NetworkX and IGraph

Graph Analysis using Python

Jonas Auel, Sven Fritz

Uni Mannheim

October 28th 2016

Overview

1. Why Python?

Why Python?

- powerful programming language
- allows clear and concise expressions of network algorithms
- growing ecosystem of packages that provide more features
- Python is an excellent "glue" language for putting together pieces of software

NetworkX

- Network creation, manipulation, analyzation (and visualization)
- available for Python
- supported platforms: Linux/Windows/Mac
- load and store networks in standard and nonstandard data formats
- nodes can be "anything" (e.g. images)
- edges can hold arbitrary data (e.g. time series)
- open source

IGraph

- Network creation, manipulation, analyzation and visualization
- available for C/R/Python
- supported platforms: Linux/Windows/Mac
- collection of graph analysis tools
- emphasis on efficiency, portability, ease of use
- open source

Graph types in NetworkX

Graph type	NetworkX class	IGraph class
Undirected	Graph	Graph
Directed	DiGraph	Graph
With self-loops	Graph, DiGraph	Graph
With parallel edges	MultiGraph, MultiDiGraph	Graph

Betweenness centrality

- Betweenness centrality of a node v: sum of the fraction of all-pairs shortest paths that pass through v
- $$c_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}$$
- V: set of nodes,
  $\sigma(s,t)$: number of shortest (s,t)-paths,
  $\sigma(s,t|v)$: number of those paths passing through some node v other than s, t
- if $s = t$, $\sigma(s,t) = 1$, and if $v \in s, t$, $\sigma(s,t|v) = 0$.