Societies, Networks, Big Data, Graphs and Algorithms

The Golden Ages of Graphs

Dr. Eng. Didier El Baz
LAAS-CNRS
Toulouse France
International Lab on Security of
Cyber-physical Systems
ITMO University
St Petersburg Russia

LAAS CNRS

ITMO UNIVERSITY

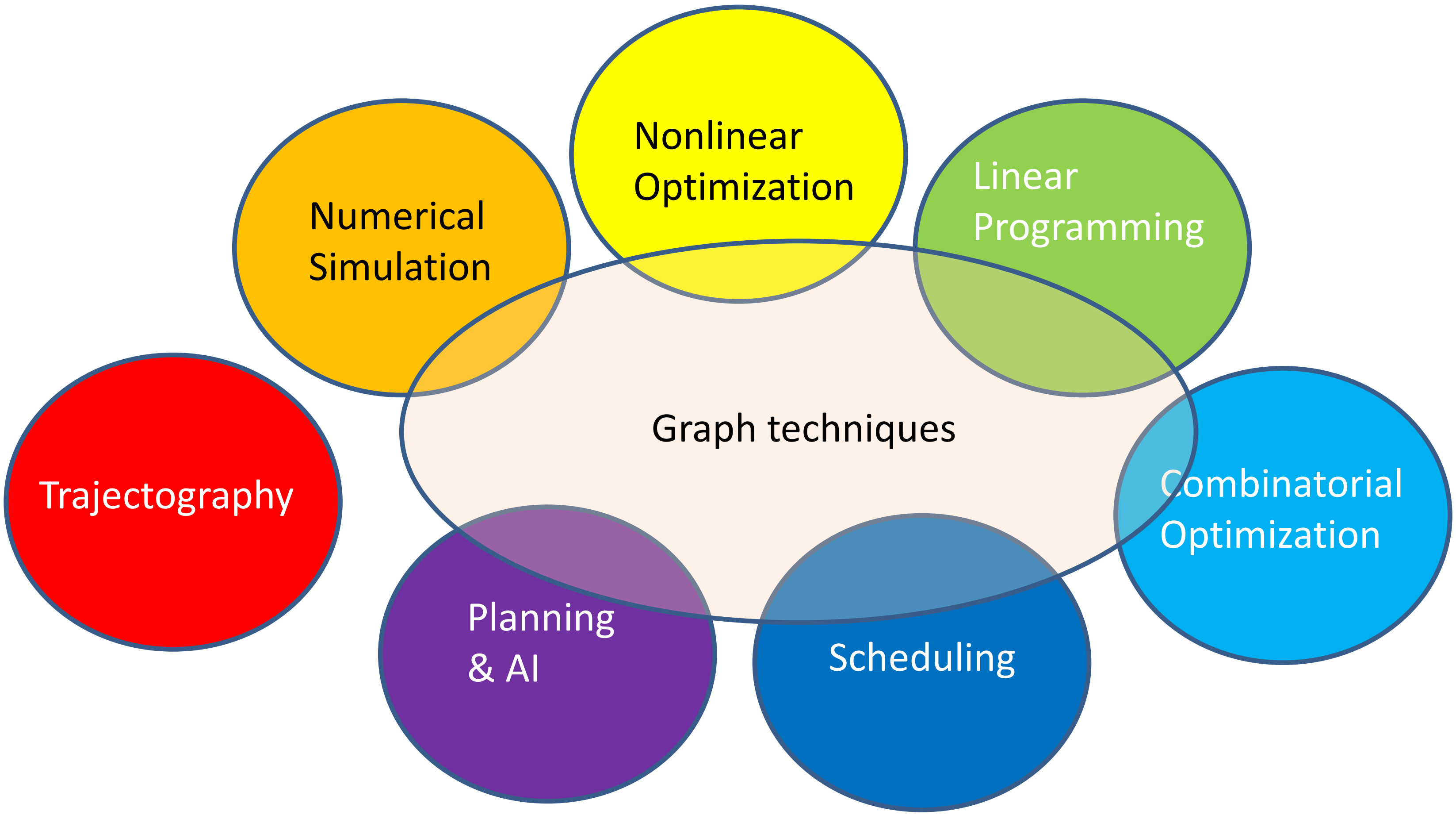
Outline

- 1 Introduction
- 2 Social networks, Big Data and graphs
- 3 Analyzing graph structure
- 4 Conclusions and future work

1. Introduction

1.1 Past work

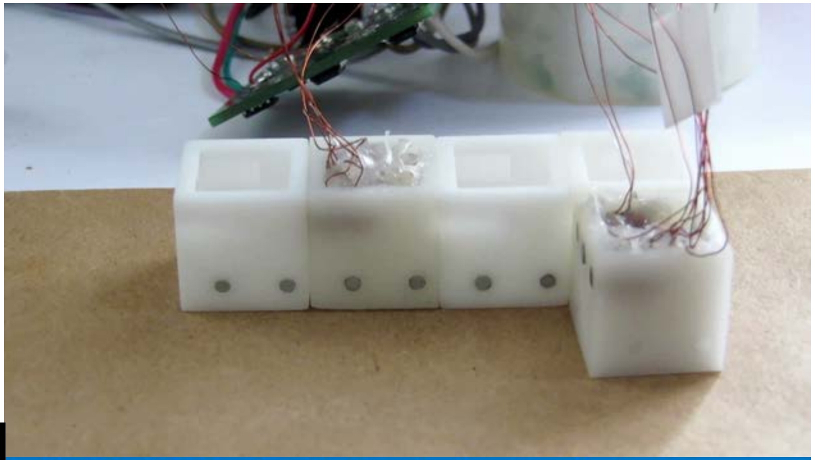
- Fields of interest: HPC, Distributed Computing

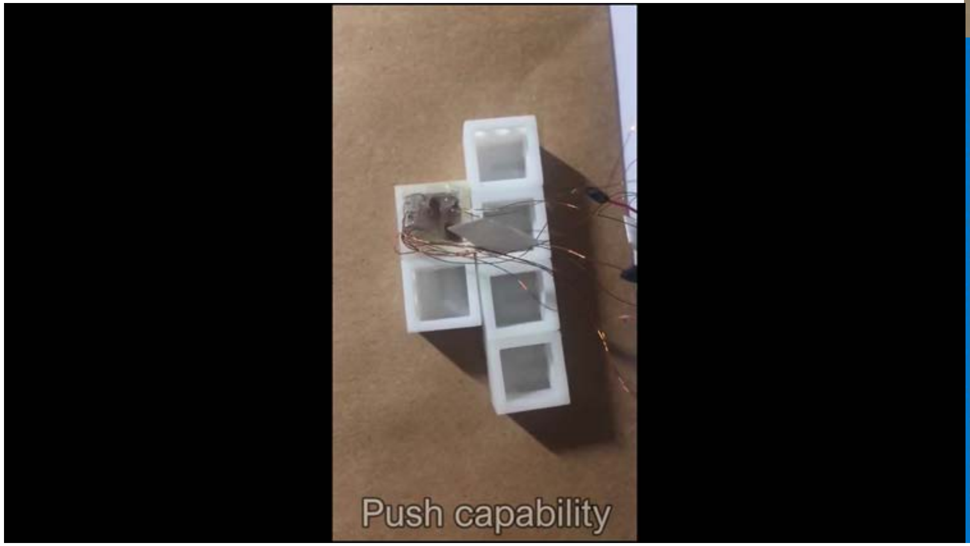


1.1 Past work

Distributed computing and distributed Cyber-Physical systems
Smart Blocks ANR-2011-BS03-005
Distributed autonomous modular system; reconfigurable conveyor.

Li Zhu, Didier El Baz, A programmable actuator for combined motion and connection and its application to modular robot, Mechatronics, Vol. 58, April 2019, 9-19.





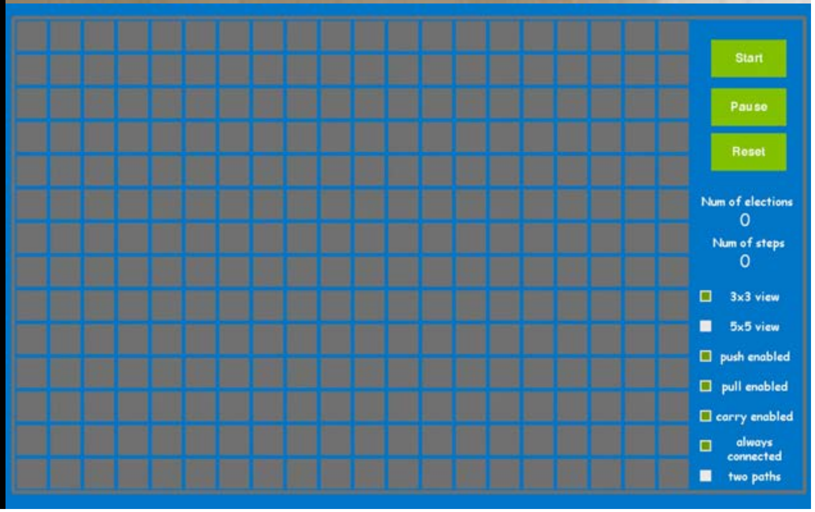


Fig. 1.1 Reconfiguriable distributed smart conveyors

1.2 Present activities

- PHC Tassili Bigreen 40160QB
  - ➢ Algorithms for Big Graphs, Application to Green City LAAS-CNRS Toulouse &Univ. of Lyon 1, France
    Univ. H. Boumediène Algiers, Algeria
- International Lab on Security of Cyber-physical Systems, ITMO University

1.3 Society, Graphs and Arts

1.3.1 Genealogy and Heraldry

- Genealogy, heraldry and trees.

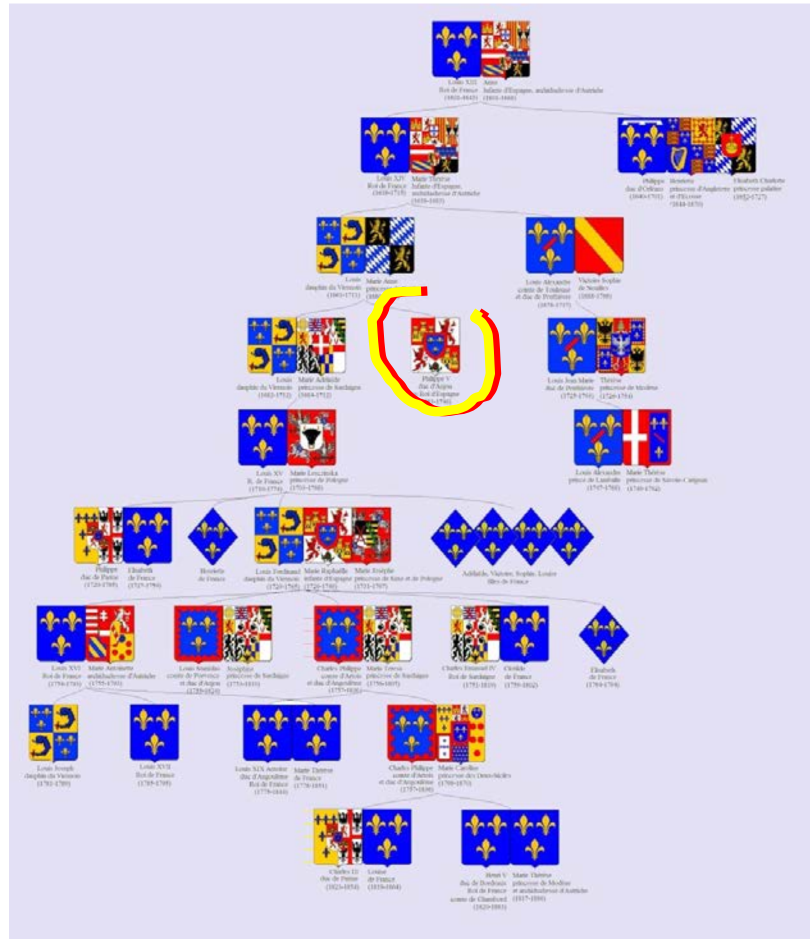
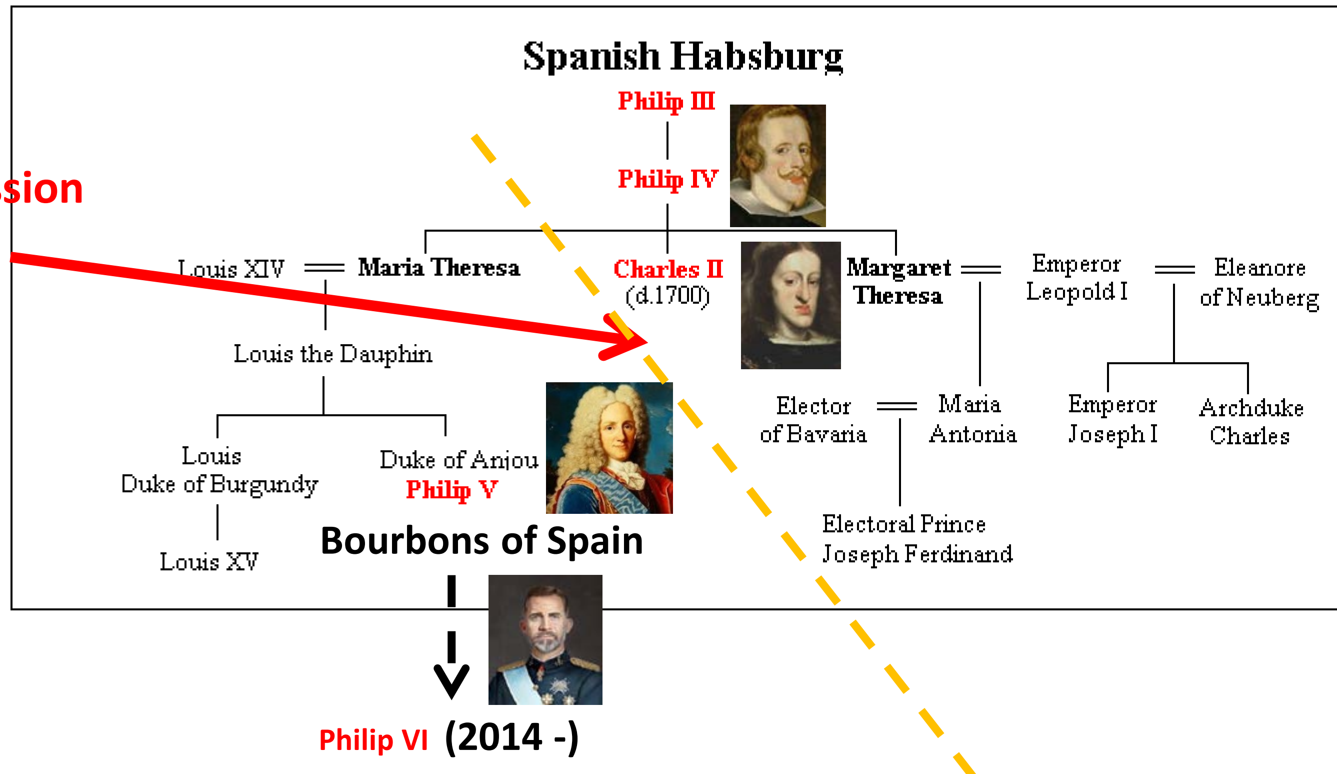


Fig. 1.2 Genealogy of
of Bourbon royal family
International influence

1.3.2 Genealogy and History

- **Trees are the key to understanding the history of countries.**
- At the time of the Old Regime, history was mainly a consequence of royal alliance and royal successions.

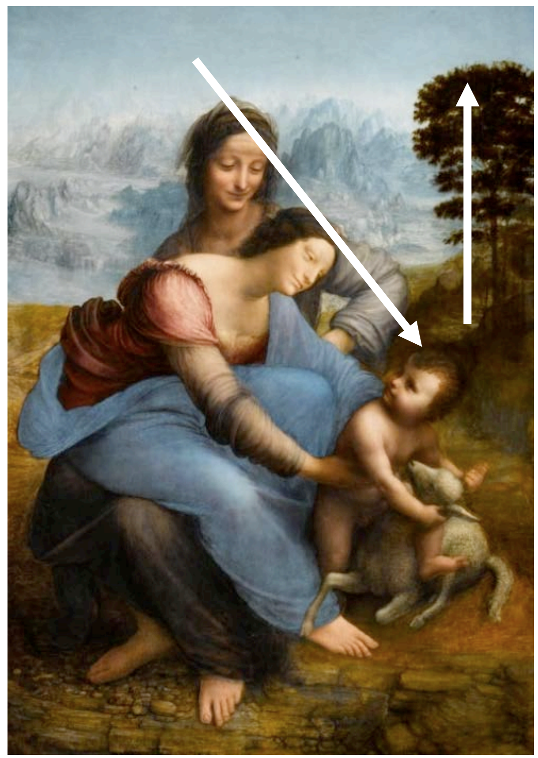
The War of Spanish Succession 1701 to 1714.



1.3.3 Genealogical tree in the Medieval & Renaissance Art

- St Anne Trinitarian

Figure 1.3
Leonardo da Vinci
St Anne, the Virgin Mary and the Christ
ca 1510-1519,
oil on wood, Paris, Le Louvre



1.3.3 Genealogical Trees in the Medieval & Renaissance Arts

- Jesse’s tree

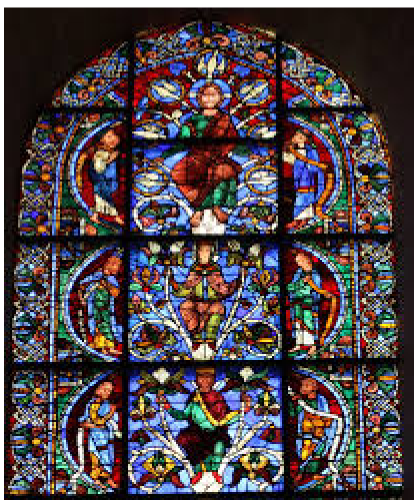


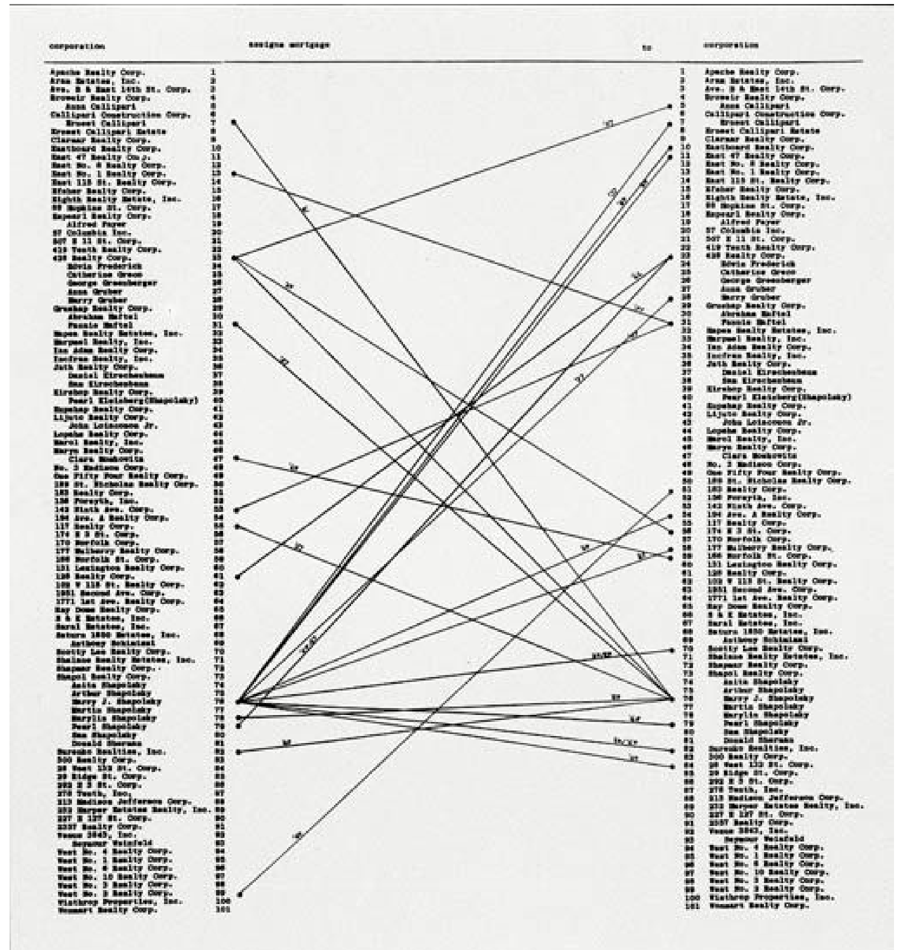
Figure 1.4 stained glass window of the cathedral of Chartres

1.3.4 Graphs in contemporary art

- Hans Haacke 1936-
- Shapolsky et al. Manhattan Real Estate Holdings,

a Real-Time Social System, as of May 1st, 1971‘

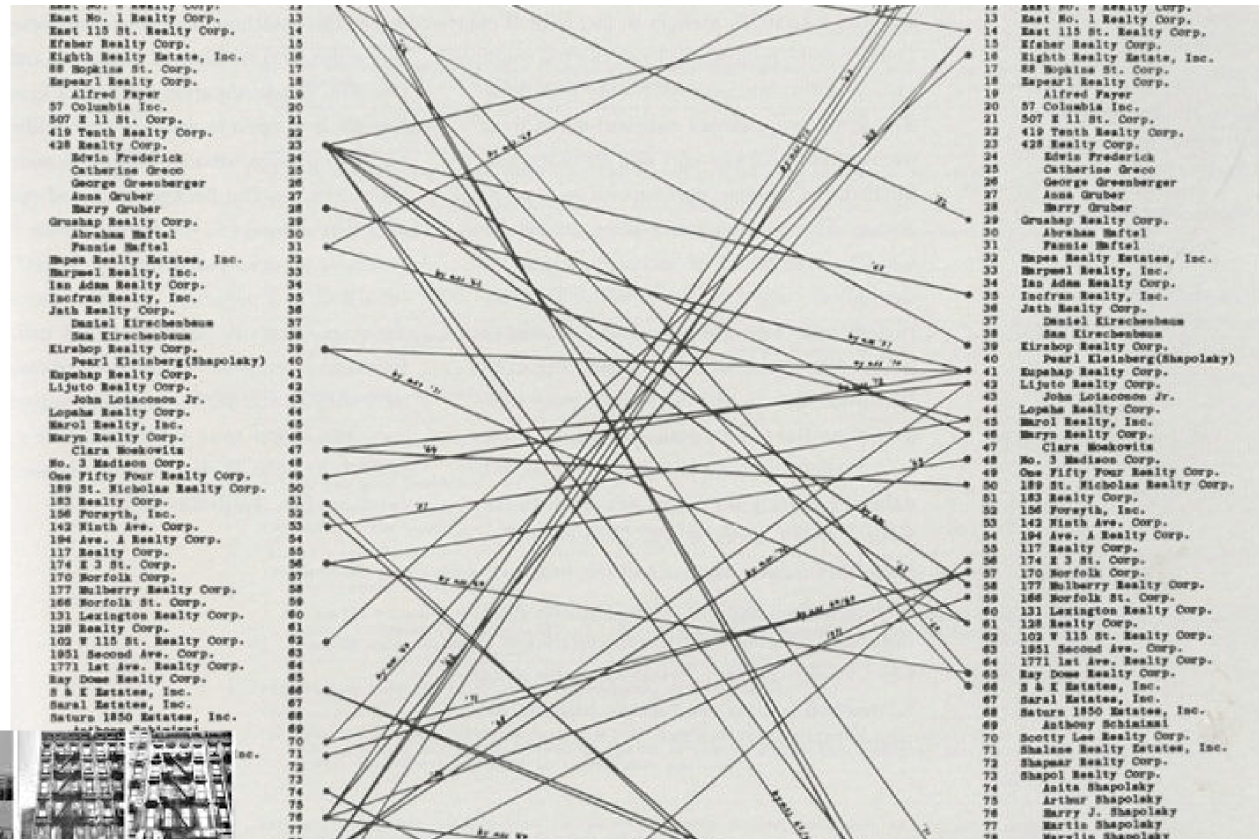
Haacke took on the real-estate holdings of one of New York City's biggest slum landlords. The work exposed, through meticulous documentation the questionable transactions of Harry Shapolsky's real-estate business between 1951 and 1971.

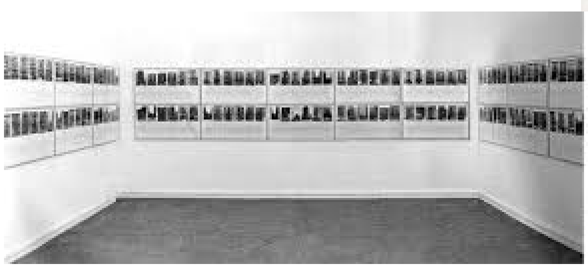


1.3.4 Graphs in contemporary art

- Hans Haacke 1936-
- Shapolsky et al.

Manhattan
Real Estate Holdings,





1.3.4 Graphs in contemporary art

- Mark Lombardi 1951 – 2000 Brooklyn, USA
- Narrative Structures
- American neo-conceptual artist who specialized in drawings that document alleged financial and political frauds by power brokers, and in general "the uses and abuses of power".

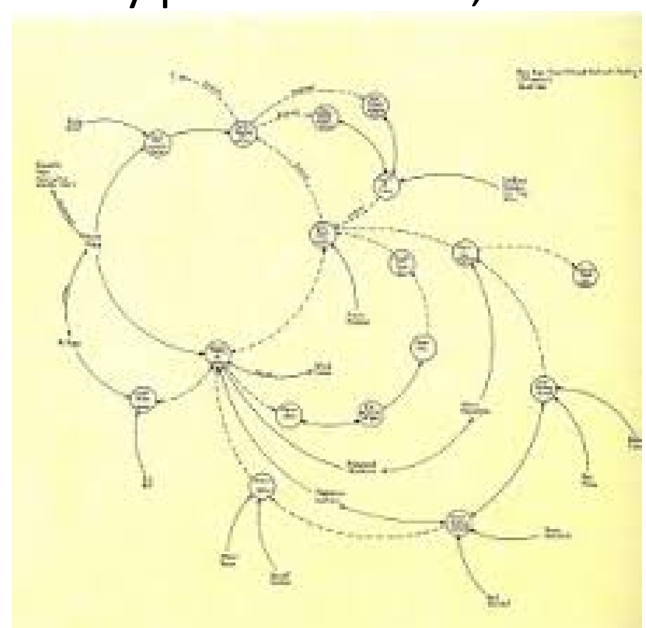
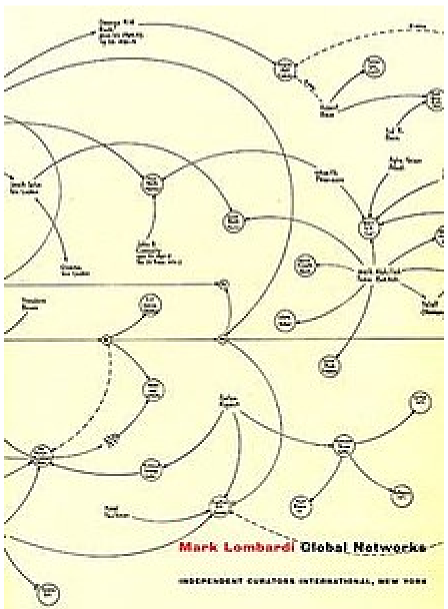


Figure 1.5 Mark Lombardi
Narrative structure

Mark Lombardi's drawings

- George W. Bush, Harken Energy, Jackson Stephens
- ca 80 - 90



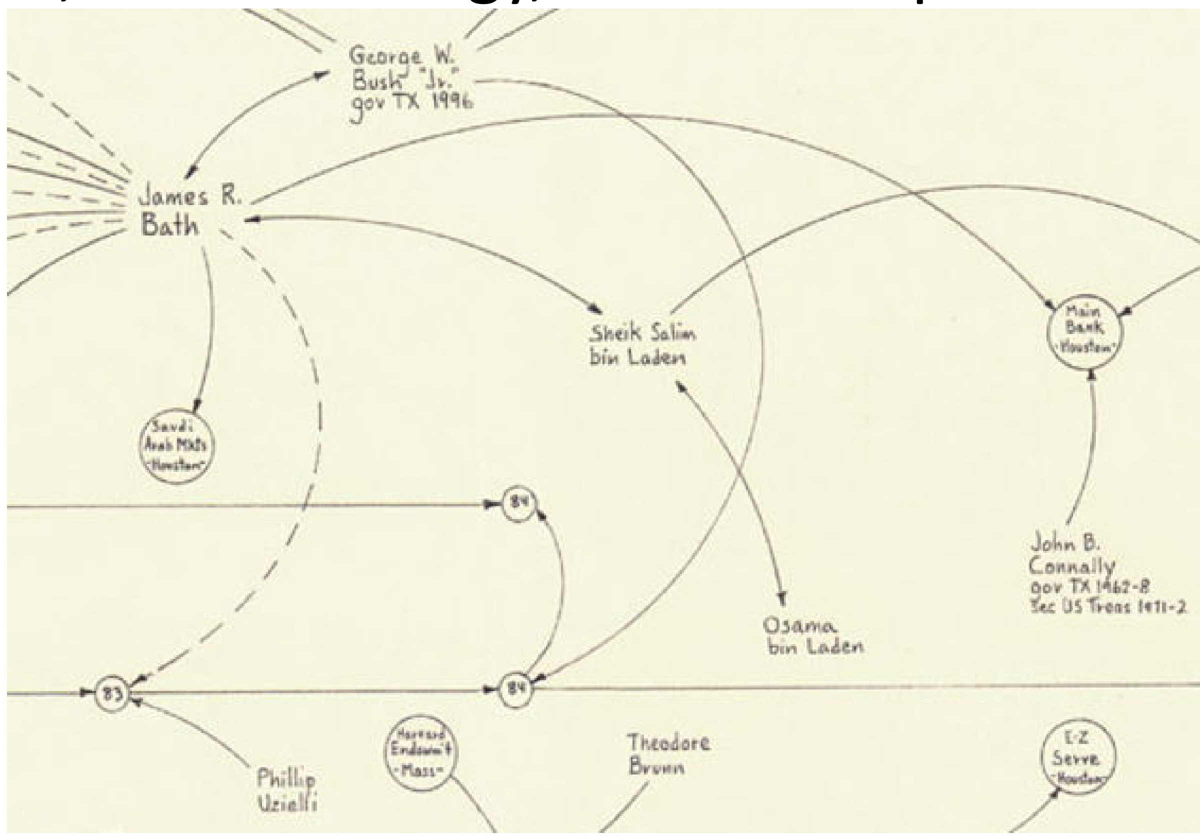


Figure 1.6 Alleged connections between James Bath, the Bush and bin Laden families, a and business deals in Texas and around the world

Mark Lombardi’s drawings

- Influences: philosopher Herbert Marcuse and visualization expert Edward Tufte.
- **Lombardi’s Narrative Structures are structurally similar to sociograms**
- Sociograms: a type of graph drawing used in the field of social network analysis.

1.4 Importance of Graphs

- **Graphs like trees have always played a major role in the past.**
- **In particular, the history of mankind was strongly related to genealogical trees.**
- **With the development of the Internet and social networks, Big Data and graph analysis is one are very important domains in Applied Maths. with many applications in sociology, security and business.**

2. Social networks, Big Data and Graphs

- 10^{30} bytes of data by day are generated and a big part of those data are social data.
- We find huge graphs in social networks, e.g., Twitter fellowship: 1,470,000,000 edges.
- Curse of dimension
- Need automated treatment.

2.1 Graphs

- Graph is a powerfull concept
- Graphs can represent many things:
  - family relationships,
  - communication networks, computer networks,
  - social networks,
  - political relationships,
  - financial influences.

2.2 Graph representation

- A graph consists of a set of vertices N

and a set of edges A, $Card(N) = n$, $Card(A) = a$.

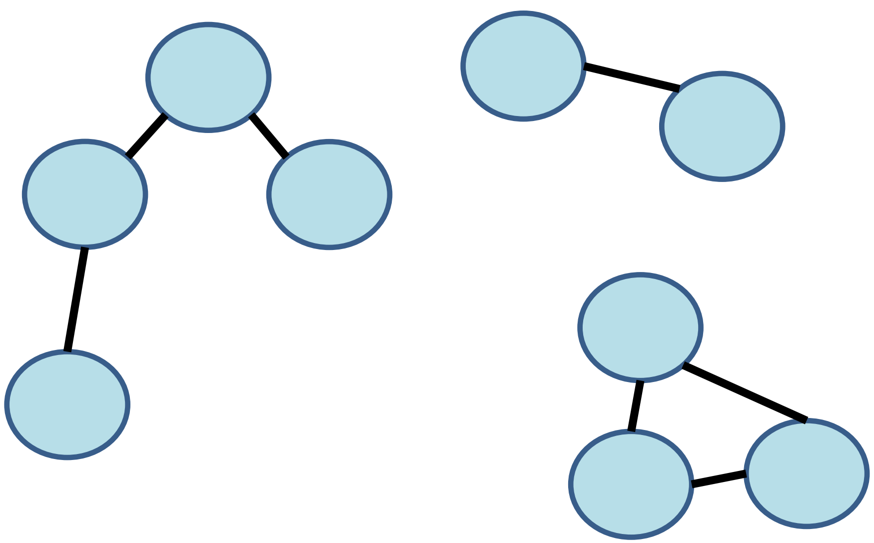


Figure 2.1 Graph

2.2.1 Undirected graphs

- An undirected graph is a graph in which edges have no orientation.

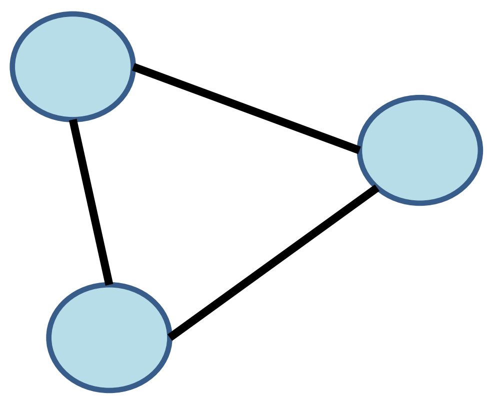


Figure 2.2 undirected graph

2.2.2 Directed graphs

- A *directed graph* or *digraph* is a graph in which edges have orientations.
- A digraph is written as an ordered pair (N, A) where N is the set of vertices and A is the set of edges.

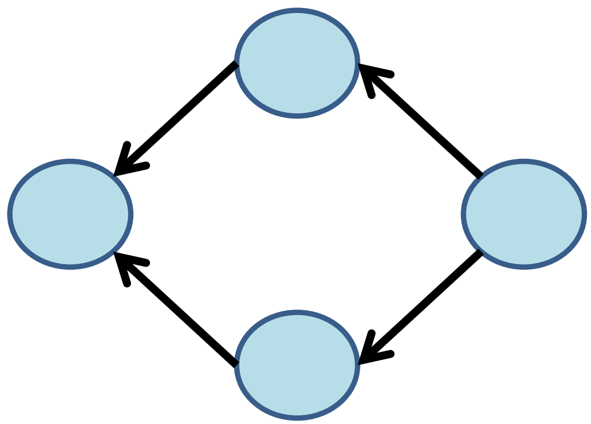


Figure 2.3 Directed graph

2.2.3 Path

- A path in a graph is a finite or infinite sequence of edges which connects a sequence of vertices.

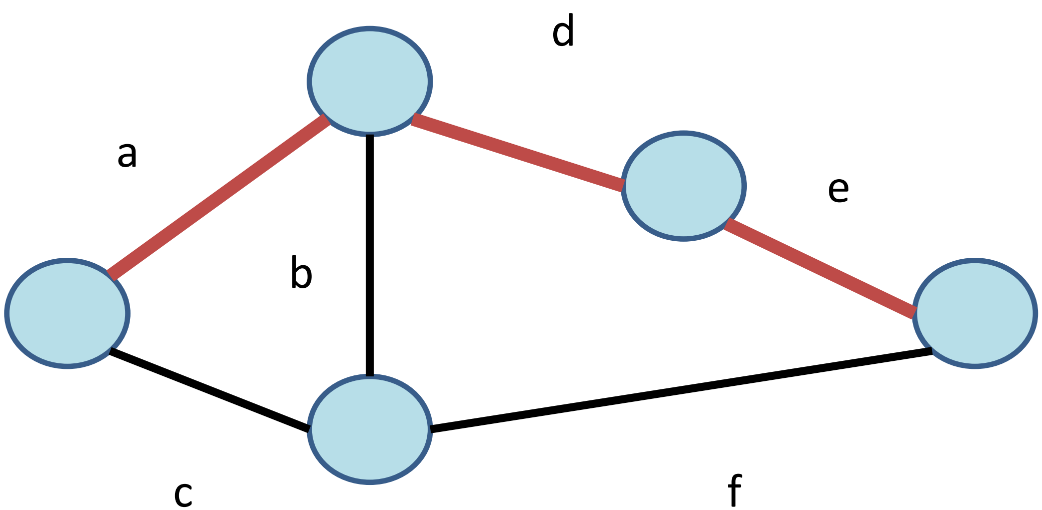


Figure 2.4 Path

- Path a,d,e

2.3 Other networks

- We find also big graphs in domains like

➢ social simulation, e.g., road networks

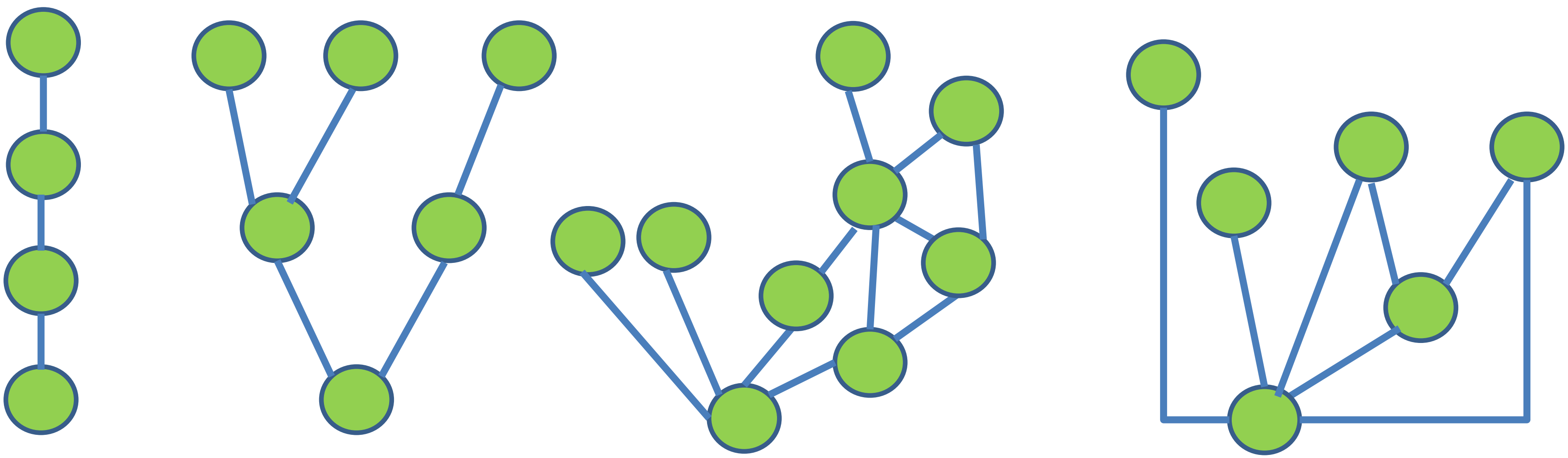
US road network: 58,000,000 edges,

➢ brain science, e.g., EU Brain project

Neural nets: 100,000,000,000 edges, 89 billion nodes.

3. Analyzing data structure

- Graphs can have different topologies / structures:



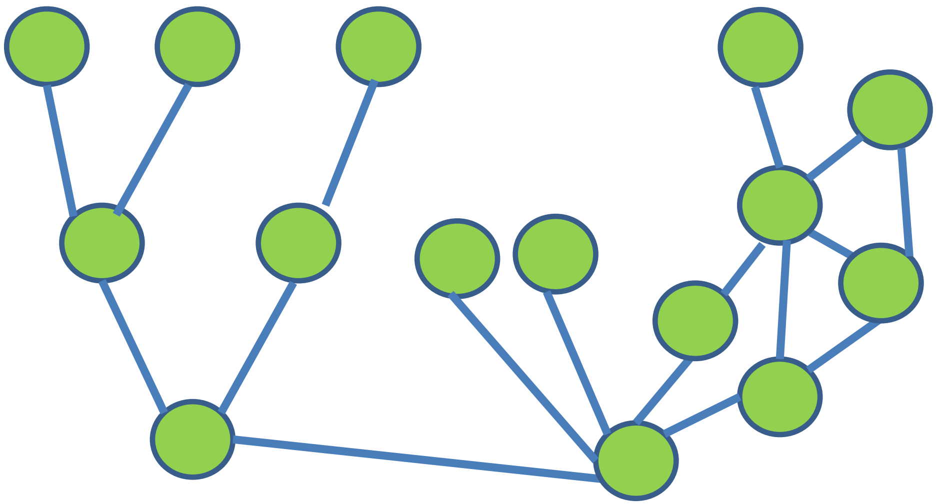
Chain Tree Unstructured Graph with central node

Figure 3.1 Graphs

3. Analyzing graph structure

- Hierarchized networks

➢ P2P networks with super peers

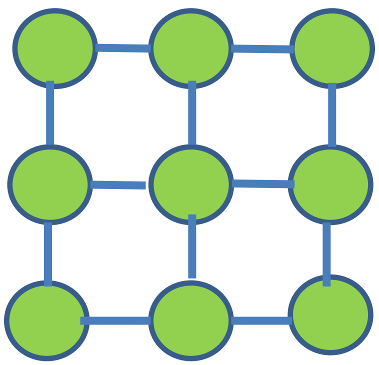


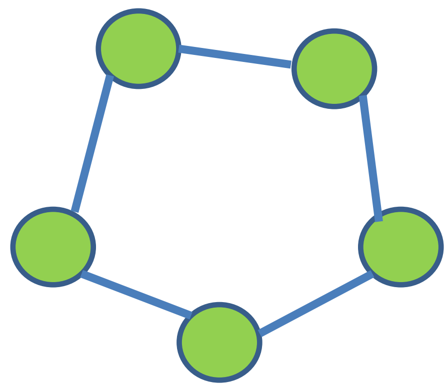
Hierarchized network

Figure 3.2 Graphs

3. Analyzing graph structure

- Mesh, cycles





3.1 Graph structure and metrics

- Degree of a node
  Number of outgoing edges

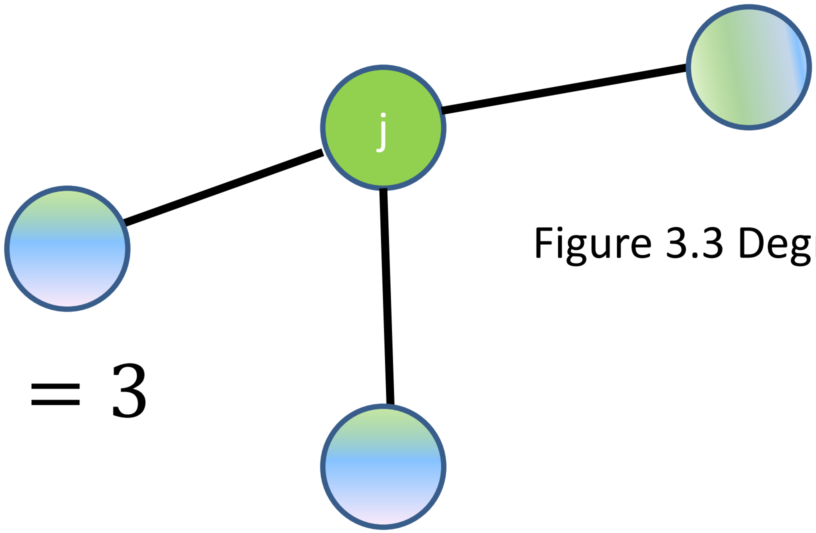


Figure 3.3 Degree of a node

- e.g., $d(j) = 3$

➢ Algorithms for opinion leaders recognition

3.1.1 Centrality

- Centrality of a node
- What is centrality?

➢ Node that is not in the periphery of the graph

This node is not central

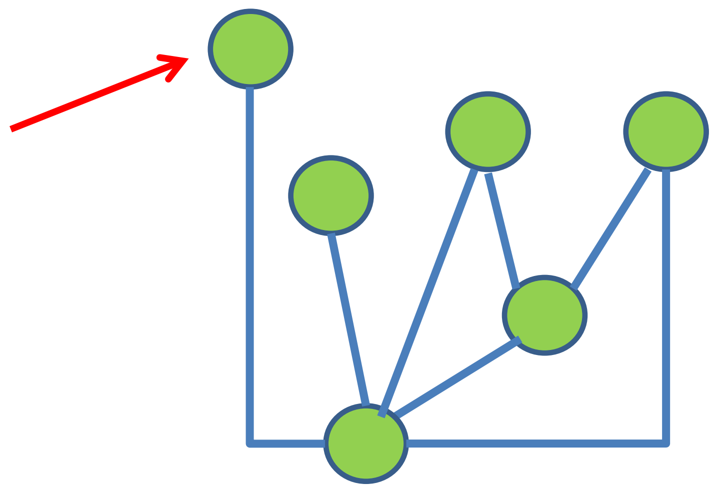


Figure 3.4 Noncentral node

3.1 Tools to analyze graph structure

- Centrality of a node
- What is centrality?

  Node that has many edges

This node is central

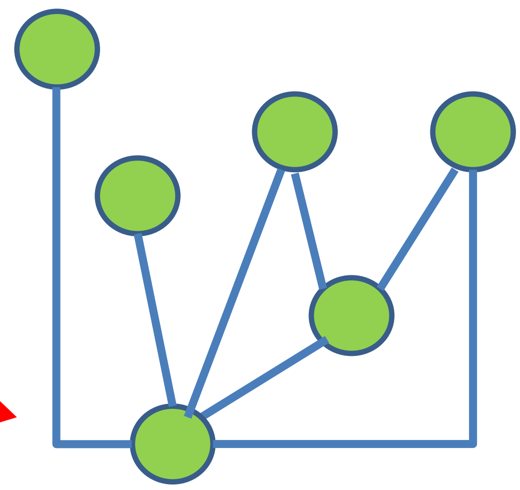


Figure 3.5 Central node

3.1 Tools to analyze graph structure

- Centrality algorithms, e.g., betweenness centrality algorithms

  Algorithms based on all-pairs shortest path calculation and counting number of shortest paths through the node.
- Closeness centrality algorithms based on all-pairs shortest path calculation

3.1 Problem

Looking for leader

Figure 3.6
Graph in the OrientDB database
Data from Vkontakte
Group of discussion on ski;
edges are friendships
Credit: M. Kolomeec

3.1 Problem

- Looking for leader

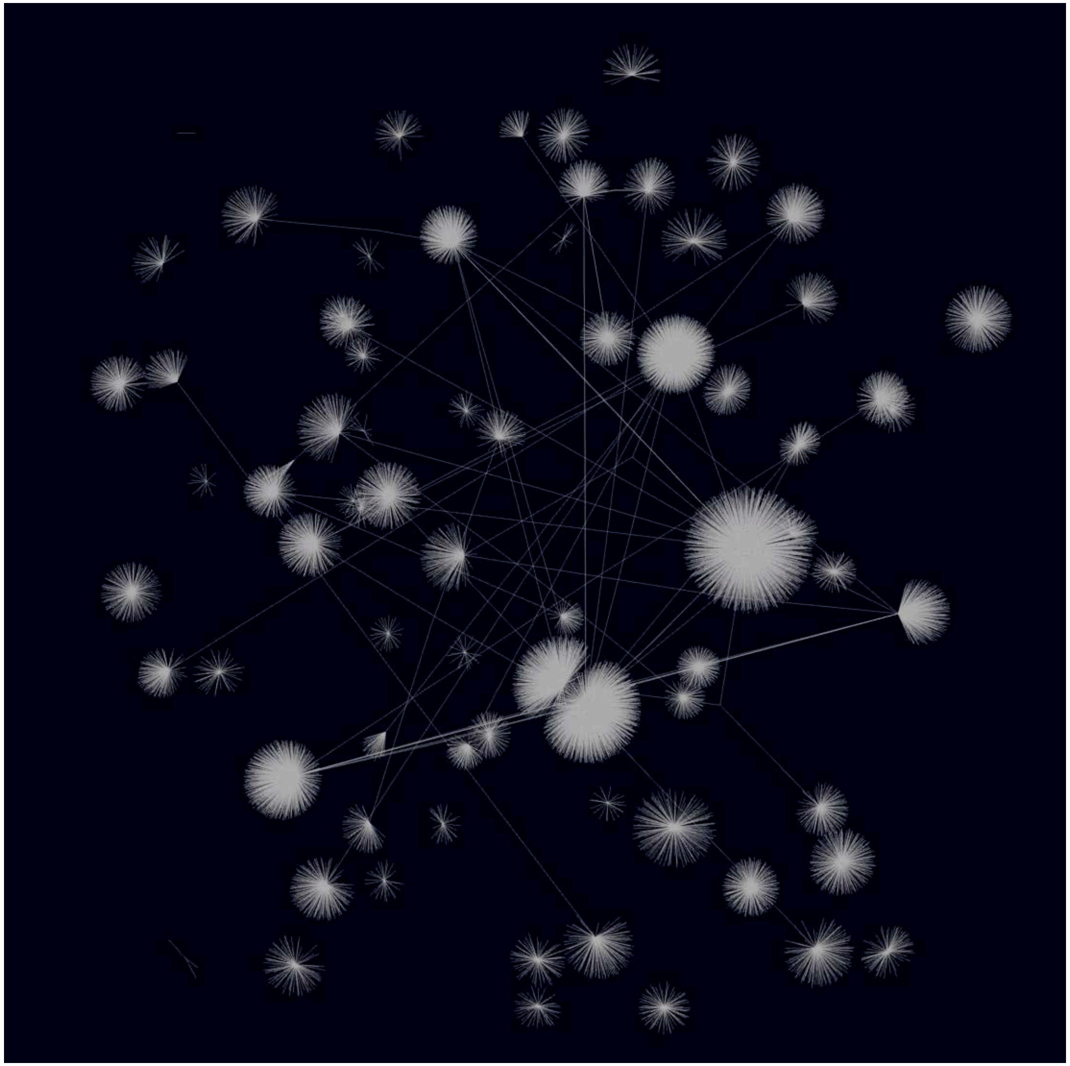


Figure 3.7
Graph in the OrientDB database
Data from Vkontakte; edges are friendships
Credit: M. Kolomeec

3.1 Problem

- Looking for best leaders or best E-fluentials

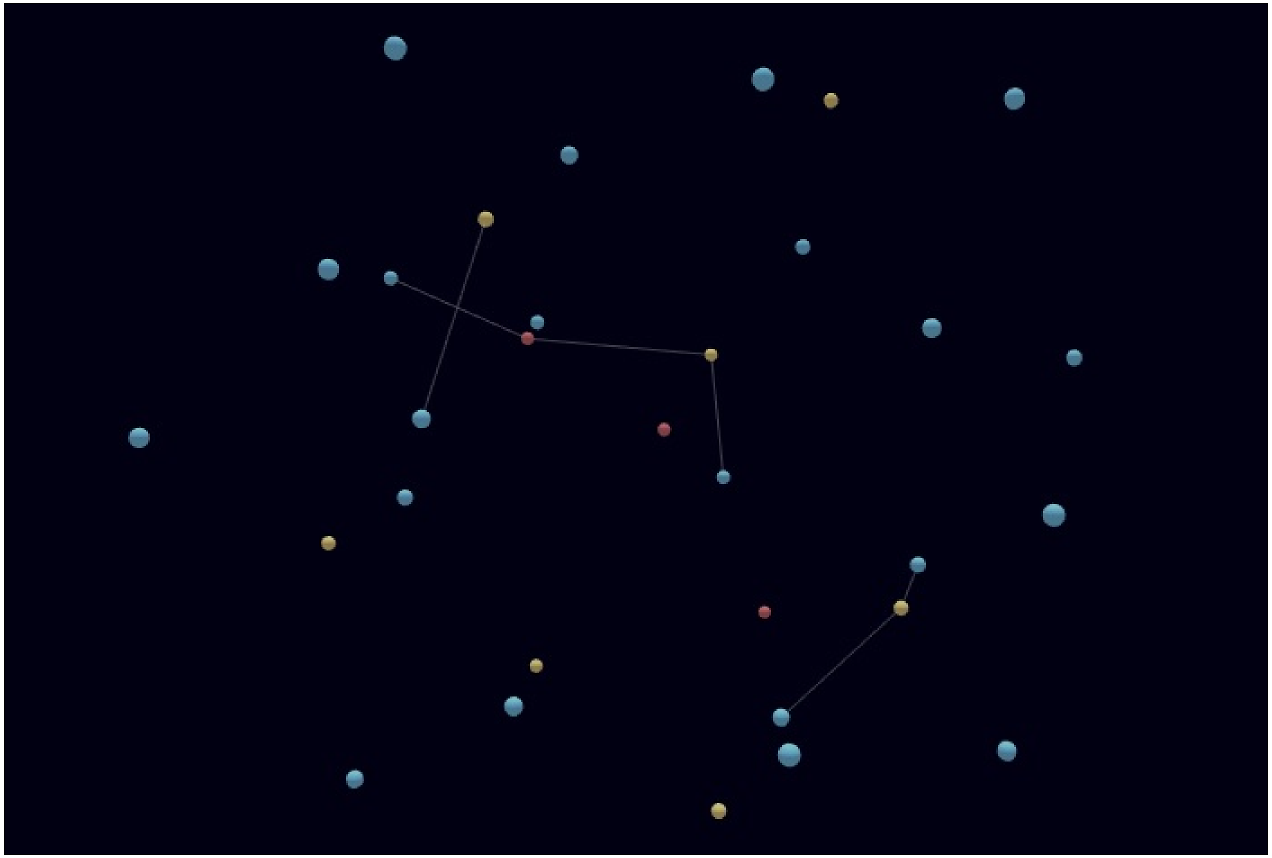


Figure 3.8 Graph in the OrientDB database Data from Vkontakte; Pushkin Data a1; Credit: M. Kolomeec

3.1 Problem

- Looking for best leader

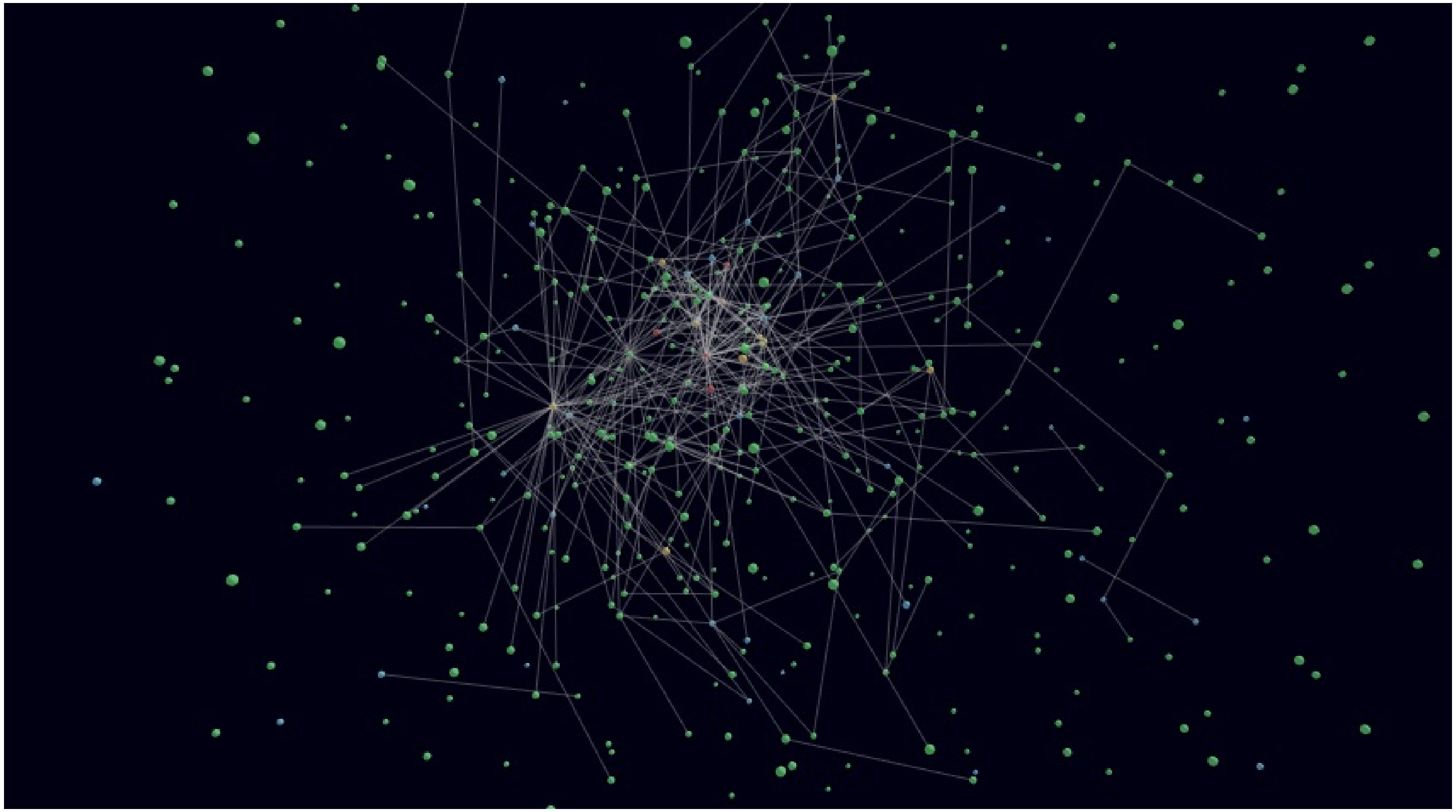


Figure 3.8b
Graph in the
OrientDB
database
Data from
Vkontakte;
Pushkin Data
B5;
Credit: M.
Kolomeec

3.1 Problem

- Looking for best leader

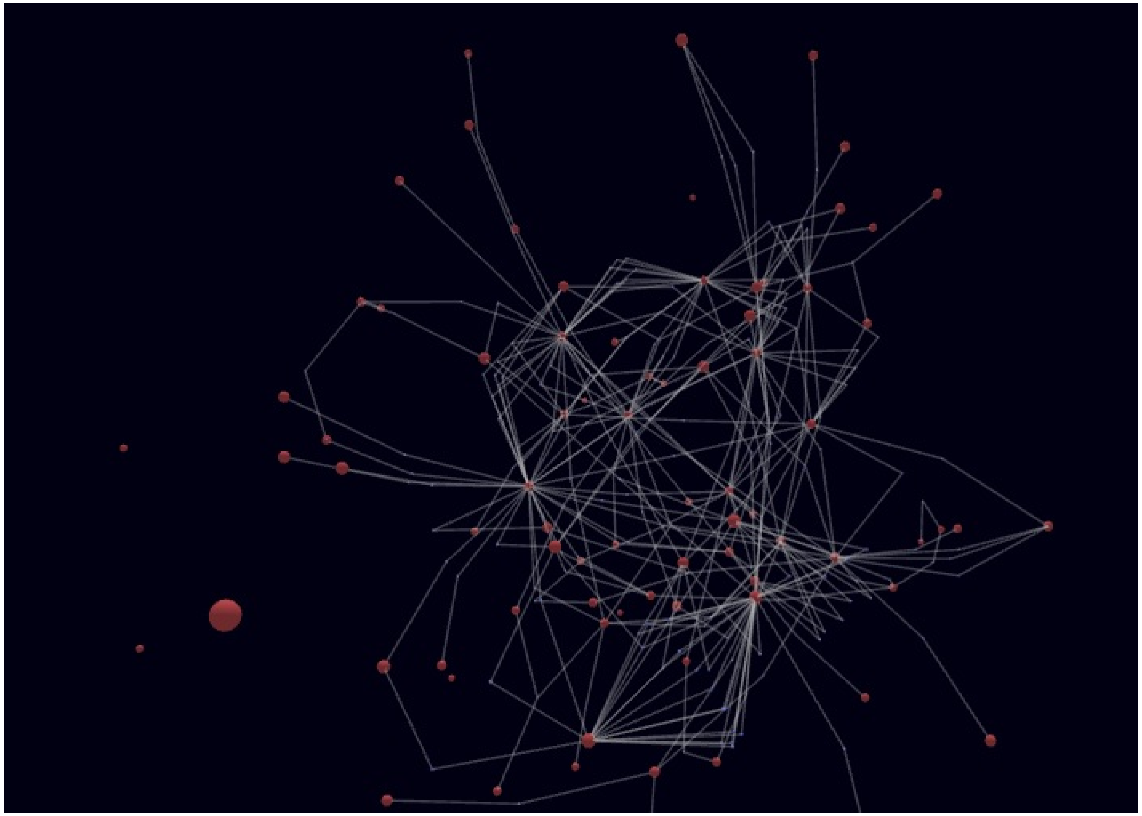


Figure 3.8b Graph in the OrientDB database
Data from Vkontakte;
Blue nodes: users who post comments
Red nodes: followers;
Credit: M. Kolomeec

3.2 Parallel hybrid centrality algorithm

- Algorithm based on a **combination of all-pairs shortest path calculations and degree of nodes.**
- Parallel algorithm designed for GPUs

3.2.1 Baseline

- **Single source shortest path problem**
- Source: node 0
- Minimum hop problem.

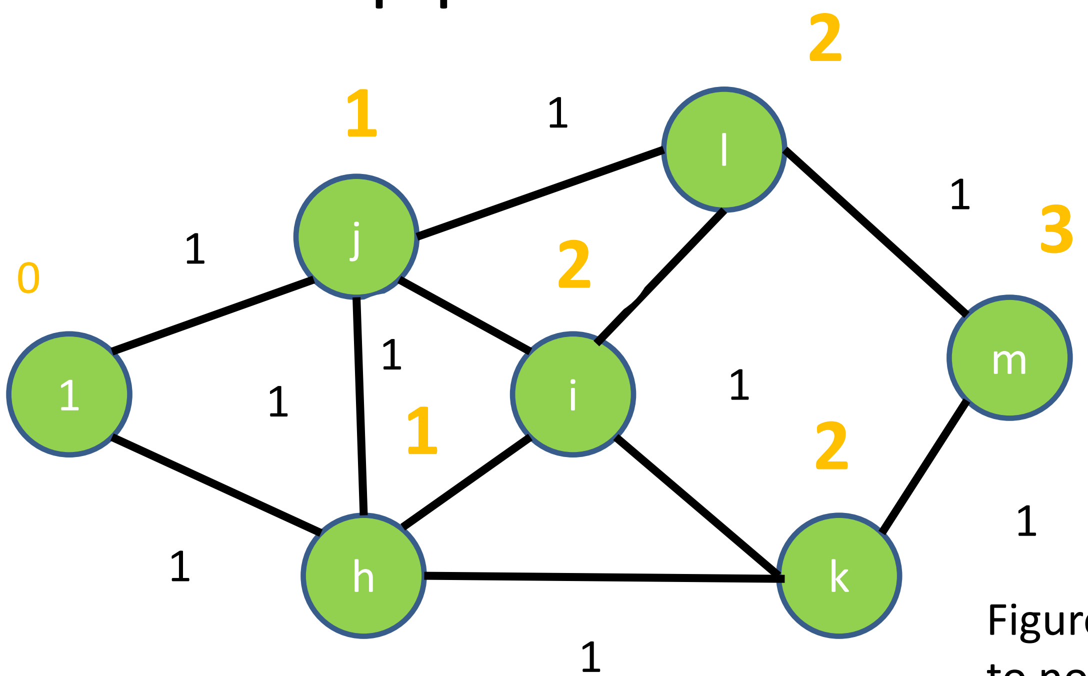


Figure 3.9 shortest paths to node 1

3.2.1 Baseline

- **All-pairs shortest path algorithm**
- compute shortest paths for all possible destinations

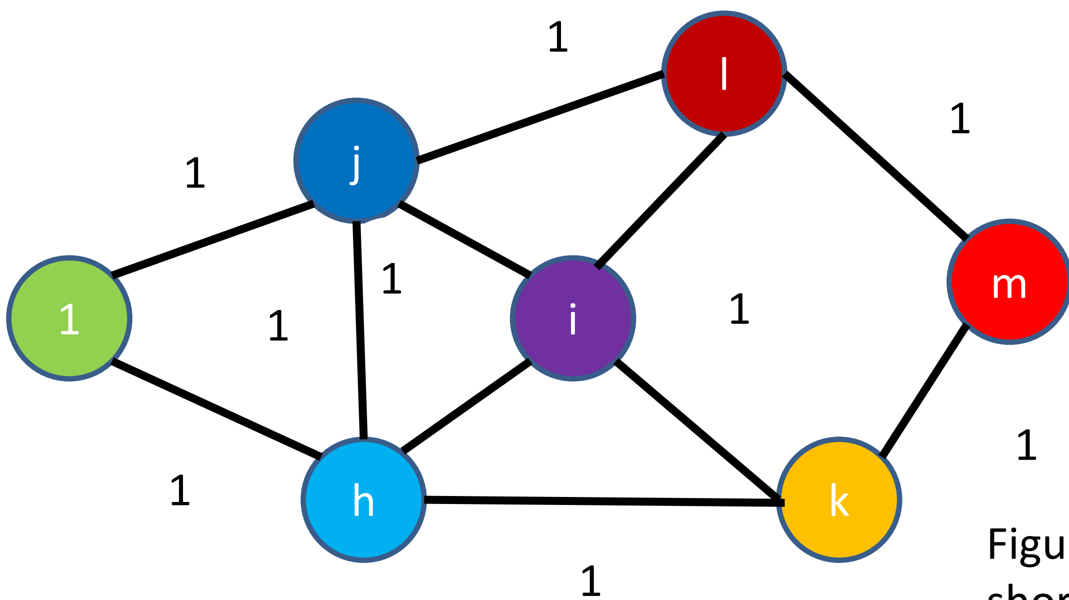
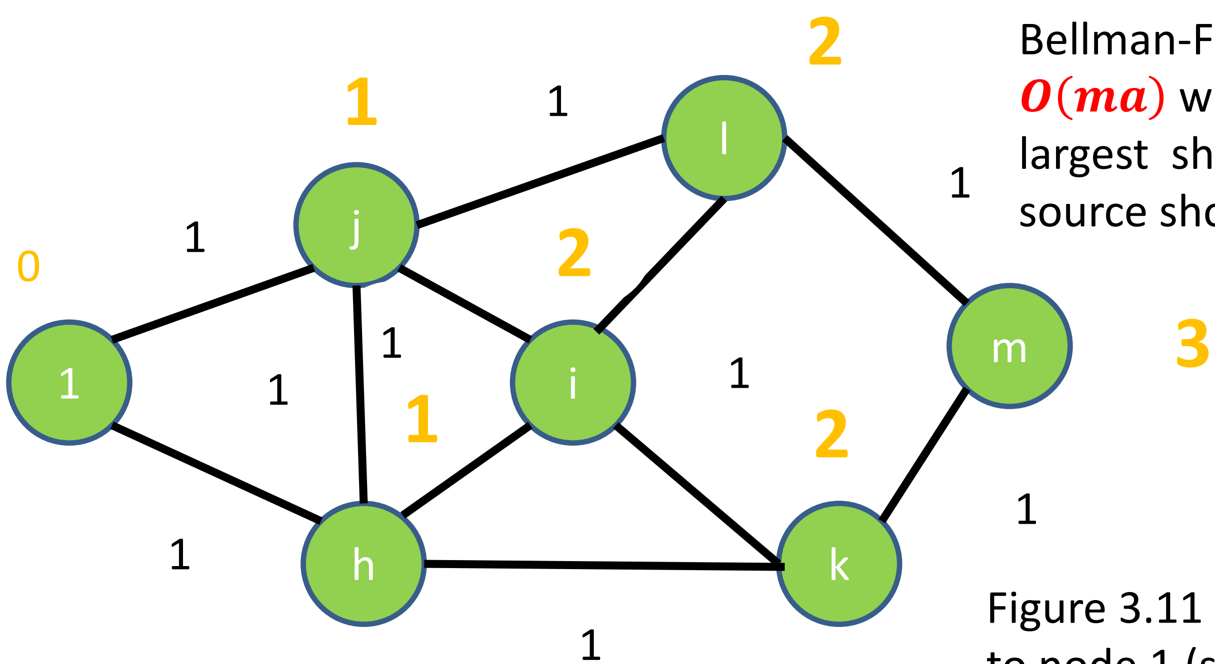


Figure 3.10 all-pairs shortest paths

3.2.2 Sequential case, step 1

- **Use a single source shortest path algorithm: Bellman-Ford**
- **Minimum hop problem**
- At iteration k, all the shortest path distances with length k are obtained.
- $x_i(k+1) = \min\left(x_i(k), \min_{j \in A(i)}\left(a_{ij} + x_j(k)\right)\right),\ i = 2, \dots n,$



Bellman-Ford time complexity $\boldsymbol{O(ma)}$ where m is the largest shortest path for single source shortest path

Figure 3.11 shortest paths to node 1 (source)

3.2.2 Sequential case, step 1

- **Solve all-pairs shortest paths thanks to Bellman-Ford algorithm**
- We solve *n* single source shortest paths problems.
- Time complexity:
- $\boldsymbol{O(nma)}$

3.2.3 Sequential case, step 2

- For each destination node $i \in N$ (source), compute $c_h(i)$ **hybrid centrality** based on the sum of distances of shortest paths from each node $j \in N$ to destination $i \epsilon N$.
- $c_h(i) = \sum_{j \in N} {x^*}_j / d(i)\ \forall i \in N$.
- Metrics that combines **degree centrality** and a kind of **closeness centrality**.

➢ Time complexity: $O(n^2)$ additions.

3.2.4 Sequential case, step 3

- The central node is the node $i \in N$ such that $c_h(i)$ is **minimal**.
- $c_h(i) = \sum_{j \in N} x^*{}_j / d(i)$.
- Time complexity: $O(n)$ comparisons.

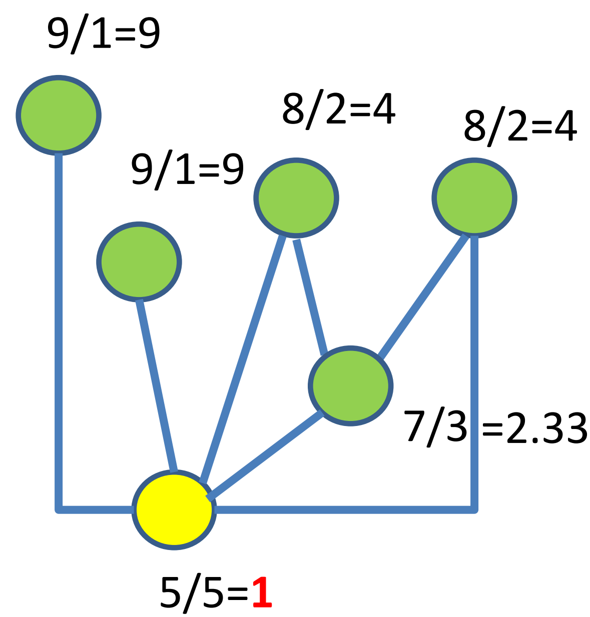


Figure 3.12 Graph with central node, values of $c_h(i)$

3.2.4 Sequential case, complexity

- Total time complexity:
- $\boldsymbol{O(nma)}$

3.3 Single source shortest paths

- The problem is to find a path with minimum length (shortest path) from each node $i \in N$ to the destination 1.

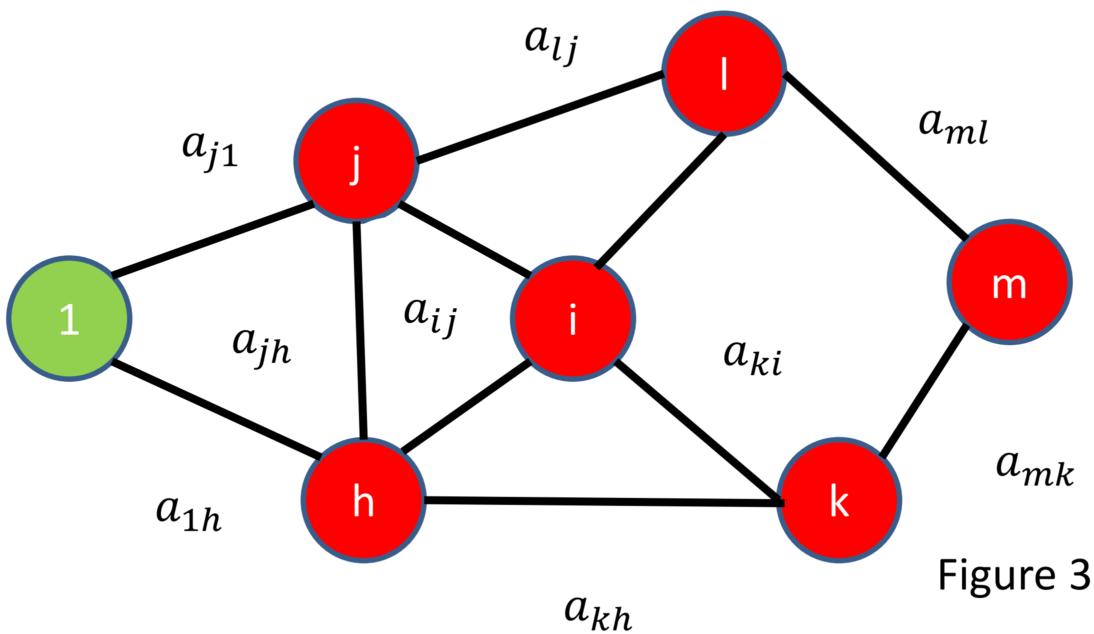


Figure 3.13 shortest path problem

3.4 All-pairs shortest paths

- We have to solve single source shortest path problem n times i.e., for each possible destination$1, 2, \dots, n$.

3.4.1 Shortest path problem formulation

- Directed graph (N, A).
- N is the set of nodes.
- Node *1*: destination node (for network traffic).
- *n* nodes numbered *1,…,n.*
- A is the set of arcs.

3.4.1 Shortest path problem formulation

- $A(i)$ set of all nodes *j* for which there is an outgoing arc $(i, j) \in A$.

  $A(i) = \{j \in N \mid \exists\ (i, j)\ \epsilon\ A\}$
- A cost a_{ij} is associated with each arc $(i, j) \in A$.
- The length of path $(i, j)(k, l)$ is $a_{ij} + a_{kl}$.
- Minimum hop shortest paths
- $a_{ij} = 1, \forall (i, j)\ \epsilon\ A$.

3.4.1 Shortest path problem formulation

- **Assumptions**

Connectivity: there exists a path from every node $i = 2, \ldots, n$ to the destination node 1.

Positive cycle: every cycle has positive length.

3.4.1 Shortest path problem formulation

- **Mathematical formulation**
- Fixed point problem
- The shortest path vector $x^*$is the unique solution of the fixed point problem:

  $$x^* = F(x^*),$$

  where

  $$x_1^* = 0,$$

  $$x_i{}^* = \min\left(x_i{}^*, \min_{j\in A(i)}(a_{ij} + \ \ x_i{}^*)\right),\ i = 2, \ldots n.$$

3.4.2 Bellman-Ford algorithm

- The Bellman-Ford iterative algorithm (1958) converges to the solution of the problem from a super solution.

$$x_1(k+1) = 0,$$

$$x_i(k+1) = \min\left(x_i(k), \min_{j \in A(i)}\left(a_{ij} + x_j(k)\right)\right),\ i = 2, \ldots n.$$

- Well suited to distributed implementation
  - simultaneous computations at each node.
  - locality of data.
- Well suited to parallelism.

3.4.2 Bellman-Ford algorithm

- **At iteration k, all the shortest path distances with length k are obtained.**

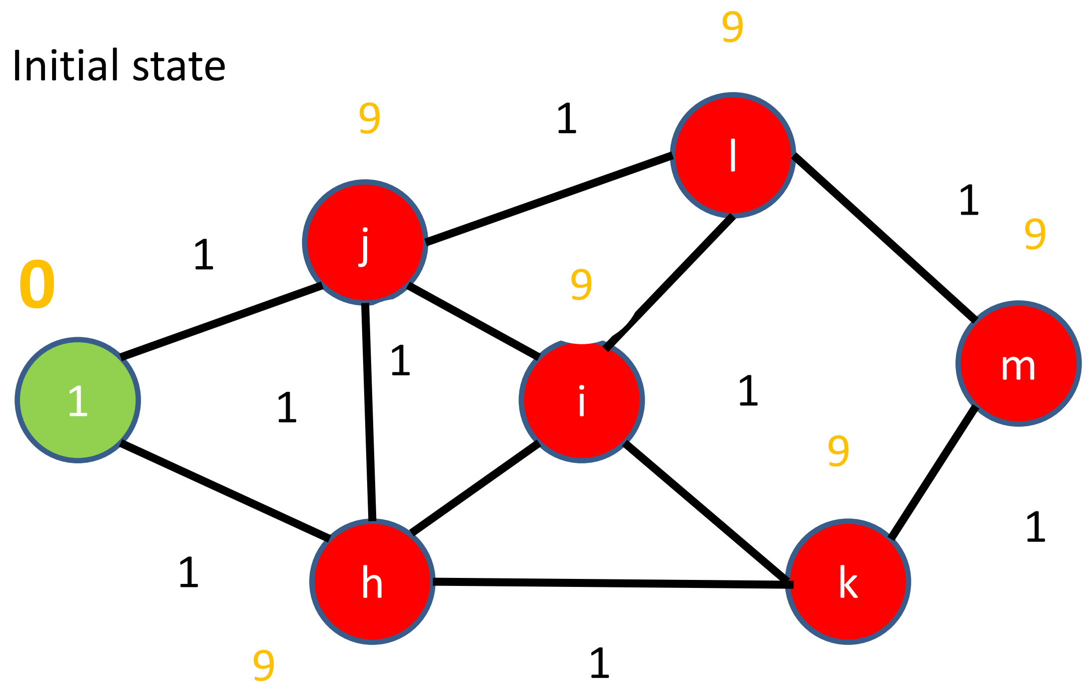


Figure 3.14 shortest path at iteration 0

3.4.2 Bellman-Ford algorithm

- At iteration k, all the shortest path distances with length k are obtained.
- $$x_i(k+1) = \min\left(x_i(k), \min_{j\in A(i)}\left(a_{ij} + x_j(k)\right)\right), i = 2, \ldots n,$$

Iteration 1

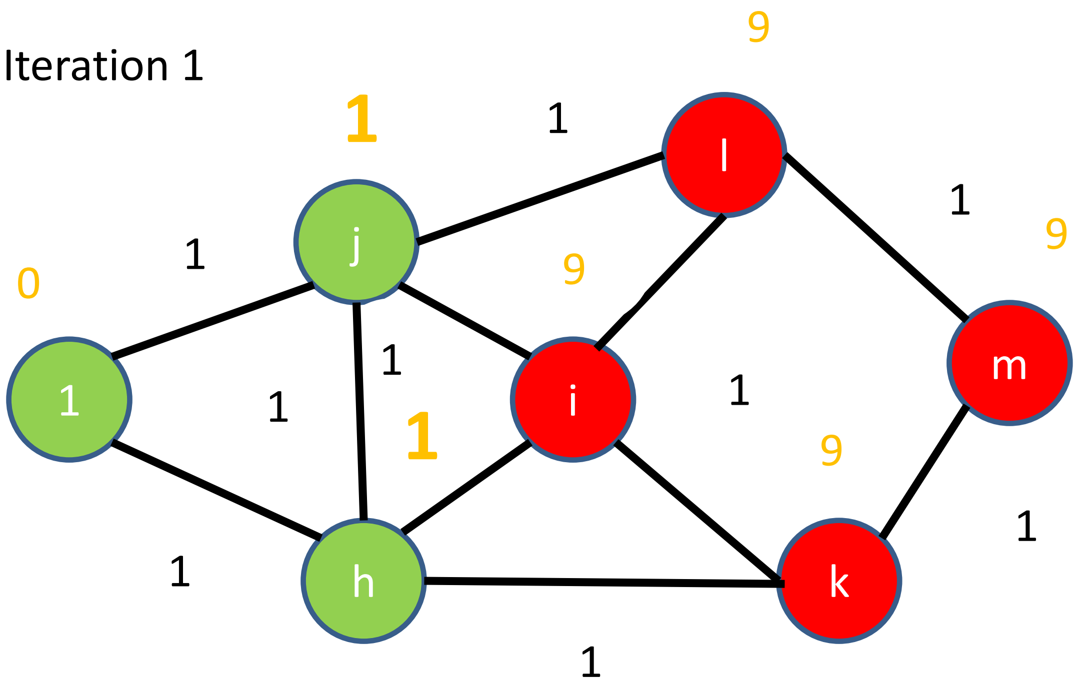


Figure 3.15 shortest path at iteration 1

3.4.2 Bellman-Ford algorithm

- At iteration k, all the shortest path distances with length k are obtained.
- $x_i(k+1) = \min\left(x_i(k), \min_{j \in A(i)}\left(a_{ij} + x_j(k)\right)\right), i = 2, \dots n,$

Iteration 2

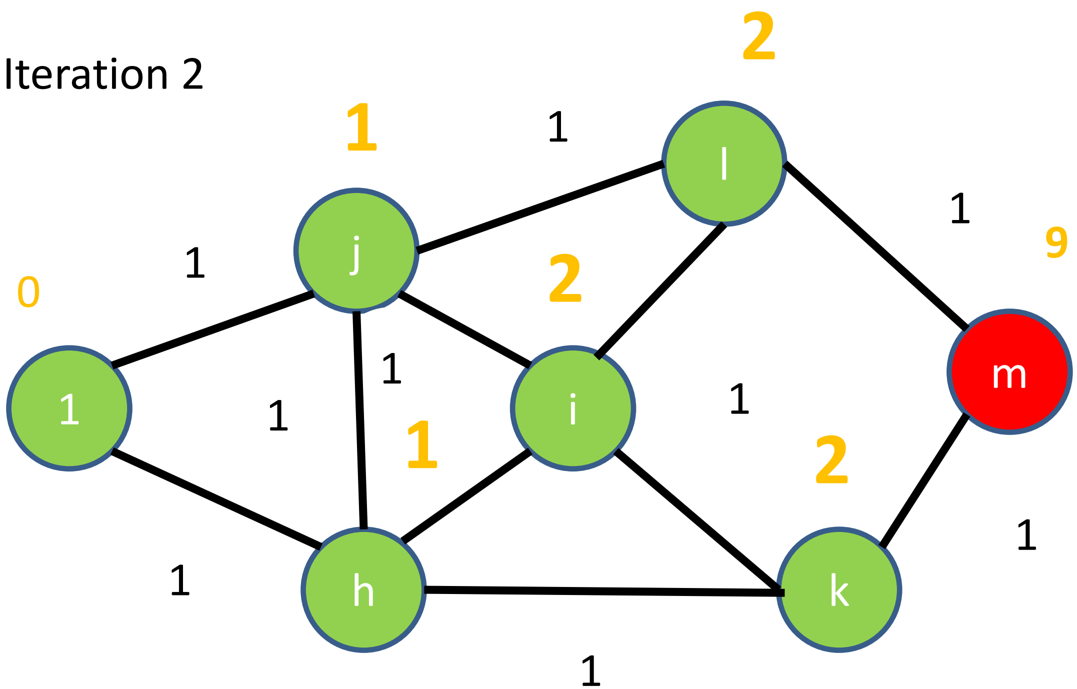


Figure 3.16 shortest path at iteration 2

3.4.2 Bellman-Ford algorithm

- At iteration k, all the shortest path distances with length k are obtained.
- $x_i(k+1) = \min\left(x_i(k), \min_{j \in A(i)}\left(a_{ij} + x_j(k)\right)\right), i = 2, \dots n,$

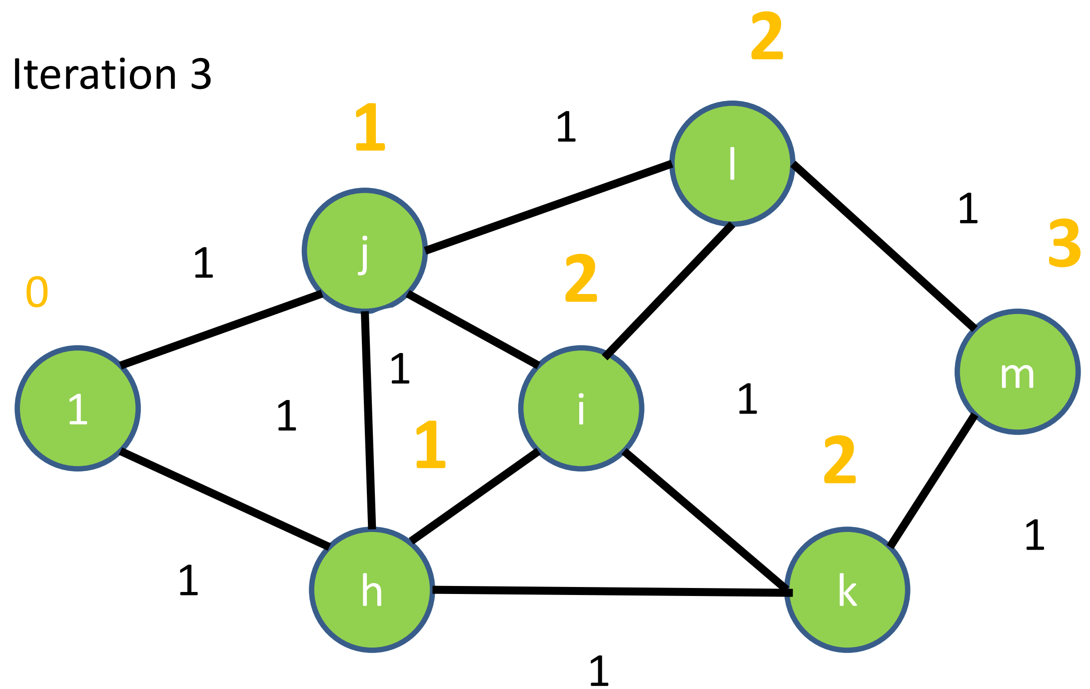


Figure 3.17 shortest path at iteration 3

3.4.3 Bellman-Ford complexity

- Complexity of the Bellman-Ford algorithm

  $\boldsymbol{O(a)}$ additions and comparisons at each iteration.

  where $a = Card(A)$.

  Time complexity

  $\boldsymbol{O(ma)}$ with $\boldsymbol{x_i(0)} = \boldsymbol{+\infty, i = 2, \ldots, n}$,

  $\boldsymbol{m}$ largest shortest path.
- **Polynomial bounded time.**

3.4.4 All-pairs shortest path

- Use Bellman-Ford algorithm for all pairs shortest path
- Total time complexity: $\boldsymbol{O(nma)}$

3.4.5 Other algorithms

- All-pairs shortest paths
- Complexity of algorithms

Weights	Time complexity	Algorithm
$\mathbb{R}$ (no negative cycles)	$O(n^3)$	Floyd–Warshall algorithm
$\mathbb{N}$	$O\left(n^3/2^{\omega(\log \mu)^{1/2}}\right)$	Williams 2014
$\mathbb{R}$ (no negative cycles)	$O(an + n^2 \log n)$	Johnson–Dijkstra
$\mathbb{R}$ (no negative cycles)	$O(an + n^2 \log \log n)$	Pettie 2004
$\mathbb{N}$	$O(an + n^2 \log \log n)$	Hagerup 2000

3.5 Parallel Algorithms

- Parallel all-pairs shortest paths algorithms
- Based on parallel Bellman-Ford.

3.5.1 Distributed asynchronous algorithm

- Bertsekas MIT 1983
- Convergence of the distributed asynchronous Bellman-Ford algorithm from initial condition:

$x_1(0)=0,$

$x_i(0)=+\infty, i = 2, \ldots, n.$

due to monotonicity property of the fixed point operator:

$$x \leq x' \Rightarrow F(x) \leq F(x').$$

3.5.2 Implementation on a computing node or Intel Xeon Phi

- Computing node with several multicore CPUs
  - On the Grid5000 testbed we cand find computing nodes with up to 4 CPUs with 16 cores each.
  - On Intel Xeon Phi computing accelerator we have around 60 to 70 computing cores and vectorization.
  - Shared memory architecture.

3.5.2 Implementation on a computing node or Intel Xeon Phi

- Threads solve independently a single destination shortest path problem.
- Reduction operations (addition and maximum).
- Parallel time complexity
- $\boldsymbol{O(nm\,a/number_t)}$

➢ One may expect that computing time will be divided by 50 on those platforms or Intel Xeon Phi computing accelerator

3.5.2 Implementation on GPUs

- Task parallelism (same as on Intel Xeon Phi)

➢ Gain difficult to predict.

(may be 500 on P100 or V100)

- Loop parallelism

each thread performs only:

$$x_i(k+1) = \min\left(x_i(k), \min_{j \in A(i)}\left(a_{ij} + x_j(k)\right)\right), i = 2, \ldots n,$$

for some $i \in N$.

➢ Gain difficult to predict.

3.5.3 GPU

- Implementation of parallel hybrid centrality algorithm on computing accelerators GPUs.

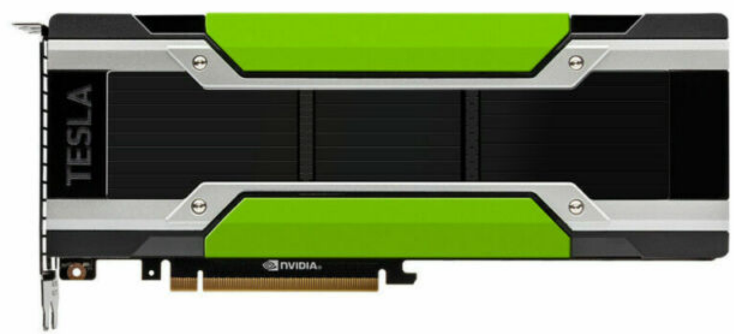


Figure 3.18 V100 GPU

3.5.3.1 GPU architecture

- Streaming Multiprocessor (SM) based GPU architecture.

Figure 3.19 NVIDIA Kepler GK110 architecture

3.5.3.1 GPU architecture

- Streaming Multiprocessor architecture

SMX Streaming Multiprocessor

Core Single Precision cuda core

DP Unit Douple preci-sion unit

LD/ST Loading and Storing unit

SFU Special function unit

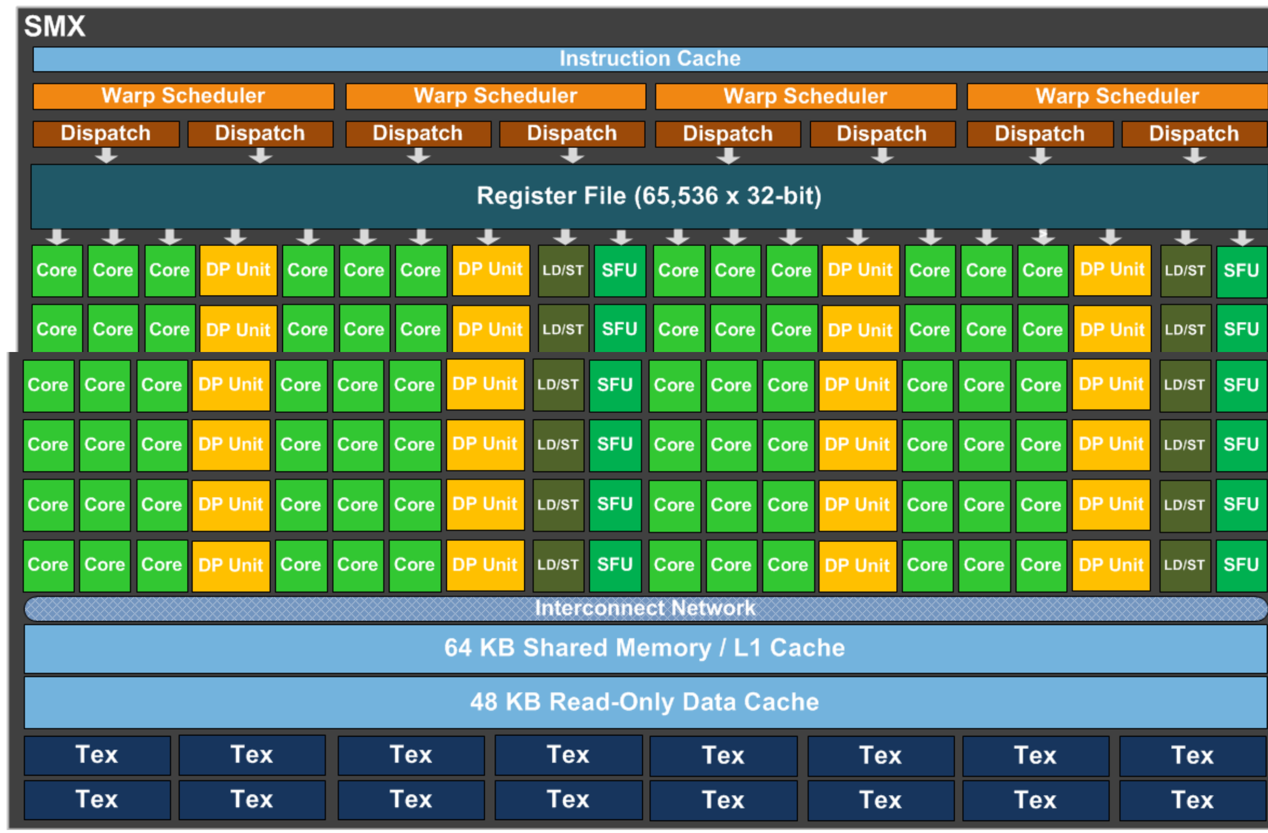


Figure 3.20. SM architecture (modified)

3.5.3.2 GPU Synthesis

- GPUs are massively parallel computing accelerators.
  - Thousands of CUDA cores.
  - GPUs provide all types of parallelism.

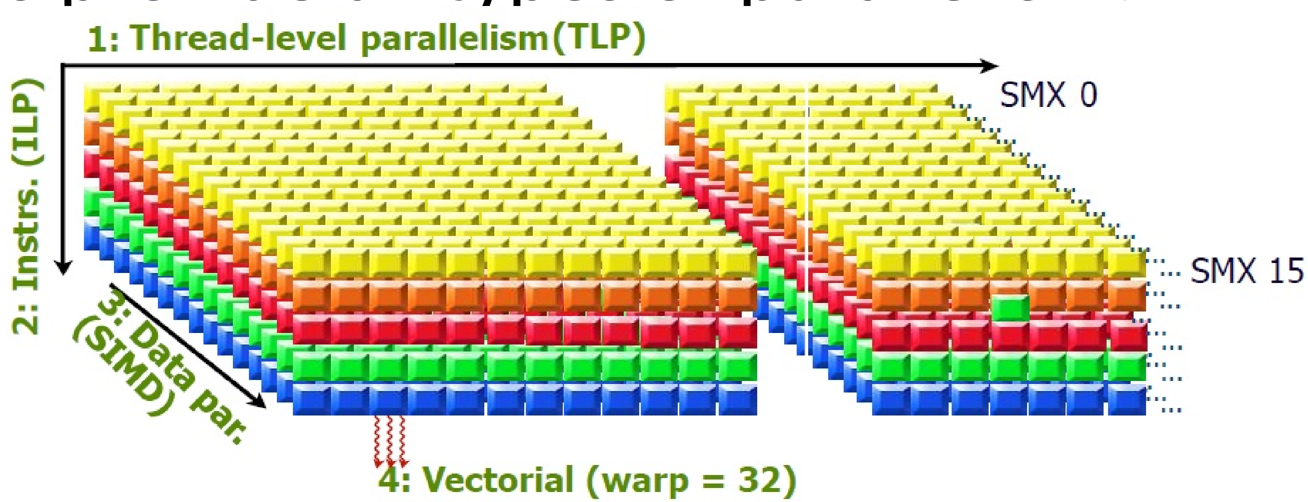


Figure 3.21 Several types of parallelsm in GPUs

3.5.3.2 Current Actions

- Parallel implementation of hybrid centrality algorithm via Gunrock
- Gunrock: a CUDA library for graph-processing designed specifically for the GPU. It uses a high-level, bulk-synchronous, data-centric abstraction focused on operations on vertex or edge frontiers.

3.5.3.2 Current Actions

- Gunrock achieves a balance between performance and expressiveness by coupling **high-performance GPU computing primitives and optimization strategies,** particularly in the area of fine-grained load balancing, with a high-level programming model that allows programmers to quickly develop new graph primitives that scale from one to many GPUs on a node with small code size and minimal GPU programming knowledge.

3.6.3 Current Actions

- Common work with:

➢ A. Benachour, LAAS-CNRS, U. of Toulouse &
U. Houari Boumediene

➢ Igor Kotenko SPIIRAS/ITMO U.
Andrey Chechulin SPIIRAS/ITMO U.
Maxim Kolomeec ITMO U. & U. Toulouse

3.6 Future work

- 2nd Phase: develop and optimize a parallel hybrid centrality CUDA code.

3.6 Future work

- 3rd phase: design and test other parallel / High Performance algorithms accelerated on GPUs for data structure analysis:
  - Diameter of the graph;
  - Maximum degree;
  - Clique number…
- Characterize groups of discussions in terms of the above metrics, e.g., politics, culture, food, sports,…

3.6 Future work

- 4th phase: graph analysis in conjunction with additional data associated with records like
  - Time of post;
  - Number of likes and number of reposts;
  - City of user.

4. Conclusions and future work

4.1 Present Work

- In this talk, I have proposed a parallel hybrid centrality algorithm based on degree of nodes and Bellman-Ford shortest paths algorithm in order to evaluate leaders in social nets

4.2 Present concern

- **With the development of the Internet and social networks, Big Data and graph analysis have become very important domains in Applied Maths.**

4.2 Present concern

- In particular, graph studies when combined with Artificial Intelligence can permit researchers in Human Sciences, journalists and politics, to understand new interactions between people, new uses, new aspirations, new needs and trends in our societies, e.g., the « Gilet Jaune » movement in France whose very nature is plural and distributed.
- They can permit also to increase security.

Acknowledgments

- Andrey Chechulin, SPIIRAS/ITMO
- Martin Strecker, IRIT
- Max Kolomeec, ITMO/UPS
- Igor Kotenko, SPIIRAS/ITMO