Socrates: A System for Scalable Graph Analytics

C. Savkli, R. Carr, M. Chapman, B. Chee, D. Minch

September 10, 2014

Cetin Savkli

Cetin.Savkli@jhuapl.edu

240 228 0115

JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

Challenges of Big Data & Analytics

Scalability Challenges

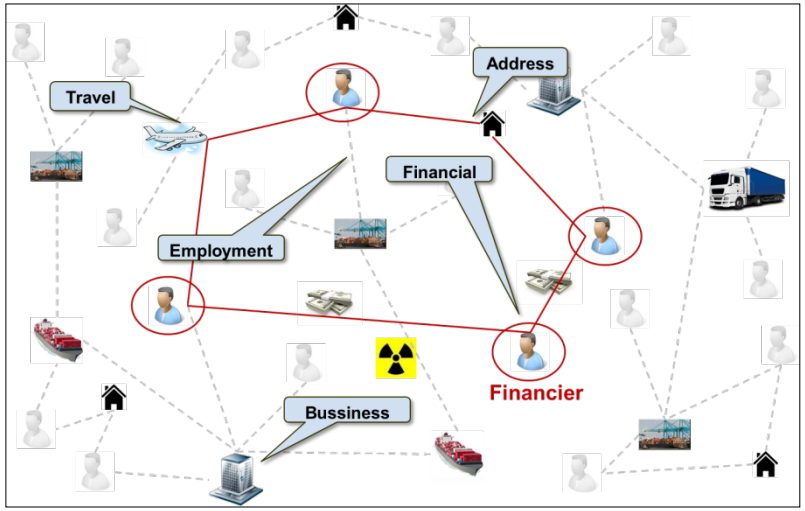
- **Storage:** Ability to handle ever growing data sizes
- **Processing:** Need a parallel processing capability that does not move data

Complexity Challenges

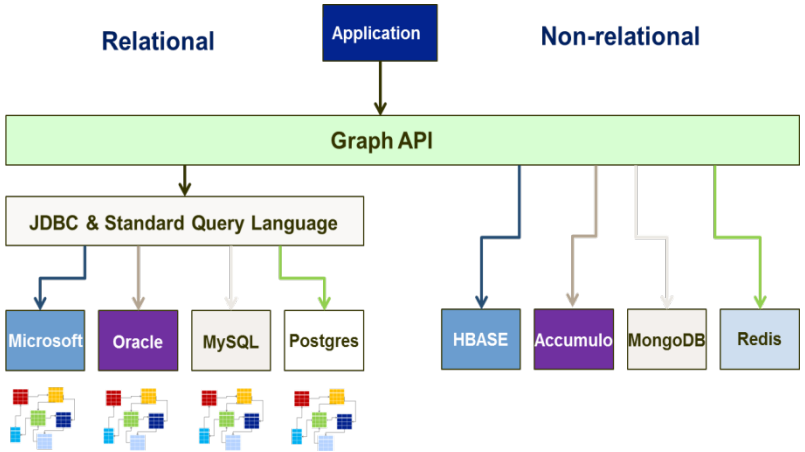
- **Data Complexity:** Often overlooked; handling changing data structures is one of the key bottlenecks
- **Software Complexity:** Ability to integrate new software and services is another challenge that determines how well the system can evolve
- **Analytics:** Making Sense and learning from complex data using mathematically robust methods.

The Role of Graphs in Big Data & Analytics

- **Graph as a Data Structure:**
  - **A flexible structure with power in expressing relationships**
  - **Facilitates integration of new data without having to deal with complex schema**
  - **Facilitates enriching data with analytics during and after data ingest- *Question Focused Data (QFD)***



- **Graph as an API:**
  - **De-couple Apps from underlying structure of data**
  - **Provides a standard interface that unifies access to non-relational as well as relational data bases**
  - **Allows higher level analysis**

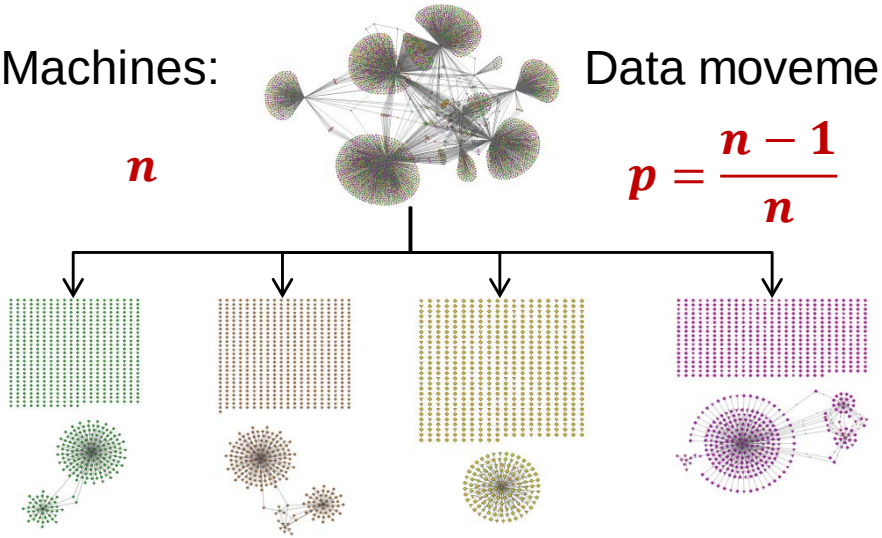


Challenges and Motivation for Graph Processing

Challenges:

- Indexing
- Locality Control
- Attribute Based Queries
- Structure Based Queries

Machines: n



Data movement: $p = \frac{n-1}{n}$

Need a solution that provides:

- Indexing & query capability of RDBMS
- Parallel processing & scalability of NOSQL

Additional Desirable Features:

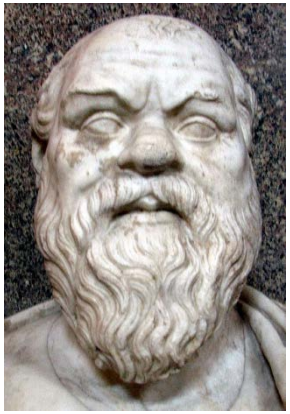
- Platform independent and simple to deploy
- Simple paradigm for development of parallelized processing

- Facilitates rapid prototyping of scalable analytics in diverse problem domains

Socrates (The Philosopher)

Socrates: 470 BC – 399 BC, Athens

- **Development of scientific method : Series of questions & answers to arrive at the truth.**
- **Executed by poisoning for asking too many questions and failing to acknowledge the gods.**



"I know that I know nothing“, Socrates.

Socrates (Software) Overview

- **Socrates general features**
  - **Stores graphs on distributed MySQL with the TokuDB engine**
  - **Indexing & query capability of RDBMS**
  - **Parallel processing & distributed storage**
  - **Locality control for partitioning data**
  - **Platform independent (runs in JVM)**
  - **No centralized management (all machines are equal)**

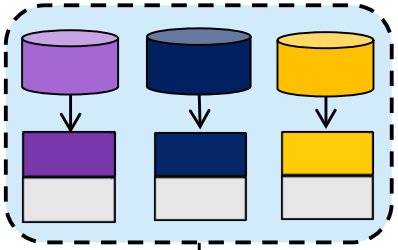
- **Socrates is under development, features not essential to most prototypes are not yet implemented**
  - **Transaction guarantees, backups, failover**
  - **If prototype needs to move into production, can reuse most/all code by switching to other Blueprints API implementation**
    - **Accumulo (APL developed)**
    - **Titan (Hbase, Cassandra, BerkleyDB)**

Socrates Overview

Analyst
OWF Web Based Interface

Scientist , Engineer
Analytics Development

Data Sources
(Inputs):
Oracle DB CSV File PCAP data

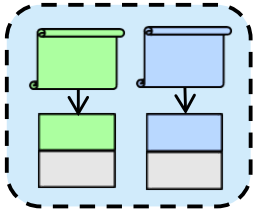


Display

User
Interfaces
& Alerting

Analytic
Services

Analytic Outputs:
Activity Patterns, Anomaly
Detection, …



Entity &
Link
Extraction

Ingest

Enrichment

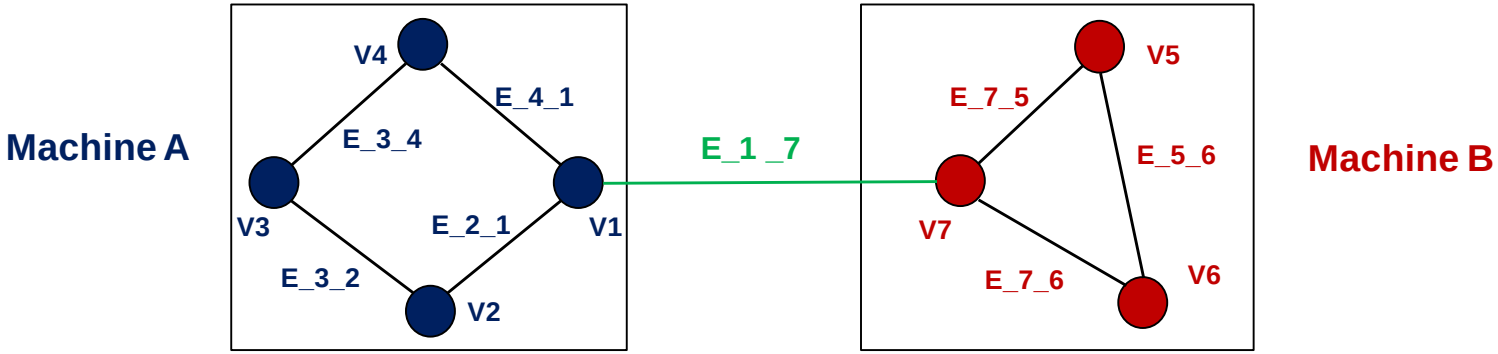
Entity 1	Link	Entity 2
Person A	Attends	Flight School B
CreditCard C	ReferencedIn	TravelRecord X
Person B	ReferencedIn	TravelRecord X

Distributed Graph Database

Graph Structure

Socrates uses the following conventions:

- ***Each graph vertex exists only on 1 machine and each graph edge can exist at most on 2 machines***
- ***Edges know the IDs of vertices they connect as well as the machine those nodes reside on.***
- ***There is no central management of location information.***



Vertices:

ID
V1
V2
V3
V4

Edges:

F	FID	EID	TID	T
A	V2	E_2_1	V1	A
A	V3	E_3_2	V2	A
A	V3	E_3_4	V4	A
A	V4	E_4_1	V1	A
A	V1	E_1_7	V7	B

Vertices:

ID
V5
V6
V7

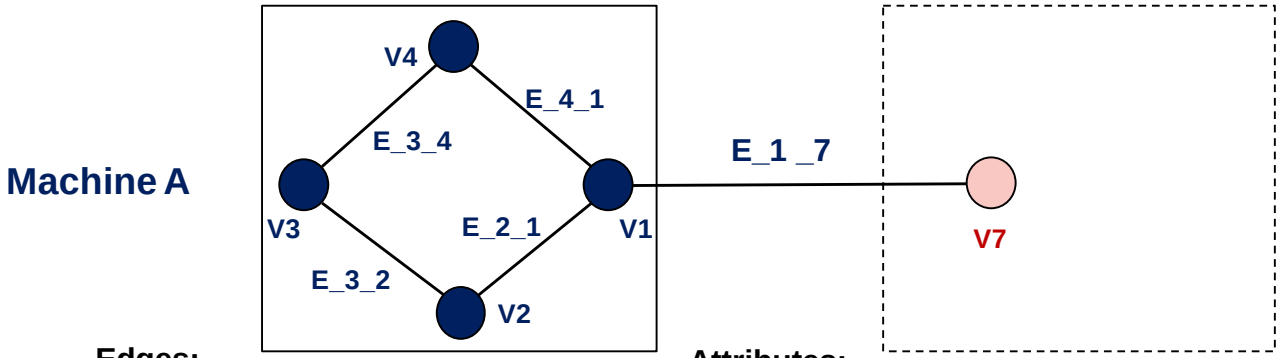
Edges:

F	FID	EID	TID	T
B	V7	E_7_5	V5	B
B	V5	E_5_6	V6	B
B	V6	E_7_6	V7	B
A	V1	E_1_7	V7	B

Indexing Graph Attributes

Socrates stores vertex and edge attributes following the following principles:

- ***Every attribute type is stored in its own two column table (primary key & value)***
- ***All attributes of a vertex resides on the machine where the vertex is. Attributes of an edge exist only on the machine where an edge originates.***
- ***Attributes are typed and the type information is automatically determined.***
- ***Attributes are indexed automatically by default with the exception of text and blob attributes.***



Vertices:

ID
V1
V2
V3
V4

Edges:

F	FID	EID	TID	T
A	V2	E_2_1	V1	A
A	V3	E_3_2	V2	A
A	V3	E_3_4	V4	A
A	V4	E_4_1	V1	A
A	V1	E_1_7	V7	B

Attributes:

A1

ID	V
V1	
V2	
V3	
V4	

A2

ID	V
V1	
V2	
V3	
V4	

A3

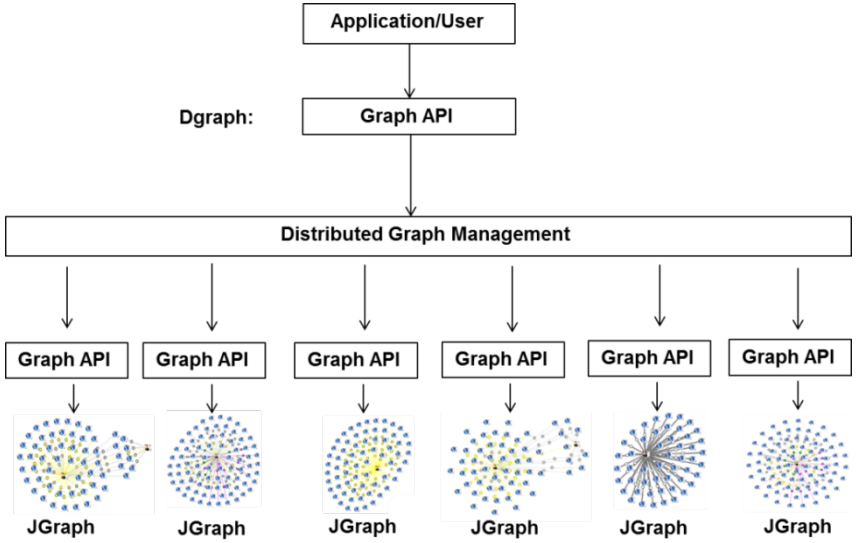
ID	V
E_2_1	
E_3_4	
E_4_1	
E_3_2	
E_1_7	

. . .

Interacting with Socrates

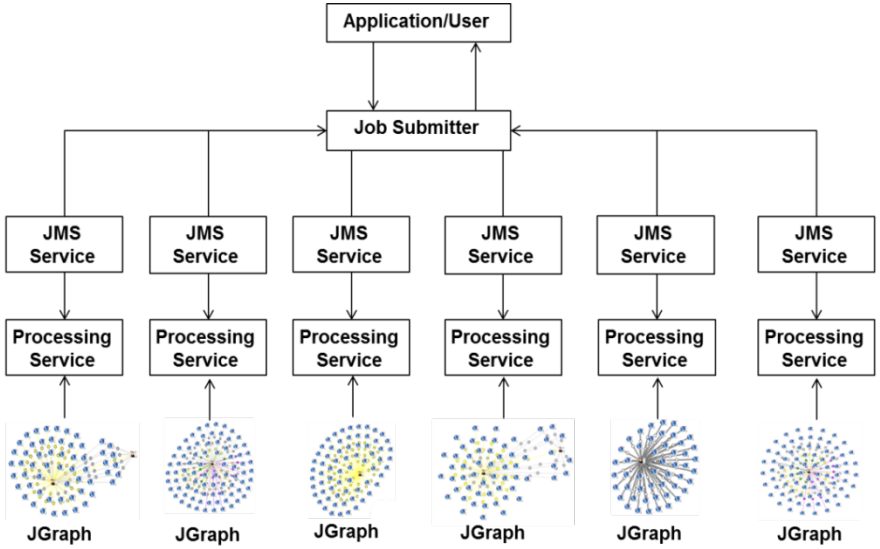
Data Access:

- Interaction with the cluster using graph API and graph query methods
- Communication provided by JDBC
- Each machine runs a server that processes the query in parallel



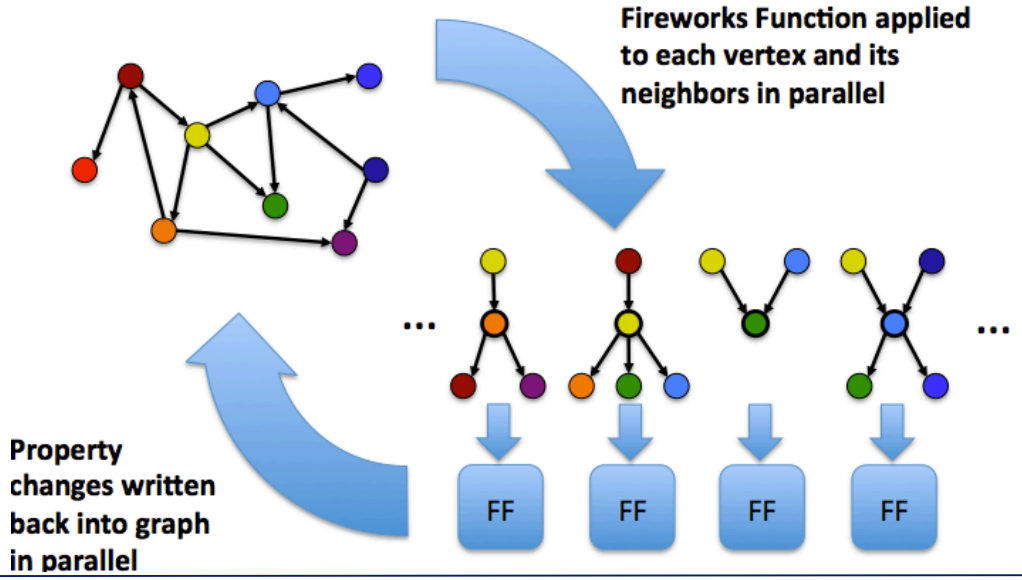
Processing:

- Submitting jobs that are executed in parallel on local graphs
- Communication provided by JMS
- Jobs are executed in parallel on each machine and results are merged at the client



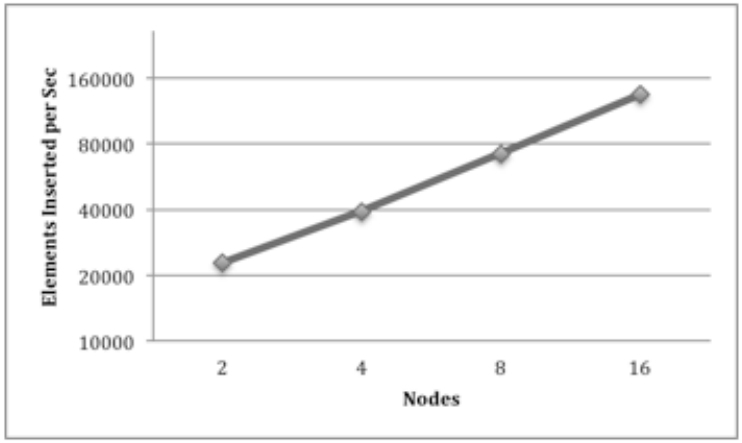
Interacting with Socrates, Cont'd

- Lowest level of parallelism: Fireworks
- Runs a given function on every vertex of the graph and its immediate neighborhood
  - Results written back into the graph
- Socrates takes care of multithreading, optimizing database queries, inserts, etc. in the background
  - Uses JGraph parallelism to run on every cluster node in parallel
  - Easily parallelize algorithms that can be performed with only local operations (PageRank, etc.)

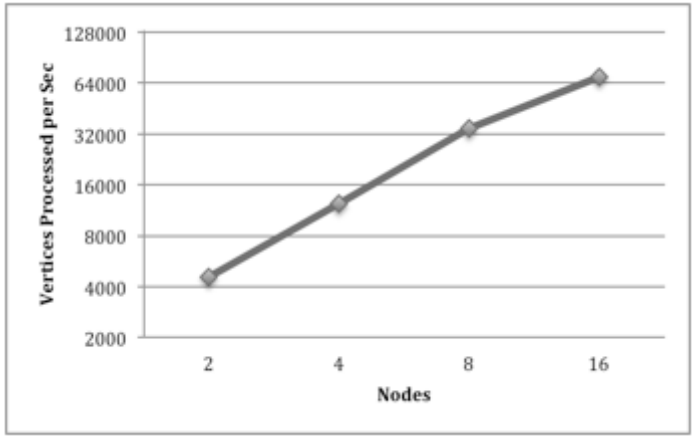


Socrates Cluster

- **APL test cluster: 16 nodes, Intel Xeon E5-2609 2.5GHz 4-core processors, 64 GB RAM, two 4.0TB HDDs in RAID 0**
  - **MySQL 5.5 with TokuDB storage engine as Socrates data store**
    - **InnoDB ingest hit severe bottleneck at about ~100m graph elements, TokuDB has scaled to over 1b elements**



Ingest speed (1.1 x 10^8 elements)

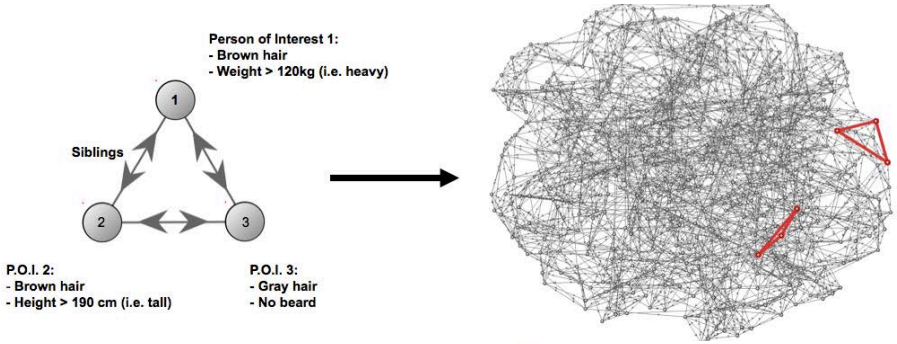


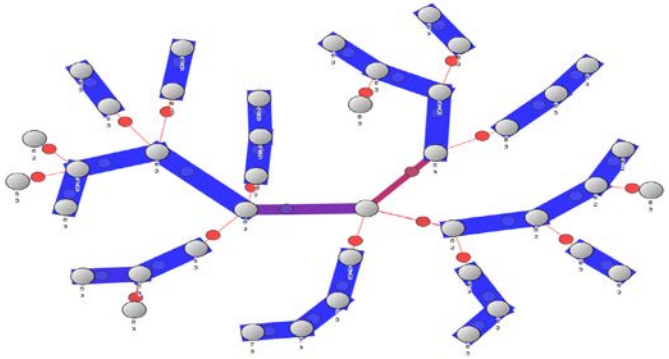
Fireworks connected component processing speed (1.1 x 10^8 elements)

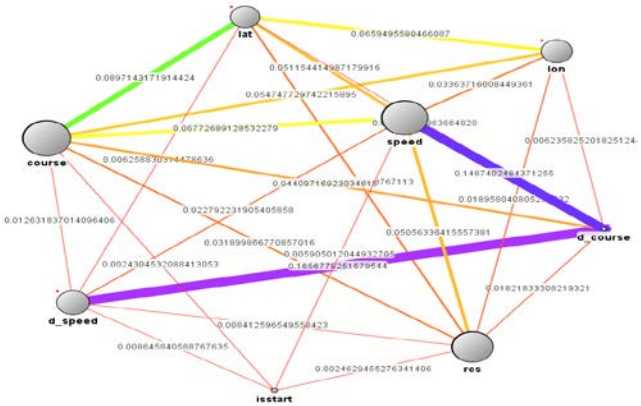
- **Primarily used to test scaling of algorithms**

General Algorithms Support Rapid Development

- Connected components
- Subgraph isomorphism
- Frequent subgraph mining
- Time series aggregation
- Activity pattern analysis
- Mutual information
- Clustering
- Anomaly detection
- Colocation analysis

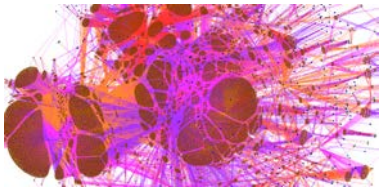




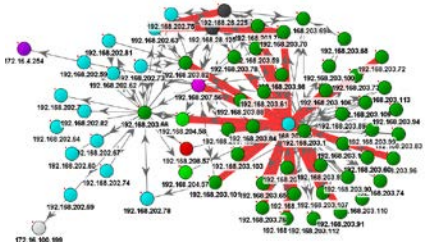


Examples of Analytics Developed in Socrates

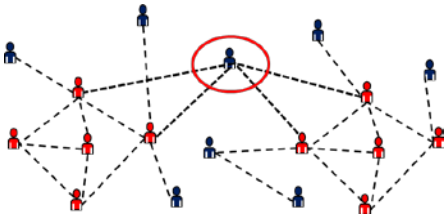
- **Detection of communities and trade patterns in international cargo shipments.**



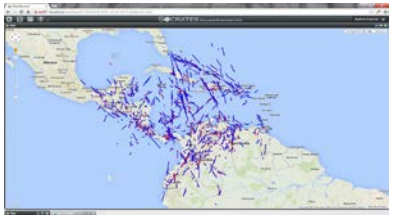
- **Analysis of cyber networks to detect anomalous activities, including identification of collaborative behavior.**



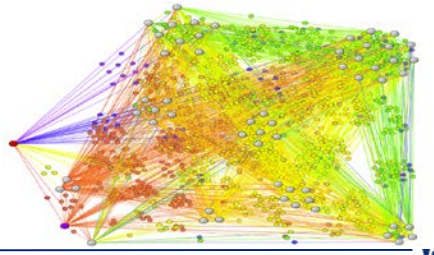
- **Discovered previously unknown associates of person of interest hidden in travel activity data**



- **Detected anomalous flight behaviors such as formation flying, loitering.**



- **Analyze genetic data to determine gene measurements correlated with various diseases**



JOHNS HOPKINS
APPLIED PHYSICS LABORATORY