A unified framework for traditional and agent-based social network modeling

# A Unified Framework for Traditional and Agent-Based Social Network Modeling

**Enrico Franchi, Michele Tomaiuolo**
*DII, University of Parma, Italy*

## ABSTRACT

In the last sixty years of research, several models have been proposed to explain (*i*) the formation and (*ii*) the evolution of networks. However, because of the specialization required for the problems, most of the agent-based models are not general. On the other hand, many of the traditional network models focus on elementary interactions that are often part of several different processes. This phenomenon is especially evident in the field of models for social networks. Therefore, we present a unified conceptual framework to express both novel agent-based and traditional social network models. This conceptual framework is essentially a meta-model that acts as a template for other models. To support our meta-model, we propose a different kind of agent-based modeling tool that we specifically created for developing social network models. The tool we propose does not aim at being a general-purpose agent-based modeling tool, thus remaining a relatively simple software system, whereas it is extensible where it really matters. Eventually, we apply our toolkit to a novel problem coming from the domain of P2P social networking platforms.

## INTRODUCTION

In the last sixty years of research, several models have been proposed to explain (*i*) the formation and (*ii*) the evolution of networks, or, more in general, (*iii*) various kinds of processes over networks.

Such models have been developed and studied by researchers coming from many different areas, such as computer science, economics, natural sciences, meteorology, medicine, pure or applied mathematics, sociology and statistics. As a consequence, a lot of material exists on the subject and it is beyond the scope of this work to review and analyze it thoroughly (Bergenti et al., 2012; Franchi & Poggi, 2011); we refer to Jackson (2010) for the economic and game-theoretic point of view, and to Snijders (2011) for a complete review of the state of art of statistical models. Newman (2010) provides an accurate presentation of the approach developed in the computer, natural and physical sciences.

Several agent-based models have also been successfully developed in order to study specific problems that, because of either (*i*) the complexity of the agent to agent interactions, or (*ii*) the richness of the underlying environment, were relatively impervious to analysis using traditional techniques (Arthur, 1996; Axtel et al., 2004; Bagni et al., 2002; Carley et al., 2006; Dean et al., 2000; Epstein & Axtell, 1996; Folcik et al., 2007; Hill et al., 2006; Ilachinski, 2000; Kohler et al., 2005; Moffat et al., 2006; North et al., 2010; Lucas & Payne, 2014).

However, most agent-based models address very specific phenomena and, as a consequence, they have relatively low reusability. On the other hand, many traditional models focus on elementary interactions that are often part of several different processes, and, consequently, the effects of these elementary interactions have been thoroughly studied. Nardin, Rosset and Sichman (2014) present a more detailed discussion regarding how to obtain more universal conclusions from agent-based modeling.

We believe that such interactions should be introduced as basic building blocks even for agent-based models, enriched with more "agent-ness" when it is the case. However, several assumptions that are perfectly legitimate in a stochastic process are not well rendered in an agent-based model. Moreover, many considerations that we derive from the conversion from traditional stochastic to agent-based models are similar to the ones that should be made by implementing the stochastic model in a generic non agent-based concurrent environment.
Considering the relevance that social networking platforms have gained in our lives, either directly or indirectly, one of the most interesting applications of agent-based models for social networks is to social networking platforms.
The structure of this Chapter is as follows: (*i*) some considerations over the epistemology of agent-based modeling are given; (*ii*) our proposal for a meta-model for social network processes is described, and also a working toolkit built over such ideas is presented; (*iii*) as an example, our platform is used to model a problem coming from the implementation of P2P social networking platforms.

## AGENT BASED MODELING FOR SOCIAL SCIENCES

ABM is a very powerful technique that has been applied increasingly often in the last years in a variety of different contexts. Examples of those contexts are (*i*) social sciences (Axelrod, 2003; Axelrod & Tesfatsion, 2006; Epstein, 1999, 2002, 2006; Kohler & Gumerman, 2000; Castelfranchi, 2014), (*ii*) economy (Arthur et al., 1997; Axtel et al., 2004; Epstein & Axtell, 1996; Arciero et al., 2014; Carbonara, 2013; Magessi & Antunes, 2014; Santos et al., 2014; Trigo, 2014), and marketing (North et al., 2010), (*iii*) epidemics and medicine (Bagni et al., 2002; Carley et al., 2006; Folcik et al., 2007), (*iv*) archeology (Dean et al., 2000; Kohler et al., 2005), (*v*) Philosophy (Coelho, da Rocha Costa & Trigo, 2014), (*vi*) energy distribution (Mota, Santos, Dimuro & Rosa, 2014), (*vii*) Game Theory (Georgalos, 2014) and (*viii*) warfare (Hill et al., 2006; Ilachinski, 2000; Moffat et al., 2006).
In ABM the subject of the modeling is described in terms of a collection of autonomous decision-making units, called agents, that are grouped together to form an agent-based model. Each agent individually assesses the situation and makes its own decision. A model consists of a group of agents, their individual behaviors and their mutual relationships.
The ideas behind agent-based modeling are rather simple: the decision process is decentralized and distributed among the simulated units, which are basically given the same information their real counterparts have. Moreover, the interactions among the agents tend to be simple and somewhat limited. Simpler interactions are usually easier to factor out from the system to model and, moreover, often agents formally act under the hypothesis of bounded rationality (Simon, 1955a), i.e., they are limited (*i*) by the information they have, (*ii*) by their cognitive abilities, and (*iii*) by the finite amount of time available to reach a decision.
Nonetheless, self-organization, patterns, structures and behaviors that have not been explicitly programmed arise from the individual interactions. In fact, emergence is one of the most peculiar features of ABM (Frisen, Gordon & McLeod, 2014). Essentially, the whole point of ABM is studying the emergence of the macro-level features of interest from the explicitly programmed micro-level interactions.
The intuitive notion of emergence, i.e., the manifestation of some property that seems to appear spontaneously in the system, is not hard to grasp and is quite a fascinating idea. Consequently, the notion has been often abused in contexts where it is not appropriate and, most of the times, is not defined at all. Researchers, such as Epstein and Axtell (1996), and Axerlrod (2007) frame the context of emergence defining an emergent phenomenon as a stable macroscopic pattern arising from the local interactions of the agents.
Another important consideration with emergence is that perfect knowledge of the micro-level interactions does not allow to predict macroscopic structure, so emergence cannot be usually predicted. However,

agent-based models specified in terms of micro-level interactions can be executed so that the macro-level structure can be actually observed.
In fact, agent-based modeling is more than just a modeling and simulation technique. Agent-based modeling is an epistemological paradigm shift in the way science is done. Traditional sciences usually follow (*i*) a deductive approach, where phenomena explanations are deduced from more general rules, assumed correct, (*ii*) an inductive approach, where the explanations are inferred from repeated observations, or, perhaps, (*iii*) a combination of the two.
On the other hand, the question of the scientist using ABM is:
*"How could the decentralized local interactions of heterogeneous autonomous agents generate the given regularity?"* (Epstein, 1999)
Epstein proposes to use the term generative to characterize this approach. He also discusses the epistemological soundness of the proposed approach (Epstein, 1999). A full discussion of the subject, even if fascinating, is beyond the scope of this work. For our purposes, it is sufficient to say that in the generative method, demonstration obtained with ABM is taken as a necessary condition. In other words, if the agent-based model did not generate the desired phenomenon, then the modeled interactions do not explain the phenomenon itself. However, if the modeled interactions generate the expected emergent behavior, then they are only a candidate explanation. If more than one candidate explanations exist, additional care should be exercised to determine which explanation is the most tenable, typically considering the plausibility of the explanation and resorting to the same ideas used when proposing a hypothesis in inductive science, e.g., Ockham's razor. In this book, Castelfranchi (2014) discusses the epistemological issues of ABM to a larger extent.

## AGENT BASED MODELING FOR SOCIAL NETWORK SIMULATIONS

Each agent-based model tends to be an *ad-hoc* product with the purpose of studying a precise and restricted family of phenomena. The sheer variety of domains where ABM has been a successfully applied prevents any sound possibility of standardization, as the price would be the flexibility that is so coveted by agent-based modelers. However, limiting ourselves to the domain of simulations of social networks, we can standardize the methodologies to specify models and simulations, so that they are easier to compare and evaluate. Moreover, in the context of social network models, several non-agent based models are extremely popular and agent-based variants of them could be important elements of more complex agent-based processes.
As a consequence, in order to simplify (*i*) the creation of new agent-based models and (*ii*) the adaptation of traditional models into agent-based ones, we designed a conceptual framework that separates the various concerns and allows to write simulations with few lines of code (Bergenti et al., 2013; Franchi, 2012b).
Analyzing the traditional models (Barabasi & Albert, 1999; Watts & Strogatz, 1998; Holme & Kim, 2002; Kumar et al., 2006), as well as some other non-generative network processes, such as the infection diffusion models described by Pastor-Satorras and Vespignani (2001), we singled out a meta-model that is suitable to implement said models as agent-based models (Franchi, 2012a; Franchi, 2012b; Bergenti et al., 2011). The meta-model also allows for features typical of agent-based models to be introduced gradually and to added to the agent-based variants of the traditional models.
Since in many simulations the nodes do not have a pro-active, goal-directed behavior, but, instead, perform actions when required by the model, we provide the meta-model with a Clock and a special agent called Activator. The Clock beats the time, which is discrete. At each step, the Activator (*i*) selects which nodes to activate, and (*ii*) decides whether nodes shall be destroyed, or (*iii*) created, negotiating the eventuality with the NodeManager. The nodes execute actions after receiving the activation message. However, if the model is fully agent-oriented, they can also perform actions autonomously, without waiting for activation. The nodes can also leave the simulation, require the creation of other agents, and send messages to the other nodes. The general structure of the execution phase is presented in Figure 1.

In essence, a simulation is fully described providing the specifications of: (*i*) the selection process of the groups of nodes to create, activate or destroy, which is performed by the Activator agent; (*ii*) the behavior of the nodes themselves. Notice that the general structure does not change significantly even when introducing some agent-ness in the simulations, e.g., introducing goal-directed behavior. For example, an agent can send messages to itself, starting from its initialization, so that, effectively autonomous behavior is obtained.

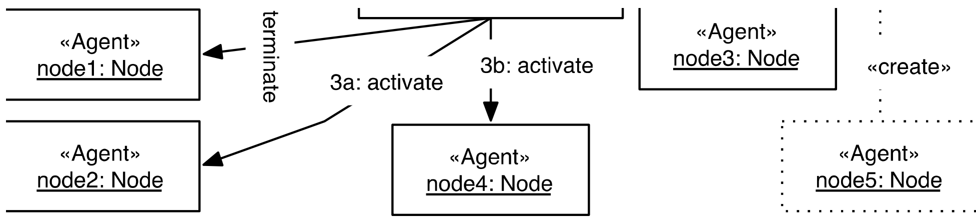


*Figure 1. Interaction diagram of the main agents in the meta-model.*

The social network simulation system we propose, PyNetSYM (PYthon NETwork Simulation-analYsis-Method), has an elaborate runtime system that supports the execution of simulations providing only brief specifications. Franchi (2012a) provides a more thorough description of the framework, and also presents some examples of how the framework can be used to develop agent-based models.

## Design and implementation of PyNetSYM

After more than 20 years of ABM for social sciences, several tools have been developed to ease the task of running the simulations; among these the most popular are Swarm (Minar et al., 1996), Mason (Luke et al., 2005), RePast (North et al., 2007) and NetLogo (Tisue & Wilensky, 2004), which, however, are not specifically tailored for social network simulations.

In this Chapter, instead, we introduce a different kind of ABM tool that we specifically created for network generation and general processes over networks (Franchi, 2012a). The tool we propose does not aim at being a general-purpose agent-based modeling tool, thus remaining a relatively simple software system, whereas it is extensible where it really matters (e.g., supporting different representations for networks, from simple memory-based ones to pure disk-based storage for huge simulations). Its theoretical foundations lie deep in the generative approach to science that we discussed in the previous section.

From a more practical point of view, the social network simulation system we propose has the following defining goals: (*i*) it must support both small and large networks; (*ii*) simulations shall be effortlessly run on remote machines; (*iii*) it must be easy to use, even for people without a strong programming background; (*iv*) deploying a large simulation should not be significantly harder than deploying a small one.

In our approach, the simulation system has four main components that can be modified independently: (*i*) the user interface, (*ii*) the simulation engine, (*iii*) the simulation model and (*iv*) the network database. The simulation model needs to be specified for each simulation and is the only part that has to be completely written by the user. Its specification is partly declarative and partly object-oriented. The user interface is responsible for taking input from the user, e.g., simulation parameters or information on the analysis to perform, and is specified declaratively. The simulation engine defines the concurrency model of the simulation, the scheduling strategies of the agents and the communication model among the agents. The network database actually holds a representation of the social network; it may be in-memory or on persistent storage, depending on the size of the simulation. Multiple network database implementations are provided and the framework can be extended with additional ones.

Large-scale simulations typically require more resources than those available on a desktop-class machine and, consequently, need to be deployed on external more powerful machines or clusters. In order to simplify the operation, we designed our system so that a simulation can be entirely specified in a single file that can be easily copied or even sent by email.

Considering that the simulations are frequently run on remote machines, we opted for a command line interface, because a traditional GUI becomes more complex in this scenario. An added benefit is that batch executions are also greatly simplified. We also support read-eval-print-loop (REPL) interaction to dynamically interact with the simulation.

In order to allow people without a strong programming background to easily write simulations, we decided to create a Domain-Specific Language (DSL). A Domain-Specific Language (DSL) is a language providing syntactic constructs over a semantic model tailored towards a specific domain (e.g., building software). The idea is that DSLs offer significant gains in productivity, because they allow the developers to write code that looks more natural with respect to the problem at hand than the code written in a general-purpose language with a suitable library.

The central element of the runtime system is the agent, since the elements under simulations and several infrastructure components of the runtime system are implemented as agents. In the following we describe the design characteristics of PyNetSYM agents.

For our purposes an agent is a bounded unit with its own thread of execution. All the communication among the agents occurs through message passing; each agent has a mailbox where the incoming messages are stored, and a unique identifier that is used to address the messages.

Agents also perceive and modify the environment. Our agents are not necessarily autonomic or goal-directed. Since they are used as a computational primitive, we need a lower-level specification that can be enriched to provide “real” agents but which does not place unnecessary overhead on the system.

Another important design decision regarding the system semantics is whether to implement cooperative or preemptive multi-tasking. Several popular languages and platform chose preemptive multi-tasking because in general purpose systems the probability and the risks of a misbehaving application consuming all the CPU time is too high. However, for a simulation oriented platform, such risks are minimal and we opted for cooperative multi-tasking because it allows a more deterministic control of complex time sequences.

As a consequence, in PyNetSYM a method handling a message can only voluntarily “give up” the execution for a while, either explicitly going to sleep or by requesting a blocking operation. In all other situations, when an agent starts processing a message, it continues until termination. This property is analogue to the semantics of the Actor Model (Agha, 1986) and simplifies formal reasoning on the system. Moreover, from the point of view of the emergent properties of the simulation it has little impact (Bergenti et al., 2013).

When an agent has an empty mailbox, it can choose to be removed from main memory and have its state saved on secondary storage. If the stored agent is subsequently sent a message, it is restored in main memory from the saved state. This behavior is extremely convenient considering that for most social network topologies, a small fraction of agents is responsible for the vast majority of the links. Since in most processes over networks the agents with few links are seldom activated, we can save memory keeping them in secondary storage and do not lose much CPU time.

Another important memory-related issue is the storage of the network itself. A possible solution is completely distributing the knowledge of the network among the agents, so that each agent only knows its neighbors and the network must be reconstructed from their interactions. In order to allow for more efficient implementation of network processes, we provide a global view of the network that is accessible by the agents. From the point of view of ABM, the decision is consistent with the interpretation of the network as the environment, as: (*i*) agents can interact with it by creating or destroying links, and (*ii*) the agents behavior is influenced by the network in several process dependent ways.

This view is presented as a software component that we call network database. The network database provides a unified interface that agents can use to modify and query the network state. We provide multiple implementations in order to be able to balance the various trade-offs. Some implementations are RAM based, and their main advantage a more efficient access when the network is not excessively large; others are backed with various secondary-storage based solutions, which results in slower operations, but they allow for simulations on larger networks.

### Structure of the simulation

The actual simulation is divided in two distinct phases (*i*) setup and (*ii*) execution. During the first phase (setup), the system is initialized so that it reaches the initial configuration specified by the simulation. First, various infrastructural agents (e.g., Activator, NodeManager) are created and started, so that they are ready to receive messages, and the Clock (if present) is also created, but not started.
Later during this phase, the Configurator agent instructs the NodeManager (*i*) to create the initial nodes in the network, (*ii*) to give them instructions to establish the connections they are meant to have at t0, and (*iii*) to provide them with any other initial information that the simulation may require .
The NodeManager is generally responsible for (*i*) creating the new agent-nodes, passing them the appropriate initialization parameters and (*ii*) monitoring them, so that their termination (exceptional or not) is managed.
We created different Configurator agents for the most frequent needs, that are (*i*) reading an initial network specification from disk and setting the system up accordingly, or (*ii*) creating n node-agents of a given kind. When reading network specifications from file, we support (*i*) popular file formats for exchanging networks, such as GraphML or Pajek, (*ii*) networks saved as sparse matrices in HDF5 files, and (*iii*) networks stored in various DBMSs, such as MongoDB.
After configuring the simulation, (*i*) the Configurator agent terminates, and (*ii*) if present, the Clock agent starts, marking the transition to the execution phase, or (*iii*) the node-agents are otherwise notified that the simulation begins.
During the execution phase (second phase) the node-agents perform the simulation according to their behavior. Although from a theoretical point of view such behavior can be completely autonomous and it does not rely on an external time schedule, in practice most network generation models and generic processes over networks can be described in terms of a relatively simple meta-model as the one previously described.

## SOCIAL NETWORKING PLATFORMS

Although in the last ten years the pervasive adoption of social networking sites has deeply changed the web, and such sites became an unprecedented social phenomenon (Boyd & Ellison, 2008), clear boundaries regarding the rights and responsibilities of service providers have not been established. Moreover, according to Stroud (2008), web sites have attracted users with very weak interest in technology, including people who, before the social networking revolution, were not even regular users of either popular Internet services or computers in general.

While most users are aware that their profile and the information they publish is essentially public, they usually harden their privacy settings only after problems arise and tend to overlook the actual impact of the information they disclose. Apparently harmless information can be exploited, and the more information the attacker has, the more severe and sophisticated the attack can be.

Privacy in online social networks is always intended as user-to-user privacy: even when the relative settings are set correctly, so that no other user in the system can access information not intended for his eyes, the system itself has full access to information. In fact, in most online social networks, the system owners actually rely on such information to make a profit, for example to improve the accuracy of target advertisement. Unfortunately, as long as they have full access to the information – i.e., the information is stored in the system without cryptography – any security issue or naivety results in privacy violations.

From a technological perspective, online social networks are mostly based on sets of web-based services that allow people to present themselves through a profile, to establish connections with other users in the system and to publish resources. Moreover, these systems use common interests and the natural transitivity of some human relationships to suggest new contacts with whom to establish a connection. Some of these aspects already appeared in other systems, but the unceasing flow of information that users pour in such systems, and their overwhelming intent to increase the number of their virtual friends and acquaintances, are unprecedented.

In other words, what sets social networking sites apart from other services is the *scale* and not the *architecture*. Distribution is used as mean for redundancy and is always *internal*: from the point of view of the users, the social networking system is a centralized system which does not behave differently from a regular website. In fact, these systems are monolithic entities owned by a single company.

As a consequence, service providers are in the position to effectively perform a-priori or a-posteriori censorship, or to disclose all the information they have, no matter how private, to other entities. They can perform such actions either motivated by selfish interests or forced under legal terms and other forms of pressure.

Considering that: (*i*) no single centralized entity can withstand the operative costs of a large scale social networking system without a solid business-plan; (*ii*) most business plans are based on targeted advertisement; and (*iii*) even if a service provider would be fair with its user's data, it would remain vulnerable to legal requests to disclose such data, we favor a P2P approach.

In the first place, P2P systems essentially achieve automatic resource scalability, in the sense that the availability of resources is proportional to the number of users. This property is especially desirable for media sharing social networking systems, considering the exceptionally high amount of resources needed.

Moreover, regarding censorship issues, a P2P system essentially solves them by design. Without a central entity, nobody is in the position of censoring data systematically, nor may be held legally responsible for the diffusion of censorable data: the sole owners and responsible of the data are the users themselves.

Building a P2P social networking platform is an open research challenge, because several simplifying assumptions that centralized platforms make cease to be valid, especially regarding the availability of the data. However, the importance of the problem is such that several researchers have proposed solutions either for full stack platforms (Buchegger & Datta 2009; Buchegger et al. 2009; Cutillo et al., 2009; Graffi et al., 2010) or to allow the distribution of certain aspects of a platform (Xu et al., 2010; Xu et al. 2010b; Perfitt & Englert, 2010)

## Blogracy

Blogracy (Franchi & Tomaiuolo, 2013) is our own implementation of a P2P social networking platform. More specifically, it is an anonymous microblogging platform, built incrementally over BitTorrent (Cohen, 2003), a popular and resilient file-sharing service. The architecture of the platform is modular and is built around a module for basic file sharing and DHT operations, possibly exploiting an existing implementation, and another module providing a set of social services to the local user through a Web interface. Moreover, the platform provides two additional agent based modules respectively providing a set of pervasive services and a set of information retrieval and pushing services. In particular, the current prototype of Blogracy takes advantage of; (*i*) Vuze (Vuze, 2012), a popular BitTorrent client implemented in Java and available as open source software, for implementing the file sharing and DHT operations, and (*ii*) Open Social (OpenSocial and Gadgets Specification Group, 2012), a set of APIs supporting the sharing of social data, for implementing the social services.

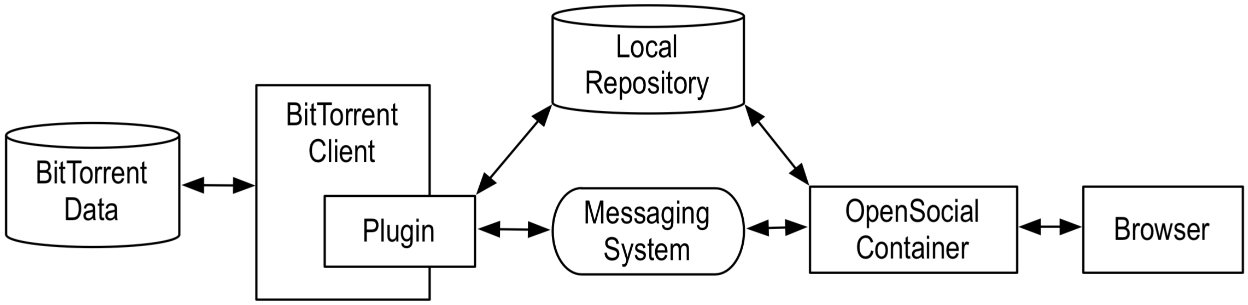


*Figure 2. The architecture of the micro-blogging social networking system Blogracy.*

For its basic operation, Blogracy uses a P2P file-sharing mechanism and two logically separated DHTs. Users have a profile and a semantically meaningful activity stream, which contains their actions in the system (e.g., add a post, tag a picture, comment). One DHT maps the user's identifier with his activity stream, which also contains a reference to the user's profile and references to user's generated resources (e.g., posts, comments). These references are keys of the second DHT, which are then resolved to the actual files. The files are delivered using the underlying P2P file-sharing mechanism. For more details on Blogracy implementation, we refer to (Franchi & Tomaiuolo, 2013)

The most fundamental problem in a P2P social network is data availability, i.e., the problem to ensure that content placed on the network is accessible after the publisher disconnected. In fact, although popular content rapidly gains lots of seeds, resources published by peripheral users, with few contacts and sparse online presence, can instead suffer poor availability to the extent that the publisher may remain the only seed for his own posts. This issue is strongly related to the churn phenomenon in P2P networks, i.e., the fact that nodes can leave and join the network at arbitrary times.

The availability of a new resource mostly depends on the connection pattern of the source node and its followers. Having a larger number of followers who share the new resources increases the availability of the data itself, especially if some of the followers have an elevated online presence. A single follower with very high availability can guarantee a very good diffusion of data, since every follower in the social graph is also a seed in the file sharing application..

Essentially, low resource availability is especially troublesome for new users with few contacts. For users in these situations, it is relatively hard to enlarge their personal social network, both in terms of (*i*) adding more followees and (*ii*) receiving the attention of new potential followers. Unfortunately, while the solution is straightforward in centralized social platforms, in a distributed setting more care is required.

Since frequent problems regarding data availability may completely doom a social networking platform, because the users would not get the desired content, we started modeling the issues accurately to gather insight from simulations.

## Data Availability Simulations

The problem of data availability for badly connected nodes is that some resources may not be available at a given time, and the follower experiences delays in the moment when he is able to actually get the resource. In this Section, we describe some simulations that we performed in order to quantitatively measure such delays. Wang et al. (2009) and Mega et al. (2012) dealt with accurately modeling the effects of churn in a generic P2P network or in a P2P network with social networks. However, our situation is slightly different, because we do not intend to measure the low level performance of the P2P network. Instead we are interested in the delays perceived by the social networking platform users as a macroscopic phenomenon. As a consequence, less sophisticated models of churn suffice as long as they capture the behavior of social networking users.

More formally, the average notification delay is measured as the average lag between the reception of a new message with respect to the optimal reception time. If a follower is online at the instant when a new message is published, then the optimal reception time is the instant of publication. Otherwise, the optimal reception time is the first time the node goes online again, after the publication. We consider two kinds of nodes: (*I*) nodes connecting occasionally (twice a day, for half an hour); and (*ii*) more stable nodes, which simulate usage in a collaborative work scenario (connecting for 8 hours a day). The intervals are normally distributed around the average values.

In our experiment, each user is modeled as an agent. The agents have "follows" relationships according to an input social network. For the purpose of the simulation, we make the simplifying assumption that the social network is static, because the exchange of messages occurs on a much shorter time scale than modifications to the social network.

At each step of the simulation, some agents are considered online, and a fraction of them also creates new messages. Both newly created and older non-received messages are then exchanged among online agents. Offline followers will receive the messages in successive steps when (and if) they are simultaneously online with another agent that has already received said messages.

Such model can be easily fitted to the meta-model that we described in the previous sections. The Activator decides which agents are to be activated at each step and whether they shall be online as "occasional" users or as "stable" users. Activated agents are online for that step, and if "stable" for some successive steps as well.

At each step, an online agent (*i*) has some tunable probability (in our case 0.01) of creating a new message and (*ii*) receives the messages from its online followees. The step timestamp is then compared to the creation timestamp to determine the lag that occurred before the message was actually received, that is our main measure of system performance.

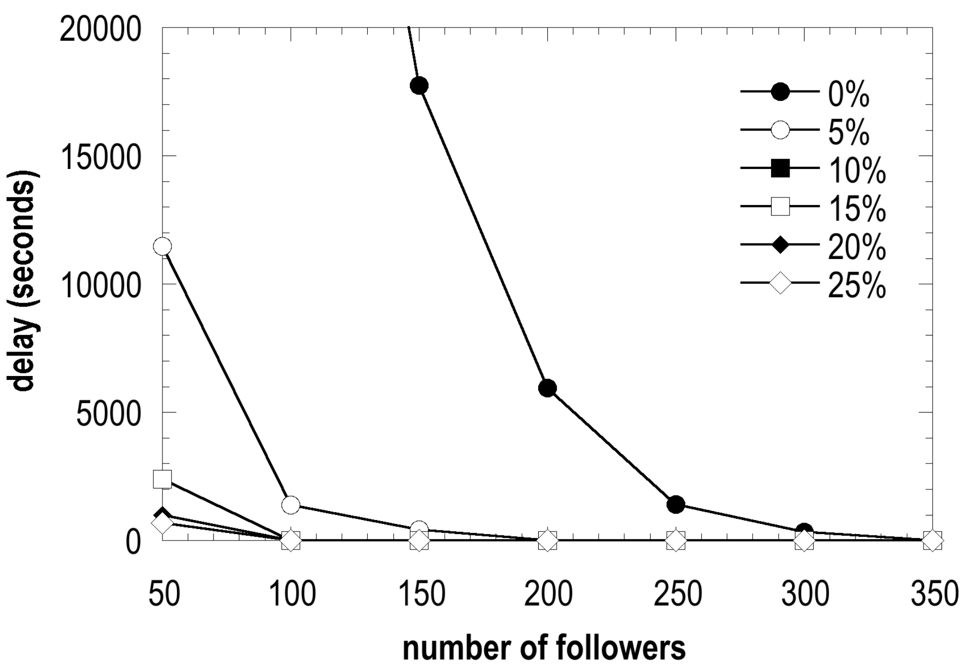


*Figure 3. Notification delay due to churn as a function of the number of followers, for different percentages (0%-25%) of stable nodes. A stable node is a node connected for ~8h a day. The solid lines are guides for the eye.*

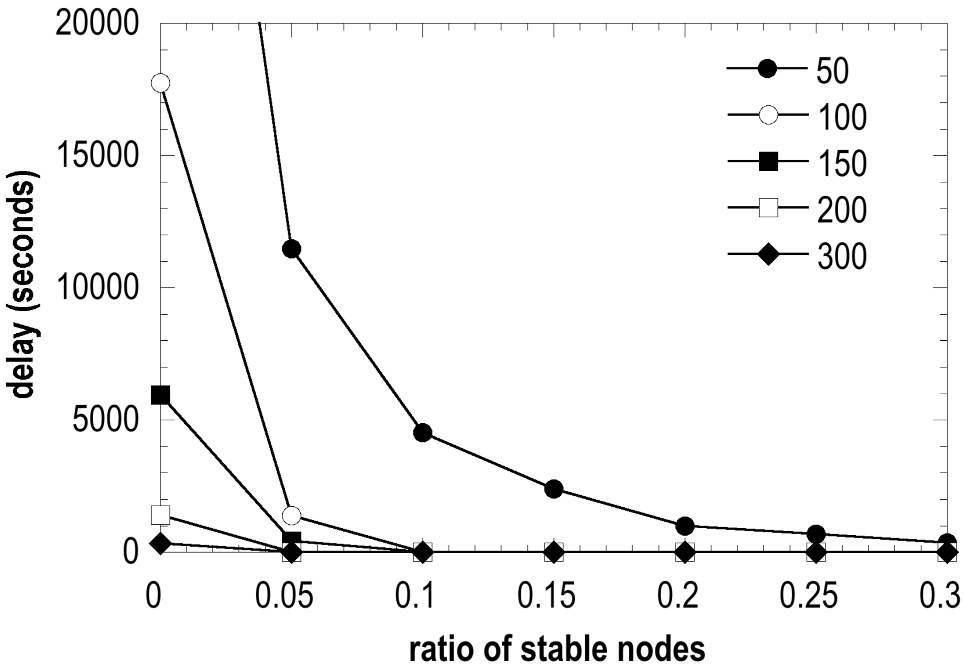


*Figure 4. Notification delay due to churn as a function of the ratio of stable nodes for different numbers of followers (50-300). A stable node is a node connected for ~8h a day. The solid lines are guides for the eye.*

We performed the experiments simulating communities of different size, from 50 to 350 agents. The plot in Figure 3 represents the notification delays due to churn as a function of the number of followers, for different percentages (0%-25%) of stable nodes. The plot in Figure 4 shows the average notification delay due to churn as a function of the ratio of stable agents for different numbers of followers (50-300).

The delays are quite severe for small communities with few stable agents, however, with 350 nodes the delay is 0 even assuming no stable nodes. On the other hand, for communities larger than 150 agents the delays are negligible with just 5% of stable nodes. Considering that in 2012 an active Twitter user has on average 235 followers and that we expect a non-null ratio of stable users, the results are encouraging (Basch, 2012). An active user is someone who created at least post in the previous 30 days.

## CONCLUSION

In this Chapter, we have presented PyNetSYM, a novel language and runtime for network specific simulations. PyNetSYM is built for the generative approach (Epstein, 1999) to science typical of agent-based modeling (ABM). We believe there is a strong need for tools that are both: (*i*) easy to use (especially for people with little programming background but with a significant expertise in other disciplines, such as social sciences) and (*ii*) able to tackle large scale simulations, using remote high-performance machines and potentially distributing the computation on multiple servers. Therefore, while our primary goal is maintaining our toolkit simple and easy to use, efficiency is our second priority, since nowadays there is a wide interest on networks of huge size.

Specifically, we created PyNetSYM: (*i*) to easily support both small and huge networks, using either in-memory and on-disk network representations, and (*ii*) to be as easy to be used both on personal machines or on remote servers.

We also used PyNetSYM to simulate the behavior of users in Blogracy, a novel fully distributed social networking platform. Our goal was to understand the condition under which the information propagates to the intended recipients. These experiments confirm our confidence on the soundness of Blogracy

architectural design and its realization over solid and widespread technologies, which are meant to provide a relatively large user-base. The results we obtained through simulation show that either (*i*) with small numbers of users (300) and no stable users or (*ii*) with some stable users and very few users the system is still expected to work. Considering that real systems have number of users orders of magnitude larger than the one simulated and that increasing the size of the network the performance improves, we expect it to be a viable option.
Moreover, the simplicity in creating simulations for novel domains shows that our approach is successful in providing a friendly and easy to use environment to perform agent-based simulations over social networks. Agent-based simulation is a powerful conceptual modeling tool for social network simulations and with the present work we contribute a natural and expressive way to specify social network simulations using a DSL.

## REFERENCES


Agha, G. (1986). *Actors: a model of concurrent computation in distributed systems*. MIT Press, Cambridge, MA, USA.

Arciero, L., Picillo, C., Solomon, S., Terna, P. (2014). Building ABMs to control the emergence of crisis analyzing agents' behavior. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.

Arthur, W. B. (1996). *Asset pricing under endogenous expectations in an artificial stock market* (Doctoral dissertation, Brunel University, London).

Arthur, W. B., Durlauf, S., and Lane, D. (1997). *The Economy as a Complex Evolving System II*. Addison-Wesley, Menlo Park.

Axelrod, R. (2003). Advancing the art of simulation in the social sciences. *Japanese Journal for Management Information System, 12*(3), 1-19.

Axelrod, R. (2007). Simulation in social sciences. In Rennard, J., editor, *Handbook of Research on Nature-inspired Computing for Economics and Management*. IGI Global.

Axelrod, R., & Tesfatsion, L. (2006). A Guide for Newcomers to Agent-Based Modeling in the Social Sciences. *Handbook of computational economics*, *2*, 1647-1659. Elsevier.

Axtel, R., Epstein, J. M., and Young, H. P. (2004). Social dynamics. In *Social Dynamics, Economic Learning and Social Evolution Series*. MIT Press, Cambridge, MA, USA..

Bagni, R., Berchi, R., and Cariello, P. (2002). A comparison of simulation models applied to epidemics. *Journal of Artificial Societies and Social Simulation, 5*(3).

Barabási, A. L. and Albert, R. (1999). Emergence of scaling in random net- works. *Science, 286*, 509–512.

Bergenti, F., Franchi, E., & Poggi, A. (2011). Selected models for agent-based simulation of social networks. In *3rd Symposium on Social Networks and Multiagent Systems (SNAMAS 2011)* (pp. 27-32).

Bergenti, F., Franchi, E., & Poggi, A. (2012). Enhancing social networks with agent and semantic web technologies. *Collaboration and the semantic web: social networks, knowledge networks, and knowledge resources. IGI Global, Hershey*, 83-100.

Bergenti, F., Franchi, E., & Poggi, A. (2013). Agent-based interpretations of classic network models. *Computational and Mathematical Organization Theory*, *19*(2), 105-127.

Boyd, D. M. & Ellison, N. B. (2008). Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication, 13*(1), 210-230.

Buchegger, S., & Datta, A. (2009, February). A case for P2P infrastructure for social networks-opportunities & challenges. In *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on* (pp. 161-168). IEEE.

Buchegger, S., Schiöberg, D., Vu, L. H., & Datta, A. (2009, March). PeerSoN: P2P social networking: early experiences and insights. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems* (pp. 46-52). ACM.

Nunzia Carbonara (2014). Knowledge-based externalities and geographical clusters: An agent-based simulation study. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global

Carley, K. M., Fridsma, D. B., Casman, E., Yahja, A., Altman, N., C., L.-C., Kaminsky, B., & Nave, D. (2006). Biowar: scalable agent-based model of bioattacks. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 36*(2), 252-265.

Castelfranchi, C. (2014). Making Visible “the Invisible Hand”. The Mission of Social Simulation. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.

Coelho, H., da Rocha Costa, A.C., & Trigo, P. (2014). On Agent Interactions Governed by Morality. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.

Cohen, B. (2008). *The bittorrent protocol specification*. http://www.bittorrent.org/beps/bep_0003.html.

Cutillo, L. A., Molva, R., and Strufe, T. (2009). Safebook: A privacy-preserving online social network leveraging on real-life trust. *Communications Magazine, 47*(12), 94-101.

Basch, D. (2012). *Some Fresh Twitter Stats (as of July 2012, Dataset Included)*. http://diegobasch.com/some-fresh-twitter-stats-as-of-july-2012.

Dean, J. S., Gumerman, G. J., Epstein, J. M., Axtell, R. L., Swedlund, A. C., Parker, M. T., & McCarroll, S. (2000). Understanding Anasazi culture change through agent-based modeling. *Dynamics in human and primate societies: Agent-based modeling of social and spatial processes*, 179-205.

Epstein, J. M. (1999). Agent-based computational models and generative social science. *Complexity, 4*(5), 41–60.

Epstein, J. M. (2002). Modeling civil violence: An agent-based computational approach. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(Suppl 3), 7243-7250.

Epstein, J. M. (2006). *Generative social science: Studies in agent-based computational modeling*. Princeton University Press.

Epstein, J. M. (1996). *Growing artificial societies: social science from the bottom up*. Brookings Institution Press.

Erdos, P. & Rényi, A. (1959). On random graphs. *Publicationes Mathematicae, 6*(26), 290-297.

Folcik, V., An, G., & Orosz, C. (2007). The basic immune simulator: An agent-based model to study the interactions between innate and adaptive immunity. *Theoretical Biology and Medical Modelling, 4*(1), 39.

Franchi, E. (2012a, August). A Domain Specific Language Approach for Agent-Based Social Network Modeling. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on* (pp. 607-612). IEEE.

Franchi, E. (2012b). Towards agent-based models for synthetic social network generation. In *Virtual and Networked Organizations, Emergent Technologies and Tools* (pp. 18-27). Springer Berlin Heidelberg.

Franchi, E., & Poggi, A. (2011). Multi-agent systems and social networks. *Business Social Networking: Organizational, Managerial, and Technological Dimensions*, 84-97.

Franchi, E., & Tomaiuolo, M. (2013). Distributed Social Platforms for Confidentiality and Resilience. *Social Network Engineering for Secure Web Data and Services*, 114.
Friesen, M.R., Gordon, R., & McLeod, R.D. (2014). Exploring Emergence within Social Systems with Agent Based Models. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.
Georgalos, K. (2014). Playing with Ambiguity: An Agent Based Model of Vague Beliefs in Games. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.
Graffi, K., Gross, C., Mukherjee, P., Kovacevic, A., & Steinmetz, R. (2010, August). LifeSocial. KOM: A P2P-based platform for secure online social networks. In *Peer-to-Peer Computing (P2P), 2010 IEEE Tenth International Conference on* (pp. 1-2). IEEE.
Hill, R., Carl, R., and Champagne, L. (2006). Using agent simulation models to examine and investigate search theory against a historical case study. *Journal of Simulation, 1*(1), 29-38.
Holme, P. and Kim, B. (2002). Growing scale-free networks with tunable clustering. *Physical Review E, 65*(2), 2-5.
Ilachinski, A. (2000). Irreducible semi-autonomous adaptive combat (ISAAC): An artificial-life approach to land combat. *Military Operations Research, 5*(3), 29-46.
Jackson, M. O. (2010). *Social and Economic Networks*. Princeton University Press. Princeton University Press.
Kohler, T. A., & Gummerman, G. J. (2001). *Dynamics of human and primate societies: agent-based modeling of social and spatial processes*. Oxford University Press.
Kohler, T. A., Gumerman, G. J., and Reynolds, R. J. (2005). Simulating ancient societies. *Scientific American, 293*, 76-84.
Kumar, R., Novak, J., & Tomkins, A. (2010). Structure and evolution of online social networks. In *Link mining: models, algorithms, and applications* (pp. 337-357). Springer New York.
Lucas, P., & Payne, D. (2014). Usefulness of agent-based simulation to test collective decision-making models. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.
Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., and Balan, G. (2005). Mason: A multiagent simulation environment. *Simulation, 81*(7), 517-527.
Magessi, N. T., & Antunes, L. (2014) Agent's risk relation on strategic tax reporting game. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global
Burkhart, R., Langton, C., & Askenazi, M. (1996, June). The swarm simulation system: A toolkit for building multi-agent simulations. Santa Fe: Santa Fe Institute.
Moffat, J., Smith, J., & Witty, S. (2006). Emergent behaviour: Theory and experimentation using the MANA model. *Journal of Applied Mathematics and Decision Sciences, 2006*, 13.
Mota, F., Santos, I., Dimuro, G. & Rosa, V. (2014). Agent-based Simulation of electric energy consumers: The NetLogo tool approach. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.
Nardin, L.G., Rosset, L.M., & Sichman, J.S. (2014). Scale and Topology Effects on Agent-Based Simulation: A Trust-Based Coalition Formation Case Study. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.
Newman, M. E. J. (2010). *Networks: An Introduction*. Oxford University Press, USA.
North, M. J., Howe, T. R., Collier, N. T., & Vos, J. R. (2007, January). A declarative model assembly infrastructure for verification and validation. In *Advancing Social Simulation: The First World Congress* (pp. 129-140). Springer Japan.

North, M. J., Macal, C. M., Aubin, J. S., Thimmapuram, P., Bragen, M., Hahn, J., Karr, J., Brigham, N., Lacy, M. E., and Hampton, D. (2010). Multiscale agent-based consumer market modeling. *Complexity, 15*(5), 37-47.

OpenSocial and Gadgets Specification Group (2012). *Open Social Specifications*. http://docs.opensocial.org/display/OSD/Specs.

Pastor-Satorras, R., & Vespignani, A. (2001). Epidemic spreading in scale-free networks. *Physical review letters*, *86*(14), 3200.

Perfitt, T., & Englert, B. (2010, May). Megaphone: Fault tolerant, scalable, and trustworthy p2p microblogging. In *Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on* (pp. 469-477). IEEE.

Santos, F., Rodrigues, T., Donancio, H., Santos, I., Adamatti, D.F., Dimuro, G.P., Dimuro, G., & De Manuel Jerez, E. (2014). Multi-Agent-Based simulation of the social production and management processes in a Urban Ecosystem: an approach based on the integration of organizational, regulatory, communication and physical artifacts in the JaCaMo framework. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.

Simon, H. A. (1955a). A behavioral model of rational choice. *The Quarterly Journal of Economics, 69*(1).

Snijders, T. A. B. (2011). Statistical models for social networks. *Annual Review of Sociology, 37*(1), 131-153.

Stroud, D. (2008). Social networking: An age-neutral commodity — Social networking becomes a mature web application. *Journal of Direct, Data and Digital Marketing Practice, 9*(3), 278-292.

Tisue, S., & Wilensky, U. (2004, May). Netlogo: A simple environment for modeling complexity. In *International Conference on Complex Systems* (pp. 16-21).

Trigo, P. (2014). Multi-agent economically motivated decision-making. In: *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*. IGI Global.

Vuze (2012). *Vuze Software*. http://www.vuze.com/.

Watts, D. J., & Strogatz, S. (1998). Collective dynamics of small-world networks. *Nature, 393*(6684), 440–442.

Wang, X., Yao, Z., & Loguinov, D. (2009). Residual-based estimation of peer and link lifetimes in P2P networks. *IEEE/ACM Transactions on Networking (TON)*, *17*(3), 726-739.

Xu, T., Chen, Y., Zhao, J., & Fu, X. (2010, June). Cuckoo: towards decentralized, socio-aware online microblogging services and data measurements. In *Proceedings of the 2nd ACM International Workshop on Hot Topics in Planet-scale Measurement* (p. 4). ACM.

Xu, T., Chen, Y., Fu, X., & Hui, P. (2011). Twittering by cuckoo: decentralized and socio-aware online microblogging services. *ACM SIGCOMM Computer Communication Review*, *41*(4), 473-474.

## TERMS AND DEFINITIONS

Agent-based model: a class of computational models for simulating interacting agents.

Multi-agent system: a loosely coupled network of software agents that interact to solve problems that are beyond the individual capacities or knowledge of each software agent.

Peer-to-peer system: a network based system in which each node can act as both client and server for the other ones of the system.

Social networking system: a network based system facilitating the building of social networks.

Software agent: a computer program that is situated in some environment and capable of autonomous action in order to meet