PROCEEDINGS OF SCIENCE

Agent-Based Modelling And Simulation For The Geospatial Network Model Of The Roman World

Jan Fousek
Masaryk University, Brno, Czech Republic
E-mail: izaak@mail.muni.cz

Eva Výtvarová
Masaryk University, Brno, Czech Republic
E-mail: eva.vytvarova@mail.muni.cz

Adam Mertel
Masaryk University, Brno, Czech Republic
E-mail: mertel.adam@gmail.com

Aleš Chalupa
Masaryk University, Brno, Czech Republic
E-mail: chalupa@phil.muni.cz

Eva Hladká*
Masaryk University, Brno, Czech Republic
E-mail: eva@fi.muni.cz

In this paper we present a computational environment called LINUM for agent-based modelling, based on a geospatial transport model similar to the one implemented in ORBIS, The Stanford Geospatial Network Model of the Roman World. We provide the implementation of the transport model, and tools for creating and analyzing agent-based models around it. We also provide web-based visualization of the results of the simulation, and enable the user to generate time-collapsed static network for further analysis by the means of complex networks measures. The functionality of the environment is demonstrated on a model of diffusion process on the transport network.

International Symposium on Grids and Clouds (ISGC) 2016
13-18 March 2016
Academia Sinica, Taipei, Taiwan

*Speaker.

PoS(ISGC 2016)007

1. Introduction

Computational models have large potential for furthering our understanding of interaction between human and environment as a factor in various social and historical phenomena [1]. One such approach are agent-based models (ABM) that provide useful model paradigm for human behavior [2]. When coupled with geospatial data, such models can be spatially explicit, and have variety of applications in computational social science [3].

The ORBIS project [4] provides a geospatial network model of travel in the Roman Empire. It combines the road network with maritime transport model derived from historical data, and provides cost and time expense prediction for given routes. The time and cost predictions take into account various influential factors, such as seasonal changes, distinguishes coastal and open sea routes, and onshore means of transport. Currently the online interface enables the researchers to examine routes between given locations, analyze distance from one location to all others, and visualize the importance of paths connecting a given location to the rest of the network.

Modelling can be used by historians and archaeologist to explicitly formulate hypothesis about historical processes, or to fill in the gaps in the sparse archaeological evidence [5] [6] [7]. Agent-based models provide an attractive framework for formulating dynamic processes such as random walks, transportation and migration, or trade.

In our case, we want to be able to model agents traveling between the cities of the Roman Empire on routes defined by the transportation model similar to the one implemented in ORBIS. As the preferred routes can change depending on season and other external factors, advancing the model from current average estimates to probabilistic distributions derived from the simulation will provide with better understanding and robustness to subsequent analysis. The agent-based approach allows for such probabilistic simulations, and it is a direct extension of the current model.

In this paper we present LINUM: a computational environment for agent-based modelling on the geospatial transport model.[1] It is designed as a set of scripts and code snippets around the transport model library—rather than one-purpose application, as is the case with ORBIS—to be able to facilitate exchange and reuse of data and models between different research projects. It enables the researchers interested in modeling dynamic processes in the context of ancient Mediterranean to formulate and run their models, while being extensible both in terms of the input geospatial data and the transport model.

We provide scaffold code to specify the agent-based model, implementation of the transport model, scripts for construction of the transport model from the geospatial data, and web-based interface to visualize the results of the simulation. We also enable the user to convert the transportation model to a static network for further analysis by the means of complex network measures. The functionality of the environment is demonstrated on a model of diffusion process on the transport network.

2. Preparing input data

The models considered in this paper are built on structured historical evidence. While there are open data sets available, many researchers improve on them, or create their own to better cover

[1] `https://github.com/i-Zaak/linum`

areas of their interest. The selection and curating of input data is difficult task often requiring compromising accuracy for generality. Therefore, instead of hard-wiring the transport model, we provide the researchers with means of constructing it from their own geospatial datasets.

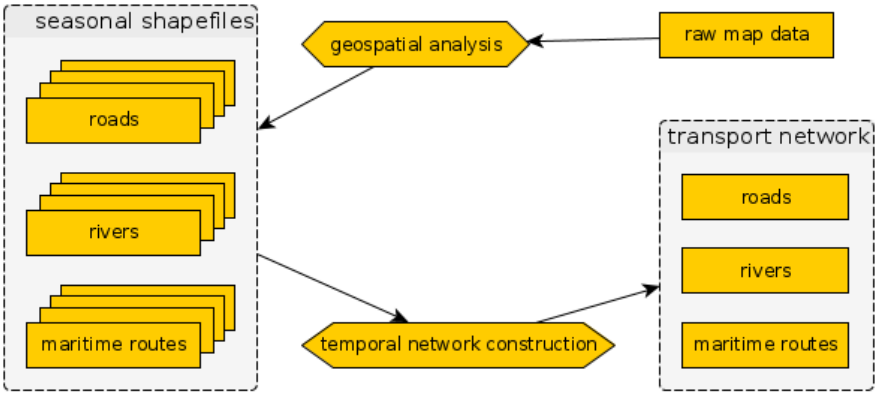

Figure 1: Overview of the workflow for creating the transport model. LINUM provides tools for the netowork construction step.

The geospatial preprocessing and analysis of the transport network is out of the scope of our framework. We start with the result of the geospatial analysis in the form of a shapefile—a common vector file format for geographic information systems [8]. Different ways of transportation can be included in the model—roads, naval/maritime routes, and river transport. The point geometries (cities, crossroads) are translated into nodes, and lines into weighted edges discarding the exact geometric shapes while keeping the distance. For more abstract models, also the crossroads can be discarded and only edges between cities with shortest direct path can be kept.

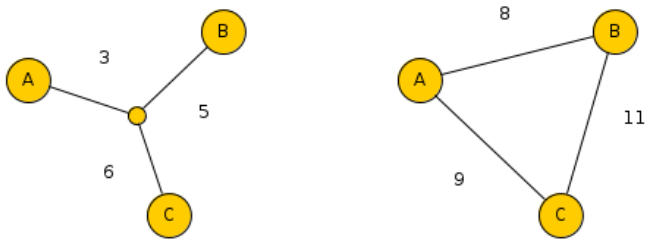

Figure 2: Two variants of transport network: left with crossroads, right with expanded crossroads.

Definition of the maritime transport network is more complicated, as the exact routes are usually unknown. Here we suggest to follow the approach taken by ORBIS, and model the open-sea and coastal routes between the harbors as shortest paths on a regular grid covering the water surface. Of course, this approximation can be improved by incorporating knowledge on e.g. sea streams (influence on speed), shallow water locations (impassable areas), etc. The generation of the maritime transport network is not yet implemented in LINUM, and will be incorporated in near future.

Rivers are also considered as a part of the transport network differing from the roads mainly in their directional asymmetry: while the travel distance in both directions is the same, upstream travel is much slower and more costly. We use the same routine as for the roads, and read the shapefile with river geometry with different constants for travel cost and speed in both directions.

All the above mentioned types of networks are combined in the last step resulting in directed transport network with edges associated with type and distance. In case of seasonal variability, particular network is duplicated in its seasonal variants, which then comprise the temporal transport network as described in next section.

3. Transport model

The transport network changes in time on several time scales. Some connections undergo seasonal changes—e.g. mountain passes being accessible only in summer—some are created (or seize to exist) in the course of years. To add the dimension of time to the transport network, we represent it as a temporal network. In this section we show how the ORBIS transport model can be constructed around a temporal network, and describe possible extensions.

3.1 Temporal transport networks

In general, the temporal networks allow for time-variable topology, and can be defined in multiple ways. We will start with the definition of *interval graphs* from [9]. The interval graph consists of a set of vertices V, set of edges E, and for every edge e a set of nonempty and non-overlapping intervals $T_e = \{(t_1, t_1'), \ldots, (t_n, t_n')\}$ giving periods of time, when the edge e is active. Further, we need to extend this definition with edge weights (e.g. cost, distance, travel time, etc.), we do so by defining the edge weighting functions w_e as piece-wise constant over the time intervals $w_e : T_e \to \mathbb{R}$. Time-respecting paths between two vertices u and v on such networks (also referred to as *journeys*) are timed sequences of edges $J(u,v) = \{(e_1\tau_1), (e_2,\tau_2), \ldots, (e_k,\tau_k)\}$ such that the timings τ_i respect the activity intervals of corresponding edges T_i [10].

While the definition above enables us to express the dynamic changes of the network in most detail, for practical purposes, we will also use (and implement) second variant of definition of temporal network—by multislice graph. Multislice graph is defined as a sequence of graphs G_i, where each slice $G_i = (V, E_i)$ consists of same set of vertices V and slice-specific edges E_i [11]. This representation is more coarse-grained, and thus simplifies both the computations, and construction from input data. The latter is important due to uncertainties and low temporal resolution of available historical evidence.

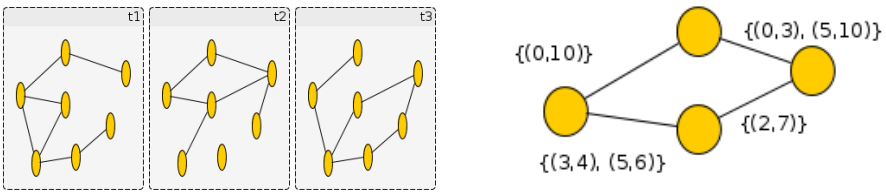

Figure 3: Two variants of representation of time-resolved network. Left: three time points of a multislice network. Right: interval graph representing a temporal network.

As mentioned above, we need to capture two main temporal characteristics of network evolution: periodic, and aperiodic. The periodically evolving network can be described in much more compact manner—we only need to write down single period—allowing for efficient modelling of long time spans. The seasonality of the ORBIS transport model is a good example of such periodic temporal network, in this case the model can be written as four seasonal or 12 monthly slices implicitly repeating indefinitely. On the other hand, the formalism of temporal networks also allows to express arbitrary aperiodic network evolution, providing a tool to model other than natural seasonal factors, such as construction of roads, establishing of trade routes, shifting geopolitical obstacles, and so on.

3.2 Traffic modelling

As we are mainly interested in spreading processes on the transport network, purely random walks may not be realistic in many cases, as they don't correspond to real traffic of people and goods within the network. To do so, we have adapted some of the techniques used in transportation forecasting to estimate volume of traffic (or intensity of information exchange) between the pairs of settlements, and the probable routes carrying this traffic. Namely, we use the modified four step model [12]: trip generation, trip distribution, mode choice, and route assignment.

The trip generation step defines the origins and destinations for the traffic. In our case, it is trivial—all settlements can be origins and destinations—however with proper historical data, these could be refined to e.g. reflect trade supply and demand locations.

The trip distribution step defines the frequency with which trips between particular origin and destination occur. To estimate the frequency T_{ij} between the nodes i and j, we opt for a gravity law [13] in a form of $T_{ij} = \frac{P_i P_j}{d^2}$, where P_i and P_j are population estimates of the settlements, and d is a distance in the transportation network. This approach generates both long-distance trips between large settlements, and short trips between local small settlements and eliminates the unrealistic long trips between small settlements. We note, that the population estimates can be hard to obtain due to incomplete, or imprecise archaeological evidence, however even coarse categorization of the settlements to classes by orders of magnitude of estimated population should be sufficient.

While we currently don't implement active choice of mode of transport, this step could be added if sufficiently backed by historical evidence. The last step—route assignment—differs in our model from the common traffic forecasting models, because the time-scale of the network changes is similar to the time-scale of the travel itself, which is obviously not the case today. Because the globally optimal algorithm is computationally demanding and complex, we have opted for approximation by the average over shortest paths in all seasons. This estimation will be improved in future versions of the framework.

The main results of the traffic modelling are twofold: trip distribution table and edge traffic density. The trip distribution can be used either as first approximation of intensity of communication between two settlements, or as an input for subsequent agent-based simulations. The edge traffic density can be used as a weight in subsequent complex network analysis.

4. Analysis and modelling

In this section we describe the agent based models and network analysis approaches we have

implemented using the transport model.

4.1 Agent based models

The agent-based models can be used on various levels of abstraction. We are mainly interested in modelling the spreading processes such as diffusion of innovation, so we first focus on agents performing random walks through the space given by the transport model. Random walk agent uses the transport model for two purposes: determine neighboring nodes to select next random destination, and to determine distance to selected destination. Note that due to the temporal character of the travel network, both these queries depend on time.

Interesting modification of the random walking agents is the addition of destination. Such agents spawn randomly according to given probability distribution over the settlements and select a random destination following the probability distribution given by the trip table described in previous section. The agent then selects the best (shortest, cheapest or fastest) path from the starting point to the destination. We allow small random diversions from the optimal path to allow the agents to better explore the space between the starting and end settlements.

While we currently don't consider any interaction between the agents in our models, the framework supports such behaviour. The agents can query the transport model for presence of other agents at given settlement.

4.2 Complex network analysis

The topology of the transport network can be quantified with the measures developed for analysis of spatial complex networks [14]. These measures however work with static network, so the transport model has to be first reduced in the time dimension. This can be done in several ways, the most straightforward is to remove the timing information from the edges, and aggregate the timed weighting function for every edge by e.g. taking minimum or average. This approach results in network overapproximation as it also includes paths, which don't correspond to valid journeys (violates the succession in time), however comparison across aggregation functions allows for at least partial control of the effect of time reduction.

Some complex network measures are based on the distribution of random walks over the edges and nodes of the network [15], such as random-walk centrality. These measures can make direct use of the transport model and the agent-based random walkers. Similarly, measures based on shortest path lengths—such as betweenness centrality—can be computed from the traces of the destination-driven walkers. Due to the substantial uncertainties in the input data and assumptions made in the construction of the model, this may be an interesting way of assessing the robustness of the results.

5. Computational environment

In this section we describe the architecture of the LINUM modelling environment, and provide the reasoning behind its design.

The LINUM computational environment consists of four main modules: the transport model, agent based models, network analysis, and visualization. The transport model component uses

multislice and temporal networks as a main data structures, and implements functions for computing shortest, fastest, or cheapest journeys, and queries for node neighbors at a given time. These functions are used by the ABM and network analysis modules: the former provides the transport model to the agents as a navigable environment, while the latter uses it to create summarizing static networks to be studied with established tools for static network analysis. Lastly, the visualization module provides basic visualization of the results of the simulations and the analysis.

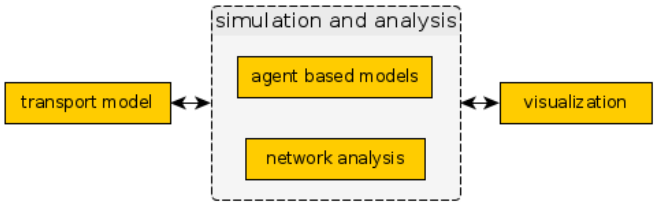

Figure 4: Schema of the components of the LINUM package.

The whole package is designed to be rather an extensible modelling library, than a single purpose application providing the researchers with enough flexibility for modifying the models and building their own *around* the shared transport model. Also, the focus on modelling and analytic workflows simplifies sharing and reuse of parts of previous studies in followup work or between otherwise independent research projects. For example the visualization module contains snippets for visualizing typical scenarios, such as geospatial plotting of the transport network, or visualizing the results of static network analysis.

As suggested above, the LINUM was designed from start with collaboration and sharing in mind. In our opinion sharing the implementation of the models and analyses should complement the ongoing efforts in sharing data [16]. Constructing a model on a level of abstraction as presented in this paper requires numerous decisions, and even if sincere effort is put into describing the model in the publication presenting the results, replication on same data set can be tricky if everything has to be reimplemented. To lower the barrier for code sharing, we have opted to host the LINUM on GitHub in a hope, that the community will be able to make use of its social networking features to improve transparency and openness of research [17].

The language of choice for implementation of LINUM is Python for two main reasons: its accessibility of the language for researchers less experienced in software engineering, and its vast number of available libraries[18]. In particular, we have used Mesa library [19] to implement the agent-based models, the network analysis module uses heavily NetworkX package [20], processing of geospatial data is possible with Fiona library,[2] and the usage examples and visualization are created as Jupyter notebooks [21]. The form of notebooks is ideal for communicating computational research on lower levels, as it allows for interweaving the code specifying the model and visualization of the inputs, intermediate steps, and results.

Lastly, we would like to note on possible opportunities for exploiting parallelism with LINUM. The models and analyses contain numerous parameters whose influence of the results is expected to be explored at least to some extent in every serious study. The resulting parameter sweeps are em-

[2] `https://github.com/Toblerity/Fiona`

barrassingly parallel and can efficiently exploit contemporary parallel computational architectures. As the LINUM package is written in Python 3, we plan to use the in-built multiprocessing module to implement local SMP parallelism, for distributed computing the master-worker architecture of the IPython module will be used.

6. Examples of use

In this section, we present the usage of the LINUM framework on two show cases: simple spreading model, and a random-walk agent simulation. The capabilities of LINUM are shown on random networks for the sake of simplicity, and all examples can be easily supplied with realistic networks derived from geospatial analysis.

6.1 Spreading process model

Here we demonstrate the usage of the transport model on a simple spreading process: we consider a susceptible-infected (SI) model [22] where the nodes can be either susceptible to disease (or information), or infected. The model starts with small number of infected nodes, and in every time-step, the infected nodes spread the disease to their neighbors with given probability.

For this demonstration, the network is Watt-Strogatz random small-world network [23]. The definition of the network model and the SI agents can be found in the `si_model.py` file in the repository. The visualization of the initial state, model evolution and summary statistic can be found in the Jupyter notebook `si_network_model.ipnb` and on a Figure 5.

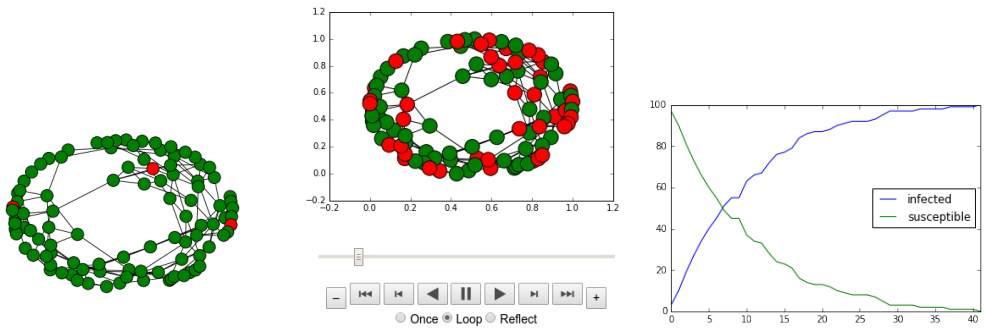

Figure 5: Visualization of a spreading process, from left to right: initial state of the network with infected nodes colored red, screenshot of the animation widget, plot of the summary statistics.

This show case can serve as a starting place for developing various spreading models. First, different networks can be supplied during the initialization. Second, the process dynamics can be defined simply by modifying the `Agent.step` function, e.g. by adding more states or changing the dynamics altogether.

6.2 Random walks on the transport model

Next we demonstrate the random-walk agent model using the temporal transport model. The temporal network consists of four random small-world networks to mimic seasonal changes. Distance weights on the edges are also set randomly and give number of days needed to cross the edge.

The length of the season in days is set to 91 so that the model approximates year cycle. 100 agents are placed randomly in the network nodes with destinations, and starting time offsets also drawn randomly so that the agents don't start all in the same time (and same season).

The simulation finishes, when all agents arrive to their destinations, or maximal number of iterations is taken. The results contain for every time step position of all agents and time in days passed from the start of the simulation. Path taken by a given agent can be visualized as can be seen in Figure 6, and in the Jupyter notebook `random_walkers.ipnb`.

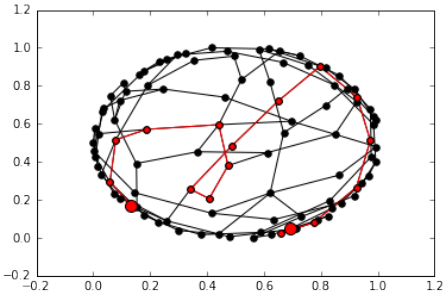

Figure 6: Path through the network taken by destination-driven random walking agent. Visited nodes and edges are colored red, larger nodes denote starting node and destination.

7. Conclusions and future work

In this paper we have presented computational framework for agent-based modelling on dynamic geospatial transportation model. The context of historical research brings unique challenges for modelling spreading processes on networks, as the time-scales of the evolution of the network and the process itself are similar.

The package contains implementation of the transport model and supporting scripts for preparing the inputs, visualizing the results, and examples of usage. It is meant not as a standalone application, but as a collection of tools and building blocks for creating transportation network models based on historical data.

While we have demonstrated the usage of LINUM on simple examples, we plan to apply it in near future in modelling concrete case studies, such as christianization of the Roman Empire [24]. Simultaneously, we plan to implement the extensions suggested in this paper, such as probabilistic complex network analysis and systematic parametrization. Lastly, we also plan on creating and sharing scripts for importing data from relevant databases with archaeological and historiographical evidence, which are currently being created within digital humanities. We would also like to invite other researchers to contribute either directly to the code, or by suggesting relevant functionality, so that the package fits best the needs of the community.

References

[1] J. M. Epstein, *Why model?*, *Journal of Artificial Societies and Social Simulation* **11** (2008) 12.

[2] R. Conte and M. Paolucci, *On agent based modelling and computational social science*, 2011.

[3] A. Crooks, *Agent-based modeling and geographical information systems*, in *Geocomputation: A Practical Primer* (C. Brunsdon and A. Singleton, eds.), pp. 63–77. Sage, London, UK, 2015.

[4] W. Scheidel, *ORBIS: the stanford geospatial network model of the roman world*, .

[5] P. Turchin, *Toward cliodynamics–an analytical, predictive science of history*, *Cliodynamics* **2** (2011) .

[6] W. H. Cegielski and J. D. Rogers, *Rethinking the role of agent-based modeling in archaeology*, *Journal of Anthropological Archaeology* **41** (2016) 283–298.

[7] M. Madella, B. Rondelli, C. Lancelotti, A. Balbo, D. Zurro, X. R. Campillo et al., *Introduction to simulating the past*, *Journal of Archaeological Method and Theory* **21** (2014) 251.

[8] ESRI, *Esri shapefile technical description*, 1998.

[9] P. Holme and J. Saramäki, *Temporal networks*, *Physics reports* **519** (2012) 97–125.

[10] B. B. Xuan, A. Ferreira and A. Jarry, *Computing shortest, fastest, and foremost journeys in dynamic networks*, *International Journal of Foundations of Computer Science* **14** (2003) 267–285.

[11] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter and J.-P. Onnela, *Community structure in time-dependent, multiscale, and multiplex networks*, *science* **328** (2010) 876–878.

[12] M. G. McNally, *The four step model*, *Center for Activity Systems Analysis* (2008) .

[13] M. Barthélemy, *Spatial networks*, *Physics Reports* **499** (2011) 1–101.

[14] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez and D.-U. Hwang, *Complex networks: Structure and dynamics*, *Physics reports* **424** (2006) 175–308.

[15] J. D. Noh and H. Rieger, *Random walks on complex networks*, *Physical review letters* **92** (2004) 118701.

[16] "Pleiades gazetteer." `http://pleiades.stoa.org/`.

[17] L. Dabbish, C. Stuart, J. Tsay and J. Herbsleb, *Social coding in GitHub: transparency and collaboration in an open software repository*, in *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, pp. 1277–1286, ACM, 2012.

[18] F. Perez, B. E. Granger and J. D. Hunter, *Python: an ecosystem for scientific computing*, *Computing in Science & Engineering* **13** (2011) 13–21.

[19] D. Masad and J. Kazil, *Mesa: An agent-based modeling framework*, .

[20] D. A. Schult and P. Swart, *Exploring network structure, dynamics, and function using NetworkX*, in *Proceedings of the 7th Python in Science Conferences (SciPy 2008)*, vol. 2008, pp. 11–16, 2008.

[21] F. Pérez and B. E. Granger, *Ipython: a system for interactive scientific computing*, *Computing in Science & Engineering* **9** (2007) 21–29.

[22] M. Barthélemy, A. Barrat, R. Pastor-Satorras and A. Vespignani, *Velocity and hierarchical spread of epidemic outbreaks in scale-free networks*, *Physical Review Letters* **92** (2004) 178701.

[23] D. J. Watts and S. H. Strogatz, *Collective dynamics of 'small-world' networks*, *nature* **393** (1998) 440–442.

[24] R. Stark, *The rise of Christianity: A sociologist reconsiders history*. Princeton University Press, 1996.