Data Mining

Lecture 11
Mining Structure from Graphs

http://www.informatik.uni-hamburg.de/WTM/

Overview

- Case Based Reasoning
- Spectral Clustering
- Clustering Graphs
  - SCAN
- Semantic Networks
- Bayesian Belief Networks
- Hidden Markov Models
- Webgraph: Google

Case Based Reasoning

- Remember k-nearest neighbours:
  - Task is to classify a new data point x_n
  - Find the k nearest points $\{x_{k'}\}$ with their class labels $\{y_{k'}\}$
  - Assign class y_n based on the majority vote of $\{y_{k'}\}$
- i.e. use existing data $\{x_{k'}, y_{k'}\}$ ("past experience") directly for future decisions
- k=1: decide as in one precedence case

CBR – A way to solve complex problems

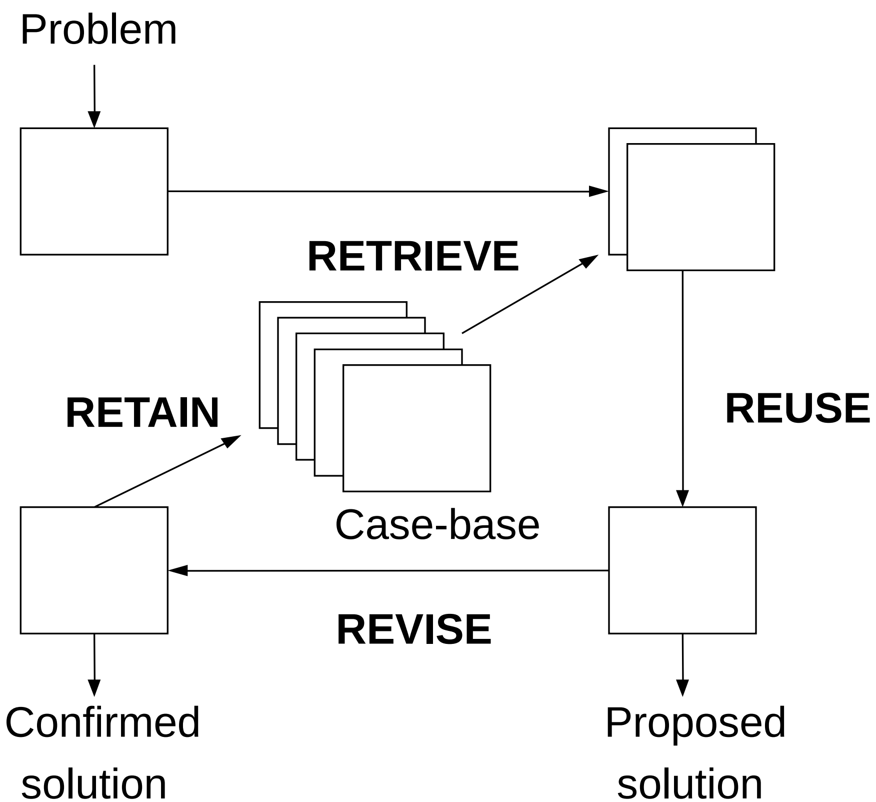
- By remembering how we solved a similar case in the past
- This is Case Based Reasoning (CBR)
  - memory-based problem-solving
  - re-using past experiences
- Experts often find it easier to relate to past cases than to formulate rules about reasoning

CBR – Problems we solve this way

- Medicine
  - doctor remembers previous patients
    - especially for rare combinations of symptoms
- Law
  - law depends on precedence
  - case histories are consulted
- Management
  - decisions are often based on past rulings
- Financial
  - performance is predicted by past results
- Robotics
  - Robot soccer – imitate good moves

CBR – Overview

- CBR provides an automated method for storing experience and reusing it to make decisions in the future



CBR Prerequisites

- Expertise is embodied in a library of past cases (experiences)
- Each case typically contains
  - a *description* of the problem
  - *goals*, and subgoals that arise in reasoning
  - *successful attempts* at achieving those goals

    → to propose solutions to new problems
  - *failed attempts*

    → to warn of possible failure

CBR Process

- Basic algorithm to solve a current problem:
  - Match the problem's features against the cases in the case base, and retrieve similar cases.
    - If multiple solutions are found then resolve any ambiguities.
  - Reuse retrieved cases to propose a solution and test for success.
  - Revise the solution, if necessary.
  - Retain the current problem as part of a new case, i.e.
    - its defining features,
    - goals & subgoals,
    - successful & failed attempts,
    - its final solution.

CBR Evaluation

- What does the CBR process depend on?
  - Appropriate methods for ***indexing cases*** using their key attributes
  - Efficient mechanisms for ***retrieving cases*** given a set of index values
  - Existing cases – a smaller case-base can be compensated for by more creativity in retrieval and revision
  - Good ***presentation*** of the information to the user

What are good CBR Applications?

- Failure prediction
  - ultrasonic non-destructive testing for Dutch railways
  - water in oil wells for Schlumberger
- Failure analysis
  - Mercedes cars for DaimlerChrysler
  - semiconductors at National Semiconductor
- Maintenance scheduling
  - Boeing 737 engines
  - TGV trains for SNCF
- Planning
  - mission planning for US navy
  - route planning for DaimlerChrysler cars

CBR in Business

“those who ignore history are doomed to repeat it“

Norwegian CBR consultant Verdande started in the oil business

CBR as Presented by “Verdande Technology”

- ”Based on the principle that similar problems have similar solutions, CBR .. analyzes data patterns in real-time, using past events to proactively predict future problems.”
- ”harvests multiple, heterogeneous data types to index and search for those past experiences and provides organizations with the information they need”
- ”transforms big data into actionable insight”
- ”offering a realistic assessment as to whether a similar scenario is likely to occur in the future”
- ”can help reduce drilling NPT” (non-productive time) by:
  - ”Identify problem precursors.
  - Interpret and resolve the drilling situation.
  - Retrieve relevant solutions and lessons learned.”

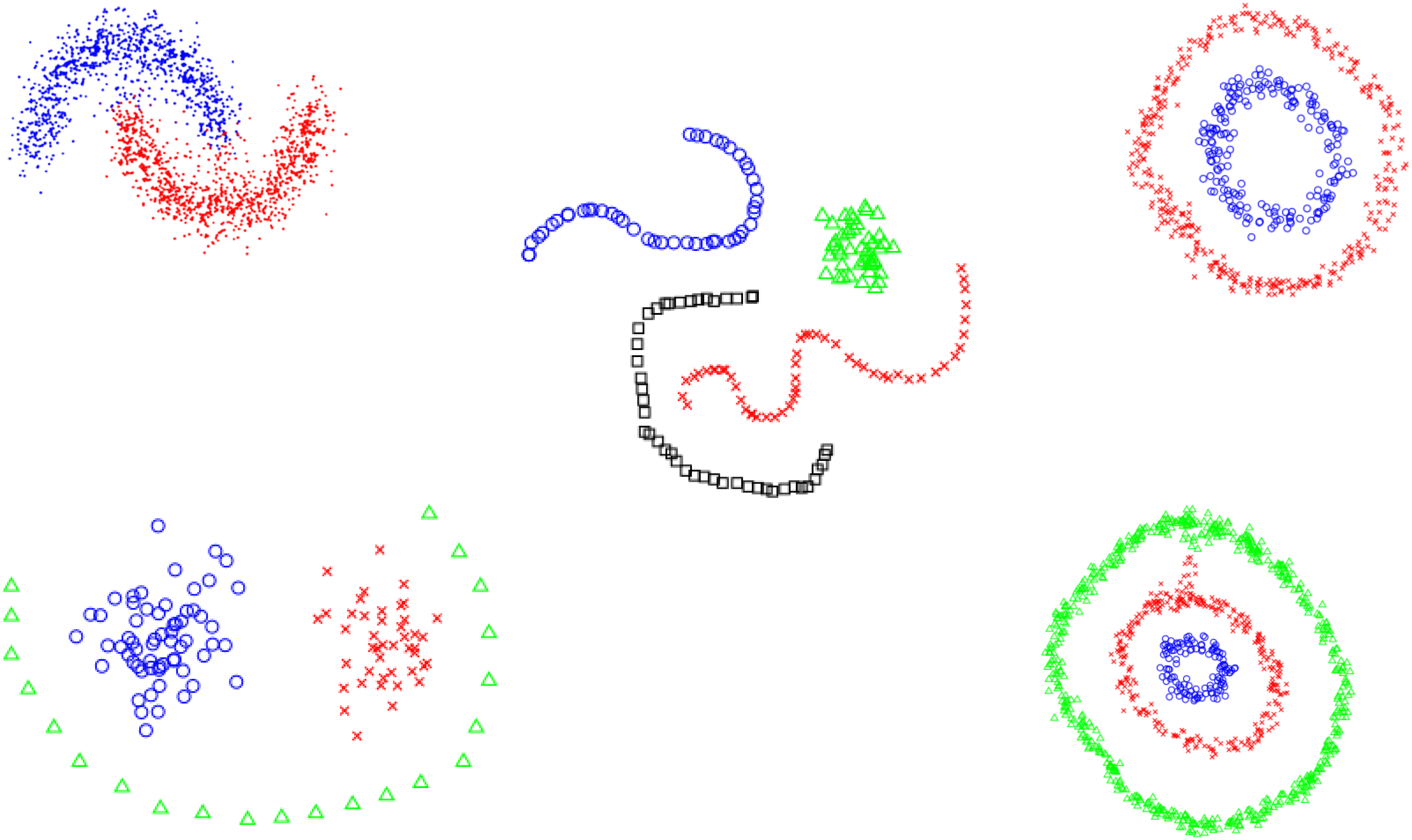
CBR – Summary

- CBR does not require an explicit domain model
  - elicitation becomes a task of gathering case histories
- Implementation is reduced to identifying significant features that describe a case
  - easier than creating an explicit model
- CBR systems ”learn” by acquiring new knowledge as cases
  - makes maintenance easy
- CBR retrieves entire cases, including context information that may not be asked for
  - may dig out unexpected problem features and solutions

Overview

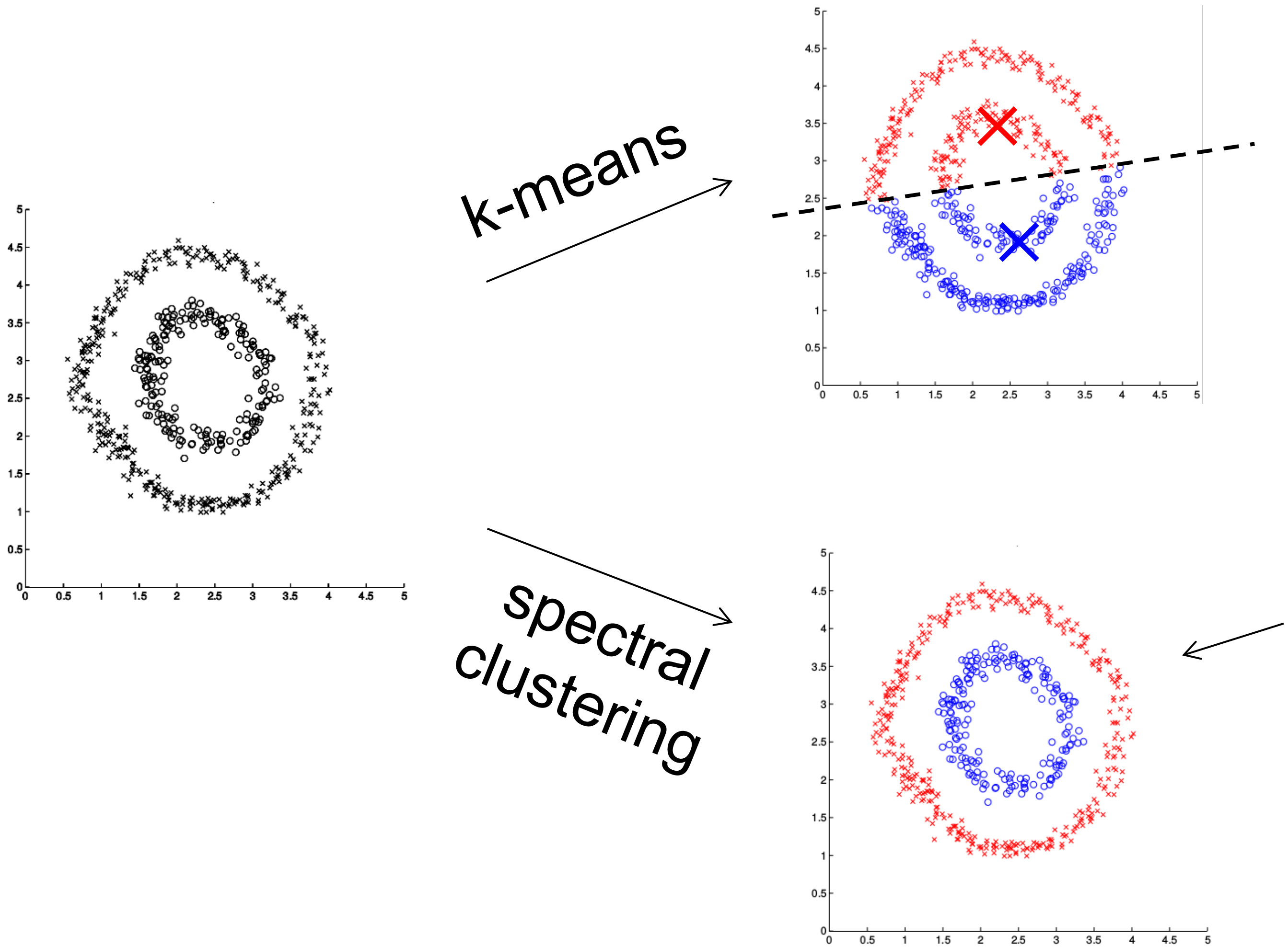
- Case Based Reasoning
- Spectral Clustering
- Clustering Graphs
  - SCAN
- Semantic Networks
- Bayesian Belief Networks
- Hidden Markov Models
- Webgraph: Google

Spectral Clustering



Ng, Jordan, Weiss. On Spectral Clustering: Analysis and an algorithm. NIPS, 2001

Spectral Clustering

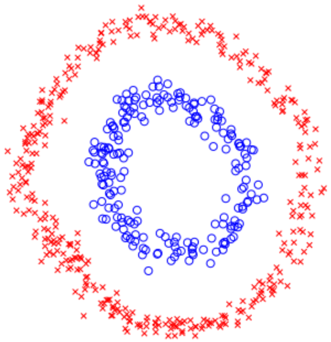


here, 2 centers and a linear boundary

not possible with 2 centers and a linear boundary

Ng, Jordan, Weiss. On Spectral Clustering: Analysis and an algorithm. NIPS, 2001

Spectral Clustering



- Idea: Information about the distances between any two data points will be needed

  $\rightarrow$ similarity (affinity) matrix $A \in R^{nxn}$

  where n = number of data points

  - Affinities A_{ij} large if data points i and j nearby
  - $A_{ij} \approx 0$ if i and j far apart (in different clusters)
  - $A_{ij} = A_{ji}$ i.e. A is symmetric

  $\rightarrow$ Eigenvectors orthogonal (for unequal eigenvalues)

Spectral Clustering

- Uses spectrum (eigenvalues) of affinity matrix

  $A \in R^{nxn}$ (*n* = number of data points)
- Mapping to R^k reduces dimensionality

  (*k = number of clusters; k << n*)
- It will turn out: mapped to R^k, data will form tight clusters at 90° to each other w.r.t. origin
- Ng, Jordan, Weiss (2001) model one of many variations

Spectral Clustering Algorithm

- Given a set of points $S = \{s_1, \ldots, s_n\} \in R^l$
- Aim: compute direct *k*–way partitioning
  - (NOT: generate two clusters, then recurse to generate more)
  - *k*–way partitioning experimentally proven better

Spectral Clustering Algorithm

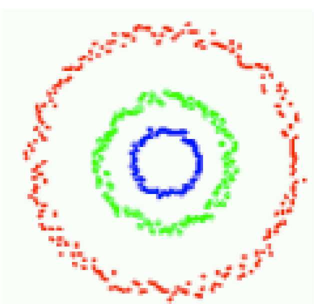
- Form the affinity matrix $A \in R^{nxn}$
- Define $A_{ij} = e^{-\|s_i - s_j\|^2 / 2\sigma^2}$ if $i \neq j$ else $A_{ii} = 0$
  - Scaling parameter σ chosen by user
  - Actual data values get lost – only proximities count!
- *Only for normalisation,* define *D* a diagonal matrix whose *(i,i)* element is the sum of *A*'s row *i*
  - D_{ii} is large for elements *i* that have many large A_{ij} (i.e. many neighbours)

Spectral Clustering Algorithm

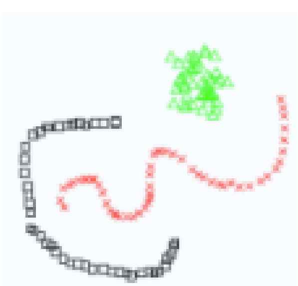
- Form the matrix $L = D^{-1/2} A D^{-1/2}$
- Find $x_1, x_2, \ldots, x_k$, the *k* eigenvectors of *L* that belong to the largest eigenvalues
  - these eigenvectors show to directions of largest, 2nd-largest, … covariance
- These form the columns of the new matrix *X*
  - Note: dimension reduces from $n \times n$ to $n \times k$

Spectral Clustering Algorithm

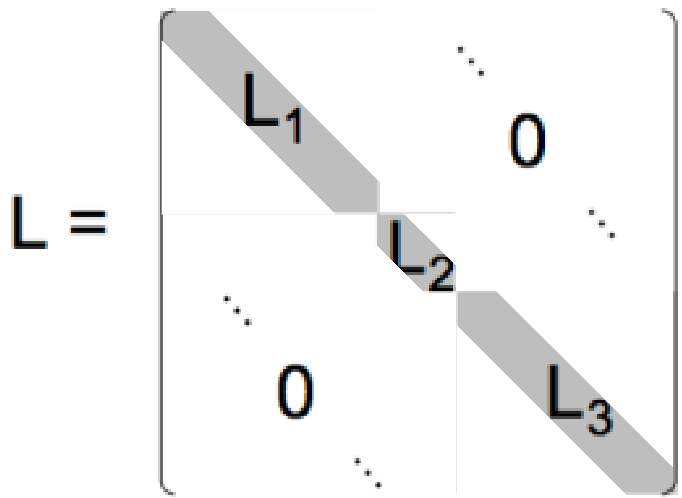
data



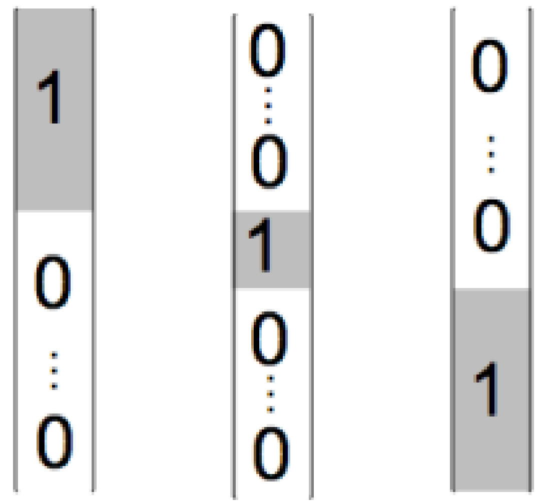
OR



matrix L



matrix X



First three eigenvectors

Spectral Clustering Algorithm

- *For normalisation,* from X form the matrix $Y \in R^{nxk}$
  - Renormalize each of X's rows (it has n rows) to have unit length

$$Y_{ij} = X_{ij} / (\sum_j X_{ij}^{\ 2})^2$$

- Treat each row of Y as a point in R^k
  - all points will lie on a unit circle around the origin
  - they will form clusters at 90° to each other

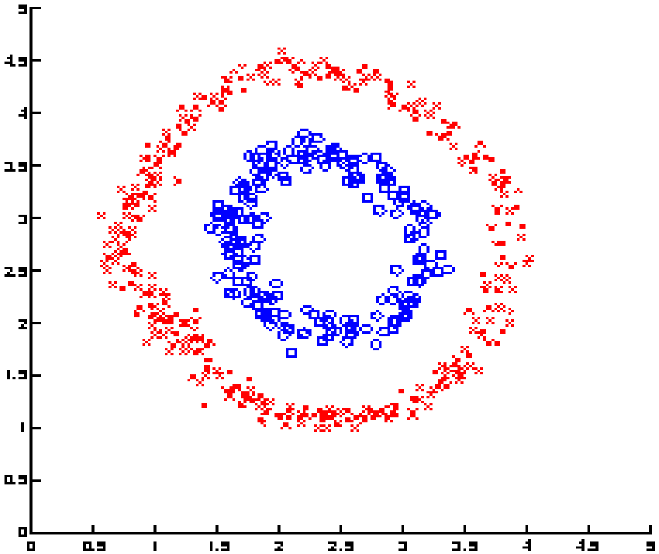
Spectral Clustering Algorithm

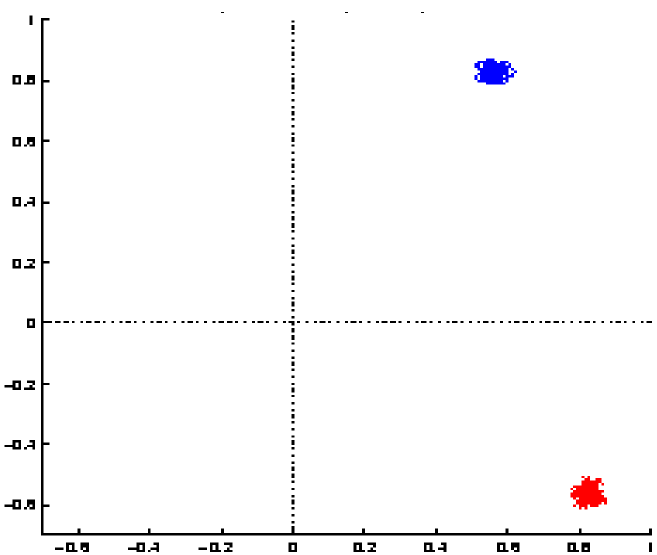
- Cluster into *k* clusters, e.g. via k-means
- Final cluster assignment
  - Assign point s_i to cluster *j*

    iff row *i* of *Y* was assigned to cluster *j*

Results

Spectral Clustering

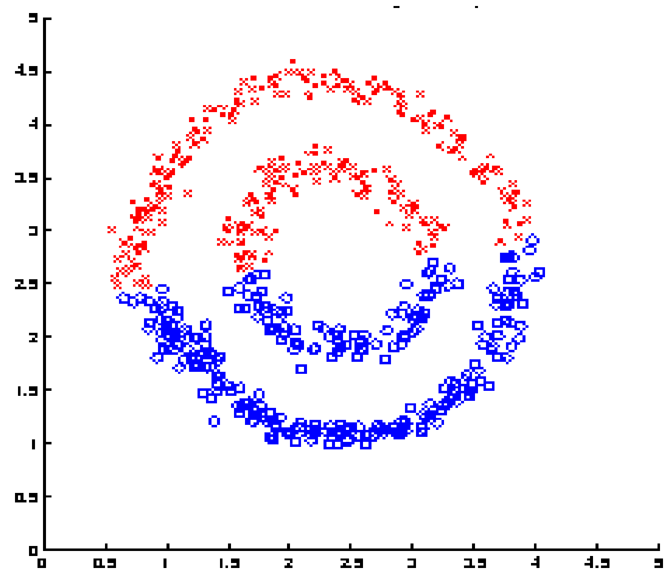




k=2

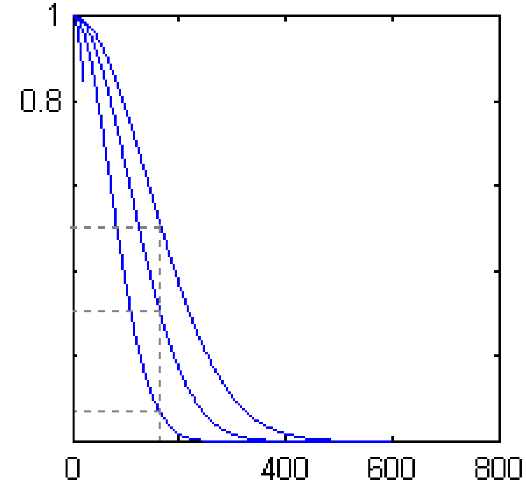
rows of Y

Comparison: k-Means (directly on the data)



Spectral Clustering: Choice of σ

$$A_{ij} = e^{-(s_i - s_j)^2/2\sigma^2} \quad i \neq j \qquad A_{ii} = 0$$



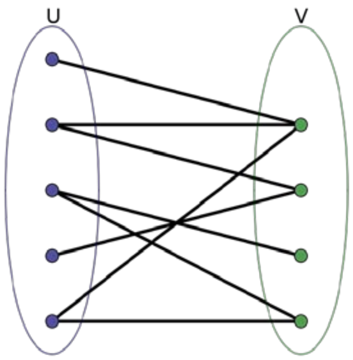
"closer" vertices get larger weight

- search over σ
  - pick the value that – after clustering Y's rows – gives the tightest (smallest distortion) clusters

Overview

- Case Based Reasoning
- Spectral Clustering
- Clustering Graphs
  - SCAN
- Semantic Networks
- Bayesian Belief Networks
- Hidden Markov Models
- Webgraph: Google

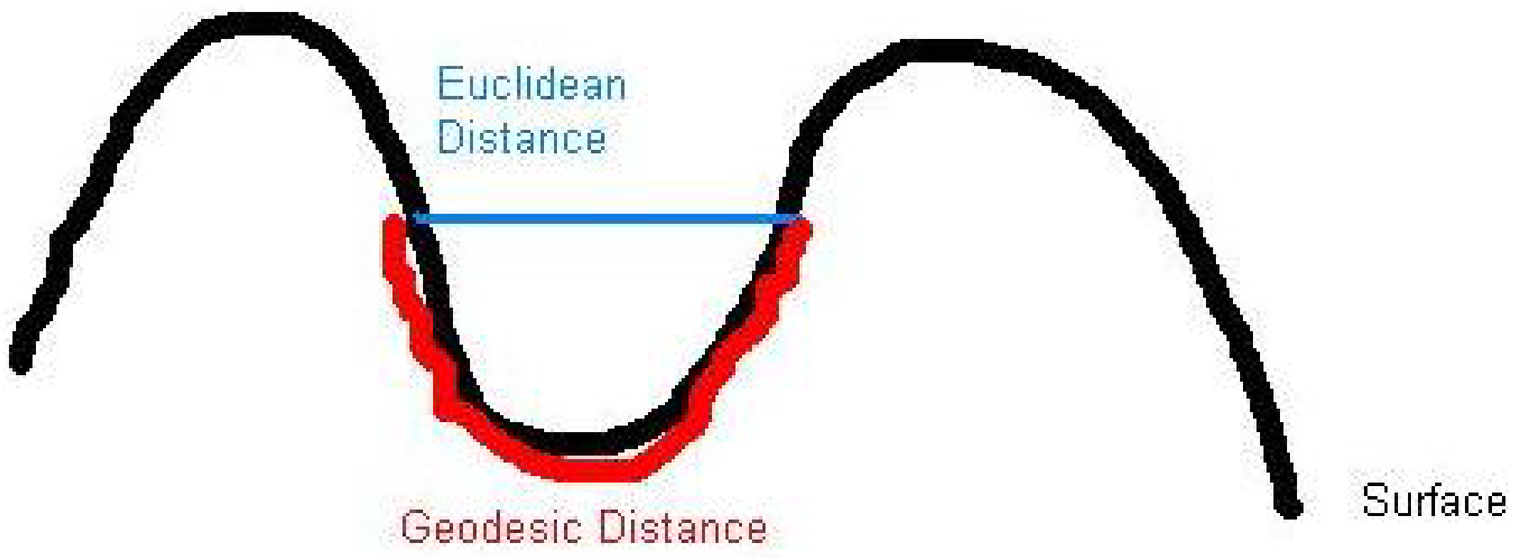
Clustering Graphs and Network Data



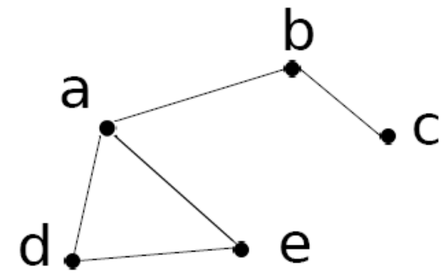
- Applications
  - Bipartite graphs, e.g.:
    - customers and products,
    - authors and conferences, …
  - Web search engines, e.g.:
    - click-through graphs, webgraph, …
  - Social networks, friendship/coauthor graphs
- Similarity measures
  - Geodesic distances
  - SimRank distance
- Graph clustering methods
  - Minimum cuts: FastModularity (Clauset, Newman & Moore, 2004)
  - Density-based clustering: SCAN (Xu et al., KDD'2007)

Similarity Measures: Geodesic Distances (1)

- ***Geodesic distance***: distance along curved spaces
- May be approximated by adding many short straight segments, using the Euclidean distance for each of these



Geodesic Distances (2)

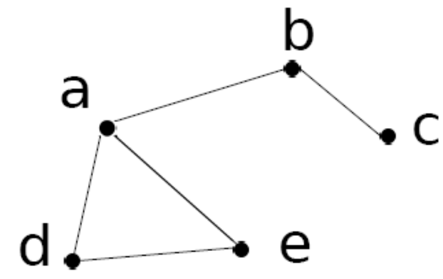


- ***Geodesic distance*** (A, B): length (i.e., # of edges) of the shortest path between A and B (if not connected, defined as infinite)
- ***Eccentricity*** of v, eccen(v): The largest geodesic distance between v and any other vertex $u \in V - \{v\}$.
  - E.g.,

    eccen(a) = eccen(b) = 2;

    eccen(c) = eccen(d) = eccen(e) = 3
- A ***peripheral vertex*** is a vertex that achieves the diameter.
  - **E.g.**, Vertices c, d, and e are peripheral vertices

Geodesic Distances (3)



- ***Radius*** of graph G:
  The minimum eccentricity of all vertices,
  i.e., the distance between the
  "most central point" and the "farthest border"
  - $r = \min_{v \in V} \text{eccen}(v)$
  - **E.g.**, radius (g) = 2

- ***Diameter*** of graph G: The maximum eccentricity of all vertices, i.e., the largest distance between any pair of vertices in G
  - $d = \max_{v \in V} \text{eccen}(v)$
  - **E.g.**, diameter (g) = 3

SimRank (1)

- SimRank: ***structural-context similarity***
  - based on similarity of its neighbors:
    Similar objects are referenced by similar objects

- In a ***directed graph*** G = (V,E),
  - individual in-neighborhood of v: $I(v) = \{u \mid (u, v) \in E\}$
  - individual out-neighborhood of v: $O(v) = \{w \mid (v, w) \in E\}$

SimRank (2) – Similarity by Fix Point Iteration

- ***Similarity*** in SimRank: $$s(u,v) = \frac{C}{|I(u)| \cdot |I(v)|} \sum_{x \in I(u)} \sum_{y \in I(v)} s(x,y)$$
  - Problem: recursive formula
- Solution: Iteration to a fixed point:
  - Initialization:
    $$s_0(u,v) = \begin{cases} 0 & \text{if } u \neq v \\ 1 & \text{if } u = v \end{cases}$$
  - Then we can compute s_{t+1} from s_t based on the definition:
    $$s_{t+1}(u,v) = \frac{C}{|I(u)| \cdot |I(v)|} \sum_{x \in I(u)} \sum_{y \in I(v)} s_t(x,y)$$
  - Typical parameters: decay factor *C=0.8,* number of iterations *T=5.*

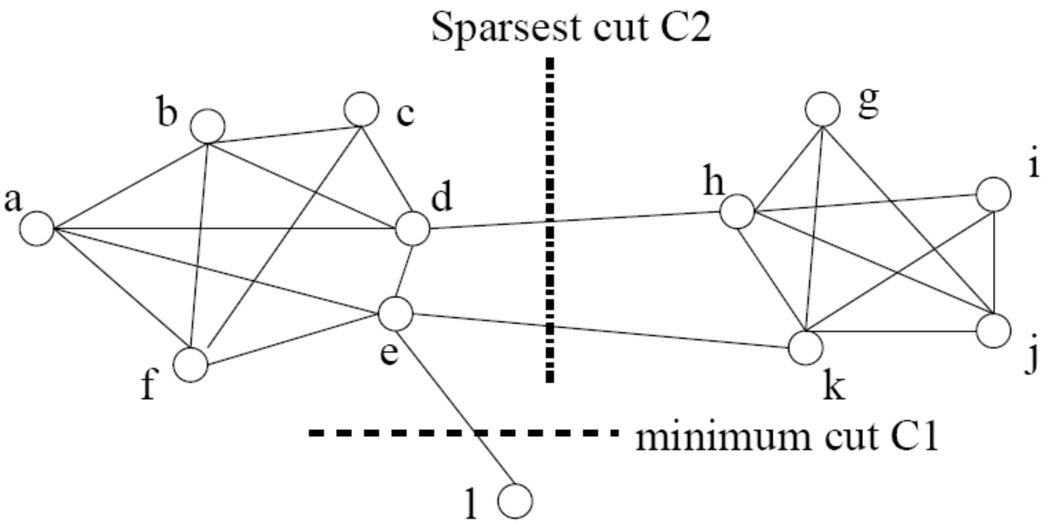
SimRank (3) – Similarity by Random Walk

- Similarities can be assessed by randomly traversing links
  - start the walks at the nodes in question
  - random walker has a given probability (not) to continue the walk
  - short tours more likely than long tours
- Highly similar nodes can be identified by:
  - small expected distance,
  - small expected meeting distance for a pair of tours,
  - high expected meeting probability.

Jeh, Widom (ca. 2002) SimRank: A Measure of Structural-Context Similarity

Graph Clustering: Sparsest Cut (1)

- ***Undirected graph*** G = (V,E). The ***cut set*** of a cut is the set of edges $\{(u, v) \in E \mid u \in S, v \in T\}$ where nodes u and v are in the two partitions S and T



- ***Size*** of the cut:
  \# of edges in the cut set

- Min-cut (e.g., C_1)
  is not a good partition

- A better measure: ***Sparsity***

$$\Phi = \frac{\text{the size of the cut}}{\min\{|S|, |T|\}}$$

Graph Clustering: Sparsest Cut (2)

- A cut is *sparsest* if its sparsity is not greater than that of any other cut
- **Ex**. Cut C2 = ({a, b, c, d, e, f, l}, {g, h, i, j, k}) is the sparsest cut
- For k clusters, the *modularity* of a clustering assesses the quality of the clustering
- The *modularity* of a clustering of a graph is the difference between the fraction of all edges *within* individual clusters (this should be large) and the fraction of all edges *between* different clusters (this should be small)
- The optimal clustering of graphs maximizes the modularity

Graph Clustering: Challenges of Finding Good Cuts

- High computational cost
  - Many graph cut problems are computationally expensive
  - The sparsest cut problem is NP-hard
  - Need to tradeoff between efficiency/scalability and quality
- Sophisticated graphs
  - May involve weights and/or cycles
- Large graphs with many vertices
  - The similarity matrix contains as many non-zero entries as there are vertices in the graph
- Sparsity
  - A large graph is often sparse, meaning each vertex on average connects to only a small number of other vertices
  - A similarity matrix from a large sparse graph will also be sparse

Two Approaches for Graph Clustering

1. Using generic clustering methods for high-dimensional data
   - Extract a similarity matrix $A \in R^{nxn}$ from a graph using a similarity measure (*n* = number of nodes)
     - e.g. similarity = 1 if nodes connected, else 0
   - Discover clusters on the similarity matrix by a generic clustering method, e.g. spectral clustering

     $\rightarrow$ approximate optimal graph cut solutions

2. Methods specifically designed for clustering graphs
   - Search the graph to find well-connected components as clusters
   - Ex. SCAN: Structural Clustering Algorithm for Networks

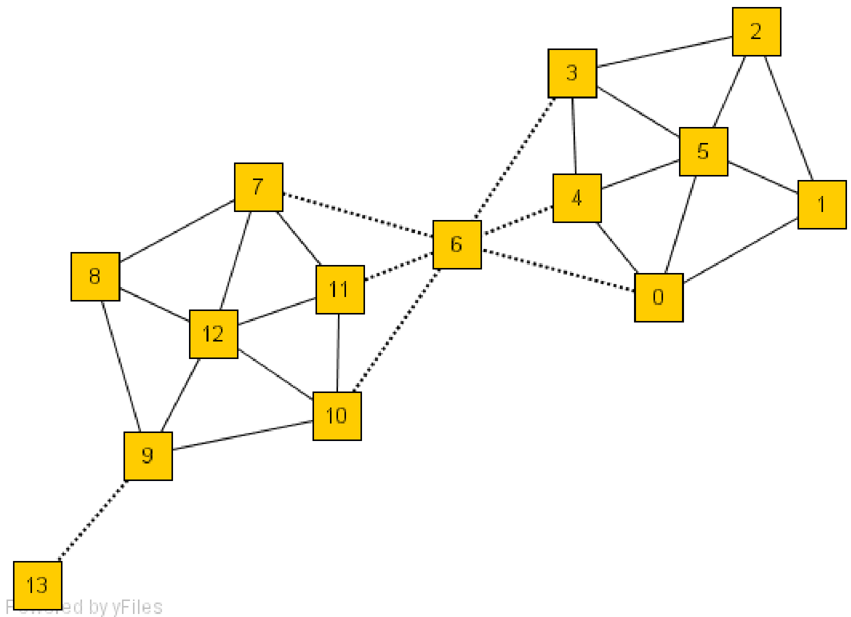
     [Xu, Yuruk, Feng, Schweiger, KDD'07]

Overview

- Case Based Reasoning
- Spectral Clustering
- Clustering Graphs
  - SCAN
- Semantic Networks
- Bayesian Belief Networks
- Hidden Markov Models
- Webgraph: Google

SCAN: Density-Based Clustering of Networks

- How many clusters?
- What size should they be?
- What is the best partitioning?
- Should some points be segregated?

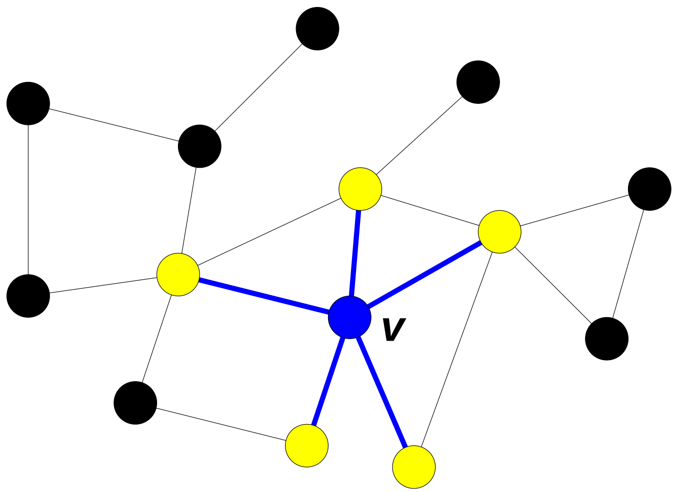


Application

- Given: information of who associates with whom
- Identify clusters of individuals with common interests or special relationships (families, cliques, terrorist cells) …

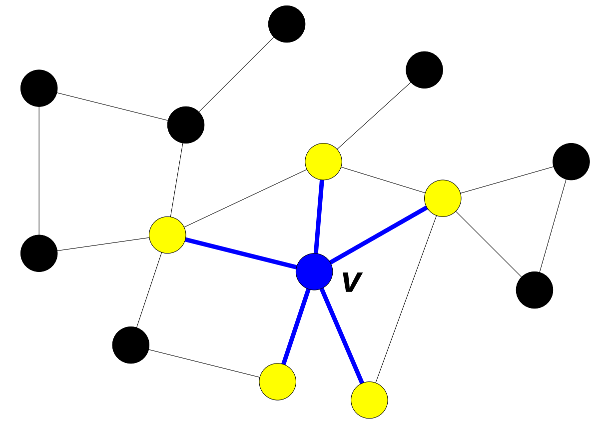
A Social Network Model

- Characteristics:
  - Individuals in a tight social group, or ***clique***, know many of the same people, regardless of the size of the group
  - Individuals who are ***hubs*** know many people in different groups but belong to no single group. E.g., politicians bridge multiple groups
  - Individuals who are ***outliers*** reside at the margins of society. E.g., hermits know few people and belong to no group
- The neighborhood of a vertex
  - Define $\Gamma(\nu)$ as the immediate neighborhood of a vertex (i.e. the set of people that an individual knows)



Structure Similarity

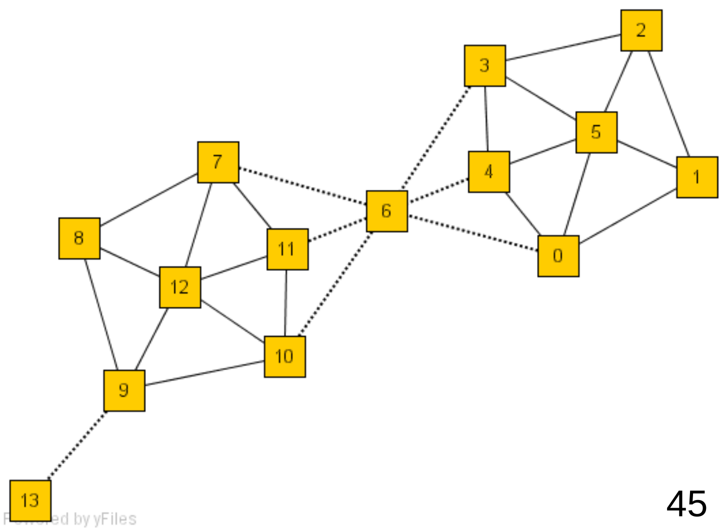
- The desired characteristics tend to be captured by a measure we call Structural Similarity



$$\sigma(v,w) = \frac{|\Gamma(v) \bigcap \Gamma(w)|}{\sqrt{|\Gamma(v)| \cdot |\Gamma(w)|}}$$

  - $0 \leq \sigma \leq 1$

- Structural similarity is large for members of a clique and small for hubs and outliers



Structural Connectivity

- ε-neighborhood: $N_\varepsilon(v) = \{w \in \Gamma(v) \mid \sigma(v,w) \geq \varepsilon\}$
- Vertex is a core: $CORE_{\varepsilon,\mu}(v) \Leftrightarrow |N_\varepsilon(v)| \geq \mu$ μ integer

  we will let structures grow starting from the core
- Direct structure reachable:

$$DirREACH_{\varepsilon,\mu}(v,w) \Leftrightarrow CORE_{\varepsilon,\mu}(v) \wedge w \in N_\varepsilon(v)$$

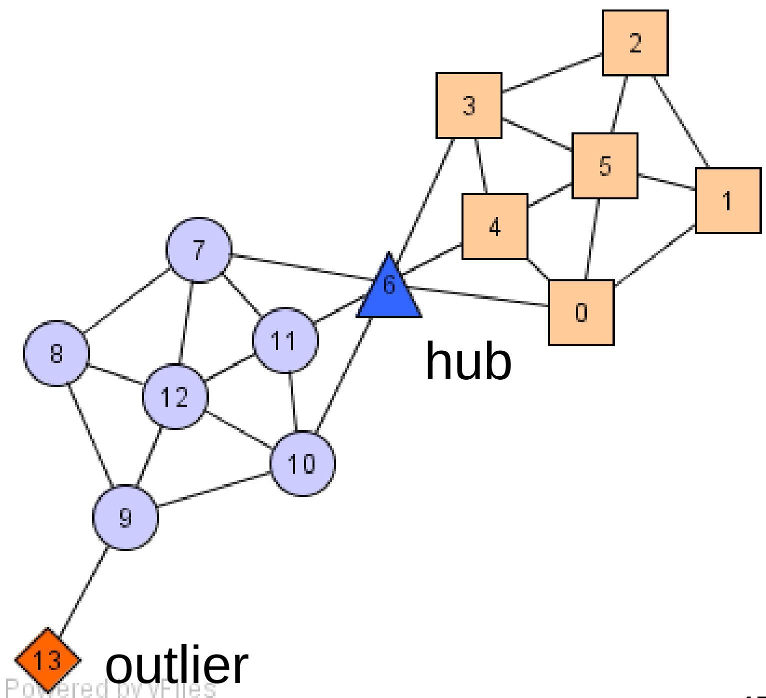
- Structure reachable: $REACH_{\varepsilon,\mu}(v,w)$ transitive closure of direct structure reachability
- Structure connected:

$$CONNECT_{\varepsilon,\mu}(v,w) \Leftrightarrow \exists u \in V : REACH_{\varepsilon,\mu}(u,v) \wedge REACH_{\varepsilon,\mu}(u,w)$$

Xu, Yuruk, Feng, Schweiger (SIGKDD'07) "SCAN: A Structural Clustering Algorithm for Networks".
See also: Ester, Kriegel, Sander, Xu (KDD'96) "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases".

Structure-Connected Clusters

- Define a structure-connected cluster *C:*
  - Connectivity: $\forall v, w \in C : CONNECT_{\varepsilon,\mu}(v, w)$
  - Maximality: $\forall v, w \in V : v \in C \wedge REACH_{\varepsilon,\mu}(v, w) \Rightarrow w \in C$
- **Hubs**:
  - Not belong to any cluster
  - Bridge to many clusters
- **Outliers**:
  - Not belong to any cluster
  - Connect to less clusters



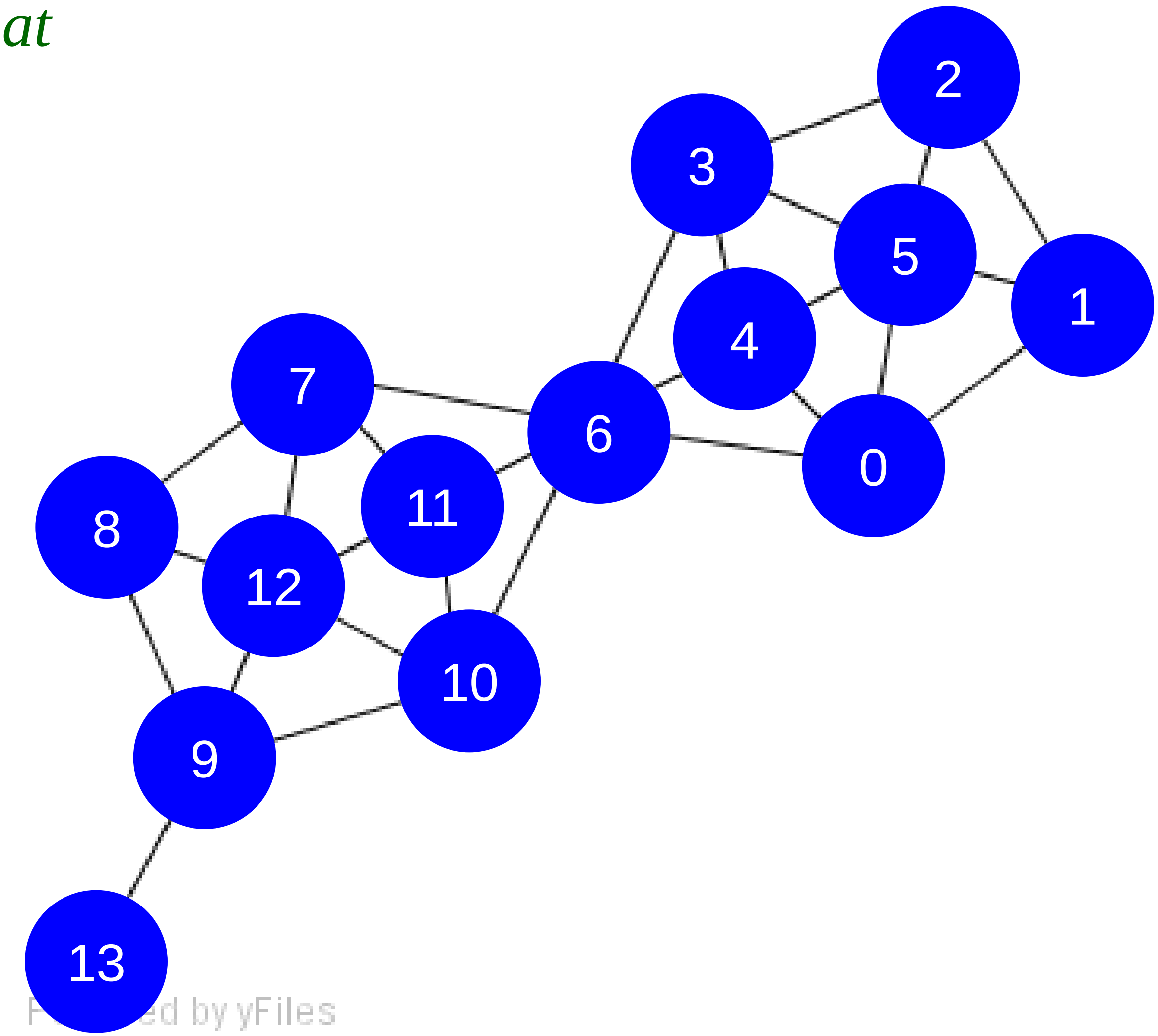
SCAN Algorithm

neighbours that vertex is core

$\mu = 2$

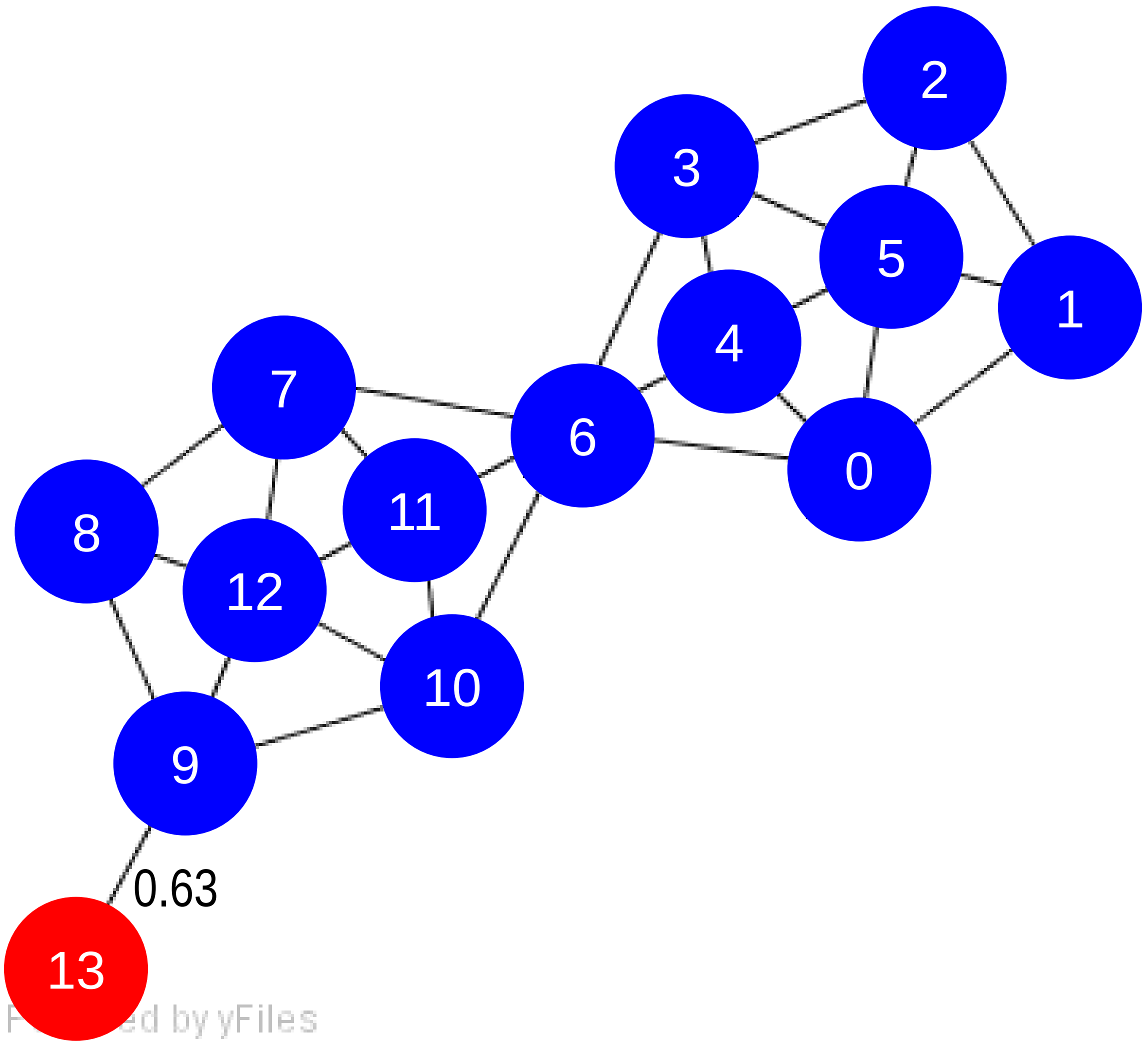
$\varepsilon = 0.7$

required similarity



SCAN Algorithm

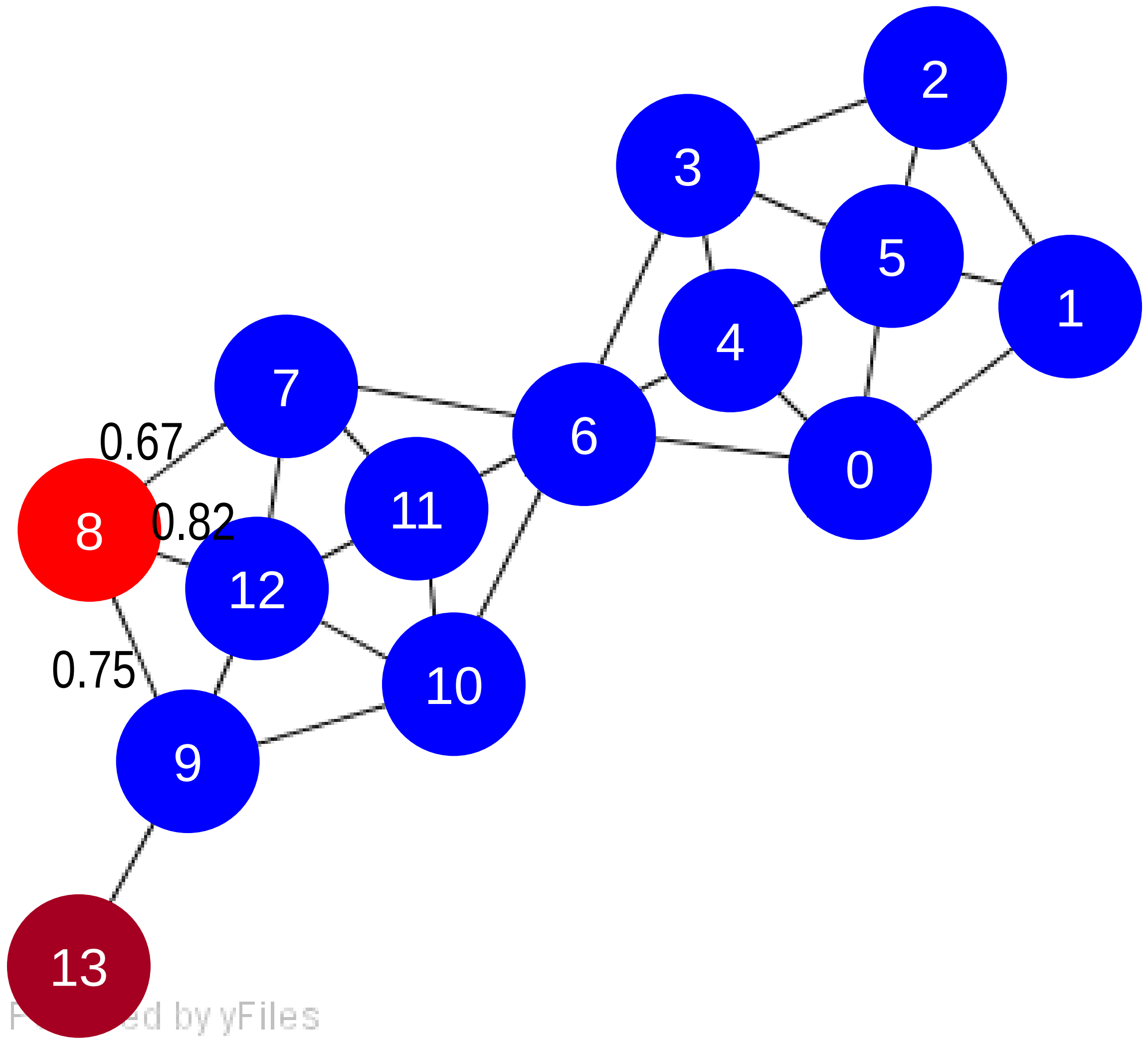
$\mu = 2$
$\varepsilon = 0.7$



SCAN Algorithm

$\mu = 2$

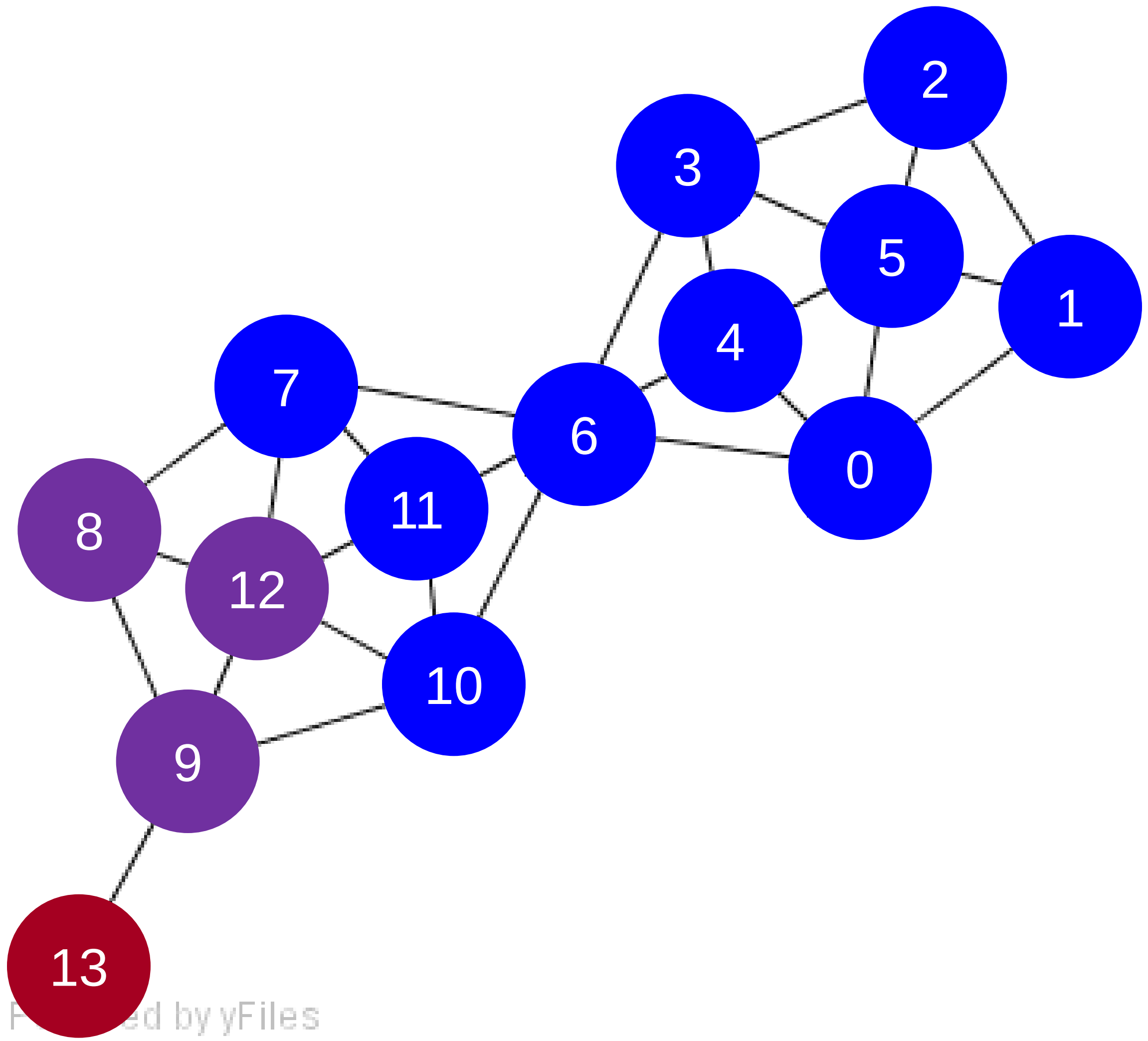
$\varepsilon = 0.7$



SCAN Algorithm

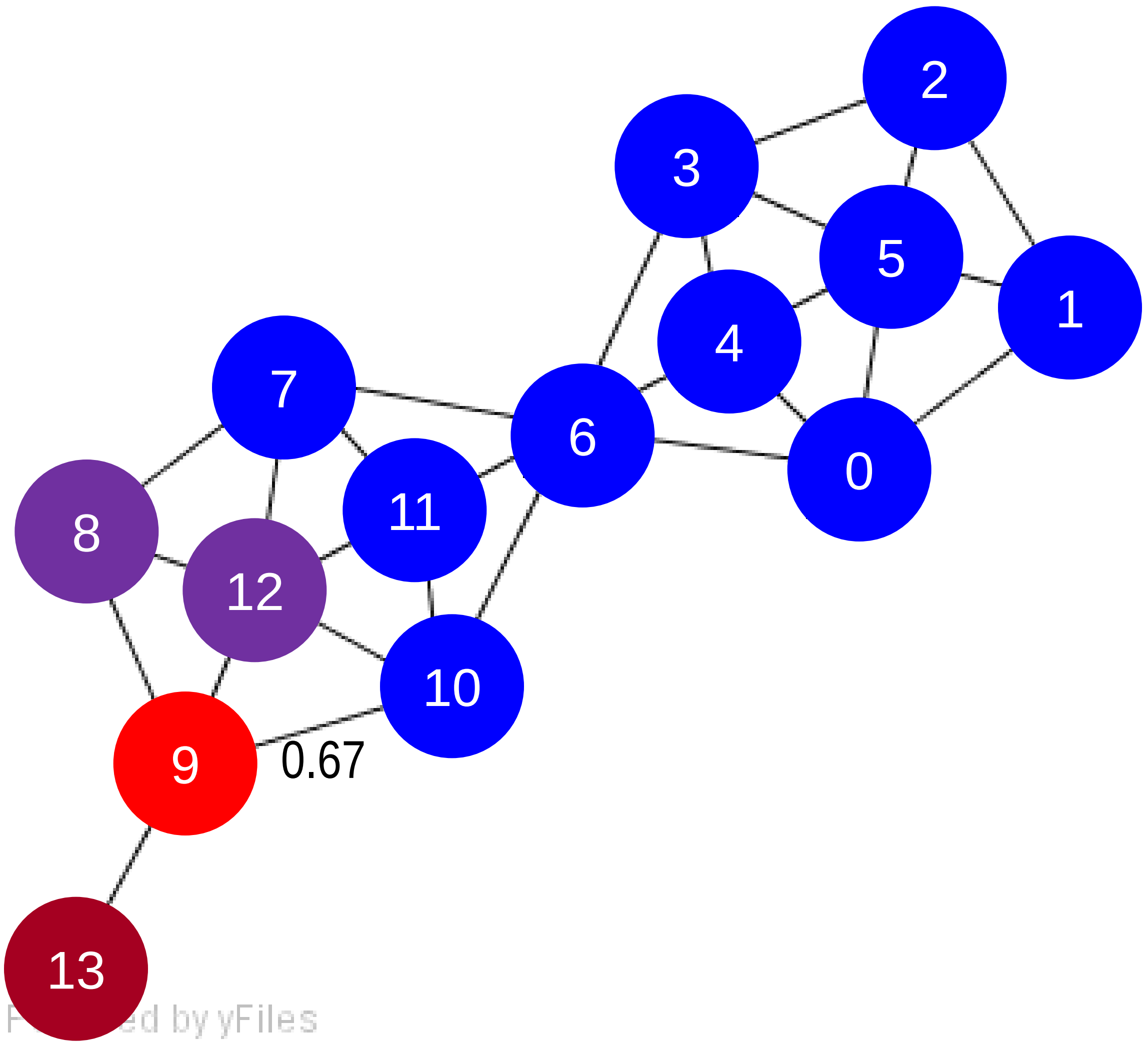
$\mu = 2$

$\varepsilon = 0.7$



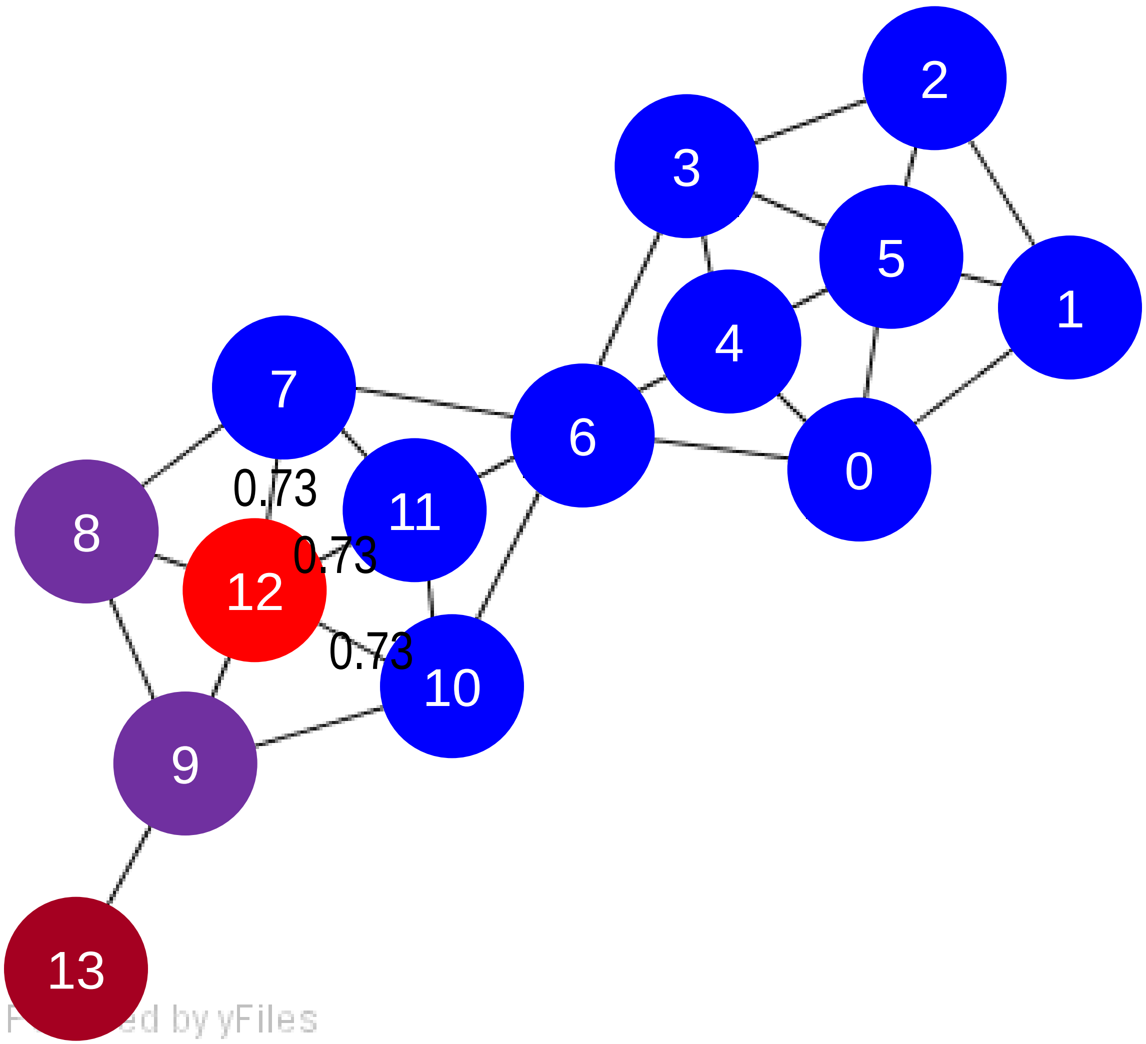
SCAN Algorithm

$\mu = 2$
$\varepsilon = 0.7$



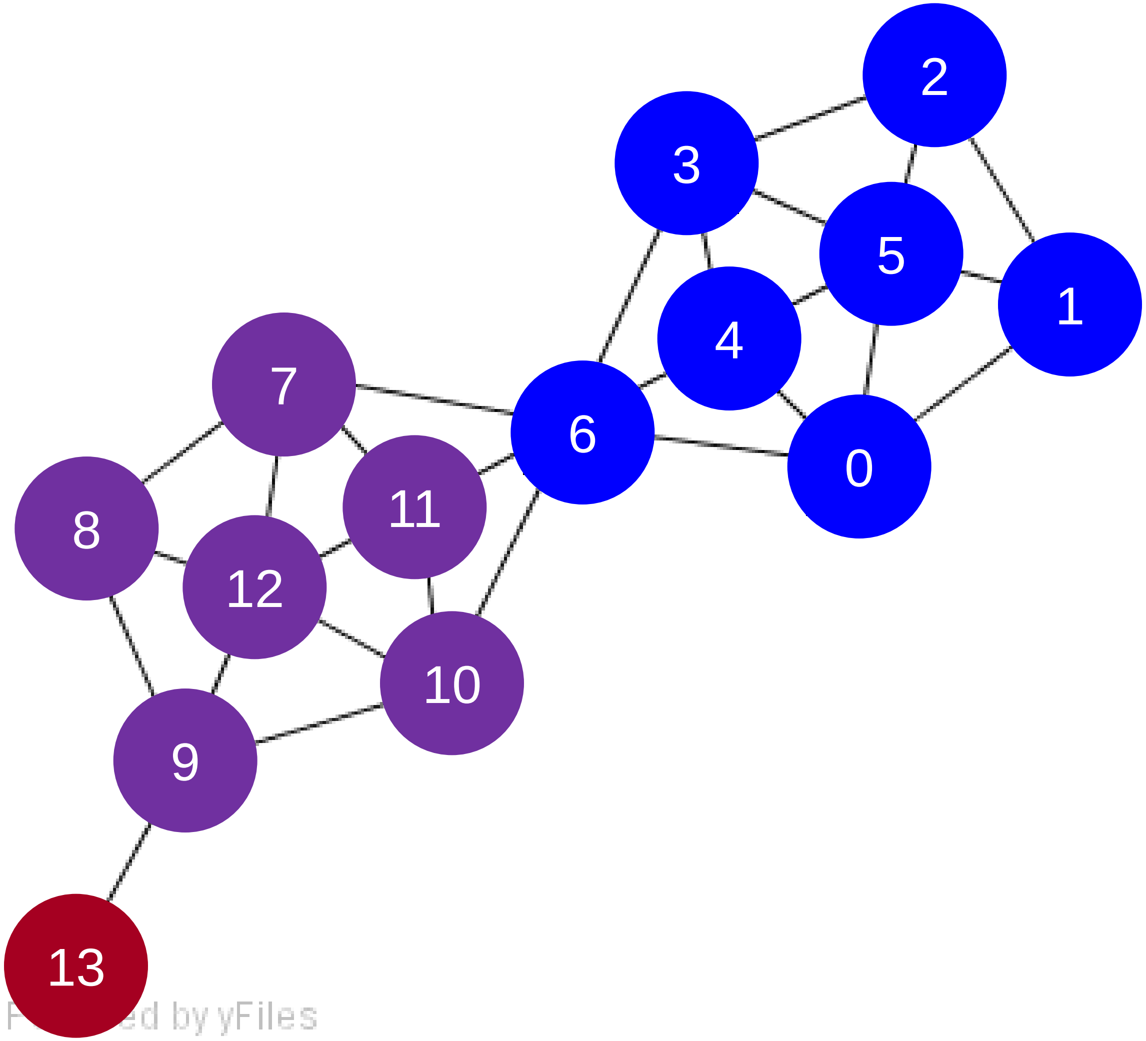
SCAN Algorithm

$\mu = 2$
$\varepsilon = 0.7$



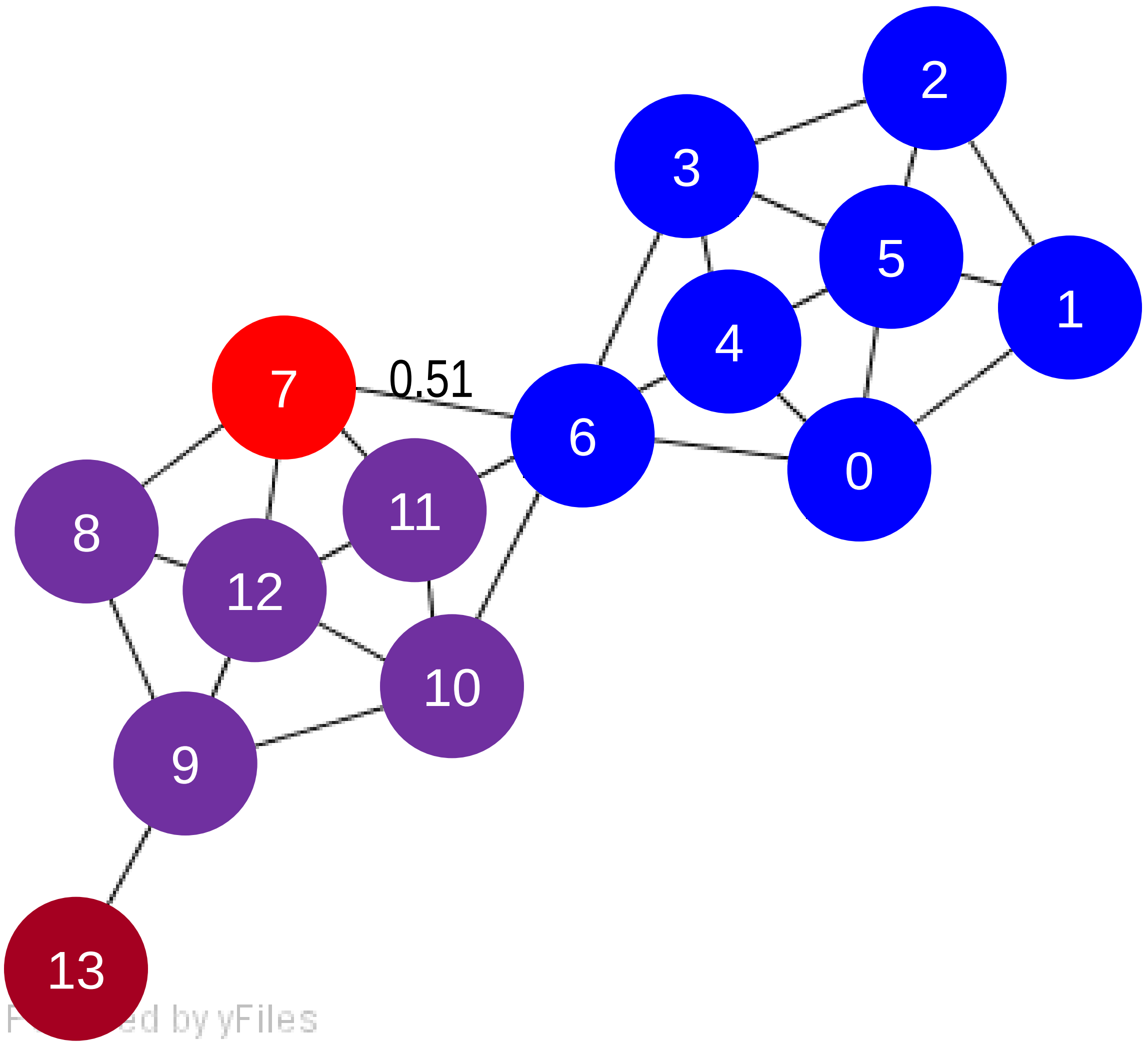
SCAN Algorithm

$\mu = 2$
$\varepsilon = 0.7$



SCAN Algorithm

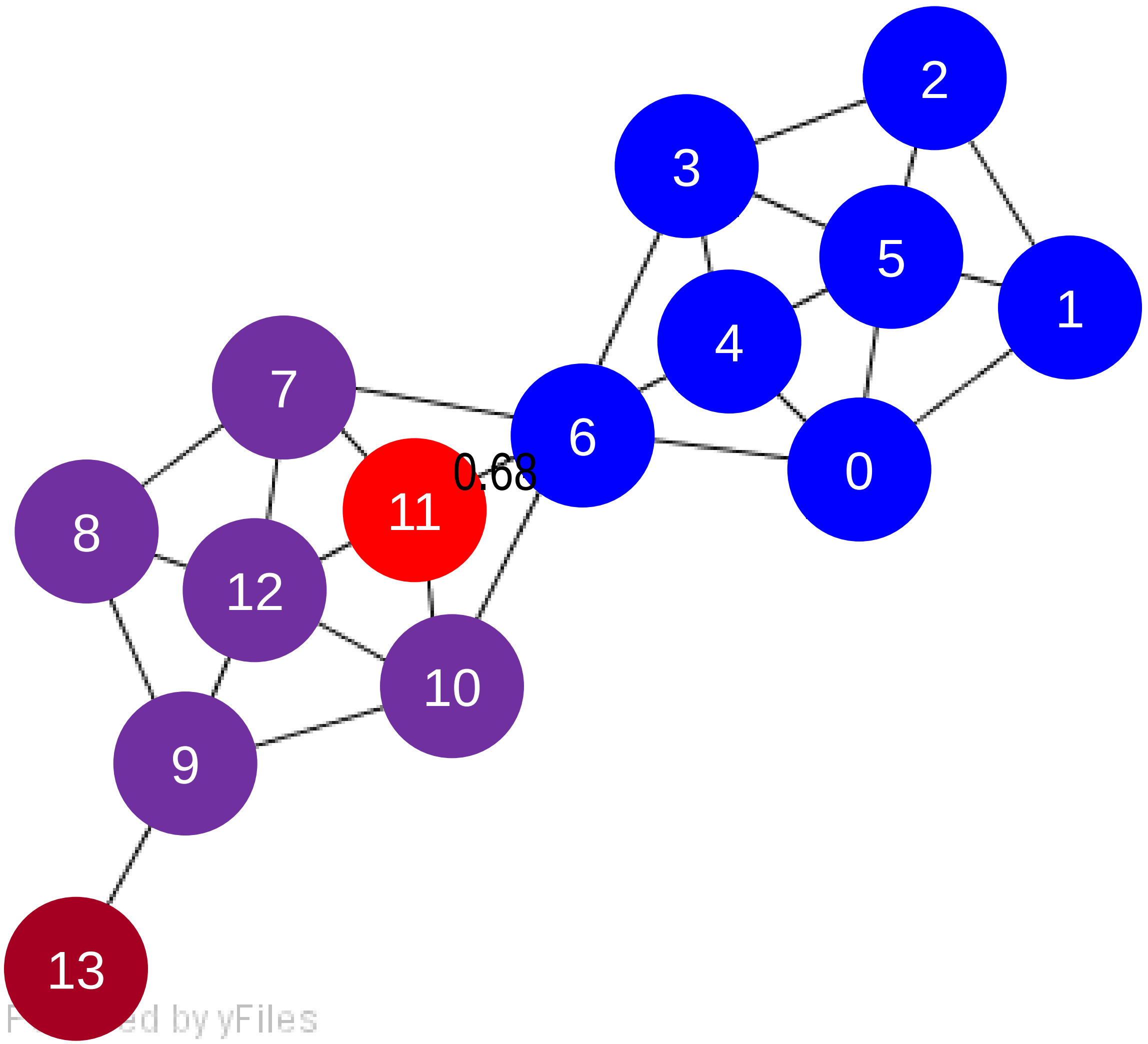
$\mu = 2$
$\varepsilon = 0.7$



SCAN Algorithm

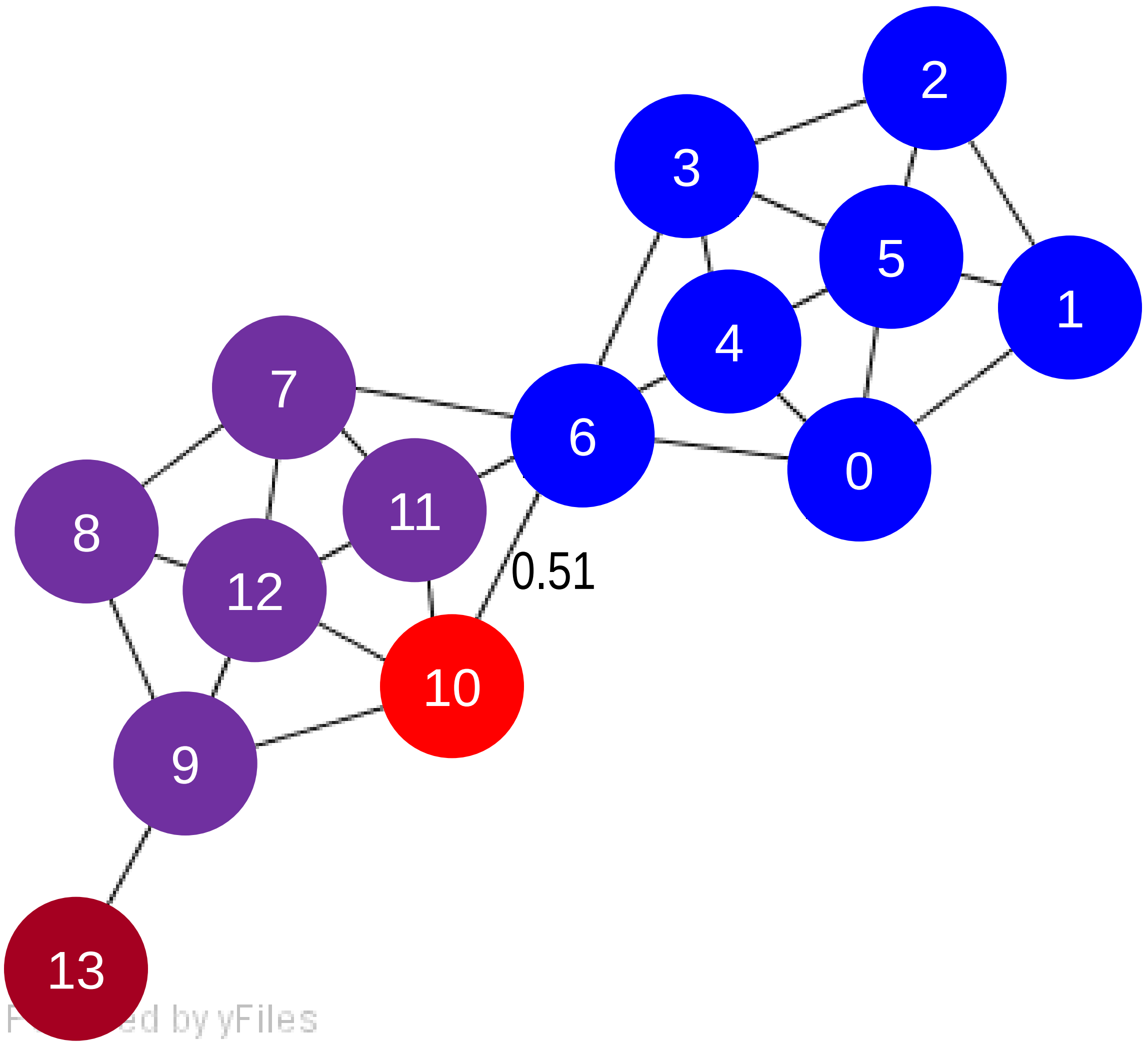
$\mu = 2$

$\varepsilon = 0.7$



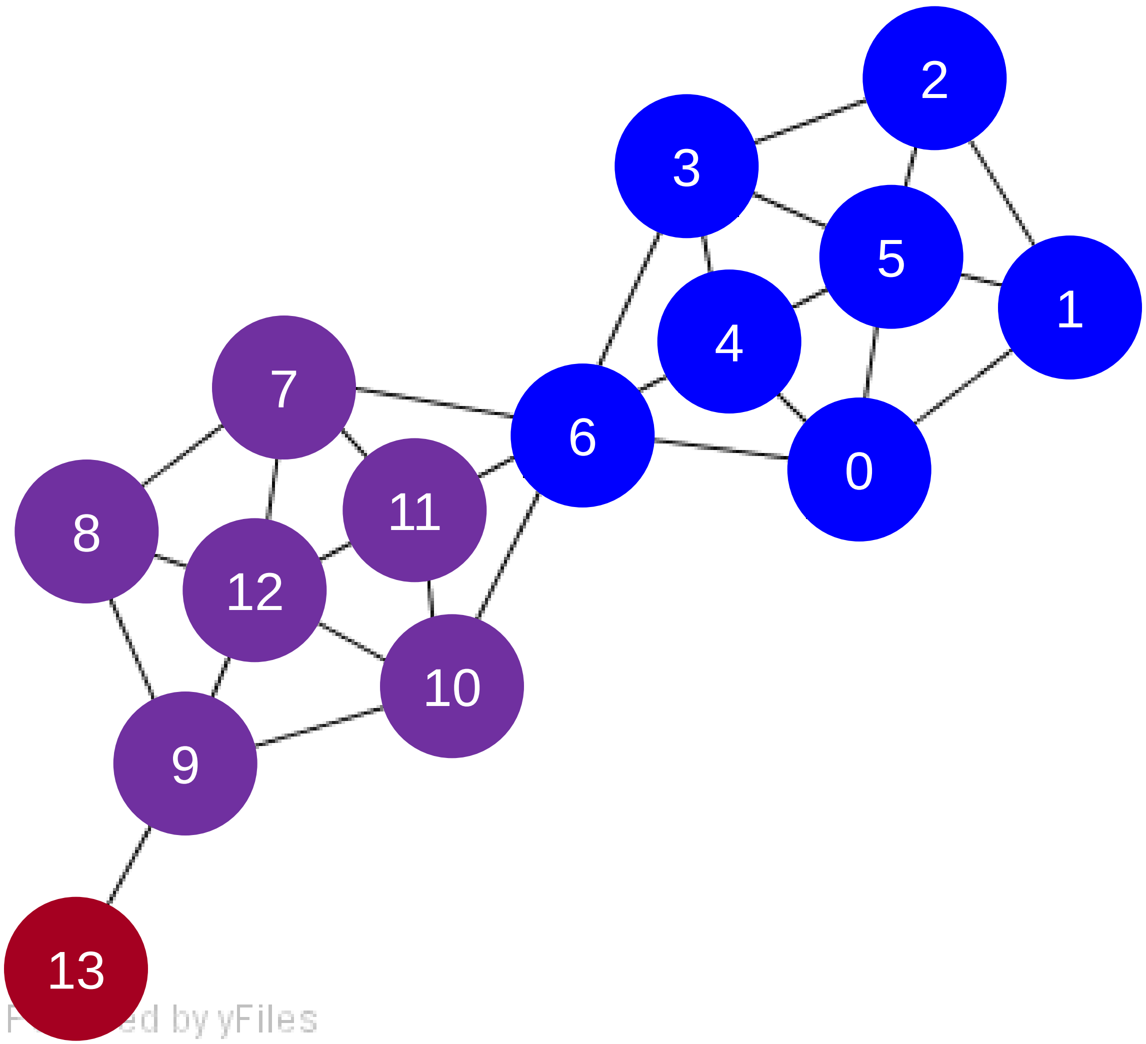
SCAN Algorithm

$\mu = 2$
$\varepsilon = 0.7$



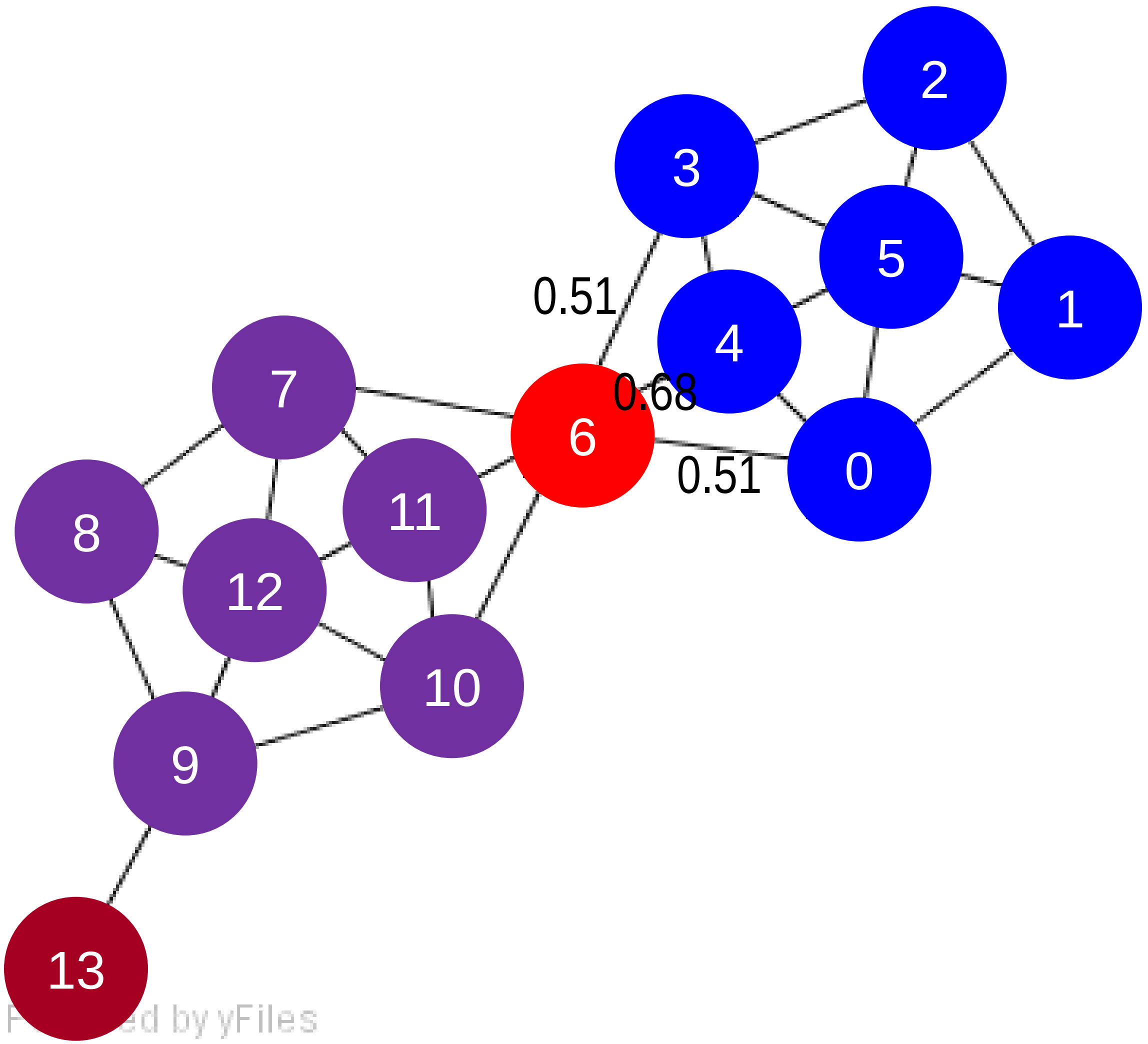
SCAN Algorithm

$\mu = 2$
$\varepsilon = 0.7$



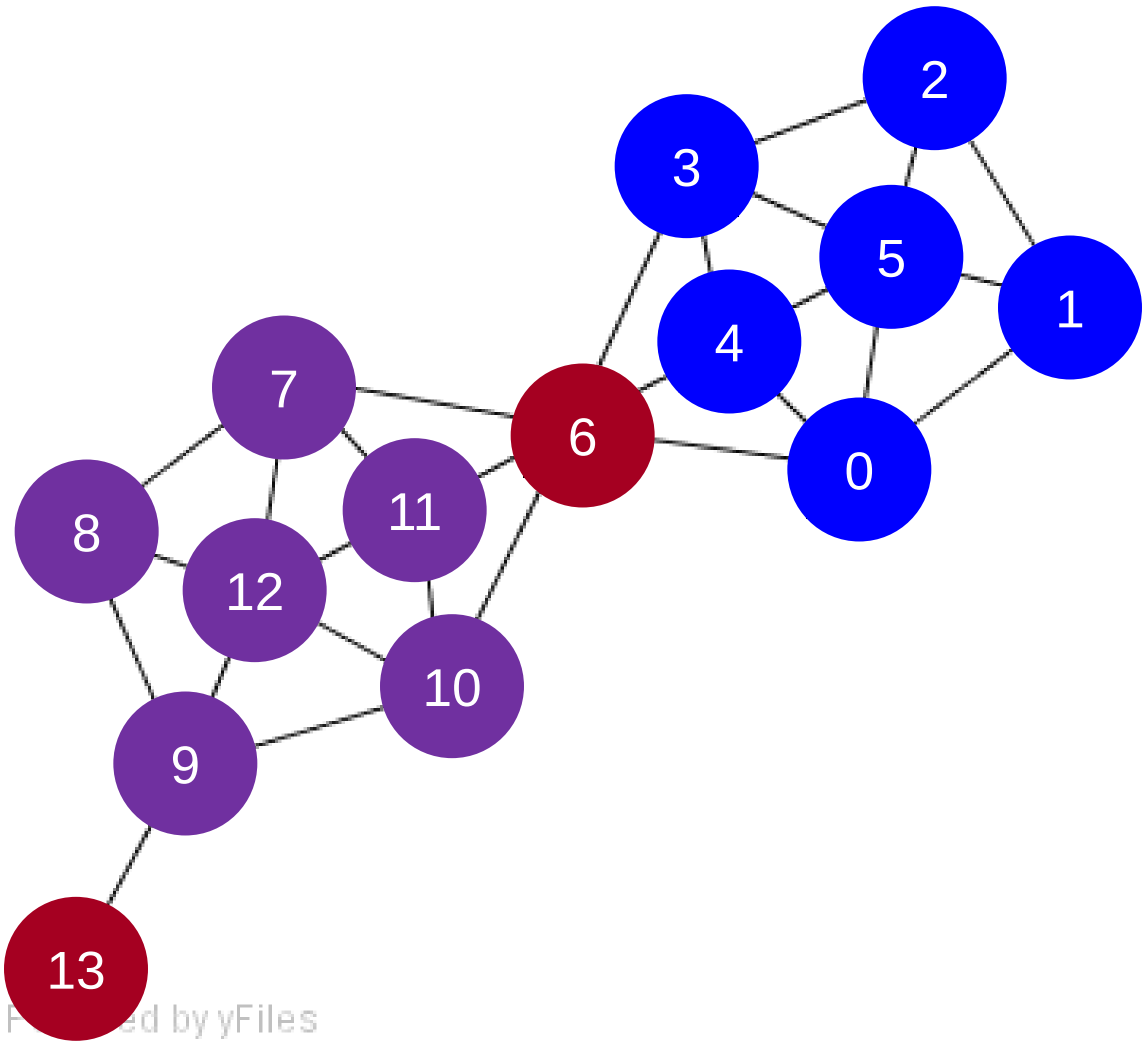
SCAN Algorithm

$\mu = 2$
$\varepsilon = 0.7$



SCAN Algorithm

$\mu = 2$
$\varepsilon = 0.7$



Summary Clustering Graphs

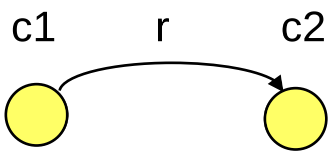
- Directed and undirected graphs
- Distances, affinities and structure similarities
- Clustering
  - Spectral clustering
  - sparsest cut
  - SCAN

Overview

- Case Based Reasoning
- Spectral Clustering
- Clustering Graphs
  - SCAN
- Semantic Networks
- Bayesian Belief Networks
- Hidden Markov Models
- Webgraph: Google

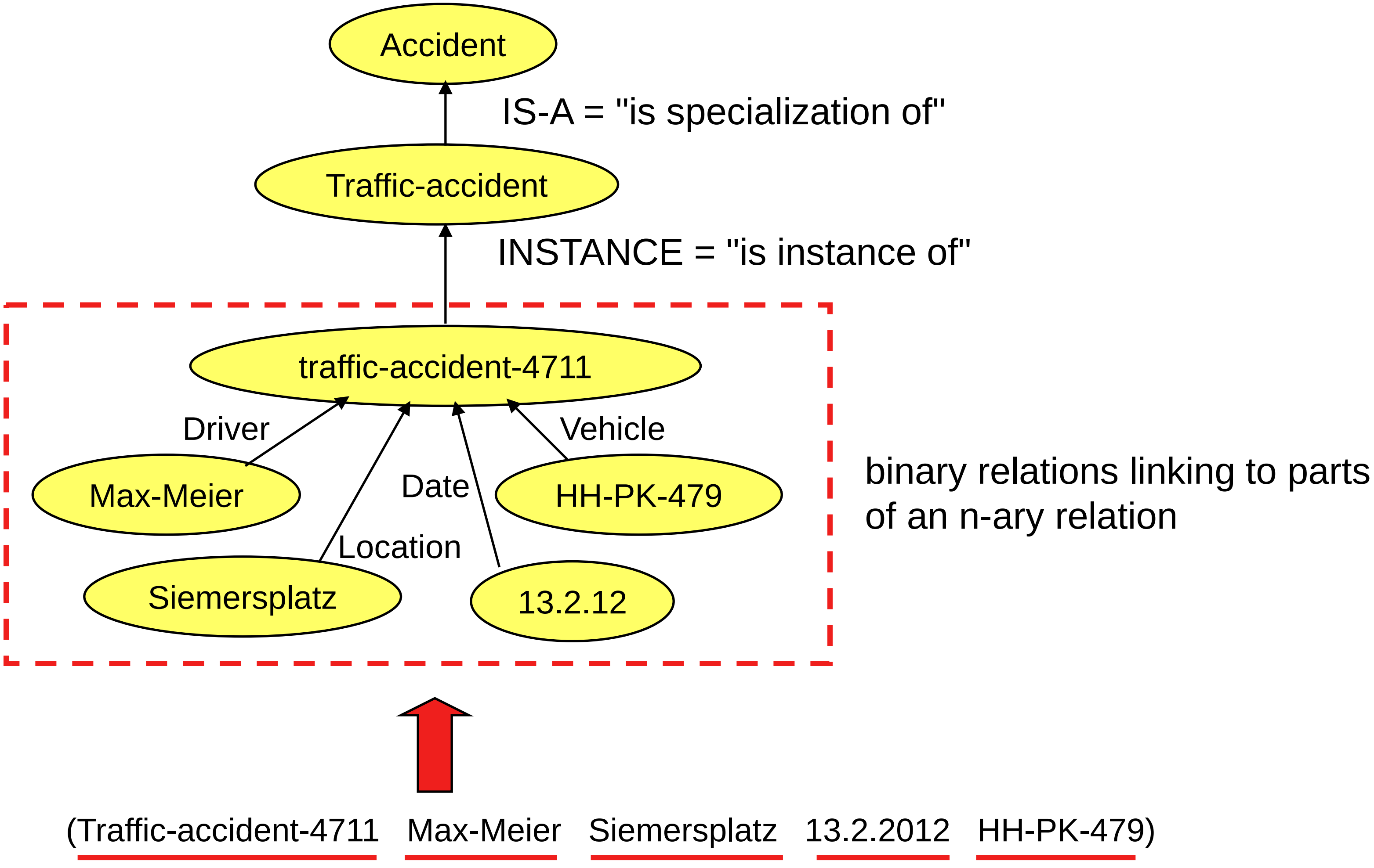
Semantic Networks

- Graphical representation of concepts and relations:
  - labeled nodes (vertices) = concepts
  - directed labeled links (edges) = binary relations



- Where is the semantics?
  - Are there any types of ***nodes*** and types of ***links*** that are valid in general, independent of a particular domain?
  - Is there any ***structuring rule*** which is valid in general, independent of a particular domain?
  - Are there ***generally valid inference procedures*** to derive knowledge which is not explicitly stated?

Basic Relations in Semantic Networks

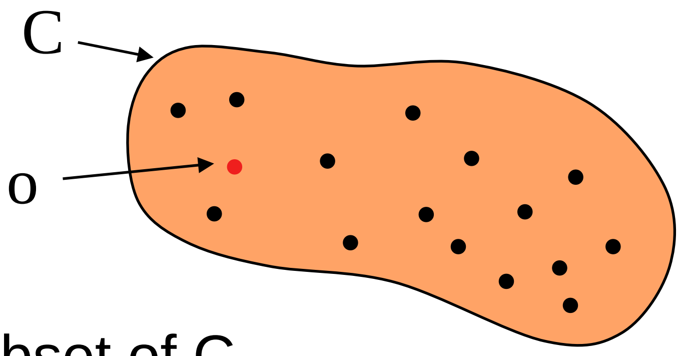


Concepts and Individuals

Nodes of a Semantic Network describe concepts and individuals.

A concept denotes a ***set of objects***.

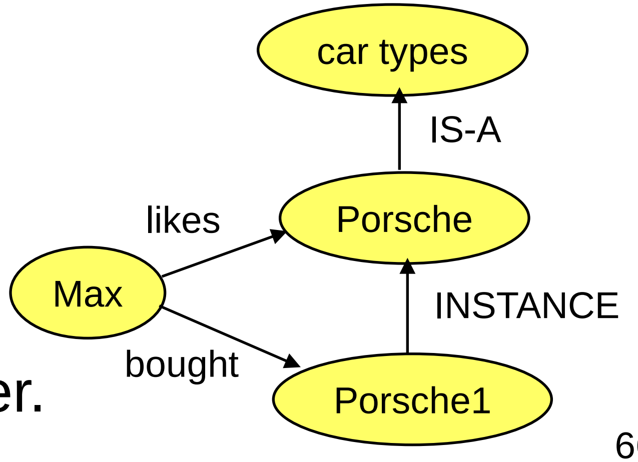
An individual denotes a ***single object***.



C_1 *IS-A* C_2	specifies that C_1 is a subset of C_2
o INSTANCE C	specifies that o is a member of C

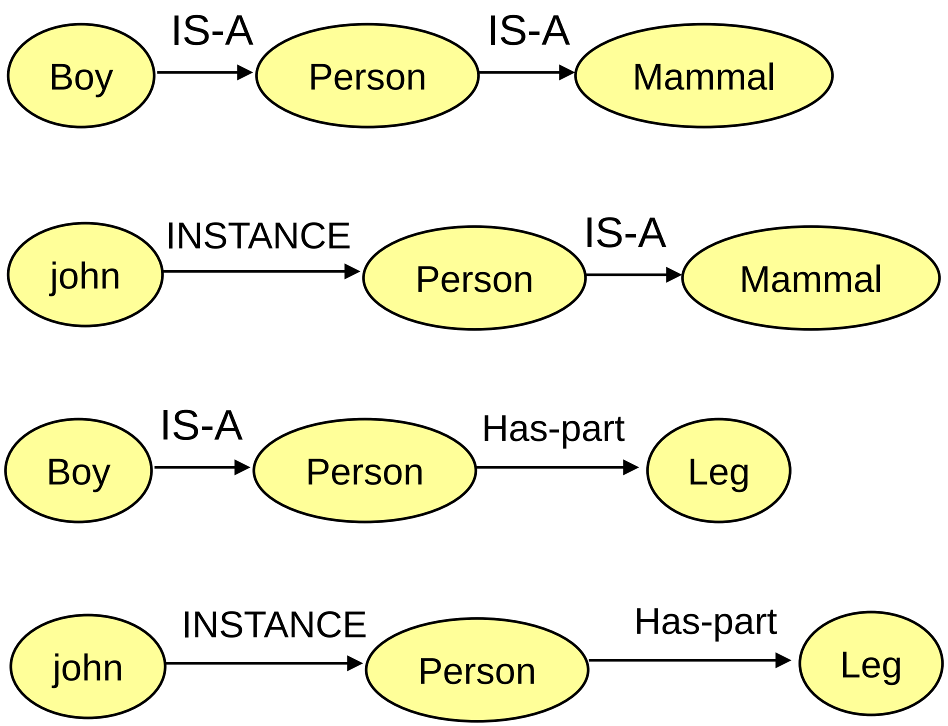
A node may represent both, an individual and a concept. Example:

Max likes a Porsche.
Max bought a Porsche at the car dealer.



Inferences in Semantic Networks

Examples



Rules

C_1 IS-A C_2
C_2 IS-A C_3 $\Rightarrow$ C_1 IS-A C_3

c INSTANCE C_1
C_1 IS-A C_2 $\Rightarrow$ c INSTANCE C_2

C_1 IS-A C_2
C_2 Rel C_3 $\Rightarrow$ C_1 Rel C_3

c INSTANCE C_2
C_2 Rel C_3 $\Rightarrow$ c Rel C_3

Special Semantics for Special Relations

- Special relations may support special inferences.

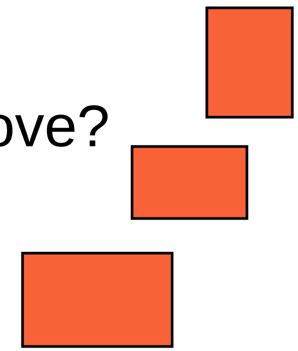
- **Examples**:

  Above(a, b) $\wedge$ Above(b, c) $\Rightarrow$ Above(a, c)

  Left(a, b) $\Rightarrow$ Right(b, a)

  Has-part(a, b) $\wedge$ Has-Part(b, c) $\Rightarrow$ Has-Part(a, c)

Above?



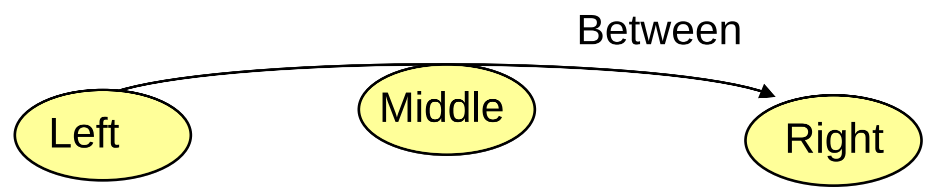
- The rules for inferences may change from domain to domain, hence they must be explicitely stated.

  ⇨ "axiomatizing a domain"

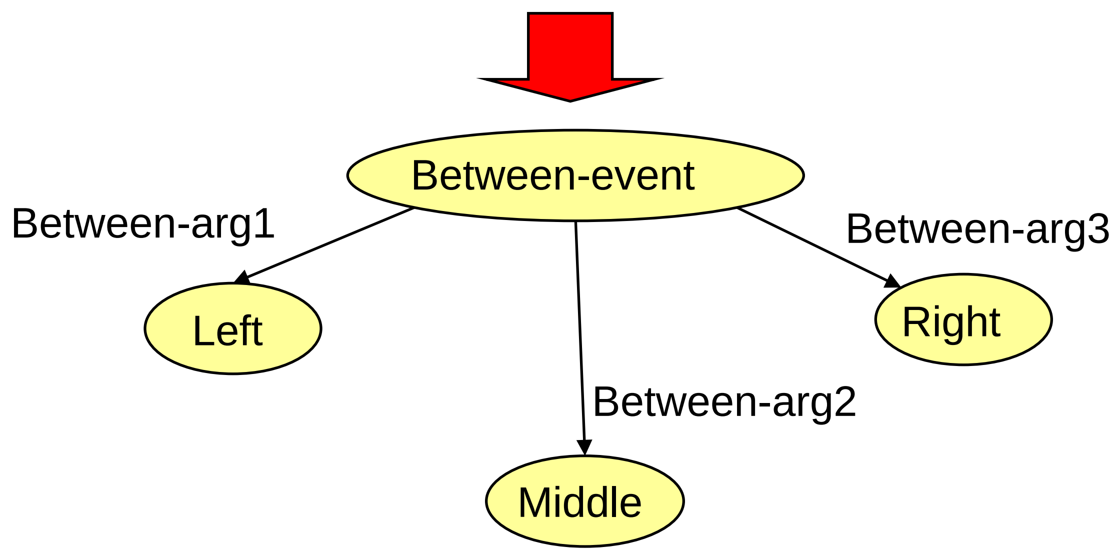
- Spatial reasoning, temporal reasoning are disciplines dealing with axiomatizations of spatial, part-of- and temporal relationships.

N-ary Relations in Semantic Networks

- Semantic Networks allow the representation of binary relations.
- Any N-ary relation can be represented by ***multiple binary relations***
- **Example**:

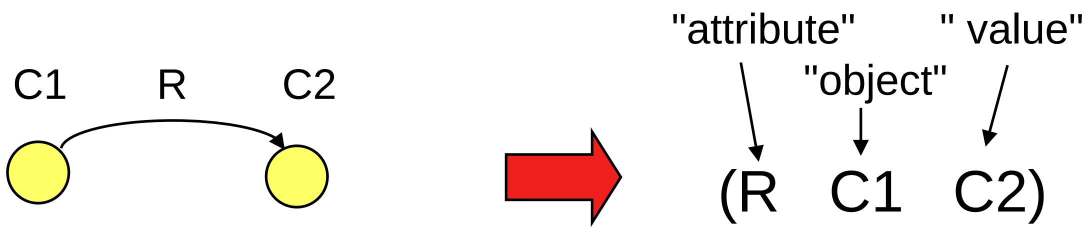


Between(Left, Middle, Right)



Attribute-Object-Value Triplets

- In knowledge representation- and programming languages, a Semantic Network can be represented by a set of triplets:



- The accident example:

  (is-a traffic-accident accident)

  (instance traffic-accident-4711 traffic-accident)

  (driver traffic-accident-4711 Max-Meier)

  (location traffic-accident-4711 Siemersplatz)

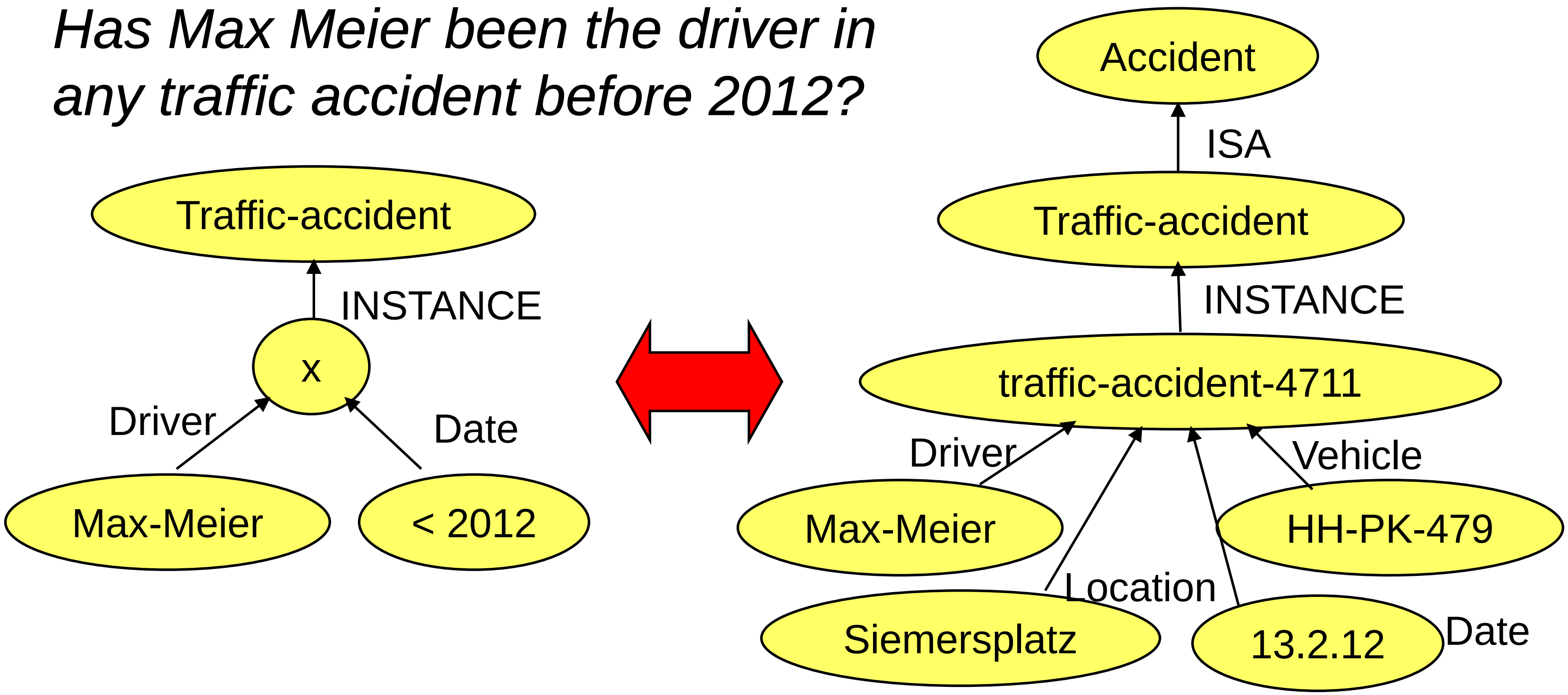
  (date traffic-accident-4711 13.2.12)

  (vehicle traffic-accident-4711 HH-PK-479)

Matching Relational Structures

- Semantic Networks applications often involve matching one network against another
- **Example:**

Has Max Meier been the driver in any traffic accident before 2012?



What services are required?

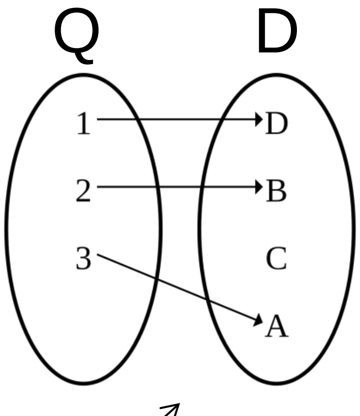
What are the matching rules?

Semantic Network (SN) Queries

- A SN query is a description of desired query responses in terms of a SN using an extended concept language.

- Typical concept language extensions:

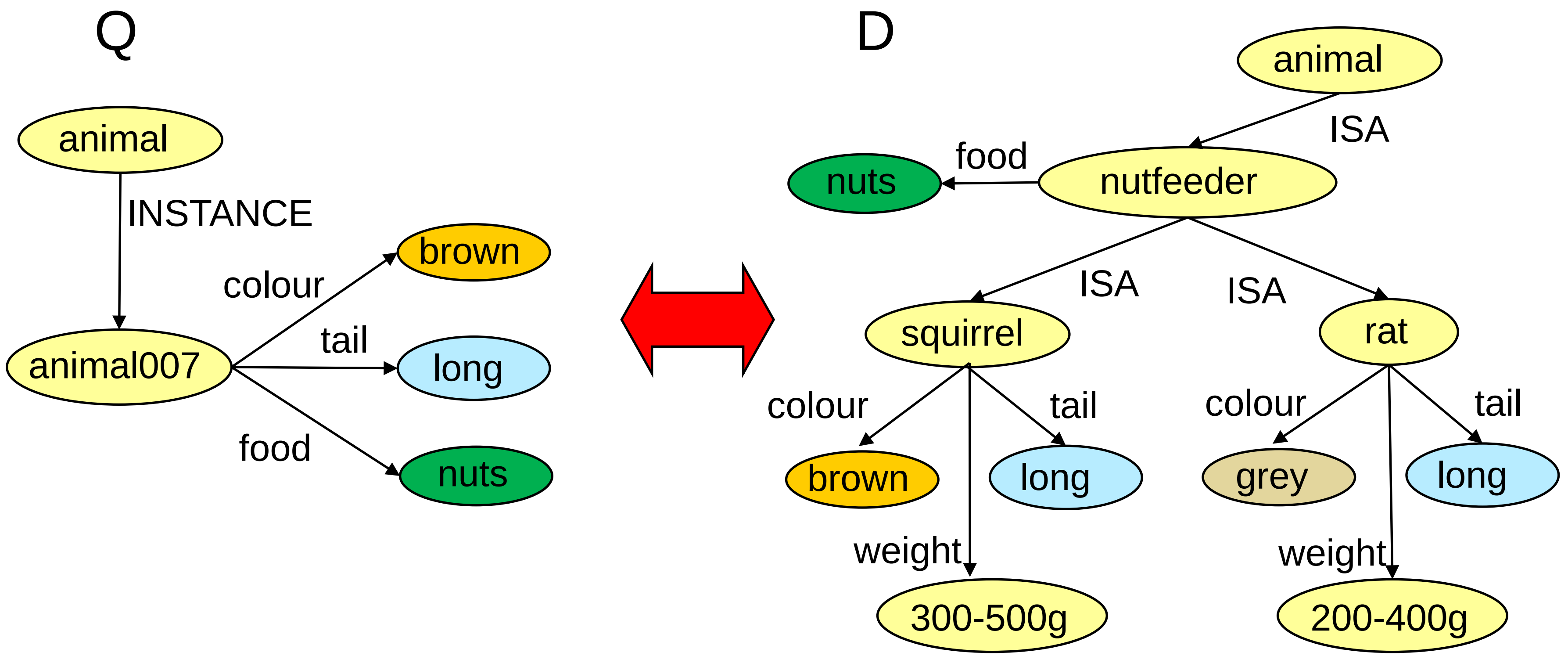
x	individual variable
X	concept variable
{a, b, c}	set of individuals
< 2012	predicate over a concrete domain individual



Matching rules:

A query Q matches a database D, if there is an *injective* mapping of all nodes and links in Q to nodes and links in D such that the corresponding nodes and links are compatible.

Object Classification by Relational Matching



- INSTANCE and ISA inheritance must be exploited for matching
- Class decriptions must be given in terms of sufficient conditions

→ graphs are classified by query matching

Semantic Networks – Summary

- Complex problems can be expressed by graphs
- Intuitive graphical knowledge representation
- Classification and information retrieval done by query matching
- Semantics of relations is well-defined for ISA and INSTANCE, but not clearly defined in general
- Need for domain-specific inference rules (“axiomatizing”)
- Relations between relations cannot be expressed
- Generally, useful services require additional formalisms such as rule-based inferences and new techniques from machine learning and automatic access, tagging, retrieval, pattern matching

Overview

- Case Based Reasoning
- Spectral Clustering
- Clustering Graphs
  - SCAN
- Semantic Networks
- Bayesian Belief Networks
- Hidden Markov Models
- Webgraph: Google

Probabilistic Graphical Models

- Modelling of observations and their relationships
- Until now: semantic networks
- Reasoning over properties inherited from other instance(s)
  - *is-a, has-a* relationships

But how certain are we about the resulting statements?

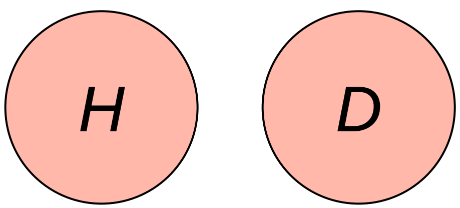
- Make inference under uncertainty
- Stochastics helps us with that
- Probabilistic Graphical Model can be ***directed*** or ***undirected***

Bayes Theorem

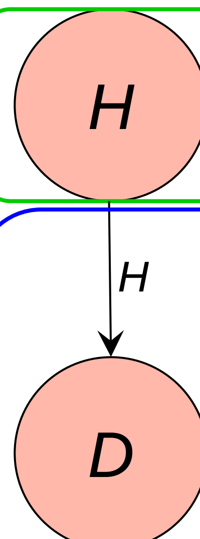
- Two stochastic events, *H* and *D*:

$$\left.\begin{aligned} \boxed{P(H,D)} &= \boxed{P(D\,|\,H)}\cdot\boxed{P(H)} \\ &= P(H\,|\,D)\cdot P(D) \end{aligned}\right\} \quad P(H\,|\,D) = \frac{P(D\,|\,H)\cdot P(H)}{P(D)}$$

joint probability of observing both *H* and *D*



(prior) probability of observing *H*



conditional probability of observing *D* given that *H* is observed

directed graphical model (Bayes net)

Bayesian Belief Network

- also known as ***Bayesian network***, probabilistic network
- Components:

  (1) A *directed acyclic graph* (called a structure)

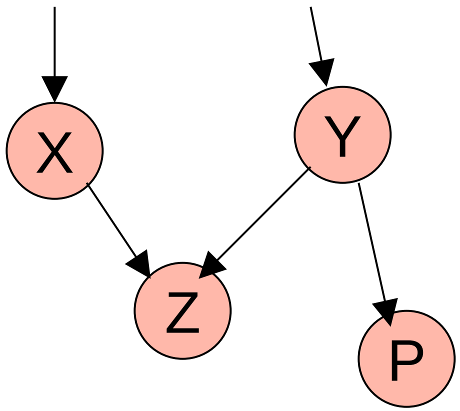
  models *causal influence* relationships

  represents *dependencies* among the variables

  allows *class conditional independencies* between *subsets* of variables

  (2) A set of *conditional probability tables* (CPTs)

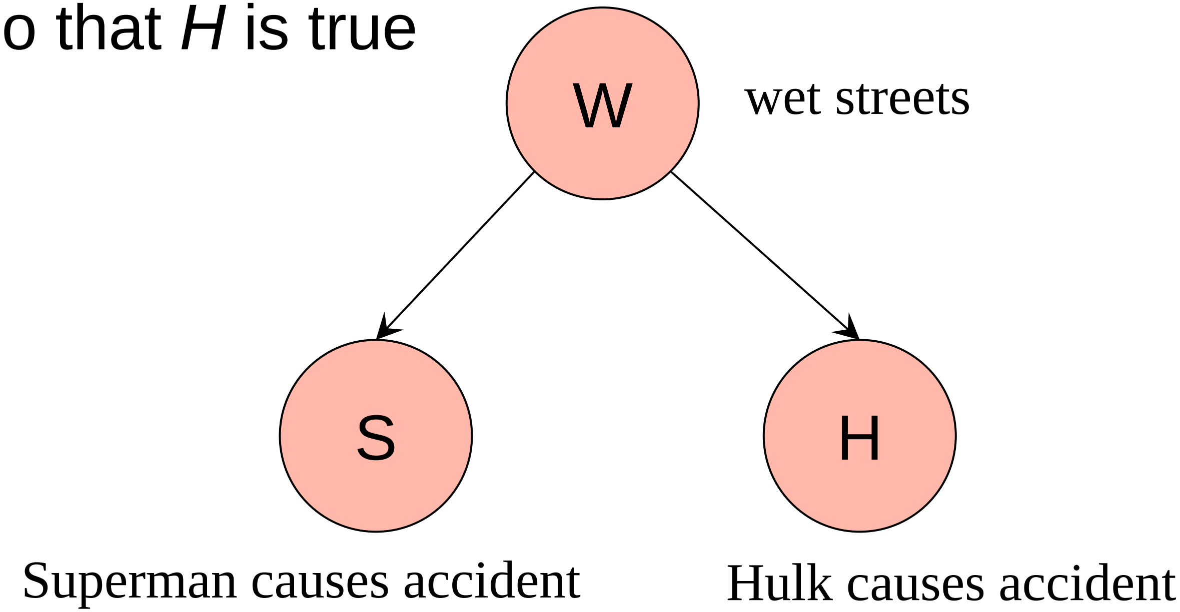
  gives a specification of joint probability distribution



- Nodes: random variables
- Links: dependency
- X and Y are the parents of Z, and Y is the parent of P
- No dependency between Z and P
- Has no loops/cycles

“Conditional Independence”

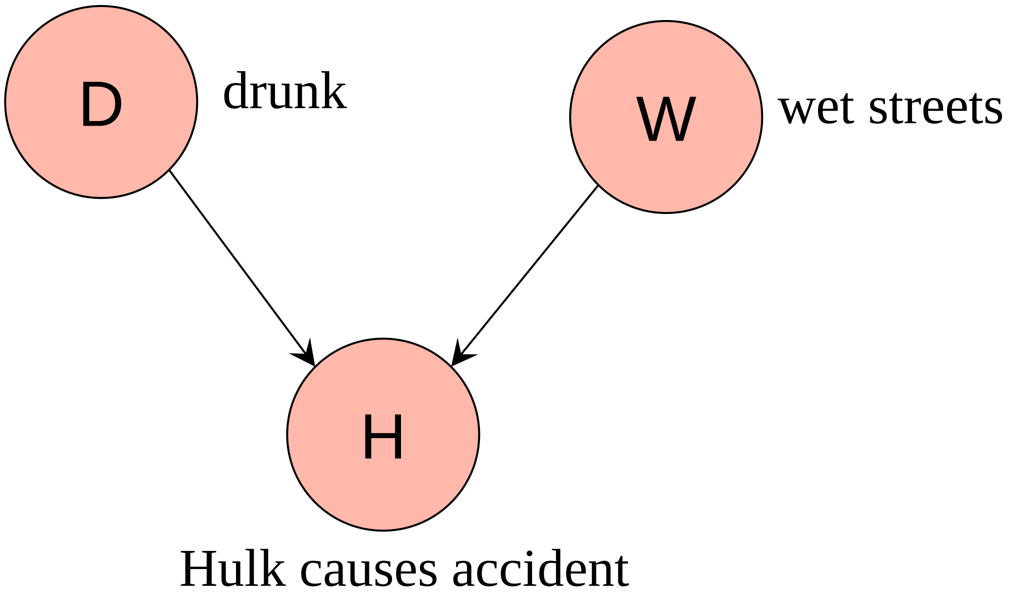
- *S* and *H* are both influenced by *W*
- If *S* is observed to be true, it is more likely that *W* is true, and then also that *H* is true



- If *W* is known (true or false), then *S* and *H* are independent → *S* and *H* are conditionally independent given *W*

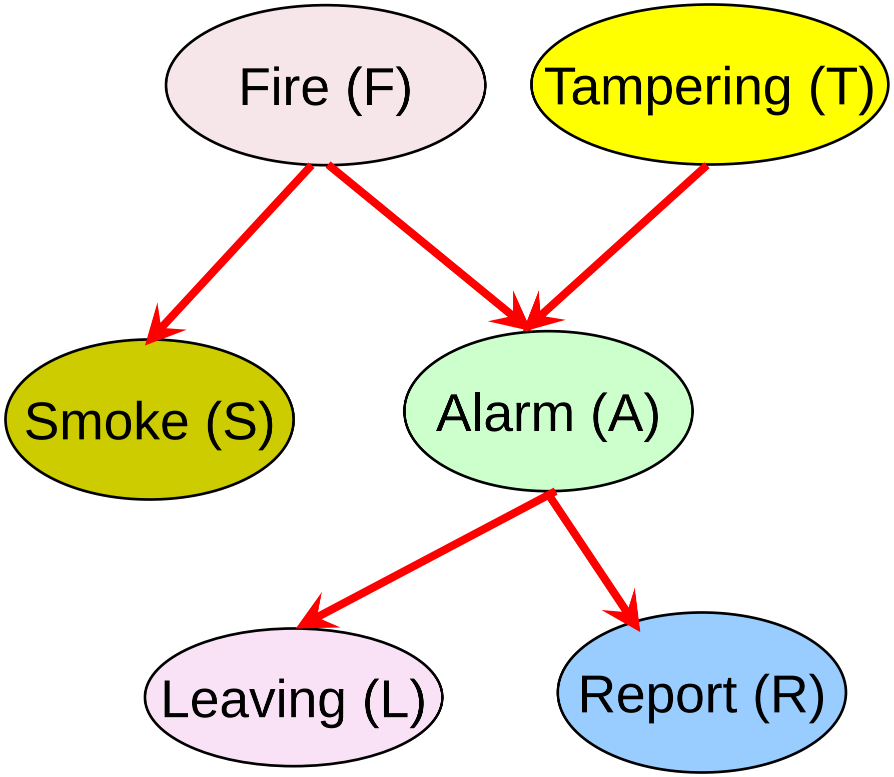
“Explaining away”

- *D* and *W* are independent
- both increase the likelihood of *H* to be true



- If *H* is observed, it is more likely that *D* or/and *W* are true
- If also *D* is observed to be true, then likelihood of *W* being true reduces again → it is “explained away”

A Bayesian Network and Some of Its CPTs



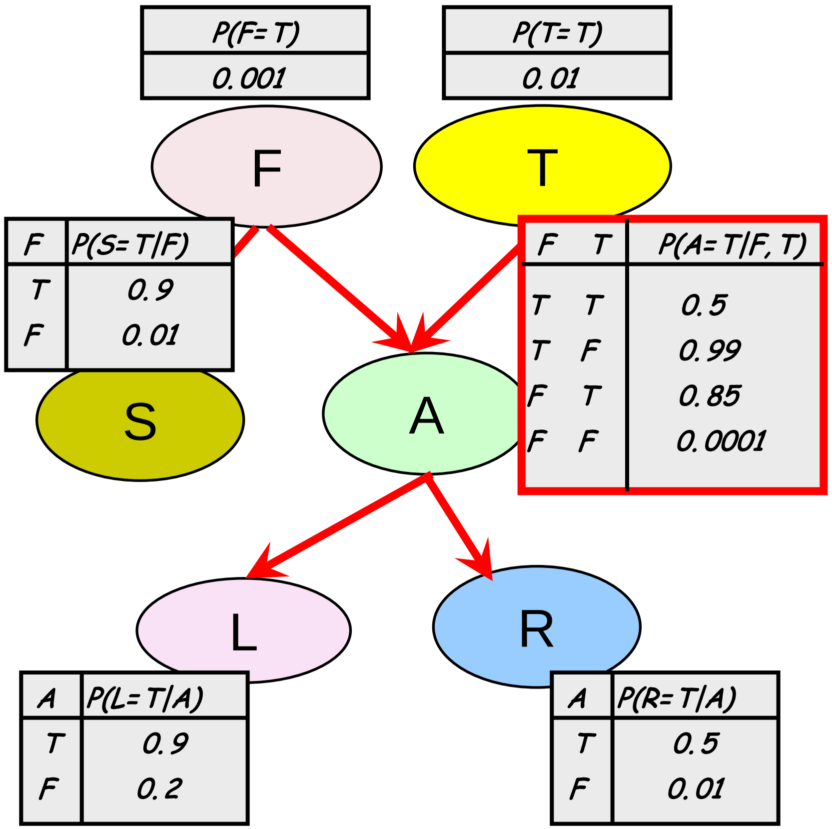
CPT: Conditional Probability Tables

Fire	Smoke	P(S|F)
True	True	.9
False	True	.1

Fire	Tampering	Alarm	P(A|F,T)
True	True	True	.5
True	False	True	.99
False	True	True	.85
False	False	True	.0001

CPT shows the conditional probability for each possible combination of its parents

A Bayesian Network and Some of Its CPTs



has $O(2^6)$ combinations

P(S,F,T,A,L,R)

product rule

= P(T) P(S,F,A,L,R|T)

F and S are independent of T

= P(T) P(S,F$\cancel{|T}$) P(A,L,R|S,F,T)

product rule

= P(T) P(F) P(S|F) P(A,L,R|F,T)

L and R are conditionally independent of F and T given A

= P(T) P(F) P(S|F) P(A|F,T) P(L,R|A$\cancel{,F,T}$)

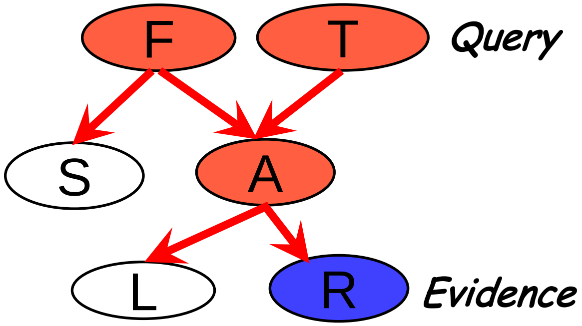
L and R are independent given A

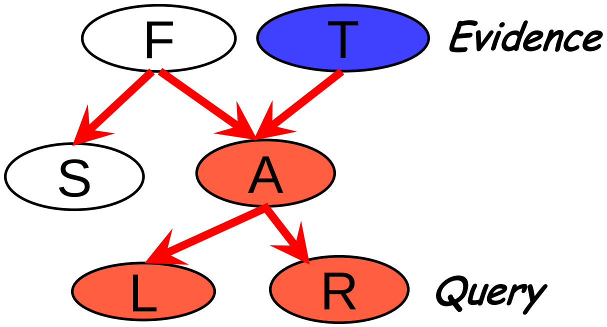
= P(T) P(F) P(S|F) P(A|F,T) P(L|A) P(R|A)

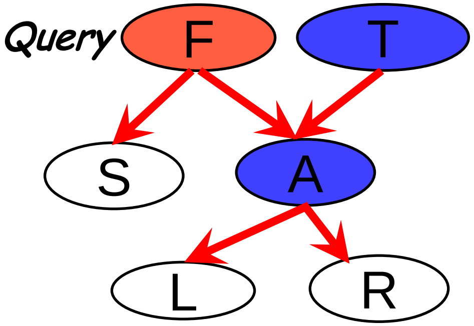
has $O(2^2)$ combinations only

Types of Reasoning in Bayesian Networks

- Diagnostic
  - From symptoms to causes, e.g., doctor infers diseases from symptoms. Reasoning occurs in opposite direction to network arcs.
- Predictive
  - Reasoning from new information about causes to new beliefs about effects, follows the directions of the networks arcs.
- Inter-causal (explaining away)
  - Mutual causes of a common effect. Initially, causes may be independent. But if a common effect is observed and we learn that one cause is true, then the other is less likely – it has been "explained away".
- Combined
  - Simultaneous use of diagnostic and predictive reasoning







How Are Bayesian Networks Constructed?

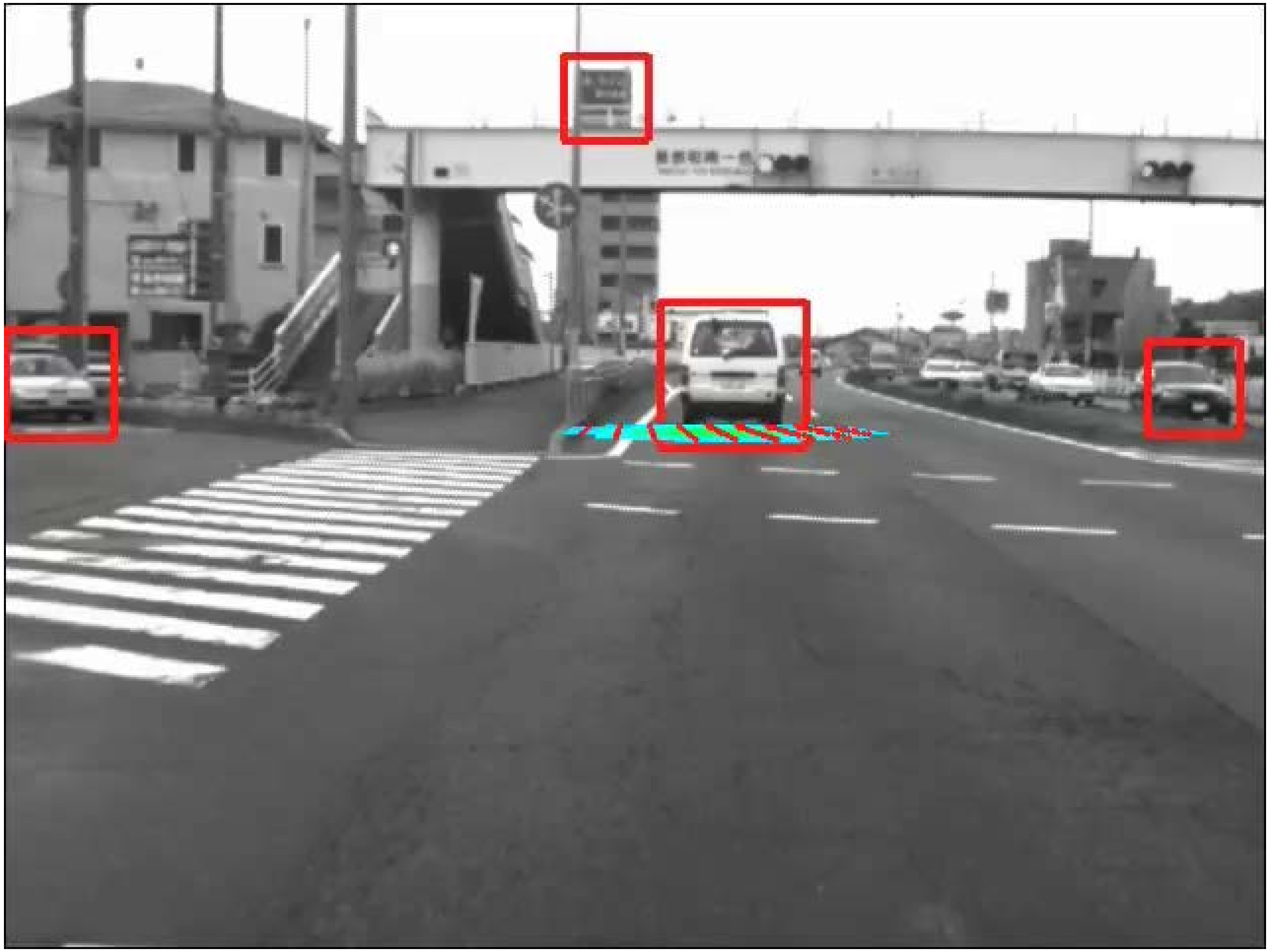
- ***Subjective construction***: Identification of (direct) causal structure
  - People are quite good at identifying direct causes from a given set of variables & whether the set contains all relevant direct causes
  - *Markovian assumption*: Each variable becomes independent of its non-effects once its direct causes are known
  - E.g., S ← F → A ← T, path S → A is blocked once we know F → A
  - HMM (Hidden Markov Model): often used to model dynamic systems whose states are not observable, yet their outputs are
- ***Synthesis*** from other specifications
  - E.g., from a formal system design: block diagrams & info flow
- ***Learning from data***
  - E.g., from medical records or student admission record
  - Learn parameters given its structure or learn both structure and params
  - Maximum likelihood principle: favors Bayesian networks that maximize the probability of observing the given data set

Training Bayesian Networks

- Scenario 1: Given the network structure and all variables observable:
  → *compute only the CPT entries*
- Scenario 2: Network structure known, some variables hidden:
  → *gradient descent* (greedy hill-climbing) method, i.e., search for a solution along the steepest descent of a criterion function
  - Weights are initialized to random probability values
  - At each iteration, it moves towards what appears to be the best solution at the moment
  - Weights are updated at each iteration & converge to local optimum
- Scenario 3: Network structure unknown, all variables observable:
  → search through the model space to *reconstruct network topology*
- Scenario 4: Unknown structure, all hidden variables:
  → no good algorithms known for this purpose

[D. Heckerman. A Tutorial on Learning with Bayesian Networks. In *Learning in Graphical Models,* M. Jordan, ed. MIT Press, 1999]

Bayesian Cars



Coue et al., 2005. Bayesian Occupancy Filtering for Multi-Target Tracking: an Automotive Application.

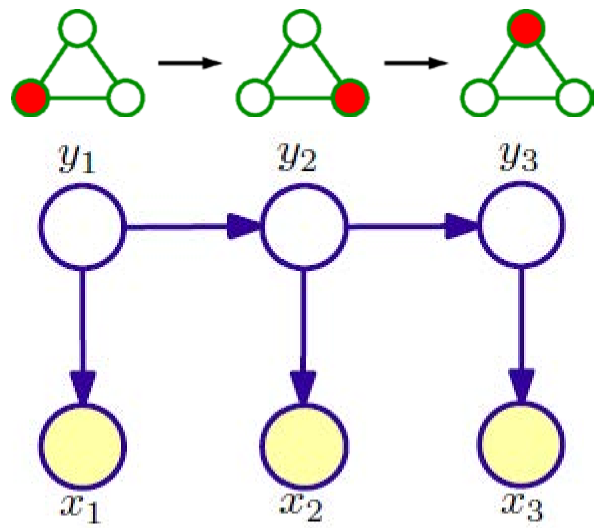
Overview

- Case Based Reasoning
- Spectral Clustering
- Clustering Graphs
  - SCAN
- Semantic Networks
- Bayesian Belief Networks
- Hidden Markov Models
- Webgraph: Google

Hidden Markov Models

- Sequential data modelling, i.e. time series
- Applications:
  - Gene prediction
  - Weather forcecasting
  - Automatic speech recognition
  - Gesture Recognition
  - EEG activity for sleep monitoring
- HMM packages in Matlab (free: Kevin Murphy‘s), R, Java

Hidden Markov Models



- Model λ:(A, B, π)
  - A: State-transition matrix
  - B: Symbol-emission matrix
  - π: initial state probability vector
- Only emissions are observable, but unknown which state produced them (so: states are ***hidden***)
- State sequence can only be inferred from observed events
- Again: Markov assumption
- State structure typically drawn "unrolled" in time

Weather example

- Given the current weather condition, predict the next possible state (Markov assumption)

		Time *t* + 1			
	State	Sunny	Cloudy	Rainy	Snowing
Time *t*	Sunny	0.6	0.3	0.1	0.0
	Cloudy	0.2	0.4	0.3	0.1
	Rainy	0.1	0.2	0.5	0.2
	Snowing	0.0	0.3	0.2	0.5

Weather Example: Emissions

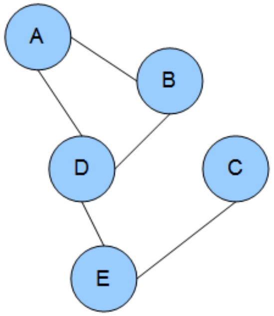
- Classification of weather conditions into {good, bad, variable} is only observable

		Weather			
		Sunny	**Cloudy**	**Rainy**	**Snowy**
Met condition	**Good**	0.5	0.3	0.2	0.0
	Variable	0.2	0.3	0.3	0.2
	Bad	0.1	0.2	0.3	0.4

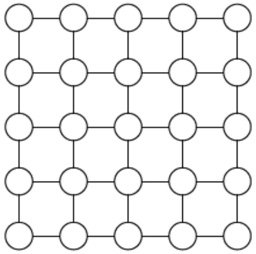
Hidden Markov Models - Problems

- ***Decoding***: Given the HMM and the observation sequence, what is the most probable state sequence?
  - Viterbi algorithm
- ***Evaluation***: Given the HMM and the observation sequence, how probable is it that this HMM generated it?
  - Forward-backward algorithm
- ***Learning***: Find the HMM parameters which best fit to the observation sequence (training data)
  - Baum-Welch algorithm

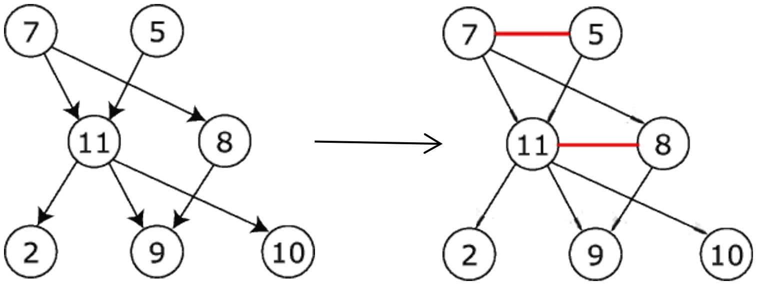
Undirected Graphical Models



- Markov Random Fields or Markov Network
- Originated from statistical physics
- Models correlations, not necessarily causality
  - Image segmentation: pixel correlation

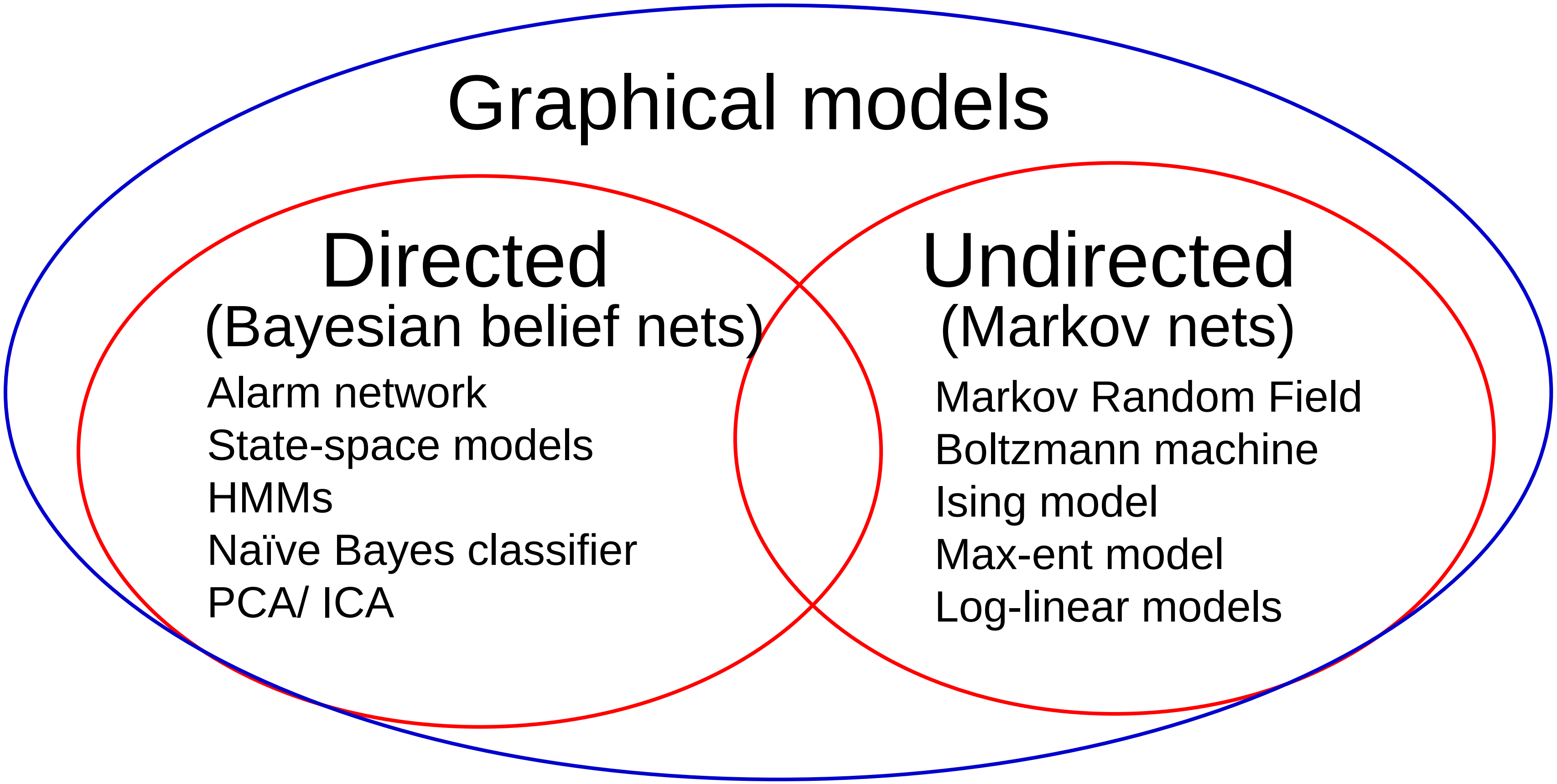


- Definition of ***potentials*** of nodes and edges
- Moralization: every directed model can be transformed into an undirected model



- Inference algorithms as for HMM still applicable

Probabilistic Graphical Models



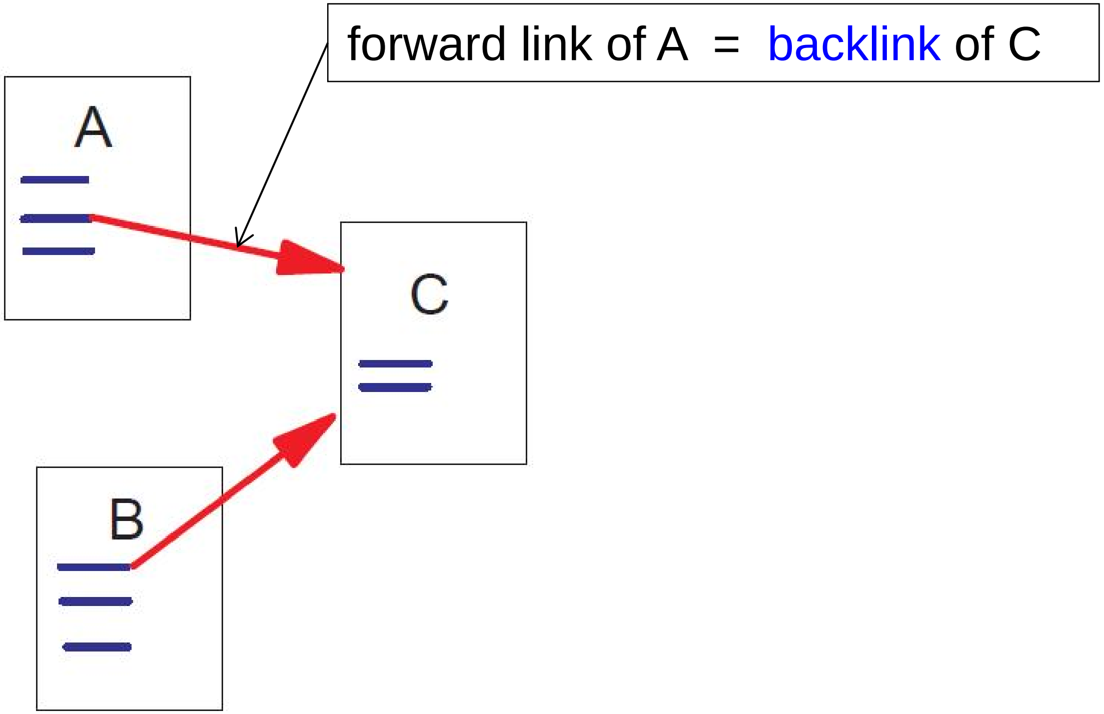
Overview

- Case Based Reasoning
- Spectral Clustering
- Clustering Graphs
  - SCAN
- Semantic Networks
- Bayesian Belief Networks
- Hidden Markov Models
- Webgraph: Google

Applications: Google (1)

Webgraph: web page = vertex, weblink = edge

A web page is important if many pages refer to it *(vote)*



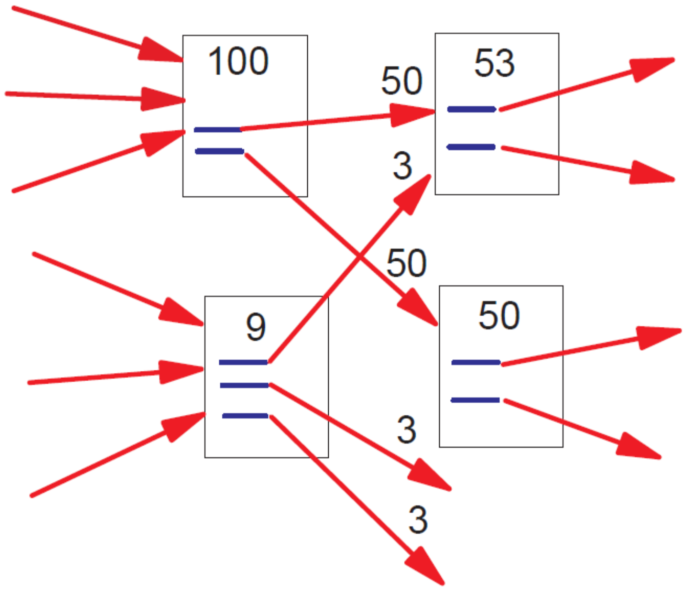
Brin, Page, et al. (1998) The PageRank Citation Ranking: Bringing Order to the Web. Tech Rep Stanford Uni

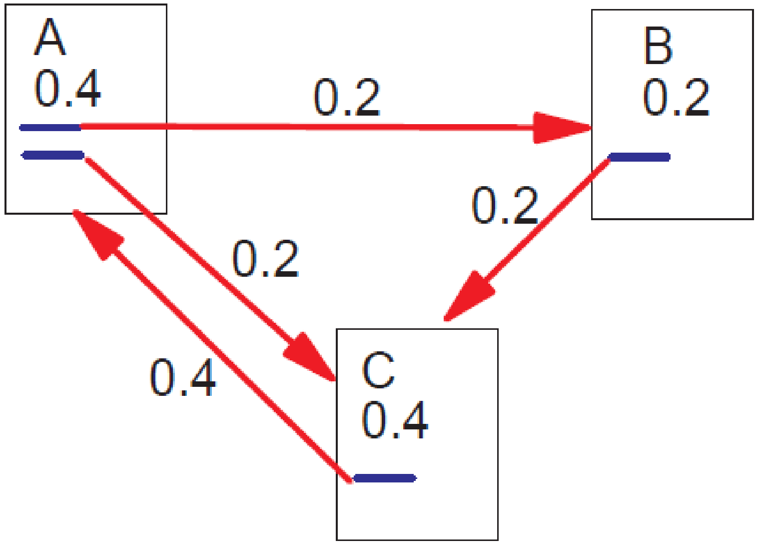
Google (2)

Ranking Function for web page *u*:

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v}$$

v: web page that links to u
B_u: backlinks
$N_v = |F_v|$: # forward links from v
c: normalization factor

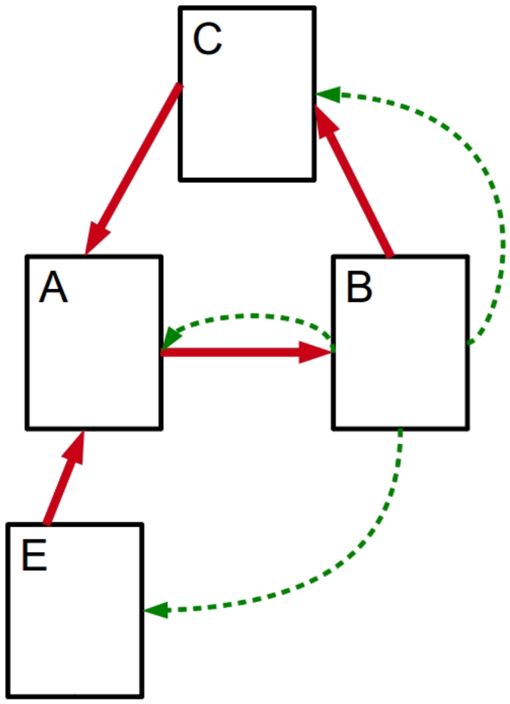




PageRanks form a probability distribution over web pages, so the sum of all web pages' PageRanks will be one

Google (3)

Problem: ***Rank Sink***
Some pages form a loop that accumulates rank to the infinity.

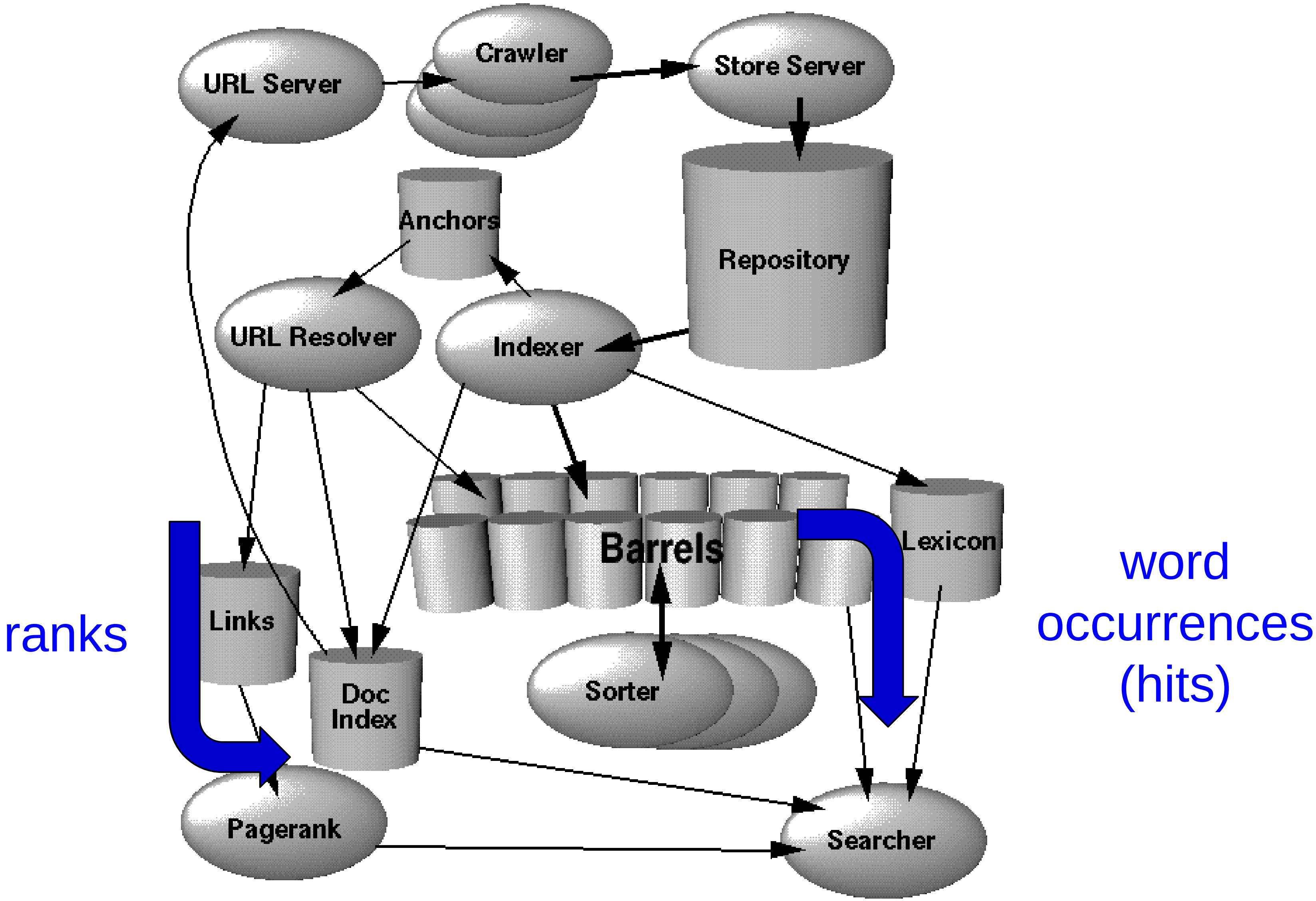


Solution: ***Random Surfer***
Jump to a random page based on some distribution *E*

$$R'(u) = c \sum_{v \in B_u} \frac{R'(v)}{N_v} + cE(u)$$

rank source

Google (4)



Brin, Page (1998) The anatomy of a large-scale hypertextual Web search engine. Comp Netw and ISDN Syst

Summary

- Case Based Reasoning – associating cases from the past
- Clustering graphs
  - spectral clustering
  - sparsest cut
  - SCAN
- Semantic Networks – graphical models for knowledge representation and classification
- Probabilistic reasoning – Bayesian Belief Networks
- Time series – Hidden Markov Models
- Google – it`s algorithm is transparent