

Swarm-Based Edge Bundling Applied to Flow Mapping

Evgheni Polisciuc^(✉) and Penousal Machado

CISUC, Department of Informatics Engineering,
University of Coimbra, Coimbra, Portugal
{evgheni,machado}@dei.uc.pt

Abstract. Applying flow maps in large datasets involves dealing with the reduction of visual cluttering. Nowadays, a technique known as edge bundling, which is geometric in nature, is often applied to reduce visual clutter and create meaningful traces that highlight the main streams of flow. This article presents an alternative approach of edge bundling for generating flow maps. Our approach uses a swarm-based system to reduce visual clutter, bundling edges in an organic fashion and improving clarity. The method takes into account the properties of data, edges and nodes, to bundle edges in a meaningful way while tracing lines that do not interfere visually with the nodes. Additionally, the Dorling cartograms technique is applied to reduce overlapping of graphical elements that render locations in geographic space. The method is demonstrated with application in the analysis of the US migration flow and transportation of products among warehouses and supermarkets of a major retail company in Portugal.

Keywords: Flow map · Edge bundling · Origin-destination map · Thematic map · Flow visualization · Geovisualization

1 Introduction

Flow mapping is a technique used to show the movement of objects from one location to another, such as the migration of people, the amount of goods being traded, the amounts of products being transported from warehouses to supermarkets, etc. Flow maps say little or nothing about the pathway, but include the information about what is flowing (moving, migrating, etc.), the direction of the flow, and how much is being transferred. In most cases, the data is represented using line width, color and spatial properties. Flow maps are advantageous in what regards to the visual clarity and ease of visual communication. This is achieved by merging edges that share similar destinations, or in some cases by tracing them through a similar path. However, this technique often fails when applied on large amounts of flow data. The visualization might become cluttered, making the map difficult to read, and difficult to distinguish the grouped individual streams.

In this work we describe a method for the generation of flow maps that is able to depict large amounts of transitions from one location to another (further expressed as *Origin-Destination* or simply *OD*). This method uses a customized swarming system to trace edges in an intuitive and organic fashion, reducing visual clutter. In particular, our system adapts the steering behaviors algorithm of autonomous agents presented by Reynolds [1], and the mechanism of indirect communication known as stigmergy. The agents of the systems communicate with each other through the environment, more precisely, the traces left by agents are used to stimulate the behavior of other agents. The method described in this article takes into account edge properties, such as directionality and the data attributes. Also, our method accounts for the presence of nodes and their properties. By routing the traces that avoid visual interference with nodes, the visualization provides clear distinction between traces and nodes. Additionally, our method employs graphic design decisions to promote clarity of visual communication in a high density environment. In order to improve clarity of representation of geographic locations, a technique, known as Dorling cartograms [2] was applied. With this technique, overlapped points were separated retaining some degree of spatial relationship. Finally, our approach supports mixed types of points – geo-referenced data points and those that have no fixed position in space. An example of the former are supermarkets with known location and of the later are warehouses with unknown geographic positions.

With that said, this work tackles the issue of depicting large amounts of products being transported from warehouses to hyper and supermarkets of a major retail company in Portugal. Our dataset is consisted of warehouse-to-supermarket transitions, over a time span of 6 months. The locations consist of approximately 60 warehouses, the majority of which located outside of Portugal and 1039 supermarkets in Portugal and 230 outside Portugal, from where only 680 are geo-located. The final graph used in this work, nonetheless comprises only the warehouses and supermarkets that are located on the continental part of Portugal.

In the following sections our approach is described in more detail. Section 3 presents the underlying idea and the method in detail. Section 4 describes an application of this technique on the available dataset. Finally, Sect. 5 presents a comparison of the results obtained by our approach and the existing one.

2 Background and Related Work

In the context of thematic mapping, an Origin-Destination representation commonly refers to the flow visualization (also known as flow maps), which is deeply rooted in the history of information visualization. Early examples, such as wine exports from France, produced by Minard [3, p. 25], represent the quantity as well as the direction of wine exports encoded by the thickness of the corresponding edges, which split from the parent edge when needed (Fig. 1). The work of Phan et al. [4] presents an automated approach to generate flow maps using a hierarchical clustering algorithm, given a series of nodes and a flow data. Generally, in the geographic context, a flow map depicts quantities of any type of

objects that move from one location to another – e.g. migrations, transportation of goods, etc. The advantage of flow maps is that they reduce visual clutter by merging edges. However, when representing large amounts of data, this technique presents a series of problems, such as poor perception of the flow directionality, high degrees of visual clutter and overlapping of graphical elements that represent locations.

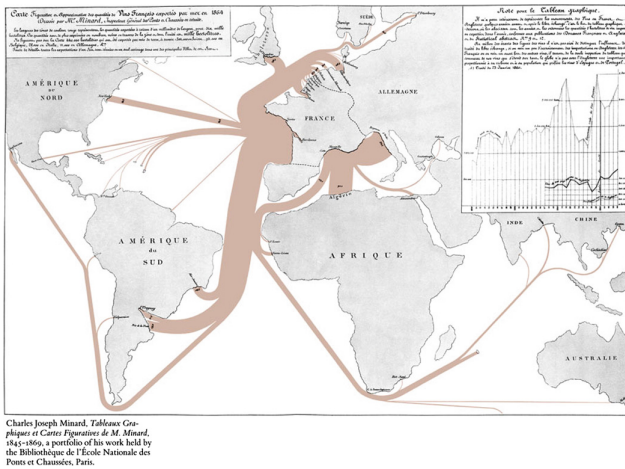


Fig. 1. Export of French wine by Charles Minard, 1864.

Direct visualization of large amounts of Origin-Destination transitions can generate high degrees of visual clutter. In these cases a reduction strategy known as edge bundling can be applied. This is characterized not only by graph simplification, but also by the revelation of the principal streams of flow. Holten introduced edge bundling for compound graphs [5]. His work consisted in routing edges through a hierarchical layout using B-Splines. Nowadays, there are several variations of edge bundling, such as, force-directed edge bundling [6] or sophisticated kernel density estimation strategies [7]. More recently, approaches that account for edge properties have changed edge bundling into a meaningful, driven by data, edge tracing [8]. However, the way existing bundling techniques trace edges tends to be purely geometric in nature. Generally, edge bundling consists of drawing similar edges on the same path, i.e., edges that are related in geometry are routed along the same path. Ultimately, existing techniques tend to ignore nodes of the graph, or explicitly do not take them into account, which can result in a visual interfere of traces with nodes.

Another important characteristic of this work is the focus on nature-inspired approaches. The underlying idea is based on self-organizing systems, more precisely on the phenomenon of emergence in such systems. As the term indicates, self-organization is a property of some complex systems, in which the structure or organization appears without interference from external sources, typically

resulting from a decentralized approach. Thus, self-organizing processes often result in the occurrence of emergent phenomena. More precisely, when the complex structure or behavior appears due to the simple low level interactions of a collection of individuals. There are several mechanisms of interaction among individuals, but the one that is used in this work is stigmergy, which consists of communication through the environment [9]. In the field of data visualization, there are techniques of graphical representation that are based on such systems. For instance *Geoboids* [10] employs a method to reveal patterns in spatial data through the use of a customized flocking system. In this system, each, so called geoboid explores the geographic space in accordance with the simple rules of interaction with other geoboids and the data found nearby. The visualization, which emerges from this simple process shows areas containing interesting information. Another series of works by Vande Moere exploit self-organization and emergence in information visualization. He introduces the idea of *infoticle*, which designates a particle that responds to data values and static forces in a particle system [11]. The visual output portrays the Internet file usage of a medium-sized company over time, conveying the patterns of file downloads. Another nature-inspired approach is *information flocking visualization* [12]. In this work, Vande Moere uses an artificial flocking system, originally proposed by [13], where the forces of attraction and repulsion are modified proportionally to the similarity between the data objects that each boid encodes. The emergent patterns analyzed at a higher-level, where each composition portrays short-term and long-term data tendencies in a time-varying dataset, convey meaningful changes over time.

3 Flow Map Flocking Model

In order to reflect the flowing nature of the data we resort to a customized flocking system. The underlying model to construct our flow map shares common characteristics with the work of Polisciuc et al. [14]. The idea is to get from point A to point B avoiding obstacles and, when possible, following existing trails that potentially lead to the destination. However, our new approach changes the way agents of the system interact with each other. Basically, the coordination is done through the environment, using stigmergy. Also, action selection rules were changed, in order to achieve meaningful behavior prioritizing obstacle avoidance and trail following.

The visualization itself can be seen as a directed graph composed by nodes and edges. The system consists of artificial agents (further referred to as *vehicles*), each one tracing an individual OD edge. A vehicle is characterized mainly by its position in space, direction and speed. During the simulation each vehicle leaves persistent traces, further referenced to as markers, which inherit the location and direction, at each simulation instant. Since the process is asynchronous, i.e., each trace is computed separately, the vehicle in the system interacts communicate with each other through the environment. While interacting with the markers, each vehicle follows simple rules: *attract* to the compatible markers; *avoid* obstacles, which are the nodes of the graph; arrive at the destination.

In order to determine the relationship between the vehicle and the marker we used a pairwise compatibility measures, including geometric properties and the weight of each one. With that said, this section describes our system in detail, including the strategy for steering behavior selection, compatibility metrics, and the modification of the environment. The output of the system is shown in Fig. 2.

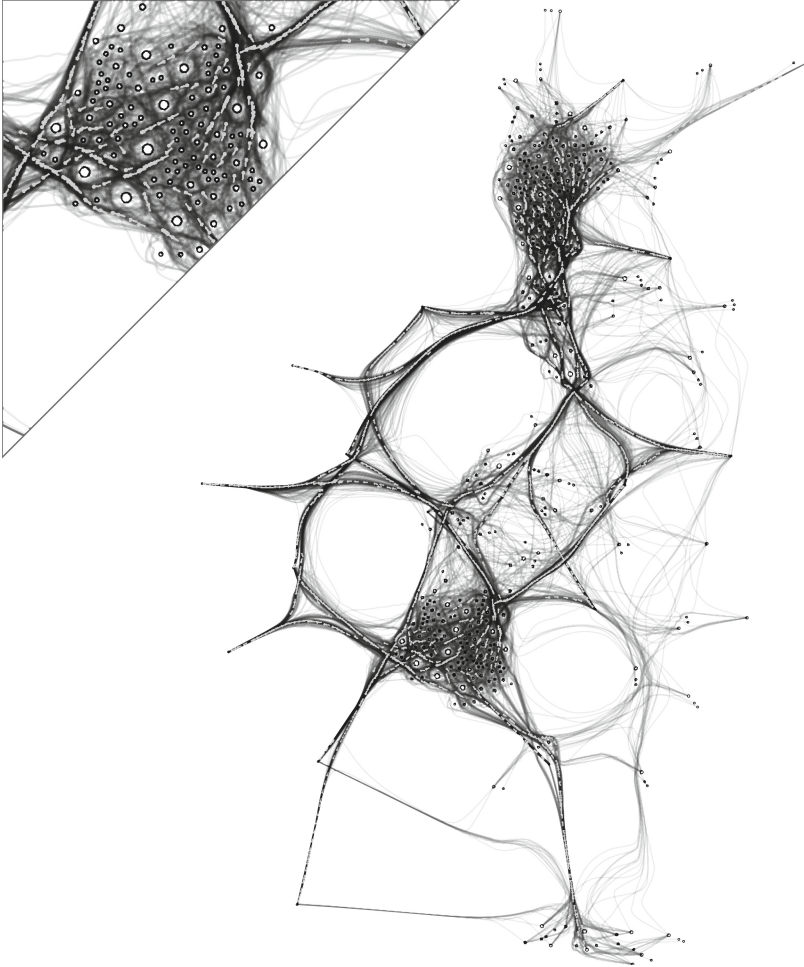


Fig. 2. Visual output from the system after 10 full cycles (bottom). Detail of the computed traces in black (top). Circles with black stroke represent nodes. White points with an arrow are markers.

3.1 Model

As previously mentioned, our system consists of a set of autonomous agents. Following Reynolds' ideas the motion behavior of an autonomous agent is defined

by three layers: action selection, steering and locomotion [1]. Regarding the locomotion we use a simple vehicle model described in his work. The simple vehicle is characterized by a position in space, mass, velocity, maximum speed and maximum force. Using Euler integration the position is altered by adding the velocity, which is modified by applying steering forces that are truncated by the maximum force. Additionally, the velocity is truncated by the maximum speed. Regarding steering we used several behaviors, such as seek, arrive, obstacle avoidance, separation and cohesion. Finally, action selection uses the sensors of the vehicle to define the strategy, setting the goal of the vehicle in response to the environment. For detailed information see the original Reynolds work [1].

In our systems each edge is traced by only one vehicle, starting at the origin node and finishing at the destination node, constantly leaving markers along its trail. Each trace is computed individually, and the computation of each one only starts when the previous one has finished, and so forth for all the traces. In each cycle the traces are updated according to the current state of the environment. More precisely, during the execution cycle each vehicle considers the markers left by other vehicles, and updates them if the specific conditions are met. This interaction determines the behavior of the vehicles. The process repeats until the visual result is acceptable and the user decides to stop it.

The behavior of our vehicles can be better understood by describing each step individually and their integration. There are four essential steps in our system: action selection, steering, locomotion and environment update.

Action Selection. The first step that defines the overall strategy of the vehicle's behavior is action selection. Using the attached sensors, vehicles perceive the environment and decide what action to take. In our model there are three types of sensors: one that detects the presence of obstacles, another that detects the presence of markers, and the last one that measures the similarity between an agent and a marker. The first sensor has the form of a cylinder, with a certain *radius* and *length*, and is aligned with the velocity vector. The length dictates how soon the obstacles will be detected and avoided. In order to create smooth traces the length should use reasonably high values, which depend on the scale of the image. The second sensor, which detects the presence of markers, is characterized by the field of sight, more precisely *angle* and *distance* of vision. The direction of the field of sight is also aligned with the velocity vector. If there are markers found in the field of sight they are considered for similarity comparison. Finally, the third sensor measures the similarity between the current vehicle and the markers in the field of vision.

The strategy of applying steering forces is based on the prioritization of behaviors and consists of simple rules. The very first step is to check if the vehicle is in its arrival distance. In the positive case the only behavior that is applied is arrive to the destination. Otherwise, we apply obstacle avoidance behavior. If the computed steering force is null, then we proceed to the second behavior, which is attraction. The attraction force consists of an average weighted vector pointing towards the markers in the field of view, using the compatibility measure described in the following paragraph. Finally, if the attraction force is null, we

proceed to the calculation of the arrival behavior. In other words, the idea behind the vehicle's behavior is to avoid obstacles if there is the possibility of collision, otherwise it tries to follow existing potential trails that lead to the destination. If there are no trails in the field of vision, it just proceeds in the direction of the destination.

The compatibility measures are composed by the set of rules, which are based on the comparison of angles. In our approach we only consider the angle in the compatibility measure. While other solutions use a geometric approach, and, therefore, there is the need to use some edge compatibility measure to compute bundles, in our case bundles are the emergent output from the simple interaction among agents and their collective behavior. So, given a maximum angle, say θ , for each marker we apply the following rules: first, we calculate the angle between the vector pointing towards the marker and the vector pointing towards the destination node; second, we compute the angle between the vector pointing from the marker towards the vehicle's destination and the marker's direction. If both of these angles are below θ , then the attraction force is calculated and weighted by the marker's weight (Fig. 3). In order to get smooth approximation to the trail, the attraction vector points towards a location a few steps ahead the marker. In other words, the vehicle is attracted to the marker that points in the direction of vehicle's destination ensuring that it will reach its destination.

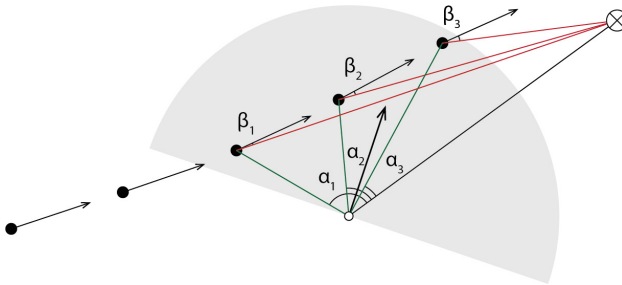


Fig. 3. Schematic representation of the vehicle and markers compatibility. Black circles represent markers. The white circle is a vehicle and the gray area is the field of sight. The white circle with a cross is the destination of vehicle. Arrows represent the direction of markers or velocity of the vehicle. Green lines are vectors pointing from the vehicle's location towards each marker. Red lines are vectors pointing from each marker's location towards the vehicle's destination. Assuming that the maximum allowed angle θ is 90 degrees, marker 1 is not considered for calculation, since the angle α is greater than θ . (Color figure online)

Steering and Locomotion. The steering behaviors used in our system are: arrive, avoid obstacles, and attract. These models and their implementation, as well as locomotion, strictly follow Reynolds ideas. However, there are some additional aspects that were introduced in order to achieve an intelligent edge bundling. In the beginning, the maximum speed limit is zero and increments with time.

This is done to achieve less chaotic behaviors near the origin location. The mass and the distance of vision are modified according to the edge's weight. So, the vehicle's that trace edges with higher weight are heavier and more "blind" than the ones that encode edges with low weights. In this case, the former traces become the central trails that suffer less modification. Finally, the attraction force is weighted according to the weight of the markers being considered in each step. Therefore, vehicles prioritize the trails with higher weights.

Environment Update. The environment of our system shares similar ideas of stigmergy, where vehicles communicate with each other through the environment. In contrast to other algorithms, such as ant colony, where there is one nest and various sources of food, in graphs there are various origins and destinations. In this case, in order to get to a specific destination there is the need to know an approximate indication about the direction of the trail. Therefore, we use the concept of marker, which is characterized not only by the location in space and its evaporation rate, but also by the direction. Each marker inherits some characteristics of the vehicle that deposited it – the location where the marker was left and the direction which corresponds to the velocity vector at that instance. With that said, the trails are updated as follows: with a certain rate, each vehicle leaves one marker; then we search for other markers that are already in the environment within a certain range; if no markers are found, the newly created marker is added to the environment; otherwise, for each found marker we check for compatibility by comparing their directions, and then update the location and the direction of each compatible marker. In this case there is no need to add the marker to the environment, since there is already one that is similar in location and direction. Both location and direction are updated with a certain rate. Otherwise, all the markers would quickly change their position, creating unwanted visual artifacts. The location is modified by pulling the marker towards the location of newly created markers, scaling down the attraction force. The direction modification is a simple summation of two directions, again scaling down the direction of the newly created marker.

An important aspect of environment modification is the rate of marker's fortification and the evaporation. At each frame (time unit) the weight of each marker in the environment is decreased by a certain degree. When the weight reaches zero, the marker is removed from the environment. Each time an existing marker is updated, the weight is increased by a certain degree. We used small values for fortification and larger ones for evaporation. In other words, the markers that are updated with frequency become more stable composing central trails of the environment. Also, there are upper limits of fortification that restrict continuous increase in the weight. Ultimately, the fortification and evaporation can be translated into the stabilization rate and how many bundling solutions are tested.

4 Application

In this section we describe a visualization application of our method for the flow map. We apply this technique to visualize the flow of products among warehouses and supermarkets, depicting movements of stocks of products of a major retail company in Portugal.

4.1 Data Description

Our dataset consists of warehouse-to-supermarket transitions of products over a time span of 6 months. Each transition has the following attributes: product id, quantity of products in transit (further referred to as *stock in transit* or *SIT*), quantity of products delivered (further referred to as *stock on hand* or *SOH*), warehouse id, supermarket id, and the date of transition. The locations consist of approximately 60 warehouses, some of which are located outside the Portugal and 1039 supermarkets in Portugal and 230 supermarkets outside the Portugal, from which only 680 are geo-located (see also [14–16]).

In order to get a graph representation of the data we proceeded with the calculation phase. First, the data were aggregated by days. Each day sums-up SIT quantities by aggregating pairs of warehouse-supermarket locations, which constitute the edges and the nodes of the graph. Finally, the total of SOH quantities per supermarket is calculated. Therefore, we get a weighted directed graph whose edges are directed from warehouse to supermarket nodes and weighted by SIT quantities. The nodes that represent supermarkets have an assigned SOH quantity, while the nodes that represent warehouse have none.

Finally, we filtered the nodes that only belong to the continental part of Portugal. We also excluded the supermarkets that have no geographic reference. For the sake of demonstration we focused only on the date of 1 of April, 2012. So, the final graph consists of 402 nodes and 2229 edges.

4.2 Flow of Products Visualization

This visualization depicts amounts of products that have been transported and the amounts of products that have been delivered during one day. It is important to describe the process to get a readable layout of a mixed graph. There are two challenges to visualize this particular graph: not all the locations have a geographical positions and the ones that do have may overlap. The first issue was solved by anchoring the position of the nodes that have a geographical location, and by using force-directed graph layout algorithm [17] to compute the location of other ones. This algorithm is efficient in what concerns the graph topology, since it considers clusters of nodes and not individual nodes. The second issue was solved by applying the Dorling cartogram technique over the pre-computed layout. The beauty of this algorithm is that it preserves the original relative positioning of geo-referenced elements, enabling the map of Portugal to be recognizable and making the locations distinguishable, increasing visual clarity (Fig. 4).

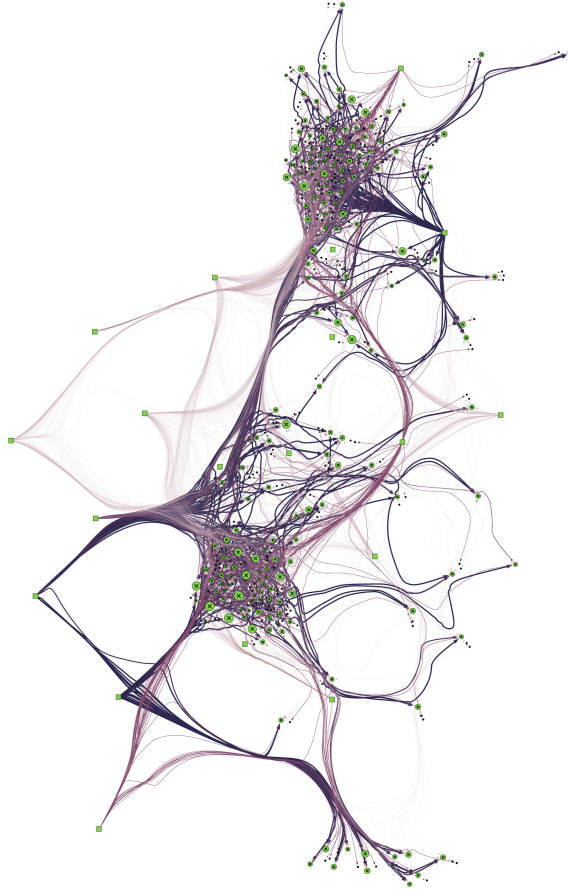


Fig. 4. Product transportation from warehouses and supermarket. (Color figure online)

In the visualization each trace represents the quantity of products in transit from warehouse to supermarket. These quantities are encoded by two means – *color* and *line thickness*. The two graphical elements are complimentary and make use of different mappings. The color uses a linear mapping. The values are mapped to a pale yellow-purple-dark blue gradient, being pale yellow and dark blue for lowest and highest values, respectively. The thickness variable, on the other hand, uses an exponential scale, emphasizing high values. Finally, we use an arrow, which is rendered at the end of a trace and scaled proportionally to the thickness of the line, in order to represent the direction of the flow (Fig. 4). The orientation of the arrow is determined in three steps. First, the length l and the width w of the arrow is computed. The w and l are equal to the $2x$ and $3x$ thickness, respectively. Second, the last vertex v of a trace that is located at the minimum distance from location of the node c plus its radius and the l is determined. Third, the arrow orientation is defined by the vector pointing from v to c .

In order to represent different types of nodes we use different graphical elements – *squares* and *circles* to represent warehouses and supermarkets, respectively. The nodes are colored according to the country they represent. Since our data contains several countries, the color could become ineffective. In this case, we calculate the total number of locations, aggregating by country, sort the countries in descendant order, and consider only the first two countries, which are Portugal and Spain, and treat the others as an unique instance. The colors used were green, red and blue for Portugal, Spain and others, respectively. Finally, the nodes that have geographic reference were rendered with the graphical shape of an X in the center of the node (Fig. 4).

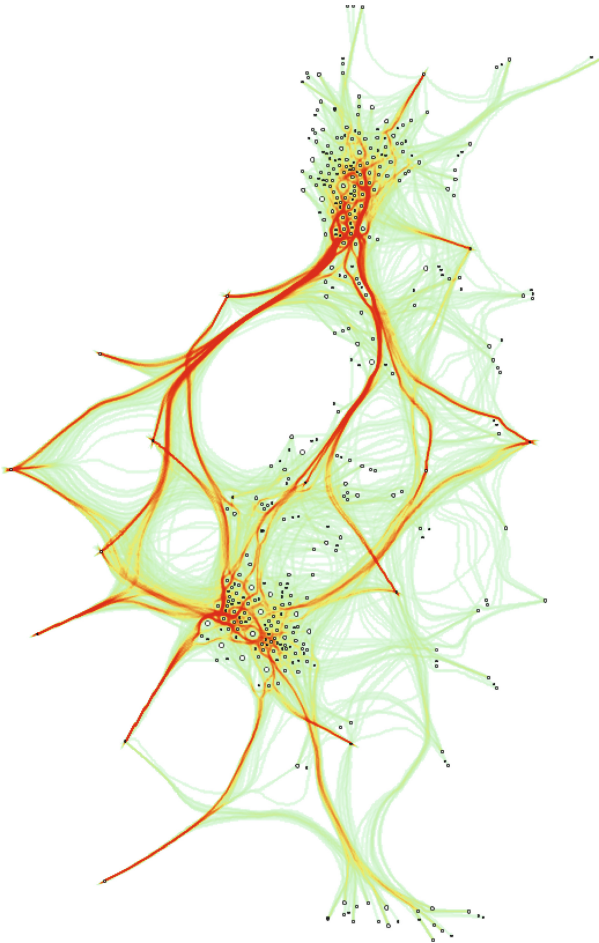


Fig. 5. A visualization of edge overlapping. The number of overlapped edges is represented with the color from pale green to intensive red. This image shows only the transitions from a selected warehouse. (Color figure online)

Density Visualization. Due to the high degree of overlapping traces, it is hard to identify main streams. For that reason, we proceed with a color-temperature visualization of the flow map. Using the same graph we apply different graphical elements. In this visualization the traces are colored according to the total number of overlapping elements. More precisely, we compute the number of overlapping segments that build-up the traces located within a defined range. Then at the render instance this value is mapped to a color scale from *intensive red* to a *pale green*, where red and green mean high and low degrees of overlaps. This representation is useful to get another perspective of such complex visualization (Fig. 5).

5 Comparison with Other Techniques

The visualizations shown in this section depict the US migration flow. The data covers migrations among cities of the United States during the period from 2014 to 2015. The graph consists of 377 nodes and 9780 edges. The weight of the edges correspond to the number of migrants normalized by the maximum value. It is important to mention that the cities not located in the continental part of the country were excluded. To encode the directionality of edges the color was used, more precisely the hue value $[0, 360]$. The absolute angle of the vector pointing from the origin node towards the destination was linearly mapped to the hue value.

With that said, this section proceeds with the comparison among straight line technique, force directed edge bundling (FDEB) method, and selected results of our technique. For the analysis of our approach we used pairs of low and high values for the parameters such as *maximum speed*, *maximum force*, *angle of vision*, *distance of vision* and *compatibility angle*. The maximum force can be translated into the agility of vehicles, which is easier to think of in the context of behavior. Higher maximum forces mean higher agility. Indicating actual values is meaningless, because they are relative to the scale of the graph. For reference, we used the values of 2 and 10 units for the graph with an approximate distance of 2500 pixels between the two most distant nodes. So, we generate all the permutations and identify combinations of values that can be divided in several groups. From the groups that achieved efficient results the representative images were selected and analyzed in detail.

The very first comparison reveals the efficiency of visual clutter reduction using the edge-bundling technique, comparing to the straight line representation (Fig. 6, image on the left). As can be observed, the force directed edge bundling (FDEB) method (Fig. 6, image in the middle) generates less visual clutter in comparison with our approach. However, the algorithms such as FDEB do not account for the presence of nodes, where traces can be drawn on top of nodes, and some of them neither account for the directionality of edges. In contrast, these properties are the emergent characteristic of our method. Since the vehicles in our system attempt to avoid obstacles, the nodes are clearly visible and do not visually interfere with the lines (see for instance Fig. 7). Also, as can be observed

in the same figure, there are no traces with opposite directions that are routed through the same path. The only moment, where the traces can overlap, can be the split or the join of streams. Additionally, when using swarms, main streams of flow become visually distinct from each other, leaving enough space for the ones with less impact. Moreover, since our approach uses the edge weights in the calculation, the generated bundles result in a meaningful representation of the data being visualized (Figs. 7 and 8).

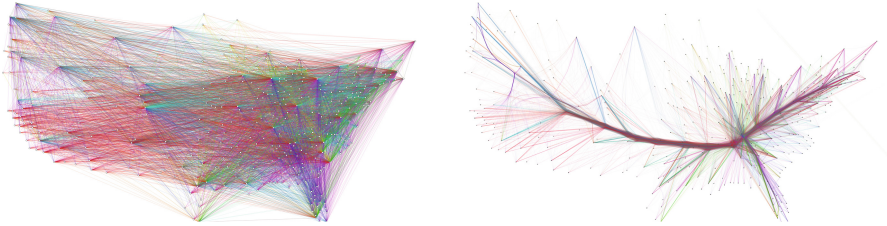


Fig. 6. Comparison between the techniques straight lines (left) and FDEB generated using 4 cycles, 50 iterations and stiffness 10 (right). The color represents the directionality of the edges.

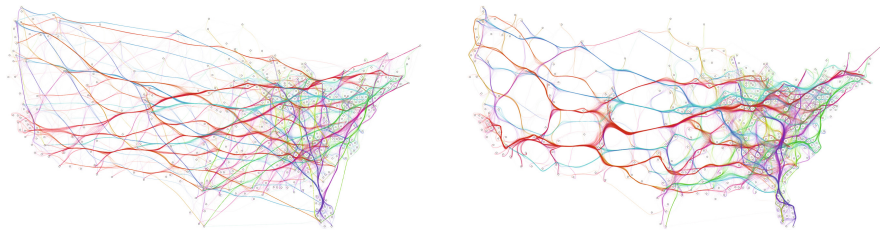


Fig. 7. Flow map generated by our method using: maximum speed equal to 10, maximum force equal to 12, mass equal to 1, vision angle equal to 135° , vision distance equal to 100 and compatibility angle equal to 45° (left); maximum speed equal to 10, maximum force equal to 12, mass equal to 8, vision angle equal to 135° , vision distance equal to 100 and compatibility angle equal to 90° (right). All values are in relative units, assuming that distance between two most distant nodes in the graph is approximately 2500 pixels.

Regarding the impact of parameters on the behavior of vehicles, and consequently over the final visual result, we established the following combinations. First of all it is important to mention that the analysis of results suggests that using low values for angle and distance of vision fails in generating an efficient visualization. In the later case the visualization becomes highly fragmented, and in many cases individual streams are hardly distinguishable, as can be observed in Fig. 8, image on the left, which was generated using low distance of sight.

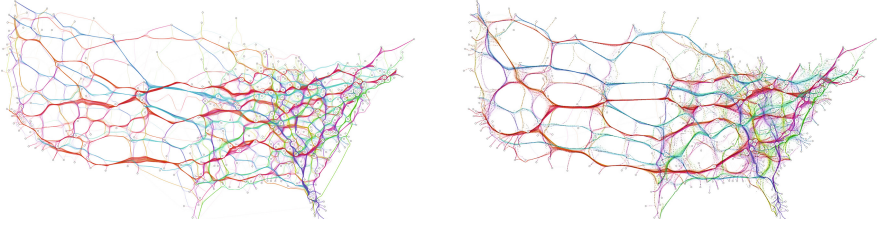


Fig. 8. Flow map generated by our method using: maximum speed equal to 2, maximum force equal to 12, mass equal to 1, vision angle equal to 135° , vision distance equal to 50 and compatibility angle equal to 90° (left); maximum speed equal to 2, maximum force equal to 2, mass equal to 1, vision angle equal to 135° , vision distance equal to 100 and compatibility angle equal to 90° (right).

Figure 7, image on the left, is the output from the system using *high* maximum speed and maximum force, and *low* mass and compatibility angle. As can be observed, meaningful bundles are clearly defined and distinguishable from each other. However, the resulting map is highly fragmented in comparison with examples that follow. Also, the representation presents constant changes between streams. This behavior is due to the low compatibility angle. The vehicles try to keep their velocity directions aligned with the destination. Finally, as can be noticed, due to high velocity and low agility the endings of traces are curly. This can be useful to distinguish origin and destination of flow.

The example shown in Fig. 7, image on the right, uses high values for all the parameters. In this example the main streams are clearly distinguishable from other traces. Due to high speed and low agility, vehicles tend to draw traces in a more organic manner. However, there is a loss in the detail of trace and overall visualization. Similar to the previous example the endings of traces present curly geometry.

The examples presented in Fig. 8 use low maximum speed. The distinction is in the agility: the example on the left uses low maximum force and the example on the right uses high maximum force. As can be observed, the traces in the representation on the left change their direction more abruptly than the ones of the image on the right. Additionally, it is clearly noticeable that curly endings disappear. This is because of low speed, therefore the vehicles have enough time to turn. Additionally, in the image on the left, we include the trails in order to illustrate the efficiency of the method to trace the streams of flow using swarms.

The method presented herein still has several limitations. The combination of high speed and mass with low agility can result in imperceptible visualization. As can be imagined with these parameters the vehicle advances fast and turns slowly, therefore making it harder to reach the destination. The vehicle spins around the destination node during a long time slowing down computational performance and creating visual artifacts. Another limitation, which can be advantageous in some cases, is that in highly dense graphs vehicles are unable to draw smooth

traces. In particular, in the areas where nodes form clusters, the vehicles attempt to avoid the entire cluster, therefore creating visual artifacts.

6 Conclusions

As previously mentioned, applying flow maps to a large amount of data is challenging, since it involves dealing with high degrees of visual clutter. In this article we presented a method for generating flow maps that overcomes the cluttering issue in visualization. Our approach relies on a nature-inspired algorithm, which creates emergent visual patterns. This approach, results in edge bundling, reducing visual clutter and to promoting visual clarity in the representation. In addition, we explored different graphical languages applied on the generated graph to give diverse perspectives over the same dataset.

Our method consists of a set of vehicles that trace a path to represent each edge in the graph. Each single vehicle follows simple rules through the interaction with the neighboring trails. The compatibility between the edges and markers of the trails determines whether vehicles are attracted to markers or not. This makes the vehicles follow the trails that eventually take them to their destination. Furthermore, the vehicles not only contribute to the environment by leaving new markers, they also update existing ones. This is done for logical reasons and to reduce computational complexity. In addition, the vehicles that represent more products have higher impact on other members of the system. Finally, every vehicle attempts to avoid obstacles, i.e. nodes, creating clear visual distinction between traces and nodes.

We described two types of graphical representation. We presented the main visualization, which depicts transitions of products from warehouses to supermarkets. The total amounts of products being transported are represented with color and line thickness. The directionality of movement is indicated by an arrow at the end of each trace. The nodes use shape to indicate either if they represent a warehouse or a supermarket. Finally, the fixed nodes are marked with an “X” in the center of the node. Additionally, the sorting order of the edges reflects the emphasis on low or high values. Then, we presented a graphical approach to distinguish main streams of flow. This is achieved by coloring the edges by their degree of overlap. In this case, the red and green colors represent a high and low number of overlapped traces, giving a visual representation of the complexity of the graph.

Finally, we compared our method with the force-directed edge bundling technique and performed a study of the impact of systems parameters on the visual output. We distinguished several groups of parameter combinations that give similar results. Therefore, we established that using high values for maximum speed results in lower waviness of the streams, and in combination with high values for mass, the method produces smooth separation of traces from main streams. Also, using low values for maximum force results in a curly ending of traces, which can be used as an indicator to distinguish between the origin and destination of streams. On the other hand, low values of maximum speed result

in a compact and definite bundling. However, the changes in the direction of streams become abrupt. Finally, low values for compatibility measure produce fragmented representation with a higher number of separate streams, but the traces become less “wavy” and the changes in the separation from main streams are less abrupt.

Acknowledgements. We would like to thank Catarina Maças, Hugo Amaro, Filipe Assunção, Antnio Cruz and Pedro Cruz for their contributions to this work. This project is partially funded by SONAE: Sonae Viz – Big Data Visualization for retail, and by Fundação para a Ciência e Tecnologia (FCT), Portugal, under the grant SFRH/BD/109745/2015.

References

1. Reynolds, C.W.: Steering behaviors for autonomous characters. In: Game Developers Conference, vol. 1999, pp. 763–782 (1999)
2. Dorling, D.: Area Cartograms: Their Use and Creation. Concepts and Techniques in Modern Geography, vol. 59. University of East Anglia: Environmental Publications (1996)
3. Tufte, E.R.: The Visual Display of Quantitative Information, vol. 2. Graphics Press, Cheshire (1983)
4. Phan, D., Xiao, L., Yeh, R.B., Hanrahan, P., Winograd, T.: Flow map layout. In: Stasko, J.T., Ward, M.O. (eds.) IEEE Symposium on Information Visualization (InfoVis 2005), 23–25 October 2005, MN, USA, p. 29. IEEE Computer Society, Minneapolis (2005)
5. Holten, D.: Hierarchical edge bundles: visualization of adjacency relations in hierarchical data. IEEE Trans. Vis. Comput. Graph. **12**, 741–748 (2006)
6. Holten, D., van Wijk, J.J.: Force-directed edge bundling for graph visualization. Comput. Graph. Forum **28**, 983–990 (2009)
7. Hurter, C., Ersoy, O., Telea, A.: Graph bundling by kernel density estimation. Comput. Graph. Forum **31**, 865–874 (2012)
8. Peysakhovich, V., Hurter, C., Telea, A.: Attribute-driven edge bundling for general graphs with applications in trail analysis. In: 2015 IEEE Pacific Visualization Symposium, PacificVis 2015, Hangzhou, China, 14–17 April 2015, pp. 39–46 (2015)
9. Di Marzo Serugendo, G., Gleizes, M.P., Karageorgos, A.: Self-organising systems. In: Di Marzo Serugendo, G., Gleizes, M.P., Karageorgos, A. (eds.) Self-Organising Software. Natural Computing Series, pp. 7–32. Springer, Heidelberg (2011). doi:[10.1007/978-3-642-17348-6_2](https://doi.org/10.1007/978-3-642-17348-6_2)
10. Macgill, J., Openshaw, S.: The use of flocks to drive a geographic analysis machine. In: International Conference on GeoComputation (1998)
11. Vande Moere, A., Mieusset, K.H., Gross, M.: Visualizing abstract information using motion properties of data-driven infoticles. In: SPIE Proceedings Series, pp. 33–44 (2004)
12. Moere, A.V.: Time-varying data visualization using information flocking boids. In: Ward, M.O., Munzner, T. (eds.) 10th IEEE Symposium on Information Visualization (InfoVis 2004), 10–12 October 2004, TX, USA, pp. 97–104. IEEE Computer Society, Austin (2004)

13. Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model. In: Stone, M.C. (ed.) Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1987, pp. 25–34. ACM (1987)
14. Polisciuc, E., Cruz, P., Amaro, H., Maças, C., Carvalho, T., Santos, F., Machado, P.: Arc and swarm-based representations of customer's flows among supermarkets. In: Proceedings of the 6th International Conference on Information Visualization Theory and Applications, pp. 300–306 (2015)
15. Maças, C., Cruz, P., Amaro, H., Polisciuc, E., Carvalho, T., Santos, F., Machado, P.: Time-series application on big data visualization of consumption in supermarkets. In: Proceedings of the 6th International Conference on Information Visualization Theory and Applications, IVAPP 2015, Berlin, Germany, 11–14 March 2015, pp. 239–246. SciTePress (2015)
16. Maças, C., Cruz, P., Martins, P., Machado, P.: Swarm systems in the visualization of consumption patterns. In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, 25–31 July 2015, pp. 2466–2472. AAAI Press (2015)
17. Fruchterman, T.M., Reingold, E.M.: Graph drawing by force-directed placement. *Softw. Pract. Exp.* **21**, 1129–1164 (1991)