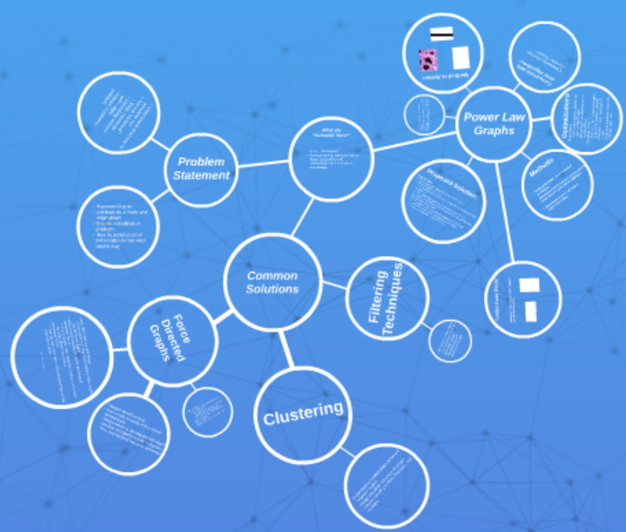
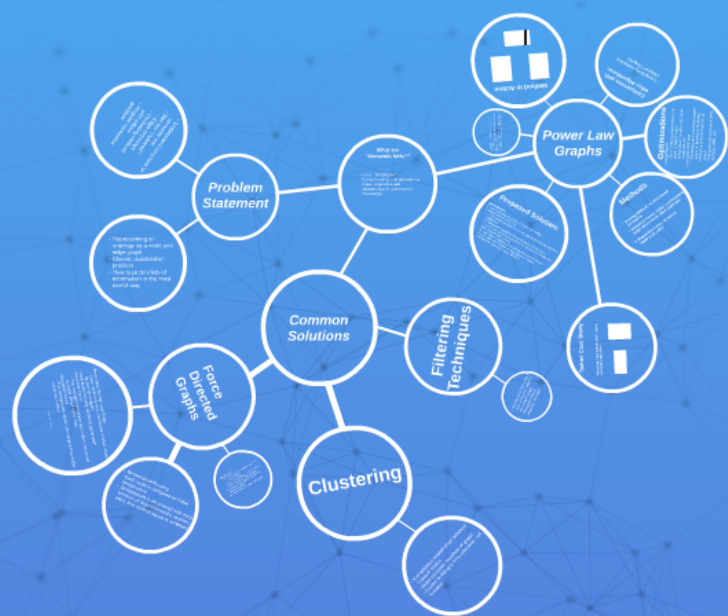
Scalable Visualization of Semantic Nets Using Power Law Graphs

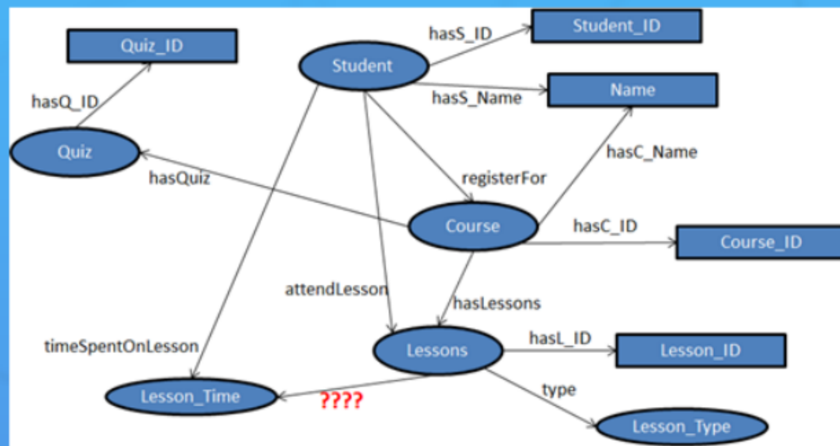


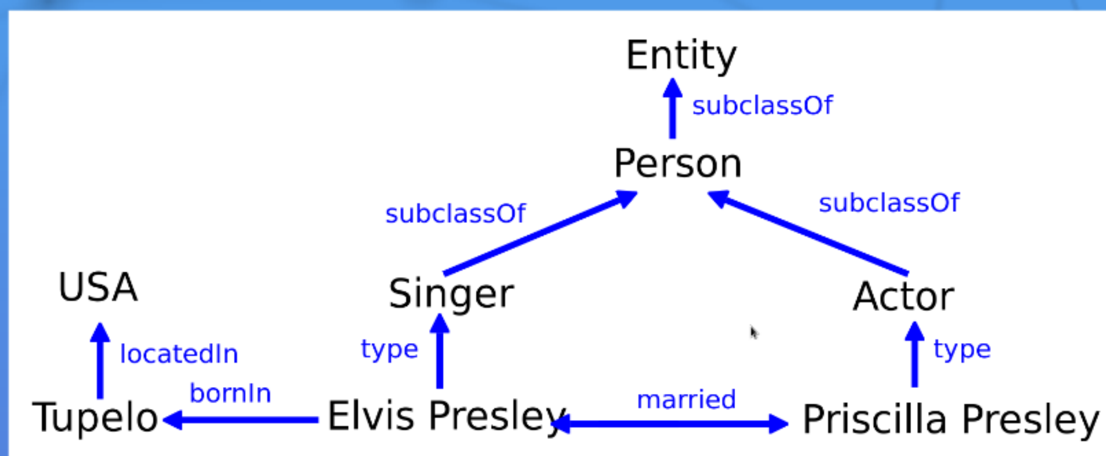
Scalable Visualization of Semantic Nets Using Power Law Graphs



What are “Semantic Nets?”

- A.k.a. “Ontologies”
- Formal naming and definition of types, properties and relationships in a domain of knowledge.


Quiz_ID
Student
hasS_ID
Student_ID
hasQ_ID
hasS_Name
Name
Quiz
hasQuiz
registerFor
hasC_Name
Course
hasC_ID
Course_ID
attendLesson
hasLessons
timeSpentOnLesson
Lessons
hasL_ID
Lesson_ID
Lesson_Time
????
type
Lesson_Type


Entity
subclassOf
Person
subclassOf
subclassOf
USA
Singer
Actor
locatedIn
type
type
bornIn
married
Tupelo
Elvis Presley
Priscilla Presley

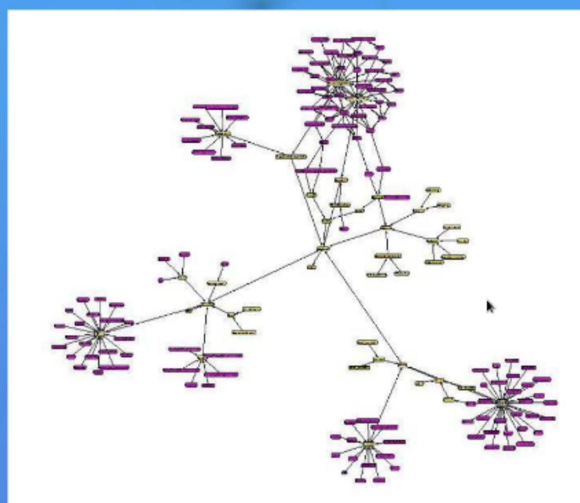
Problem Statement

- Representing an ontology as a node and edge graph
- Classic visualization problem
- How to picture lots of information in the most useful way

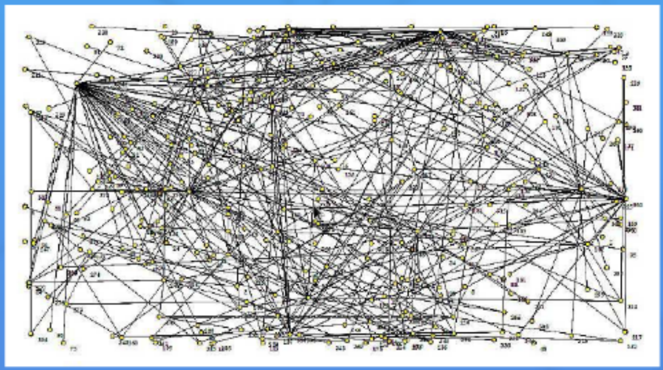
- Understand structure of semantic net
- Not too “cluttered”
  - Edge crossings
  - Occluding vertices with edges
  - Angular resolution problem

Amino Acid Ontology

Good Layout



Bad Layout



Common Solutions

Force Directed Graphs

- Basic idea: Springs and Rings
  - Each node is a ring, connected to other nodes by springs (edges)
- Initial layout usually randomly generated*
- Attractive Force
  - The strength with which two nodes connected by edge attract each other
- Repulsive Force
  - The strength with which non-neighboring nodes repel each other

http://bl.ocks.org/mbostock/1062288

nodes

- Temperature/Cooling
- Each node is assigned an initial temperature
- Temperature is decreased with each iteration of algorithm until it reaches zero, and optimal layout is achieved

- Major issues
  - Occluding vertices with edges on complex graph
  - Angular resolution, edge crossing
  - Not scalable for large graphs
- Repulsive force requires O(|V|*|V|) computation time
- Not well-suited for large ontologies

Clustering

- Extrapolates relationships between “related” nodes
- Helps elucidate structure of graph
- Cluster building is time intensive, not scalable

Filtering Techniques

- Get rid of certain edges and vertices in order to de-clutter* graph
- Preserve overall structure of graph

Power Law Graphs

- As degree of node increases, frequency decreases
- Few “core” nodes with many edges with many non-core nodes
- Many graphs of interest follow this distribution
  - Interaction networks and ONTOLOGIES

Proposed Solution:

General Idea

1. Sort nodes by degree to extract core nodes
2. While temperature != 0
    a. Calculate attraction force among Power-Nodes and their neighbors
    b. Calculate Repulsive Force among Power-Nodes
    c. Calculate attraction force among Non-power Nodes and their neighboring Nodes
    d. Calculate repulsive force among Non-power Nodes
    e. Calculate and Update (x,y) position of nodes
    f. Reduce temperature each iteration

Methods

- **Scaling Method: re-size based on degree**
- **Attraction Force: nodes connected by edge attract each other (springs)**
- **Repulsion Force: all nodes repel each other**

$$\sigma_i = \left[\frac{d_i}{\Delta(G)} \right] \times \kappa \qquad (1)$$

where,

σ_i = scale of node i.
d_i = degree of node i.
$\Delta(G)$ = maximum degree of graph G.
and $\sigma_i \leq \kappa$; where κ is a defined constant.

Methods

- **Scaling Method: re-size based on degree**
- **Attraction Force: nodes connected by edge attract each other (springs)**
- **Repulsion Force: all nodes repel each other**

Algorithm 1: AttractionForce

Data: n $\longrightarrow$ *node*; d $\longrightarrow$ *degree*;
N $\longrightarrow$ *Nodes*; E $\longrightarrow$ *Edges*; k $\leftarrow$ *StretchConstant*;

Input : The graph $G < N, V >$ and $<$ n, d $> \longrightarrow$ set of node-degree pairs;

Description: Attraction force among connected nodes, by updating their (x, y) coordinates to bring them closer to each other.

begin
    for $i \leftarrow 1$ **to** $|N|$ **do**
        for $j \leftarrow 1$ **to** $|E_{n_i}|$ **do**
            $n_1 \leftarrow i$ and $n_2 \leftarrow$ Other end node of n_1
            $\Delta x \leftarrow n_{1x} - n_{2x}$
            $\Delta y \leftarrow n_{1y} - n_{2y}$
            $\text{Length} \leftarrow \sqrt{\Delta x \times \Delta x + \Delta y \times \Delta y}$
            $\text{force} \leftarrow \frac{\text{Length} - k}{k \times (100)}$
            $d_x \leftarrow \text{force} \times \Delta x$
            $d_y \leftarrow \text{force} \times \Delta y$
            $n_{1x} \leftarrow n_{1x} - d_x$
            $n_{1y} \leftarrow n_{1y} - d_y$
            $n_{2x} \leftarrow n_{2x} + d_x$
            $n_{2y} \leftarrow n_{2y} + d_y$
        end
    end
end

Methods

- Scaling Method: re-size based on degree
- Attraction Force: nodes connected by edge attract each other (springs)
- Repulsion Force: all nodes repel each other

• Tempera
• Core n
given
te

Algorithm 2: RepulsionForce

Data: n $\longrightarrow$ *node*; d $\longrightarrow$ degree;
N $\longrightarrow$ *Nodes*; E $\longrightarrow$ Edges ;
k $\longrightarrow$ Repulsion Constant;
$d_x \longrightarrow$ distance co-efficient of n_1;
$d_y \longrightarrow$ distance co-efficient of n_2;R = Random Value;
$\lambda \longrightarrow$ a constanct initially set to 700;
Input : $<$ n,d $> \longrightarrow$ nodes along their degrees;
Description: Repulsive force between non-connected nodes, by updating their (x,y) coordinates to move them away from each other.

begin
    for $i \leftarrow 1$ **to** N **do**
        $n_1 \leftarrow i$
        for $j \leftarrow i+1$ **to** N **do**
            $n_2 \leftarrow j$ $d_x = 0$ and $d_y = 0$
            $\Delta x \leftarrow n_{1\,x} - n_{2\,x}$
            $\Delta y \leftarrow n_{1\,y} - n_{2\,y}$
            Length $\leftarrow \sqrt{\Delta x \times \Delta x + \Delta y \times \Delta y}$
            if Length *equal to 0* **then** ;
            `// Collision Detection`
                $d_x = R$ and $d_y = R$
            end
        end
        else if Length $< \lambda^2$ **then** ; `// Distance Limit`
            $d_x \leftarrow \frac{\Delta x}{\text{Length}}$ and $d_y \leftarrow \frac{\Delta y}{\text{Length}}$
        end
        force $\leftarrow \frac{(n_{1k} \times n_{2k})}{80}$
        $n_{1x} \leftarrow n_{1x} + d_x *$ force
        $n_{1y} \leftarrow n_{1y} + d_y *$ force
        $n_{2x} \leftarrow n_{2x} - d_x *$ force
        $n_{2y} \leftarrow n_{2y} - d_y *$ force
    end
end

Methods

- **Scaling Method: re-size based on degree**
- **Attraction Force: nodes connected by edge attract each other (springs)**
- **Repulsion Force: all nodes repel each other**

• Tempera
• Core n
given
te

Optimizations

- Temperature
  - Core nodes (power nodes) are given a higher initial temperature, to allow for more readjustment
- Semantic Filtering
  - Removing non-essential edges/ nodes to decrease cluttering
  - Preserve overall structure of graph
  - Structural primitives from XML, RDF(S), OWL, etc.

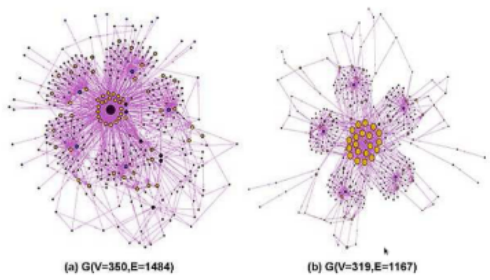


Fig. 5: Semantic filtration (a) Unfiltered graph, (b) Filtered graph

Table 1: Filtration statistics on nodes and edges

Triples	Unfiltered Graph		Filtered Graph	
	Nodes	**Edges**	**Nodes**	**Edges**
1,515	474	1,515	246	1,245
5,527	3,045	5,527	1,738	3,467
7,330	3,090	7,330	1,052	2,149
10,893	5,937	10,893	3,446	6,830
16,229	8,697	16,629	5,097	10,250
47,003	34,291	47,003	11,767	23,490

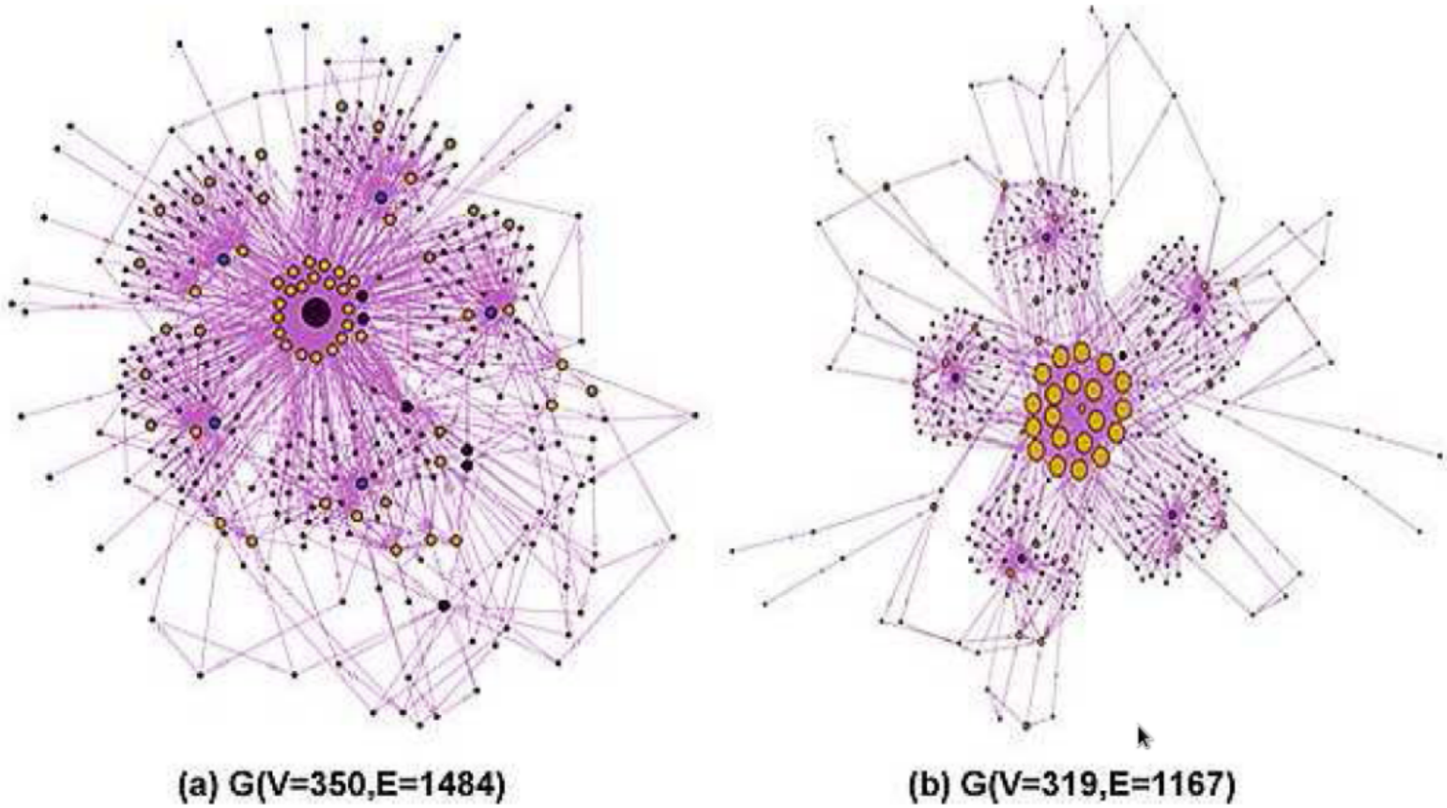


Fig. 5: Semantic filtration (a) Unfiltered graph, (b) Filtered graph

Table 1: Filtration statistics on nodes and edges

Triples	Unfiltered Graph		Filtered Graph	
	Nodes	Edges	Nodes	Edges
1,515	474	1,515	246	1,245
5,527	3,045	5,527	1,738	3,467
7,330	3,090	7,330	1,052	2,149
10,893	5,937	10,893	3,446	6,830
16,229	8,697	16,629	5,097	10,250
47,003	34,291	47,003	11,767	23,490

Comparison with other Algorithms:

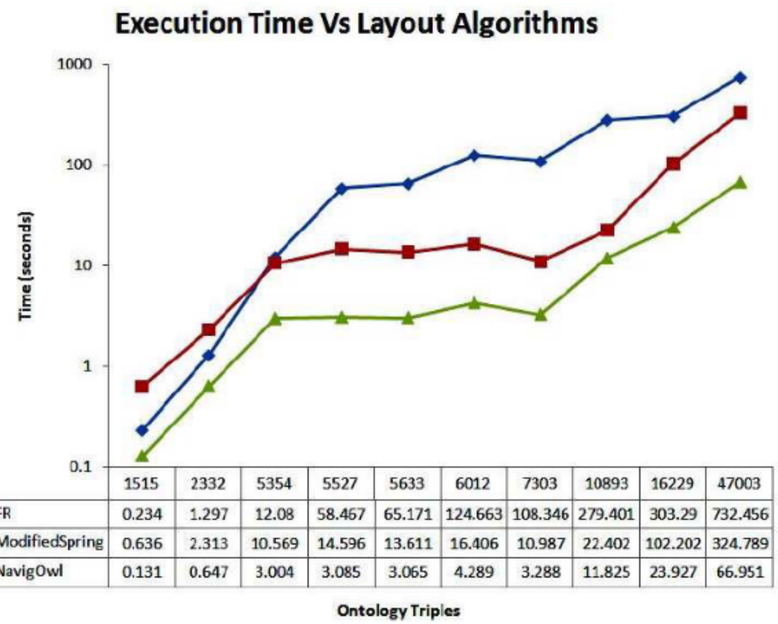
- Complexity reduction
- Clearer* Graphs

Improvement (?)

Attractive Force $\Rightarrow \Theta(|V_p||E_p|)$
Repulsive Force $\Rightarrow \Theta(|V_p{}^2|)$
Forces Complexity $\Rightarrow \Theta(|V_p| \cdot (|V_p| + |E_p|))$

- $V_p \rightarrow$ Number of Power Nodes.
- $E_p \rightarrow$ Number of Edges connected to Power Nodes
- Moreover, $V_p \ll V$ and $E_p \ll E$

Improvement



	1515	2332	5354	5527	5633	6012	7303	10893	16229	47003
FR	0.234	1.297	12.08	58.467	65.171	124.663	108.346	279.401	303.29	732.456
ModifiedSpring	0.636	2.313	10.569	14.596	13.611	16.406	10.987	22.402	102.202	324.789
NavigOwl	0.131	0.647	3.004	3.085	3.065	4.289	3.288	11.825	23.927	66.951

Fig. 10: Comparison of time to layout (in logrithmic scale) of various graph layout algorithms.

ations

c) are

Comparison with other Algorithms:

- Complexity reduction
- Clearer* Graphs

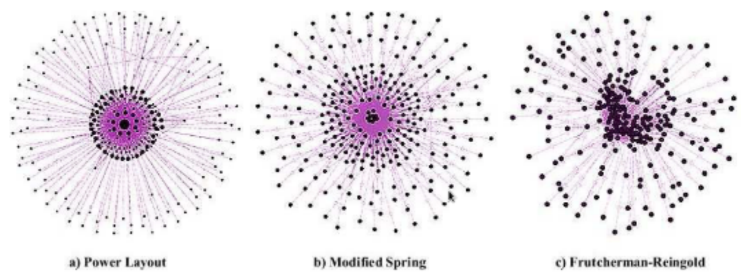


Fig. 11: Layout comparison on OCW Ontology of 1,515 triples filtered graph G(V=246,E=1,245).

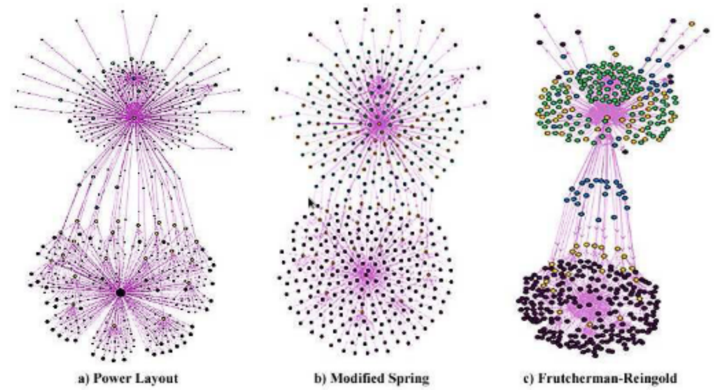


Fig. 12: Layout comparison on Food ontology of 870 triples filtered graph G(V=339,E=604).

Comparison with other Algorithms:

- Complexity reduction
- Clearer* Graphs

Method in Action

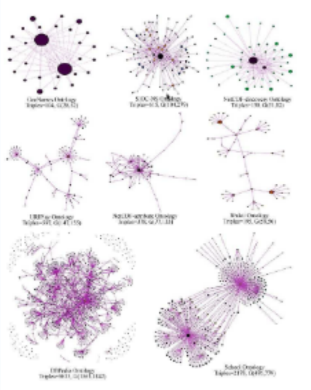


Fig. 13: Symmetrical and clustered graphs of small ontologies.

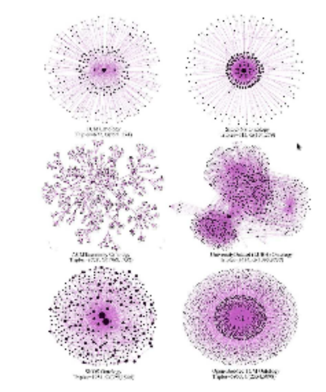


Fig. 14: Large scale symmetrical, dense, clustered visualizations.

Table 2: NavigOwl Results on power-layout

Ontology	Triples	$\lvert V \rvert$	$\lvert E \rvert$	*Time(s)*
GeoNames	104	28	52	0.037
TransOntology Bhakti	195	58	56	0.042
IRI Library CF	378	77	133	0.047
URIplay	597	147	155	0.232
SIOC-NS	615	104	279	0.039
SKOS	1,954	399	1,544	0.146
School	2,178	476	779	0.231
University (LUBH)	5,454	1,095	3,737	2.103
DBPedia	5,633	1,563	1,842	3.198
Barton Subgraph	5,863	1,902	3,691	4.593
Open-BioMed TCM	5,950	2,554	5,098	6.768
TDWG Geography	7,303	1,052	2,149	3.807
LOID OrdnanceSurvey	47,003	11,767	23,490	17.595

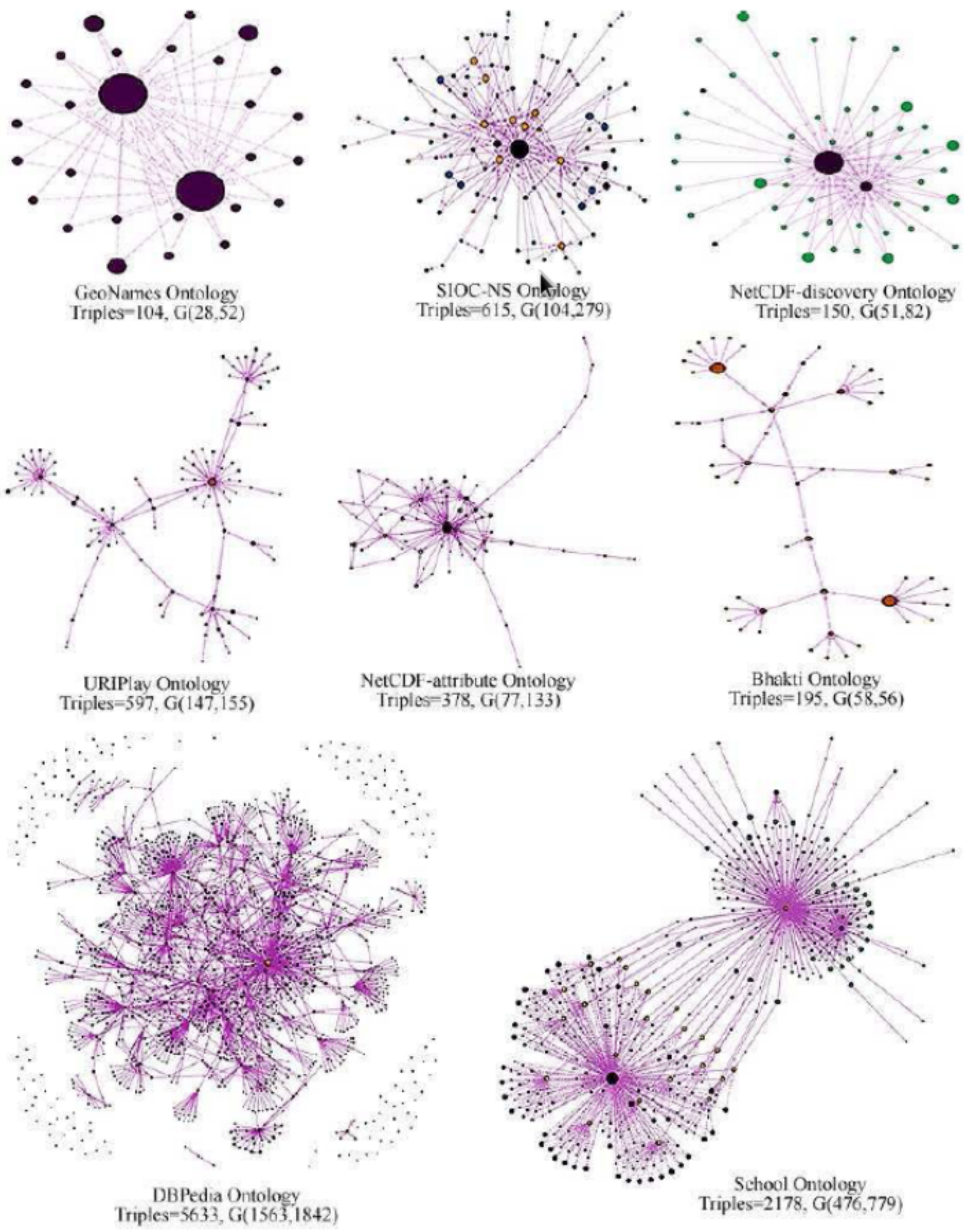


Fig. 13: Symmetrical and clustered graphs of small ontologies.

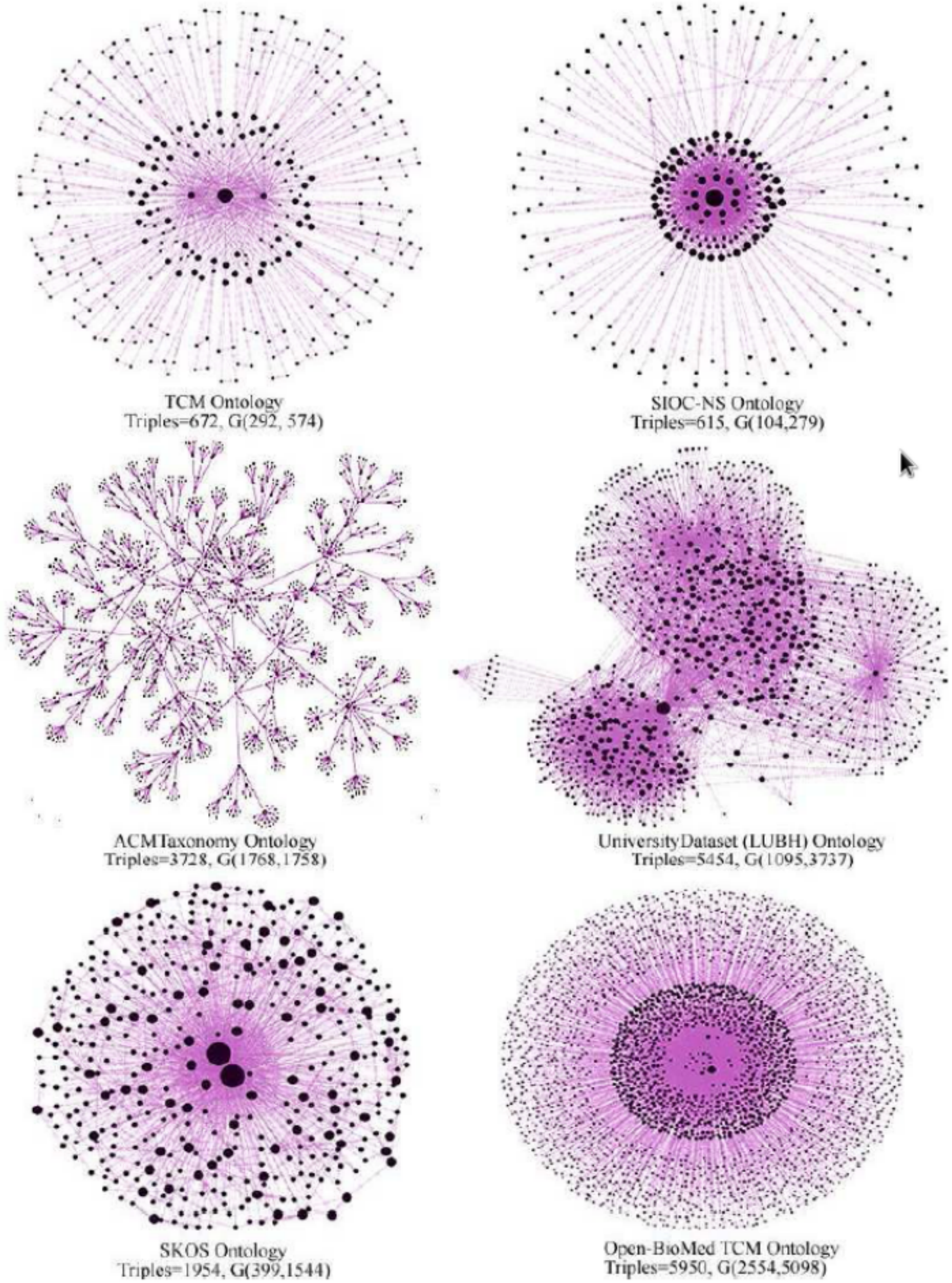


Fig. 14: Large scale symmetrical, dense, clustered visualizations.

Table 2: NavigOwl Results on power-layout

Ontology	**Triples**	$|V|$	$|E|$	$Time(s)$
GeoNames	104	28	52	0.037
TransOntology Bhakti	195	58	56	0.042
IRI Library CF	378	77	133	0.047
URIplay	597	147	155	0.232
SIOC-NS	615	104	279	0.039
SKOS	1,954	399	1,544	0.146
School	2,178	476	779	0.231
University (LUBH)	5,454	1,095	3,737	2.103
DBPedia	5,633	1,563	1,842	3.198
Barton Subgraph	5,863	1,902	3,691	4.593
Open-BioMed TCM	5,950	2,554	5,098	6.768
TDWG Geography	7,303	1,052	2,149	3.807
LOID OrdnanceSurvey	47,003	11,767	23,490	17.595

Twitter Case Study

- Modeling "who follows who" tuples using this algorithm

Table 4: Mapping of Twitter dataset to ontology schema.

Dataset Records	**Ontology Triples**	$\lvert V \rvert$	$\lvert E \rvert$
5,000	532	280	528
10,000	906	473	902
15,000	5,393	2,706	5,389
20,000	11,346	5,663	11,342
30,000	20,533	10,250	20,529
40,000	28,504	14,161	28,500
50,000	36,230	17,929	36,226
60,000	42,649	21,004	42,645

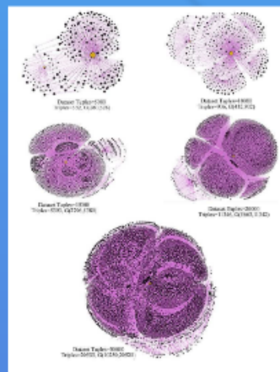


Fig. 15: Twitter Dataset Visualizations on NavigOWL.

Table 3: Tuples represnting *'who follows who?'* in Twitter

Twitter User ID	Twitter Follower ID
6353282	783214
6633812	6353282
7017692	6633812
14951565	7017692
14681199	7017692
8195652	14681199
15015170	8195652
68998614	15015170
3785461	68998614
40887009	3785461
53268444	40887009
——	——

Table 4: Mapping of Twitter dataset to ontology schema.

Dataset Records	**Ontology Triples**	$\|V\|$	$\|E\|$
5,000	532	280	528
10,000	906	473	902
15,000	5,393	2,706	5,389
20,000	11,346	5,663	11,342
30,000	20,533	10,250	20,529
40,000	28,504	14,161	28,500
50,000	36,230	17,929	36,226
60,000	42,649	21,004	42,645

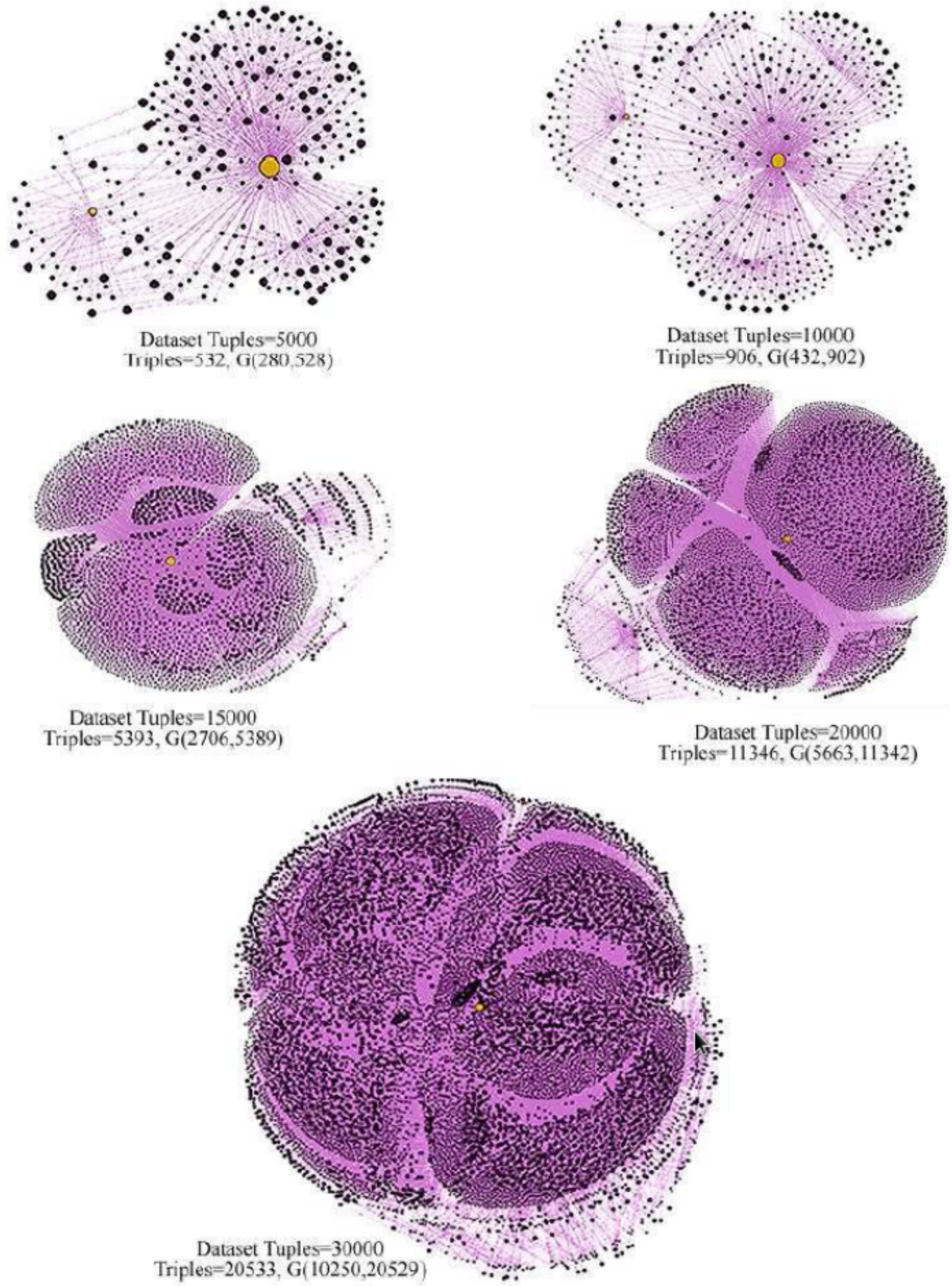


Fig. 15: Twitter Dataset Visualizations on NavigOWL.

Scalable Visualization of Semantic Nets Using Power Law Graphs