Studying Network Diffusion Dynamics

Giulio Rossetti, Letizia Milli, Salvatore Rinzivillo, Alina Sirbu, Dino Pedreschi and Fosca Giannotti

University of Pisa, Italy
ISTI-CNR, Italy
Diffusive Phenomena are everywhere

Epidemic Spreading, Diffusion of Innovations and Ideas, Gossip...

All those phenomena can be modeled as diffusive processes
Diffusive Processes and Networks

Diffusion implies network structure!
- It happens only when the carries of the diseases/virus/idea are connected to each other.

Diffusive phenomena can be modeled by describing "node statuses" and "transition rules".

Example SIR model:
- Three node statuses: (S)usceptible, (I)nfected, (R)ecovered
- Two transition available: S→I; I→R
Available Models

Epidemics
(10 Models)

• SI / SIS / SIR
• SEIS / SEIR
• Threshold / Profile / Profile-Threshold / Threshold-Blocked
• Independent Cascades

Opinion Dynamics
(5 Models)

• Majority Rule
• Voter / Q-Voter
• Sznajd
• Cognitive Opinion Dynamics
A framework for everyone!
Analytics as-a-service for...

... an heterogeneous audience

Researchers
Analysts
Students

... coming from several disciplines

Computer science
Physics
Social Sciences

.... having different goals!

Test Diffusive models
Perform Experiment
Visualize and Compare models

...
NDlib provide a common workflow to both programmers and analysts:

**Programmers:**
- Unified interface for several diffusion model
- Results Visualization facilities
- I/O standardization
- Extensibility

**Analysts:**
- Visual (web-based) platform
- Experiment configuration/execution
- Analytics as-a-service
A simple, unified, interface:

- Load the Graph
- Select and configure the model
- Run the simulation

All models follow the same programmatic pattern and produce standardized results

Programmer: SIR Code Example

```python
import networkx as nx
import ndlib.models.ModelConfig as mc
import ndlib.models.epidemics.SIRMModel as sir

# Network topology
G = nx.erdos_renyi_graph(1000, 0.1)

# Model selection
model = sir.SIRMModel(G)

# Model Configuration
cfg = mc.Configuration()
cfg.add_model_parameter('beta', 0.01)
cfg.add_model_parameter('gamma', 0.005)
cfg.add_model_parameter('percentage_infected', 0.05)
model.set_initial_status(cfg)

# Simulation execution
iterations = model.iteration_bunch(200)
```
ndlib.viz implements visualization facilities

**Base Viz**
- Diffusion Trends
- Prevelence

**Advanced Viz**
- Compare Models
- Multiple Run
Programmer: Describe New Models

New models can be added to NDlib easily:

1. Extend the base class `ndlib.models.DiffusionModel`

2. Specify model **metadata** (i.e., available status, required parameters)

3. Describe the **transition rules** (i.e., under which circumstances a node becomes infected?)

Metadata
- [S, I, R]
- [alpha, gamma]

Transition Rules
- S->I
- I->R
**NDlib** offers a remote experiment server that, using a REST-full API, allows to:

- Create Ndlib experiments
- Configure them
- Execute them remotely

**NDlib-REST** aims to:

1. Decouple experiment definition/execution
2. Increase scalability
Analyst: Visual Simulation
Present and Future

Dynamic Network Topologies
What happens when diffusive phenomena occur on top of a Temporal Network?

Integration

CIMPLEX (H2020)
“One tool to rule them all…” Gleam, NDlib... all under a same umbrella!

SoBigData (H2020)
NDlib-REST as-a-service!

DyNetX
Integrated in NDlib v3.0.0

Models, Visualizations, Evaluation...

Join us! Implement your model within Ndlib, propose new features
When:
Right now, NDlib v3.0.1 is out!

Where:
- Pypi: https://pypi.python.org/pypi/ndlib
- GitHub NDlib: https://github.com/GiulioRossetti/ndlib
- GitHub NDlib-REST: https://github.com/GiulioRossetti/ndlib-rest
- Documentation: http://ndlib.readthedocs.io/
- SoBigData: http://www.sobigdata.eu
For further information, collaborations, suggestions...

✉️ giulio.rossetti@isti.cnr.it