of the network created between food points. Models of this organism have been constructed, which show the same properties. With certain parameters, these models can be used to approximate existing transport networks. Because real-life transport networks are solutions to the network design problem, an algorithm which approximates these networks well enough can also be used to design transport networks. We hope to increase the quality of network design by implementing the notion of targets in an existing Physarum-algorithm, creating a target-oriented particle-based model of Physarum Polycephalum. With this adaptation, particles will have a randomly assigned target, which is a food point. We show that this adaptation does improve the approximation of existing transport networks. Thus, we have created a model which is better at designing an efficient transport network than the regular Physarum-algorithm. We also provide some useful applications and further adaptations, which could further increase the quality of network design by Physarum-algorithms.

TABLE OF CONTENTS

1 Introduction
2 Jones' Physarum model
2.1 Initiation
2.2 Sensory stage
2.3 Movement stage
2.4 Parameters
3 Methods
3.1 Adaptations to Jones' Physarum model 4
3.2 Comparing the models
4 Results
4.1 Abstract environments
4.2 The Netherlands
4.3 Poland
5 Discussion
References
Appendix
1. Countries
A. The Netherlands
B. Poland
2. Graphs
A. The Netherlands
B. Poland
3. Code repository

1 INTRODUCTION

Hundreds of millions of people drive on 64 million kilometers of road every single day (Central Intelligence Agency, 2013). This is equivalent to 46 times the diameter of the sun. Many countries on earth have vast road networks, and almost all of them are constantly expanding. The total size of rail transport networks worldwide is increasing as well, recently passing a million kilometers (The World Bank, 2016). With many countries investing in high-speed rail, rail networks will keep changing. Rapidly expanding infrastructure requires rapid solutions. Both rail and road networks have to be very efficient, keeping costs low, but capacity high. With these rapid expansions, automation is desired. Shortest path-algorithms can be of great help here.

One way of finding shortest path-algorithms suitable for providing efficient networks is by looking at nature. Nature has shown many efficient ways of solving some difficult problems. 'Physarum Polycephalum', a type of slime mold, is an example of an organism capable of solving the shortest path-problem. This slime mold creates paths between food sources, maximizing its food intake, but keeping the network length at a minimum. By placing these food sources in strategic places, this organism can be used to create networks. For example, food sources can serve as existing cities. When Physarum Polycephalum is used to create road networks between these food sources, the result often approaches real-life road networks (Adamatazky, Lees, & Sloot, 2013). The way in which Physarum finds a path, can be modeled. This results in an efficient shortest path-algorithm (Zhang, et al., 2014).

Jones' particle-based slime mold-algorithm has demonstrated spontaneous transport network formation (Jones, 2011), making it useful in approaching existing networks. The algorithm can be considered as a virtual computing material. Adaptations have been made to Jones' particle-based slime mold-algorithm in the past. These adaptations can be used to improve some of the qualities the model has. One example is the addition of the notion of terrain (van Dorsten, 2015). Another quality which could be improved is real-life transport network approximation. Slime mold-algorithms usually perform well in this area on a smaller scale. This is useful, since real-life transport networks can be seen as solutions to the 'Network Design Problem' (Yang & Bell, 1998). This shows us that real-life transport network approximation is great for measuring the performance of slime mold-algorithms on forming efficient transport networks.

The problem with constructing efficient networks, or network design, is that it is a very complicated process. It is a process which requires many days or months of work by many different people, depending on the scale on which it is done. There is much decision-making involved, which can be strategic, tactical and operational. All these decision-making situations make it hard for algorithms to replace humans in this process. Many algorithms designed to tackle this problem have a lot of limitations (Magnanti & Wong, 1984). This makes the Network Design Problem one on of the most difficult problems in transport (Yang & Bell, 1998). With the large increase of interest in Artificial Intelligence, lots of AI-related algorithms have been improved or developed. These algorithms might make it possible to finally automate the process of network design. The algorithm previously described, the slime mold-algorithm, is an algorithm which shows potential to solve the network design problem. However, current versions of the slime mold-algorithm make networks which are not complex enough. Cities often have a very limited amount of connections with other cities, which is usually not the case in real life, especially in countries with relatively high population density. Also, with the regular slime mold-algorithm, remote cities often have no connections with the created network. Because of this, slime mold-algorithms do not perform that well on existing transport network-approximation on a larger scale, where a lot of cities are present. This makes the networks created by regular slime mold-algorithms unsuitable for network design. In this paper, we try to fix this problem, by improving the network approximation-capabilities of the slime moldalgorithm. By improving the network approximation-capabilities, we also improve the network designcapabilities of the slime mold-algorithm.

We try to achieve the improvements in network approximation-capabilities of the slime mold-algorithm by adding target-orientation to Jones' particle-based slime mold-algorithm. We investigate the performance of this modification on transport network-approximation. Normally, particles in Jones' particle-based model do not have any destinations. In our adaptation, we provide the particles with destinations, hoping it will approximate real-life traffic. In real life, transport networks will be utilized by some type of vehicle, such as motorized vehicles or trains. All of these individual vehicles have a destination, which lies somewhere on the network. Since Jones' model makes use of particles, we can take these particles as a sample of real-life traffic. This way, we can still ensure the rapidness of the original model. We will compare our target-oriented particle-based Physarum model with Jones' original model. Can the use of target-orientation for particles in particle-based slime mold-algorithms improve the creation of efficient transport networks? By comparing the performance of the standard model and the model with targets on approximating real-life transport networks, we are hoping to find that the target-oriented particle-based model is better in transport network design.

2 JONES' PHYSARUM MODEL

This section covers the essentials of Jones' particle-based Physarum model (Jones, 2011), on which our targetoriented model is based. Jones' Physarum model is based on an existing organism, *Physarum Polycephalum*. This organism is a slime mold and it forms paths between food spots. This behavior can be used to approximate existing road networks (Adamatazky, Lees, & Sloot, 2013). Jones' Physarum model tries to approximate the complex network formation and evolution seen in this organism. The model is particle-based, meaning that it consists out of multiple particles. Every particle has its own sensory stage and movement stage. Each particle drops pheromone, which influences decisions in the sensory stage of every particle. We make the pheromone visible on a pre-defined scale, meaning that places with lots of pheromone will light up, and places with little pheromone will not. The result of running the algorithm works. The first subsection, 'Initiation', covers how Jones' Physarum model is initialized. The second subsection, 'Sensory stage', covers how each individual particle locates pheromone and chooses its direction. The third subsection, 'Movement stage', covers how each particle moves after choosing a direction in the sensory stage. The fourth subsection, 'Parameters', covers the parameters of the model, and how they are involved in updating the environment.

2.1 INITIATION

As stated before, Jones' particle-based model consists of multiple particles that behave on their own. From now on, we will refer to these particles as ants. Each ant is able to drop pheromone, which attracts other ants. This will lead to high attractiveness of paths with a high number of ants. Pheromone can be stacked, meaning that pheromone can be increased any number of times by all ants. Every ant moves on a field, which in our case consists out of patches. A patch is a small square area, which has customizable size. Every patch has its own pheromone value. Ants move in-between food points. Food points are spots on the field which resemble food in real life. The patches underneath the food points have a pheromone-multiplier, meaning that every amount of pheromone is multiplied by a parameter, which we will call the Pheromone-intensifier (pherIn). The amount of patches that multiply is affected by the size of the food point. At initiation, all patches have zero pheromone, including the patches underneath food points. Ants can start in different positions. They can all start in in the center, facing a random direction, or every ant can start on a random food point. There is also a third option, where every ant starts at random coordinates, facing a random direction. On the field, every ant is shown as a small green dot. The amount of ants at the start is determined by coverage, the population is a percentage of the field size. Coverage is noted as %p. Ticks are used to run the simulation. Every tick, each ant goes into the sensory stage, then the movement stage. One tick consists out of running both stages for all ants, and updating the environment, as explained in the 'Parameters'-subsection.

2.2 SENSORY STAGE

In the sensory stage, every ant chooses its heading. Each ant has a set of three antennas, called Front (F), Front Left (FL) and Front Right (FR). Where these antennas are positioned relative to position the ant (C), is determined by the Sensor Angle (SA) and the Sensor Offset Distance (SO). These antennas can be used to measure the amount of pheromone on the patch underneath them. This will help the ant choose a direction to face in the sensory stage. Optionally, a Sensor Width (SW) can be used. Every time the ant tries to decide its next position, it moves through the sensory stage once. It checks the amount of pheromone at each antenna, and either keeps facing the same direction, rotate left by its Rotation Angle (RA), or rotates right by its Rotation Angle. Just like the SA and SO, the RA is a parameter that can be customized. The way each ant chooses a direction, can be seen in Figure 2.1. F, FL and FR also represent the amount of pheromone under their corresponding sensors.



Figure 2.1: Particle sensory behavior and particle morphology (Jones, 2011 p. 1349)

2.3 MOVEMENT STAGE

After the sensory stage, the ant goes into the movement stage. This stage moves the ant forward by Step Size (SS), a parameter which can be customized. When the ant is moved forward, it lands somewhere on a patch. On this patch, pheromone is deposited, which is determined by depT, a parameter for pheromone deposition. Optionally, co-location can be added. When co-location is not allowed, no more than one ant is allowed on a patch. Whenever an ant wants to move forward, but tries to move to a patch which already has an ant on it, it will not move forward. Instead, it will face a random direction, and do nothing for the rest of the tick. It will also not deposit any pheromone. The next tick, it will go through all the stages again. When co-location is allowed, this notion does not play a role, and more than one ant is allowed per patch.

2.4 PARAMETERS

Additionally, there are parameters involved in the simulation other than the parameters explained above. These parameters play a role in updating the environment, which consists out of the field with all of its patches. Each tick, the environment is updated. Two parameters are used for this, the diffusion factor (diffK) and the damping factor (dampT). Pheromone can spread to neighboring patches, by an amount determined by diffK. Also, pheromone can evaporate, which will make a portion of the pheromone on all patches disappear. The amount by which it evaporates is determined by dampT. Each tick, pheromone is diffused and evaporated.

Also optional is the addition of barricades in the model. Several adaptations on Jones' Physarum model have used this notion. Ants cannot land on patches marked as barricades, and will try to avoid them by facing opposite directions. Our model will also use this notion, since barricades can be used to draw the outlines of countries. It can also be used to add rivers and other non-traversable objects.

Our adaptation to Jones' Physarum particle-based model will be presented in the next section. This adaptation uses all concepts described above.

3 METHODS

In this section, our adaptation to Jones' Physarum model will be presented. After this, our methods of comparing Jones' Physarum model to the target-oriented model will be presented.

3.1 Adaptations to Jones' Physarum model

In our adaptation to Jones' Physarum model, targets will be introduced. This modification is made to better approach existing transport networks. This makes sense, since a Physarum-algorithm using ants can be seen as a very simplified model of real traffic. Ants can serve as (simplified) cars, and food spots can serve as cities or other destinations. Ants move between food spots, in a similar fashion to cars moving between cities. This is also why both biological Physarum and Physarum-algorithms are used to approach existing transportation networks. The existing Physarum-algorithm by Jones is simple. By adding an extra layer of realism to this algorithm, we might get better results in approaching existing transportation networks.

In order to get this extra layer of realism, targets will be introduced in our modification to Jones' Physarum model. In real traffic, each car or other vehicle has a destination, which is something we can also give to ants. This way, ants can be seen as a sample of existing traffic. In real traffic within a county, hundreds of thousands to millions of cars utilize the road network to get from A to B. In our model, hundreds or thousands of ants will do the same thing. In order to make this modification as simple as possible, we give all ants targets. A target is one of the food spots in our environment. All targets are randomly assigned. Whenever an ant reaches a target, another target is given. This target is always different from the previous target.

Since ants now have targets, this extra information can be used in traversing existing paths and making new paths. Each ant knows the position of the target relative to itself. It knows in which heading its target is. Using this information, we increase the sensitivity of the antenna for this ant relative to the position of its target. This is done using the positioning of the antennas of each ant. The modified sensory stage can be seen in Figure 3.1. If the target lies between FL and FR of the ant, the antenna for F will have increased sensitivity, meaning that pheromone is perceived TSF (TargetScaleFactor) times stronger. When the target lies between FL and the back of the ant (when F is at 0°, the back is at exactly 180°), the antenna for FL will have increased sensitivity. When the target lies between FR and the back of the ant, the antenna for FR will have increased sensitivity.



Figure 3.1: The sensory stage of our modified Physarum model is the same as that of the regular model, but F, FL or FR changes according to where the target is relative to the ant. The cones extend to the whole field. When the field is a torus, the ants take the direction with the shortest path to the target. Here, TSF is the Target Scale Factor.

Only existing pheromone is multiplied. There is no addition of pheromone. So, when there is absolutely no pheromone, it cannot be multiplied. This makes sense, since the regular Physarum-algorithm already tends to explore by itself, without providing aid for it. Also, adding pheromone instead of multiplying existing pheromone would cause the ants to sometimes walk straight to their targets, especially after initialization. This is undesirable, because it goes against the basic principles of regular Physarum-algorithms.

3.2 COMPARING THE MODELS

To analyze the modification on the Physarum Polycephalum-model explained earlier, several scenarios will be implemented in NetLogo. The first step is to implement the regular particle-based Physarum Polycephalum-model based on earlier work from Jones. The modification is added to this model. Performance of both models can easily be compared, using several criteria and scenarios, which will be explained later. The model which uses food points as targets will be referred to as *the Physarum target-model* or *target model*.

ABSTRACT ENVIRONMENTS

In the first scenario, both models will be compared in some generic situations, using visual inspection. We will call these generic situations 'abstract environments'. These abstract environments can be seen in Figure 3.2. We will use these scenarios to show some changes in behavior between the regular model and the target model. By using simple scenarios, these behavioral changes will be clearly visible. By running both the regular model and the target model and the target model the same amount of ticks, with the same parameter-settings, we can compare the results of both models. The method which we will use to create a network is the filamentous condensation method, presented by Jones. In this method, a relatively small population is initialized, %p=2%, and all the ants start on random coordinates, facing random directions. Using the right parameters, we can get a network with characteristics of a Steiner tree. In filamentous condensation, we choose SA to have the same value as RA. We also do not allow co-location.



Figure 3.2: Three generic scenarios used to compare the behavior of the Physarum Polycephalum-model and the Physarum target-model. Food spots are shown as red dots.

EXISTING TRANSPORT NETWORKS

Since the Physarum target-model is designed to be a better approach to real-world traffic, it will be compared to existing transport networks. Connections between major cities, mainly highways, will be compared to the connections made by the regular Physarum Polycephalum-model and the connections made by the Physarum target-model. We look at the connections made in both models, and we compare them using the same criteria for both models. By using the exact same criteria to evaluate our connections, we hope to eliminate the arbitrary factor involved in visual inspection. In order to use the models on existing combinations of cities, existing borders and existing waterways, maps of the area will be converted to a special environment, which the models can traverse. Barriers, which the individual ants cannot traverse, represent borders and waterways. Cities are represented as food points. In the Physarum target-model, the cities serve as targets for the individual ants. Our models will be run in NetLogo, a multi-agent programmable modeling environment.



Figure 3.3: Top row: Maps of the Netherlands and Poland generated from existing maps. Bottom row: Environments based on these maps, suitable for both Physarum-models to traverse.

By choosing two countries to analyze, we can easily compare existing transportation networks with results produced by our two models. The first country that was chosen for this analysis was the Netherlands. With its large density of cities in the west (Randstad), and its lower density in the east, the Netherlands makes for an interesting choice, since the Physarum target-model could solve some difficulties which the regular Physarum Polycephalum-model faces, such as not making connections from and to isolated cities. The transportation network that was analyzed was the railroad network. This network is a lot less dense than the Dutch road network, so analyzing the performance of both models will be a lot easier. The second country that was chosen was Poland. With its pretty evenly distributed cities and its advanced planned highway network, Poland could easily show the differences the Physarum target-model makes on transport network-approximation in comparison to the regular Physarum Polycephalum-model. The transportation network used here is Poland's motorway network, with roads known as the A-roads.

In order to compare the performance of the two models on approximating existing networks, the existing connections between a manually chosen set of cities will be noted. For the Netherlands, the 15 most populous cities were chosen according to the population of their municipalities in May 2018, plus province capitals 's-Hertogenbosch, Zwolle, Maastricht and Leeuwarden (province capitals up to the 25th most populous cities). For Poland, the 14 most populous cities were chosen, plus province capitals (both governor and assembly) Toruń, Kielce, Rzeszów, Olsztyn and Zielona Góra (province capitals up to the 25th most populous cities). The cities that were chosen for both countries can be seen in the appendix of this paper. The environments that were generated using these sets of cities can be seen in Figure 3.3. To make the existing (or in the case of Poland, planned) connections between the chosen cities clearer, the network will be shown as a graph. The same graphs will be made from the results of running the models on our created environments, making it easy to compare them. These graphs can be seen in Figure 4.3 and 4.6 (in the 'results'-section) as (0). Magnified versions can be seen in the Appendix of this paper. Performance of the models will be graded in the amount of existing connections they created, and the amount of connections which are correct, when compared to the original network.

GRAPH SETUP

The graphs are made by looking at existing (or in the case of Poland, planned) connections between the cities that will be used in the environments we created for this experiment. When it is possible to get from A to B, without passing C, a connection between A and B is made in the graph. When it is not possible to get from A to B, without passing C, a connection between A and C and C and B is made. Using this method, graphs for both Poland and the Netherlands are created. Again, magnified versions of these graphs can be found in the appendix, but will also be shown in the results section. When two connections in the graph cross, it can be assumed that these two connections cross in the existing network, creating a connection between the two points. For the Dutch train network, connections exist when they are the fastest possible connection. The fastest connection from Dordrecht to 's-Hertogenbosch is via Breda and Tilburg, but there is another way, without crossing any of the cities that were chosen in our network. However, this way is slower than the first one, and thus no connection is made. This is why the connection 's-Hertogenbosch-Dordrecht does not exist, and connections such as Apeldoorn-Arnhem do exist. Also, when two connections in the railway network cross, it can be assumed that these connections are connected in the existing network.

RUNNING CRITERIA

We will use both the filamentous condensation method and the filamentous foraging method, both as presented by Jones. In filamentous foraging, we use a relatively small population, %p=2%, in which all ants start from the food sources. In this method, RA will have a fixed value, but SA will vary. We will start with a low value for SA, and we increase it after fine branching networks have formed. After this, we set SA to be the same as RA. With our target model, this will also be the moment we give all ants their targets. Before changing the SA, the model will behave the same way as the regular model. For both models, we run the algorithm for the same amount of ticks. We do not look for a completely converged (or minimized) network per se, since by stopping the algorithms after a fixed amount of ticks, we can also say something about how quickly both models converge, if they do. Also, a fully converged network does not necessarily give us the best approximation. Especially with the regular model, the algorithm might decide to skip a few isolated points. Because of these characteristics, we stop the algorithms after a fixed amount of ticks. We also do not allow co-location.

4 RESULTS

This section will cover the results of comparing the models using the methods discussed in the previous section. The first subsection will cover the results obtained by running both models on our abstract environments. The second subsection will cover the results obtained by running both models on maps of the Netherlands, comparing them to the existing rail network. The third subsection will cover the results obtained by running both models on maps of Poland, comparing them to the planned highway network.

4.1 ABSTRACT ENVIRONMENTS

Using the abstract environments, we can see some core behavioral changes between the regular model and the target model. We have 3 distinct abstract environments, which are labeled (a), (b) and (c). Parameter-settings are the same for all three environments. The letter following the label for the environment stands for the model that was used, where 'R' stands for the regular particle-based model, and 'T' stands for the same model with target orientation. The results can be found in Figure 4.1.

Using the parameter-settings seen in the description of Figure 4.1, we can see some clear visual differences. Most notable is the difference between (aR) and (aT). (aR) is the result one would expect from running Jones' particlebased model on this type of abstract environment, and (aT) gives us a cross, similar to (bT). In (bR) and (cR), we can see that some obvious connections are missing. This might change with different parameter-settings, but the same parameter-settings were used in (bT) and (cT), where these connections do exist. All in all, when compared to the regular model, the target model shows slight changes in behavior. These changes might be useful in approximating existing transportation networks.



Figure 4.1: Top row: Results of running Jones' particle-based model on the abstract environments. Bottom row: Results of running the target model on the same abstract environments. The following parameter-settings were used for both models and all three environments: SS 2, SO 10, SA 45, RA 45, depT 15, dampT 0.4, diffK 0.04, pherIn 20, TSF 5 (for the target model). For both models, environment (a), (b) and (c) were run for 8000, 14000 and 20000 ticks respectively.

4.2 THE NETHERLANDS

In this section, we present the results of both models on a map of the Netherlands, with the 15 most populous cities according to the population of their municipalities in May 2018, plus province capitals 's-Hertogenbosch, Zwolle, Maastricht and Leeuwarden (province capitals up to the 25th most populous cities). We will show the results of both models, using both the filamentous condensation and filamentous foraging methods, described in the methods-section of this paper. We will also present the graphs corresponding to the networks made using our algorithms. These graphs will be shown together with the graph corresponding to the existing railroad network in the Netherlands, which has been adapted to only show the connections between our chosen set of cities. All graphs can be found in the appendix, in higher quality and increased size. Also, a table will be provided, showing some statistics of the created graphs.

From the images in Figure 4.2 on the next page, we can immediately see that the target model creates a more vast and complex network, and also seems to be in a more advanced stage of convergence after the same amount of ticks. While the regular model skips remote food sources with these parameters, our target model does not. The behavior shown here is similar to the behavior found in the abstract environments. The same results can be seen in the graphs created from these images, shown in Figure 4.3 on the next page. In filamentous condensation, our target model (aT) creates more vast networks, closer to the real-life network shown in (0). Using the same method, the regular model (aR) skips a few food sources after 8000 ticks, and the target model does not. The same can be seen when comparing both models after using filamentous foraging. The regular model (bR) creates a simple and less complex network, when compared to the target model (bT). Again, with (bT), all points are interconnected at this timestamp in our target model, which gives us a network which is more closely related to (0). Many of the connections seen in graph (0), reappear in the graphs based off networks created by our models.



Figure 4.2: (aR) & (aT): regular model and target model using filamentous condensation, stopped after 8000 ticks, SS 2, SO 10, SA 45, RA 45, depT 15, dampT 0.4, diffK 0.02, pherIn 20, TSF 5 (for the target model). (bR) & (bT): regular model and target model using filamentous foraging, stopped after 20000 ticks, SS 2, SO 10, SA 22.5 increased to 45 after 8000 ticks, RA 45, depT 20, dampT 0.4, diffK 0.02, pherIn 20, TSF 5 (for the target model).



Figure 4.3: (0): The graph created from the existing rail network in the Netherlands. (aR), (aT) (regular model and target model after 8000 ticks, using filamentous condensation), (bR), (bT) (regular model and target model after 20000 ticks, using filamentous foraging): graphs created from the results shown in images from Figure 4.2.

The performance of both models using both methods can be found in Table 4.1 on the next page. We can see that the target model outperforms the regular model in both the total connections made and the correct connections made. A correct connections is a connection which can also be seen in graph (0). An incorrect connection is a connection which cannot be seen in graph (0). The target model with filamentous condensation shows a few incorrect connections, but it also made more connections in the first place.

	(0)	(aR)	(aT)	(bR)	(bT)
Total	34	19	28	19	23
connections					
Correct	-	19	22	17	23
connections					
Incorrect	-	0	4	2	0
connections					

Table 4.1: Performance of the models, where correct connections are the number of connections which can be found in graph (0), and incorrect connections are the number of connections which cannot be found in graph (0). When a connection between two cities exists through a line, it is added to the amount of total connections.

4.3 POLAND

In this section, we present the results of both models on a map of Poland, with the 14 most populous cities, plus province capitals (both governor and assembly) Toruń, Kielce, Rzeszów, Olsztyn and Zielona Góra (province capitals up to the 25th most populous cities). Again, we will show the results of both models, using both filamentous condensation and filamentous foraging, as described in the methods-section of this paper. We will again present the graphs corresponding to the networks made using our algorithms. These graphs will be shown together with the graph corresponding to the planned highway network in Poland, which has been adapted to only show the connections between our chosen cities. All graphs can be found in the appendix, in higher quality and increased size.



Figure 4.4: (aR) & (aT): regular model and target model using filamentous condensation, stopped after 8000 ticks, SS 2, SO 10, SA 45, RA 45, depT 15, dampT 0.4, diffK 0.02, pherIn 20, TSF 5 (for the target model). (bR) & (bT): regular model and target model using filamentous foraging, stopped after 40000 ticks, SS 2, SO 10, SA 15 increased to 45 after 10000 ticks, RA 45, depT 20, dampT 0.4, diffK 0.02, pherIn 20, TSF 5 (for the target model).

From the images in Figure 4.4, we can see that the usage of targets gives us a far more advanced network when stopping at 8000 ticks, with filamentous condensation. We can see that the target model is in a more advanced stage of converging, and less remote food sources are skipped in the target model (aT), when compared to the regular model (aR). These results are similar to the results found when running the algorithms for the environment representing the Netherlands. The same can be found when using filamentous foraging, which has converged after 40000 ticks. In the regular model (bR), a few remote food sources are skipped once again. The target model (bT) does not show this behavior, and connects even the most remote food sources. Overall, the network created by the target model is more complex, and is more related to the real-life network. In the graphs from Figure 4.5 on the next page, the same results show. We can clearly see the differences in complexity of the created networks, and the total absence of connections for remote food sources in the regular model. Many of the connections seen in (0), reappear in the networks created by our models.





Figure 4.5: (0): The graph created from the planned highway network of Poland. (aR), (aT) (regular model and target model after 8000 ticks, using filamentous condensation), (bR), (bT) (regular model and target model after 40000 ticks, using filamentous foraging): graphs created from the results shown in images from Figure 4.4.

From Table 4.2, we can see that the target model outperforms the regular model, the same way it did for the Netherlands. For both methods, the target model has more total connections, creating a more advanced network. It also has more correct connections, creating networks which are closer related to the real-life transport network.

	(0)	(aR)	(aT)	(bR)	(bT)
Total	42	29	34	22	26
connections					
Correct	-	26	32	19	24
connections					
Incorrect	-	3	2	3	2
connections					

Table 4.2: Performance of the models, where correct connections are the number of connections which can be found in graph (0), and incorrect connections are the number of connections which cannot be found in graph (0). When a connection between two cities exists through a line, it is added to the amount of total connections.

5 DISCUSSION

In this paper, we provided an adaptation to Jones' Physarum model. Jones' Physarum model is an algorithm based on a slime mold, Physarum Polycephalum. This organism has shown to be useful in approximating real-life transport networks. Jones' Physarum model shows the same properties as the organism it is based on. To further improve the quality of transport network approximation, we introduce the notion of targets to Jones' Physarum

model, creating a target-oriented particle-based model of Physarum Polycephalum. Since real-life transport networks have shown to be solutions to the 'Network design problem', an algorithm which approximates reallife transport networks well can also be used to create efficient networks. With our adaptation, we hoped to improve the way Jones' Physarum algorithm creates networks, hoping they will become more efficient and realistic. To test the performance of our adaptation, we compared the results of Jones' Physarum model and our so-called target model on approximating real-life transport networks.

Transport networks in both the Netherlands and Poland were compared. Using the exact same parameters and using both filamentous condensation and filamentous foraging, we could see an increase in the total amount of connections made with our target model. This created far more advanced networks, closer to those seen in real life. Our target model also creates more connections which also exist in the real-life transport network, when compared to the regular model. We can also see that our target model has a less likely chance of making no connections to and from a food point at all, when using the same parameters as the regular model. All these improvements suggest that the notion of targets does increase the quality of approximating real-life transport networks. Thus, we can say that our target model increases the quality of creating efficient transport networks.

We have shown that our adaptation improves the complexity of the resulting networks, and remote cities are often connected with the created network. This increase in quality improves the performance of the particlebased slime mold-algorithm in solving the network design problem. With further improvements, this problem might be completely solvable by this Artificial Intelligence-algorithm. If network design can be partially or fully automated, many manhours could be spared, and many costs would vanish. There is still a lot of potential in this field, and with further improvements, the slime mold-algorithm could become a solution to one of the most difficult problems in transport.

To further improve the network design-capabilities of the particle-based slime mold-algorithm, more notions could be added in the future, such as the notion of terrain. We could also add the notion of city size. It would be possible to increase the size of a food source proportional to the population it represents. When using targets, we could make target assignment less random and make it proportional to the size of the food point. The bigger a food point, the more likely it is to be a target for an individual particle. Using targets, we could also implement the notion of ring roads (or beltways). It would be possible to disallow ants to step on non-target food points, which would make them go around them. We could also add more cities to increase the realism of our networks. In this paper, we used simplified networks, created by taking connections to and from the cities we have chosen. However, some cities which could have had an influence on the creation of this transport network, might not have been chosen. This would make it impossible for our algorithm to make the exact same decisions in network design. By increasing the amount of cities (usually based on population), we would decrease the chances of there being a city which had an effect on the network design, but was not picked in our simplified version of the transport network. We could also create more realistic networks by adding some outside connections. Especially in the European Union, where many large highways and railroads cross many borders, this would increase the realism of created networks, and thus improving network design.

All in all, our adaptation to Jones' Physarum model yields a lot of further research. This could all be used to increase the efficiency and realism of networks created by Physarum-algorithms. This could have many uses, since partially or fully automating the process of network design could be very useful, especially in an era where millions of kilometers of roads and rail are traversed each day. The worldwide transport network is expanded every single day, which calls for algorithms capable of improving network design. With our model, we increased the network design-capabilities of particle-based Physarum-algorithms, hoping that one day, we might be driving on a transport network which was designed by a very advanced version of a target-oriented particle-based model of Physarum Polycephalum.

REFERENCES

- Adamatazky, A., Lees, M., & Sloot, P. (2013). Bio-development of motorway network in the Netherlands: a slime mould approach. *Advances in Complex Systems*, 2013, *16*(02n03), 1250034.
- CBS. (2018). Regionale kerncijfers Nederland. Retrieved from CBS Statline: https://opendata.cbs.nl/statline/#/CBS/nl/dataset/70072ned/table?ts=1543245923598
- Central Intelligence Agency. (2013). The World Factbook: Roadways. Retrieved from Central Intelligence Agency: https://www.cia.gov/library/publications/the-world-factbook/fields/2085.html
- GDDKiA. (n.d.). Mapa Stanu Budowy Dróg. Retrieved from GDDKiA.gov.pl: https://www.gddkia.gov.pl/pl/1077/mapa-stanu-budowy-drog
- Główny Urząd Statystyczny. (2018). Wyniki badań bieżących . Retrieved from Główny Urząd Statystyczny: http://demografia.stat.gov.pl/bazademografia/Tables.aspx
- Jones, J. (2011). Influences on the formation and evolution of Physarum polycephalum inspired emergent transport networks. *Natural Computing*, *10*(4), 1345-1369.
- Magnanti, T. L., & Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation science*, *18*(1), 1-55.
- Nederlandse Spoorwegen. (2018). Spoorkaart 2019. Retrieved from NS.nl: https://www.ns.nl/binaries/_ht_1541689522035/content/assets/ns-nl/dienstregeling/nieuwedienstregeling/spoorkaart-2019.pdf
- The World Bank. (2016). Rail lines (total route-km). Retrieved from The World Bank: https://data.worldbank.org/indicator/IS.RRS.TOTL.KM
- van Dorsten, S. (2015). Permeability of terrain in particle-based models of Physarum Polycephalum (Bachelor's thesis).
- Yang, H., & Bell, M. (1998). Models and algorithms for road network design: a review and some new developments. *Transport Reviews*, *18*(3), 257-278.
- Zhang, X., Wang, Q., Adamatzky, A., Chan, F. T., Mahadevan, S., & Deng, Y. (2014). An improved physarum polycephalum algorithm for the shortest path problem. *The Scientific World Journal, 2014*.

APPENDIX

1. COUNTRIES

The following countries were used in the experiments, with the following cities.

A. THE NETHERLANDS

For the Netherlands, the 15 most populous cities were chosen according to the population of their municipalities in May 2018, plus province capitals 's-Hertogenbosch, Zwolle, Maastricht and Leeuwarden (province capitals up to the 25th most populous cities). These statistics were taken from CBS, known as Statistics Netherlands outside of the Netherlands (CBS, 2018).



- 5. Arnhem
- 6. Breda
- 7. Den Haag

- 12. 's-Hertogenbosch
- 13. Leeuwarden
- 14. Maastricht

19. Zwolle

B. POLAND

For Poland, the 14 most populous cities were chosen, plus province capitals (both governor and assembly) Toruń, Kielce, Rzeszów, Olsztyn and Zielona Góra (province capitals up to the 25th most populous cities), according to data from GUS, known outside of Poland as the Central Statistical Office of Poland (Główny Urząd Statystyczny, 2018).



- 1. Białystok
- 2. Bydgoszcz
- 3. Częstochowa
- 4. Gdańsk
- 5. Gdynia
- 6. Katowice
- 7. Kraków
- 8. Łódź
- 9. Lublin

- 10. Olsztyn
- 11. Poznań
- 12. Radom
- 13. Rzeszów
- 14. Szczecin
- 15. Toruń
- 16. Warsaw
- 17. Wrocław
- 18. Zielona Góra

2. GRAPHS

The following pages are dedicated to the graphs created from existing road networks, as well as the ones created from our two models. These graphs can also be found in the results-section, but they are magnified here.

A. THE NETHERLANDS

This graph was created using information from Nederlandse Spoorwegen, also known as NS (Nederlandse Spoorwegen, 2018). Shown here are all existing connections via train routes between the chosen cities, excluding high-speed lines.



Figure 4.3, graph (0)



Figure 4.3, graph (aR)



Figure 4.3, graph (aT)



Figure 4.3, graph (bR)



Figure 4.3, graph (bT)

B. POLAND

This graph was created using information from the Polish Motorways Authority, also known as the GDDKiA (GDDKiA, sd). Shown here are all planned and constructed connections via motorways between the chosen cities, known in Poland as A-roads.



Figure 4.5, graph (0)



Figure 4.5, graph (aR)



Figure 4.5, graph (aT)



Figure 4.5, graph (bR)



Figure 4.5, graph (bT)

3. CODE REPOSITORY

The code used for the experiments in this paper can be found in a Github code repository, which is accessible with the following link: <u>https://github.com/FSSlijkhuis/Target-oriented-slime-mold/</u>

Our model uses NetLogo 6.0.4, a multi-agent programmable modeling environment. More information on NetLogo and its dictionary can be found here: <u>https://ccl.northwestern.edu/netlogo/</u>

In the repository, the following files are present:

- 'Target-oriented slime mold by F.S. Slijkhuis.nlogo': This is the main file used in the experiments. This file features the code and UI for the model. This file also features a description on how to use the model, found under the 'Info'-tab. This file can be opened using NetLogo 6.0.4, which can be downloaded on the website provided above.
- A Readme-file: This file describes the contents of the repository.
- A 'Scenarios'-folder: In here, some of the essential scenarios used in the model are present. How to import them is provided in the description of the .nlogo-file.
- An 'Image'-folder: In here the images used for the experiments are present. How to import them is provided in the description of the .nlogo-file.

For more information on the model, and additional files, please contact the author at <u>filip8250@gmail.com</u>. If the previously provided link does not work, please do not hesitate to contact me as well.