Social Network Visualization: Graph Drawing Basics

Lecture will mainly follow [1], [4], [5], so citations to these sources are sometimes omitted in order to enhance the readability

Social Network Visualization: Graph Drawing Basics

- Ultimate goal of (empirical) Analysis & (mathematical) Modeling of Social Networks in Computer Science: Constructing useful services for users in Social Networking

- In Social Computing: Services are ultimately Communication Services: Information is transferred between individua

- More fine grained subdivision: Awareness Services, Communication Services (per se), Information Services; (subdivision is only coarse: every communication act transfers information and requires awareness about state of receiver; every use of an information service is ultimately a communication act)

Social Network Visualization: Graph Drawing Basics

- Awareness-Services: Inform Users about the activities or general state of other users

- Communication-Service (per se): Establish channels for all kinds of communication acts: (synchronous vs. asynchronous, direct vs. indirect, 1:1 vs n:m, textual, audio-visual, symbolical etc.)

- Information Service: Allow users to manage information spaces and other users to access those information spaces

Social Network Visualization: Graph Drawing Basics

- Most basic Awareness-Service in Social Networking:
Visualize the social network

- Purposes of social network visualizations:
  - Get an overview about a particular social network
  - Perceive structures (clusters etc.) in the SN via human visual system
  - Show dynamic changes in a network

Social Network Visualization: Graph Drawing Basics

- We model SN as graphs → Investigate discipline of graph drawing

Part 5: Theoretical Background;
Drawing of static graphs

Part 5 B: Drawing of dynamic graphs;
SN visualization applications

Social Network Visualization: Graph Drawing Basics [1]

- Given: Graph G=(V,E); Definition: 2-dim. Graph drawing (abstract): Mapping $D:(V,E) \rightarrow (\mathbb{R}^2, C)$ where each node v_i is mapped to a point in $\mathbb{R}^2$ and each edge e is mapped to a curve c over $\mathbb{R}^2$ connecting the two incident nodes. The space of such curves is denoted as C

- Analogous: 3-dim. Graph drawing; Here we need one or more additional projection functions (possibly parametrizable (→ interactive viewing)) to project the $\mathbb{R}^3$ on the drawing canvas ($\subset \mathbb{R}^2$)

- Abstract goal for information visualization (informal): Display set of data with complex internal structure and relations so that a human can visually perceive structures and relations that would be harder to perceive with alternative data-display techniques

Social Network Visualization: Graph Drawing Basics [1]

- Given: Graph G=(V,E); Definition: 2-dim. Graph drawing (abstract): Mapping $D:(V,E) \rightarrow (\mathbb{R}^2, C)$ where each node v_i is mapped to a point in $\mathbb{R}^2$ and each edge e is mapped to a curve c over $\mathbb{R}^2$ connecting the two incident nodes. The space of such curves is denoted as C

- Formally: A curve c is a mapping of a real interval to a topological space X. $c: I \subseteq \mathbb{R} \rightarrow X$. In our case X is the real plane $\mathbb{R}^2$
- Curves that we have in mind: non-self intersecting ← "open Jordan curve"
- Topology: (closed) Jordan curve: Any curve which is homeomorphic to S^1 ; open Jordan curve: Not closed
- Homeomorphism: "Topological isomorphism"

Criteria for good graph drawings [1]

- Clearly: “Good” drawing criteria for subway map graph different from “good” drawing criteria for genealogy map graph
- Clearly: Various different aesthetics / taste criteria / viewing customs exist

Is it possible at all to formulate criteria for “good” graph drawing?

- Answer is yes: There are certain criteria for “good” drawings of graphs that can be assumed to apply to a vast majority of applications (distinguish technical and ‘aesthetic’ (information viz) criteria)

Fig. 1.1. Subway map, New York City (MTA, 1999).

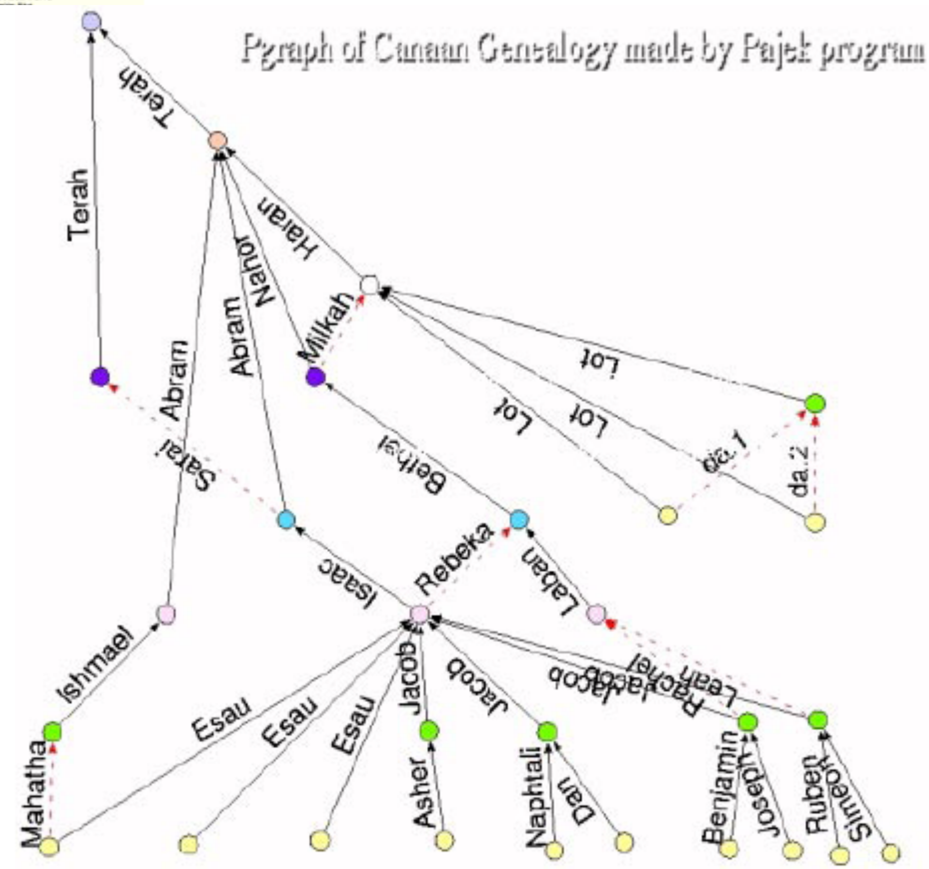


Fig. 1.2. Canaan Genealogy. [1]

Speech Recognition

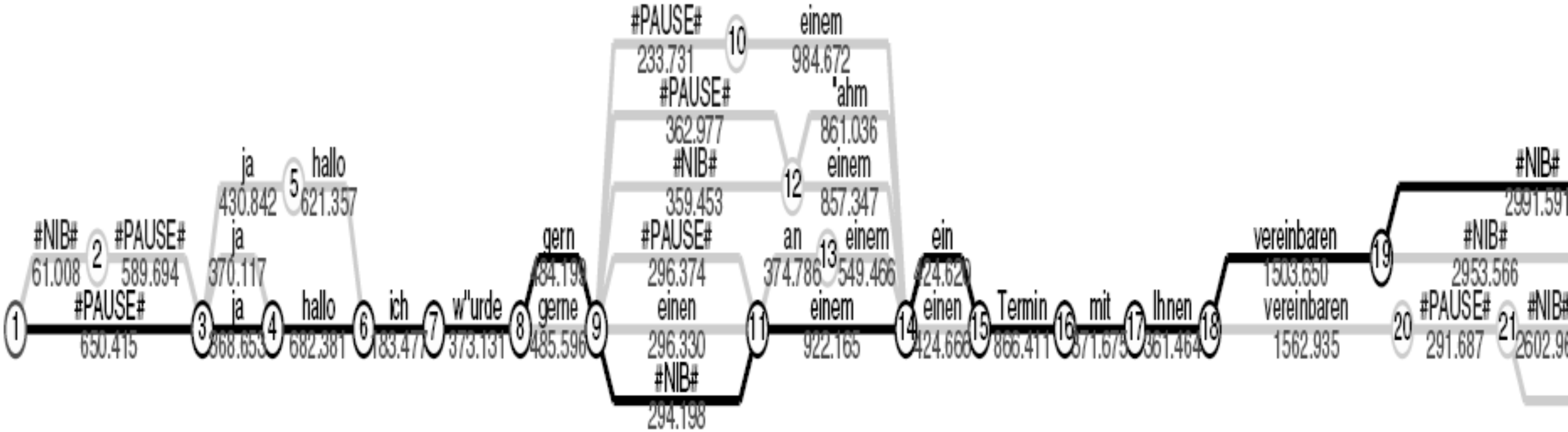


Fig. 1.3. Word graph. [1]

Specification / Verification / Modelling of concurrent systems / ...

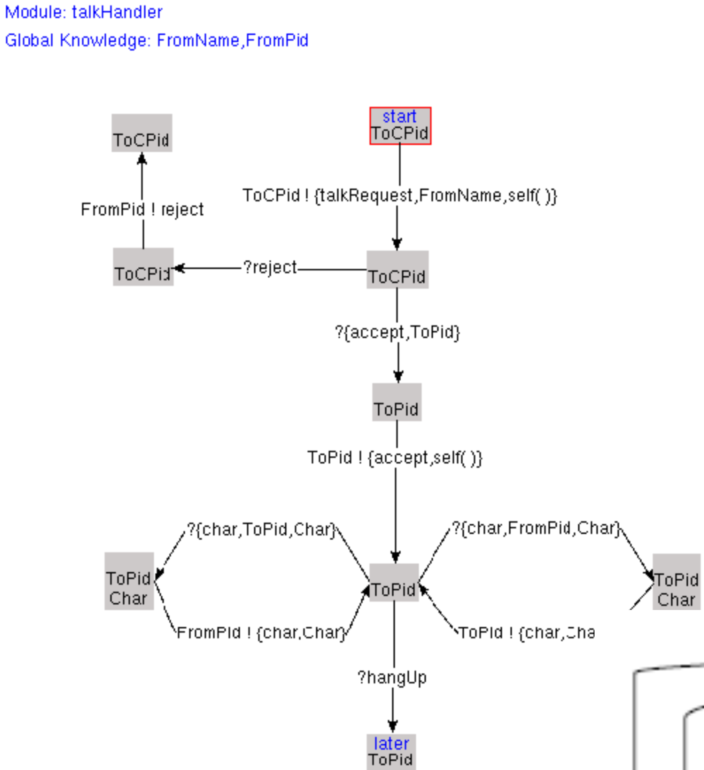


Fig. 1.5. Component transition diagram. [1]

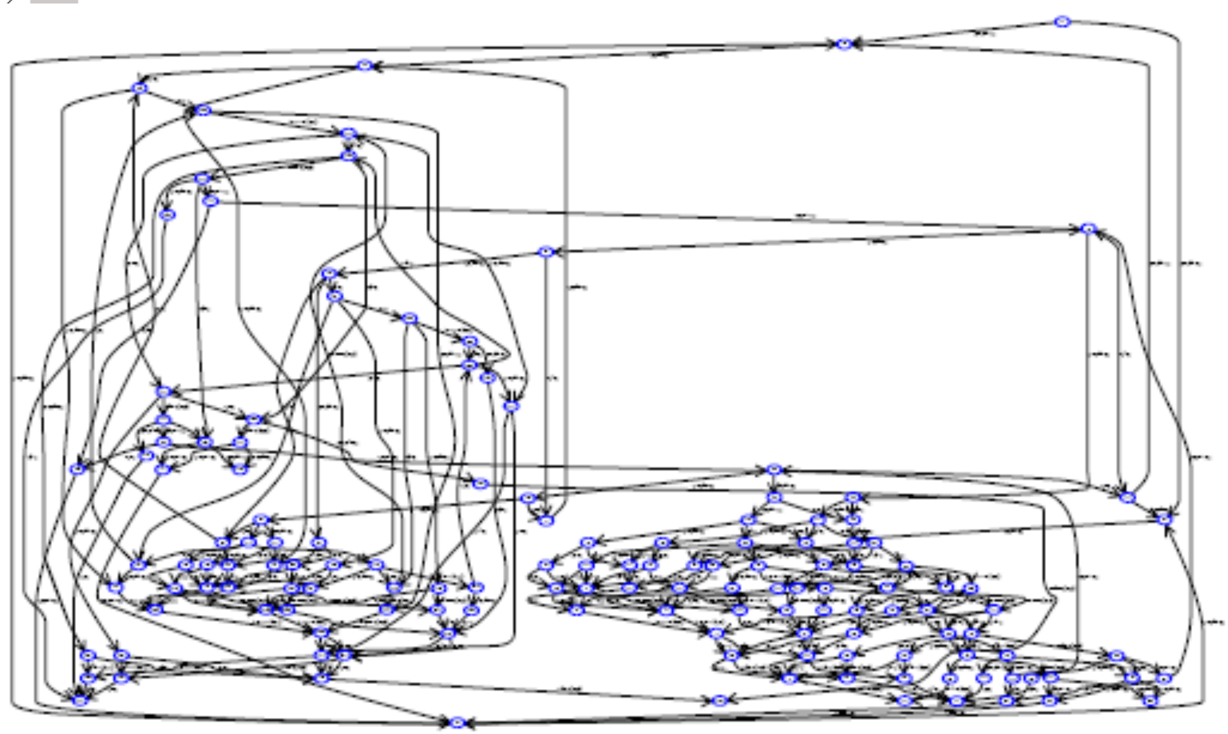


Fig. 1.6. Full transition diagram. [1]

Workflow Modeling

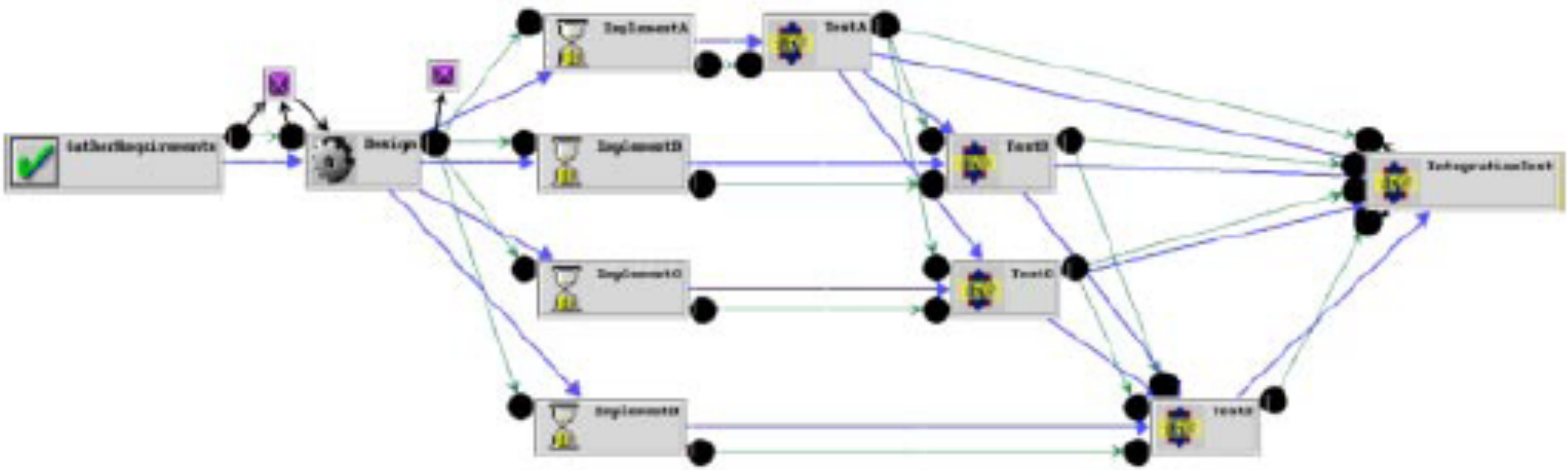


Fig. 1.7. Process graph, laid out manually. [1]

Data Modeling

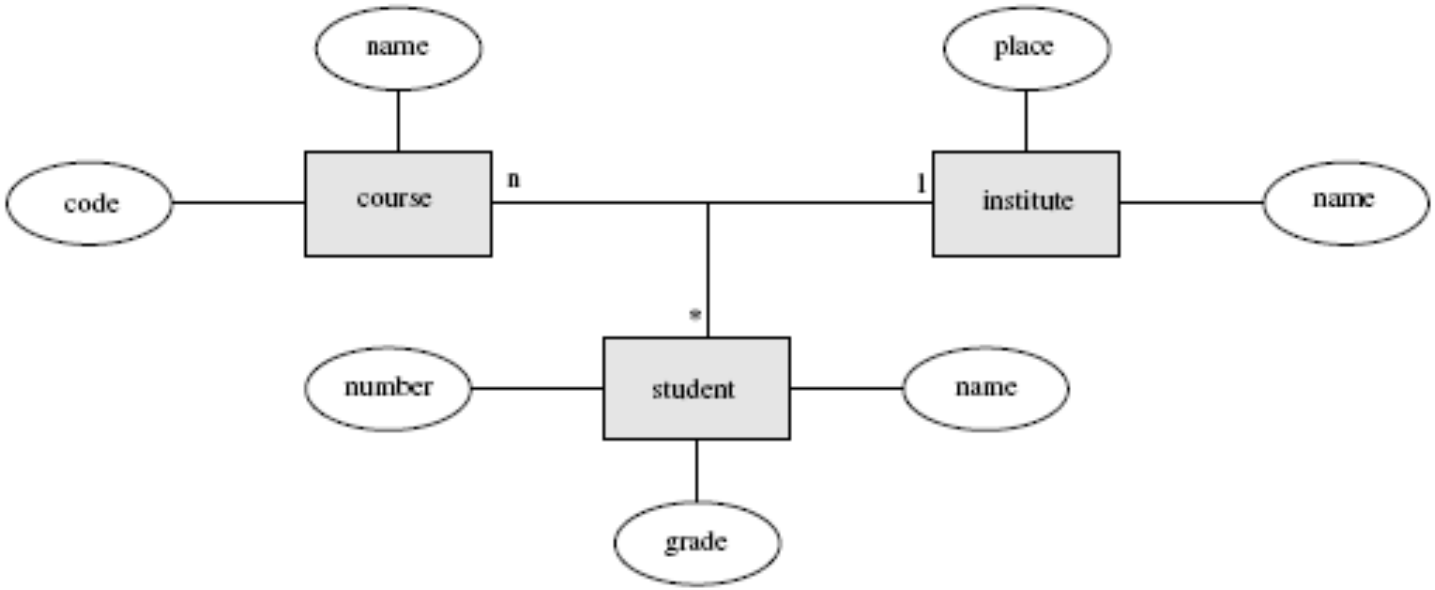
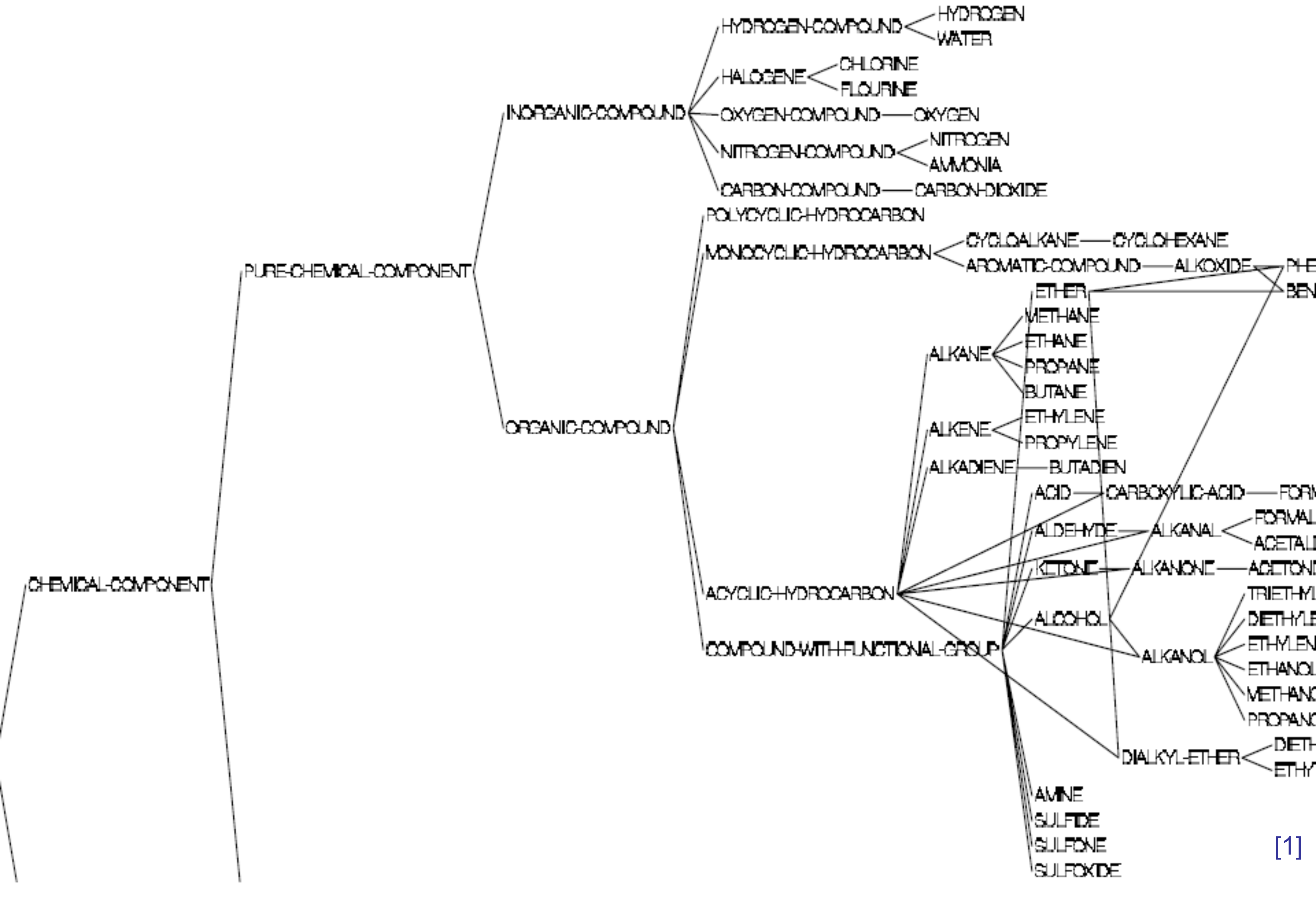


Fig. 1.8. Entity-relationship diagram. [1]

Subsumption hierarchy



[1]

Social Networks

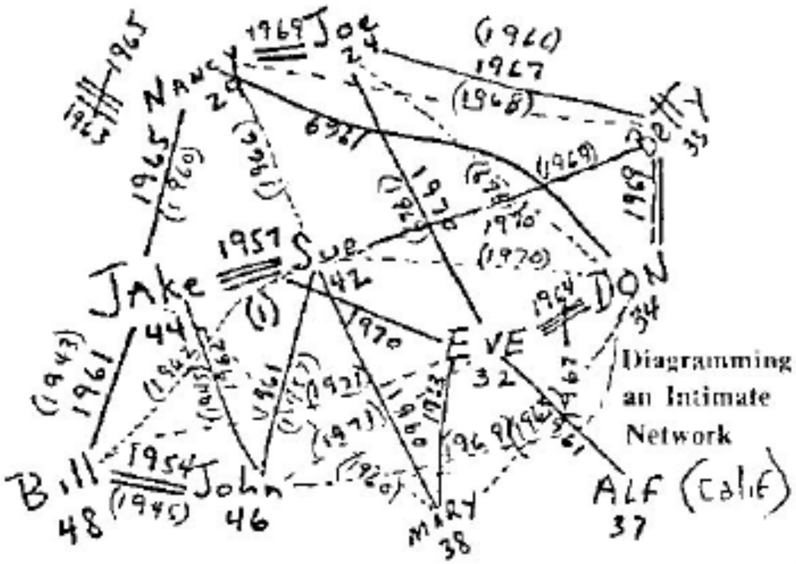


Fig. 1.10. Hand drawn social network. [1]

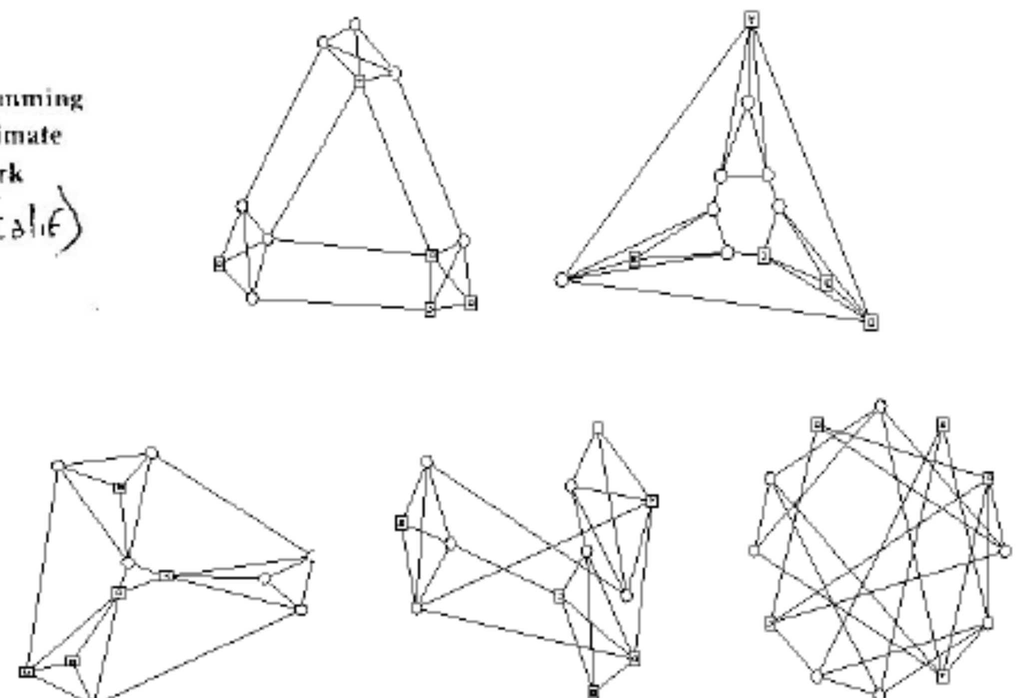


Fig. 1.11. Five drawings of a single network (Blythe et al., 1996). [1]

Criteria for good graph drawings [1]

- Crossing minimization: too many crossings → eye cannot follow easily; also possible: Technical reasons (VLSI design: electrical isolation)
- Bend minimization: too many bends in an edge → eye cannot follow easily; also poss.: Technical reasons (VLSI design: bends are error prone)
- Area minimization / Homogeneous use of area : Only use the amount of space that is necessary to visualize the (part of the) graph; use this area homogeneously → eye can dissolve structures more efficiently
- Angle maximization: Maximize angle between edges → eye can dissolve structures better

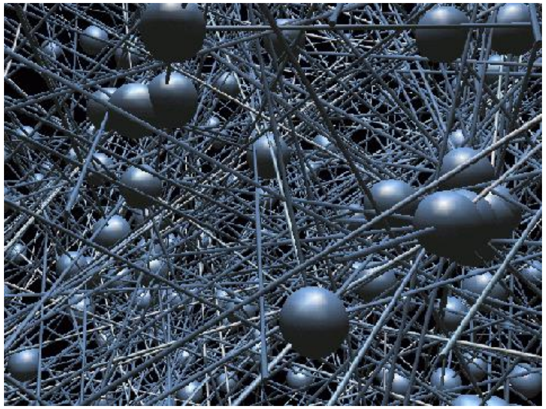
Criteria for good graph drawings [1]

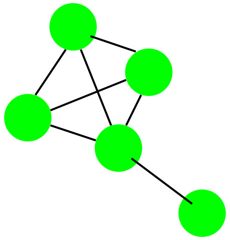
- Length minimization: Edges too long → eye can less easily follow
- Symmetry: Graph symmetries should be symmetrically visualized accordingly ←→ general goal of visualization
- Clusters: Clusters in the graph can be marked visually or collapsed into hypernodes to aid the structural overview
- Respect other implicit structures in the data → use layered drawings if layer-like orders are present

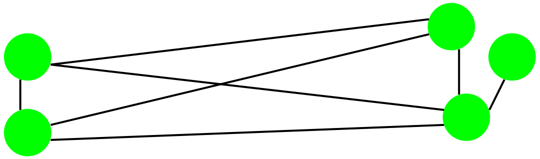
Criteria for good graph drawings [1]

- Some of these criteria are conflicting → compromise has to be found
- Further “general guidelines”:
  - choose number of nodes and edges depicted on the momentary canvas minimal so that the aspects that are intended to be visualized can be seen
  - choose visualization so that no artifical perception effects are induced in the viewer w.r.t. structures that are not structurally present in the graph

[2]







Planar Graphs [4]

- Some definitions / recapitulations from DS1:

  - Planar Representation of G=(V,E): Mapping of V to distinct points in $\mathbb{R}^2$ and edges E to open Jordan curves $I \to \mathbb{R}^2$ with properties:
    - Representation of each edge $e=(v_1,v_2)$ connects the representation of v_1 with the repr. of v_2
    - Representations of two different edges have no common points except common end-/starting-points
    - No vertex representations lie on any edge representation

Planar Graphs [4]

- Numerous efficient algorithms for drawing planar graphs exist [4]
We don't go into detail about these here.

- Planar graphs are visually appealing AND adhere perfectly to the "Minimize crossings" principle if drawn "correctly" (planar graphs may also be drawn WITH crossings)

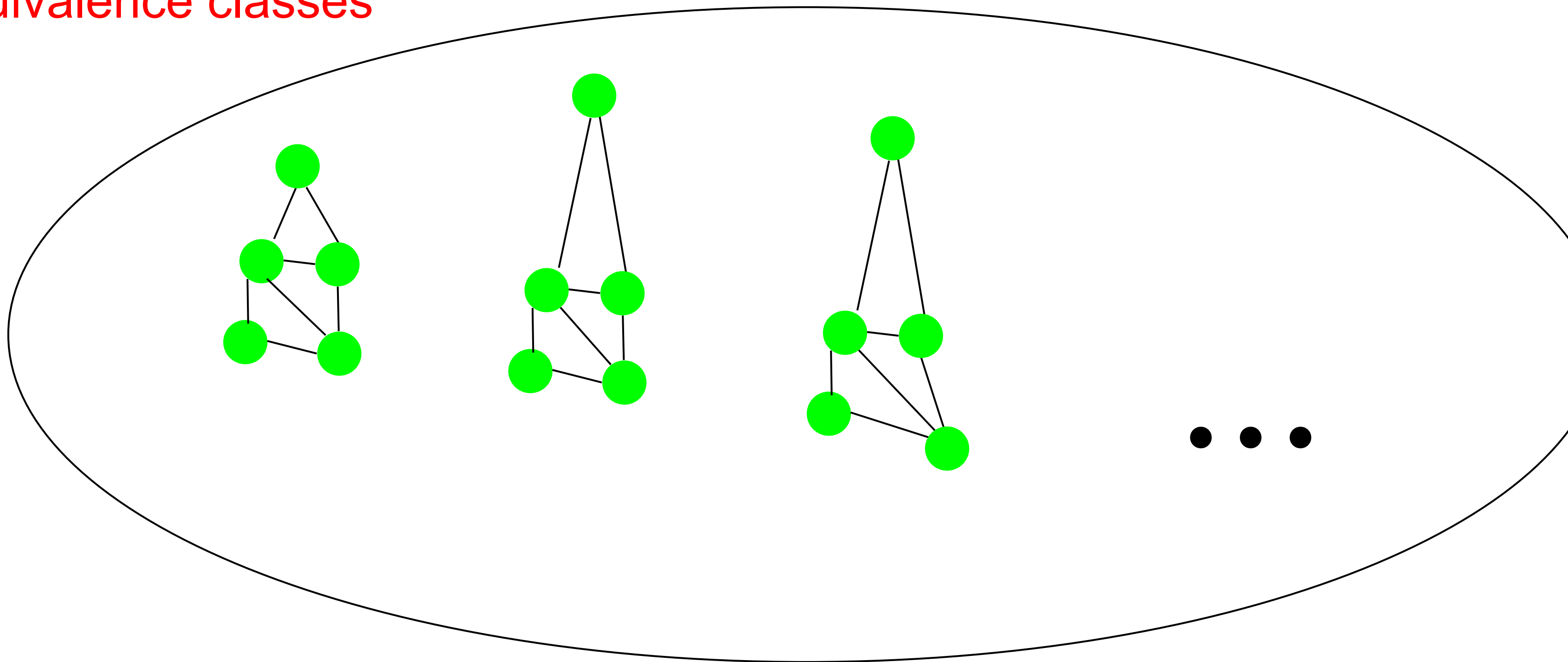
- → It may be rewarding to check for planarity

- Drawback: Usually Social Network graphs are non-planar

Planar Graphs [4]

- Some definitions / recapitulations from DS1:

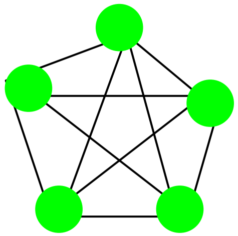
  - Planar Graph: Graph that has a planar representation
  - Each planar graph has infinitely many planar representations
  - Planar representations of planar graphs may be partitioned into equivalence classes

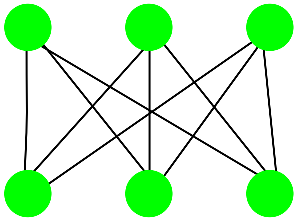


Planar Graphs [4]

- Some definitions / recapitulations from DS1:

  - Subdivision of Graph G: On G perform one or more instances of the following action: Add new node u; replace an edge e=(v1,v2) with the two edges e=(v1,u) , e=(u,v2)
  - Theorem of Kuratowski: G planar iff it does not contain a subdivision of K_5 or $K_{3,3}$
  - K_5 (complete graph with 5 nodes) and $K_{3,3}$ (bipartite complete graph with 3 nodes in each partition): prototypes of non-planar graphs



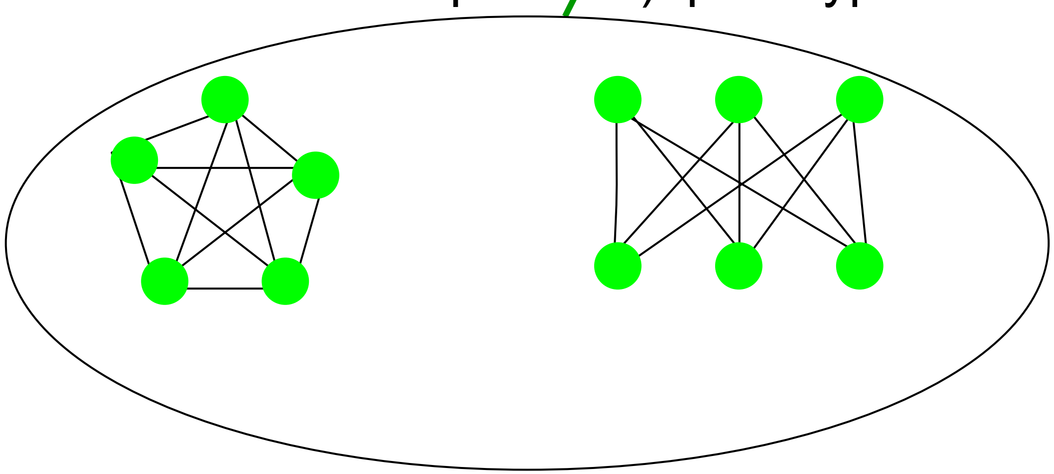


Planar Graphs [4]

- Some definitions / recapitulations from DS1:

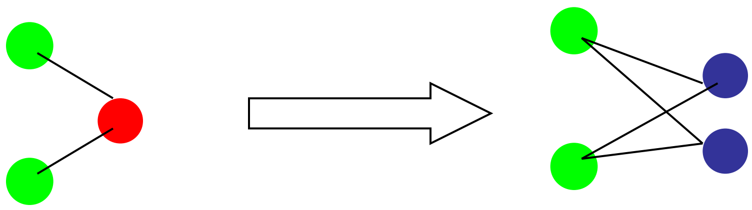
  - Su … of the fo … with t
  - Th … subd
  - K_5 … graph … graphs

Unfortunately, in a dense social network with many cliques, many such structures may exist → use planarity checks, planarity modifications and planar graph drawing only if network is rather sparse and only a small number of small cliques exist.



Planar Graphs [4]

- Linear time algorithms for planarity checking exist (e.g. Hopcroft & Tarjan (1974) or Lempel, Even & Cederbaum (1967)) (see [4])
- If G is non planar, one can try to make it planar by:
  - Deleting nodes (drastic operation AND deciding if the deletion of at most k vertices makes the graph planar is NP-complete (see [4])); But: In Social Networks maybe acceptable to "hide" actors which are not interesting for some reason
  - Splitting nodes (also quite drastic AND Deciding whether at most k vertex splits make G planar is NP-complete (see [4]); In social networks maybe acceptable if actor has more than one role



Planar Graphs [4]

- If G is non planar, one can try to make it planar by:
  - Inserting vertices (e.g. at crossings of a drawing of the non-planar original graph); For a Social Network visualization in most cases not suitable
  - Deleting Edges. Since spanning trees are planar, we can make graph planar by deleting edges In Social Networks maybe acceptable to “hide” ties which are not interesting for some reason
    - Deciding for non-planar G if ex. planar subgraph with at least k <|E| edges is NP-complete (see [4])
    - Finding one Maximal Planar Subgraph (no edges of G can be added without losing planarity) is in P-time (see [4] for alg)
    - Finding the Maximum Planar Subgraph (no other planar subgraph retains more edges of G) is NP hard (see[4])

Planar Graphs [4]:

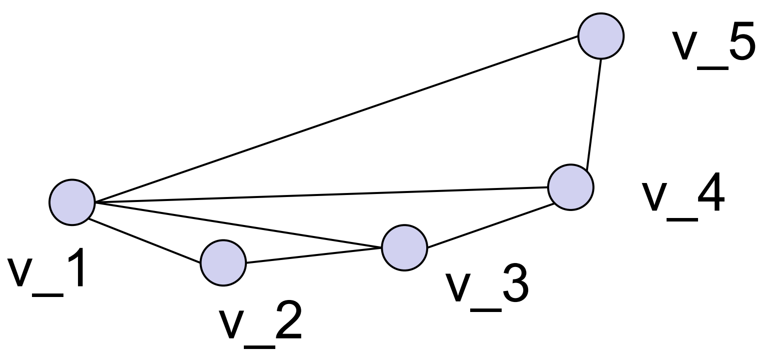
- 2-connected planar graphs are especially convenient to draw [4]

→ Inserting edges (Augmentation) in a planar graph so that the resulting graph is 2-connected and still planar. Finding the minimum number of such edges together with the edges themselves is NP-hard (see [4])

For Social Networks: Adding ties generally not desirable.

Pathfinder Networks [3]:

- → get graphs planar / “nearly” planar → leaving out less important edges is one of the better alternatives
- → similar idea: Pathfinder networks:
  - required: distance measure d(v_1, v_2) (alg is easily transferrable for similarities (e.g. tie strengths)) → use as weight of edges: w(e) = w(v_1,v_2) = d(v_1, v_2)
  - idea: remove edges that violate triangle inequality $d(v_1,v_{q+1}) \leq d(v_1, v_2) + d(v_2,v_3) + \ldots d(v_q, v_{q+1})$



- → parameter q: limits the number of links in the paths for which the triangle inequality is ensured in the final PFNET

q = upper limit on the number of nodes considered for determining the min-distance betw. two nodes

Pathfinder Networks [3]:

- → get graphs planar / “nearly” planar → leaving out less important edges is one of the better alternatives
- → similar idea: Pathfinder networks:
  - furthermore: use Minkowski metric to compute distance along paths: $d = \left(\sum_i (d_i)^r\right)^{1/r}$ → parameter r: defines how „pessimistically“ distances are computed
  - the larger r and q are, the sparser the pathfinder network → if r = ∞ and q = n = |V| → sparsest pathfinder network → union of MSTs of graph

With the two parameters r and q, a particular PFNET can be identified as PFNET(r,q). We can now state the definition of Pathfinder networks precisely. Given a DATANET (proximities) with adjacency matrix $A = [a_{ij}]$ and a distance matrix $D^{r,q} = [d_{ij}]$ computed with parameters r and q: A link (i,j) in the DATANET is a link in the PFNET(r,q) if and only if

$$a_{ij} \neq \infty \qquad \text{and} \qquad d_{ij} = a_{ij},\ i \neq j$$

Because different values of r and q result in different weights of paths, Pathfinder can produce several different PFNETs.

[3]

Pathfinder Networks [3]:

- Algorithm:
  - Matrices $W^{(i)}_{jk}$: store minimum distance from j to k using exactly i edges
  - Matrices $D^{(i)}_{jk}$: store minimum distance from j to k using at most i edges

  iterate:
  - compute $W^{(i)}$ by setting $W^{(i+1)}_{jk} = \min_m ((W^{(1)}_{jm})^r + (W^{(i)}_{mk})^r)^{1/r}$
  - compute $D^{(i)}$ by setting $D^{(i)}_{jk} = \min (W^{(1)}_{jk}, \dots, W^{(i)}_{jk})$

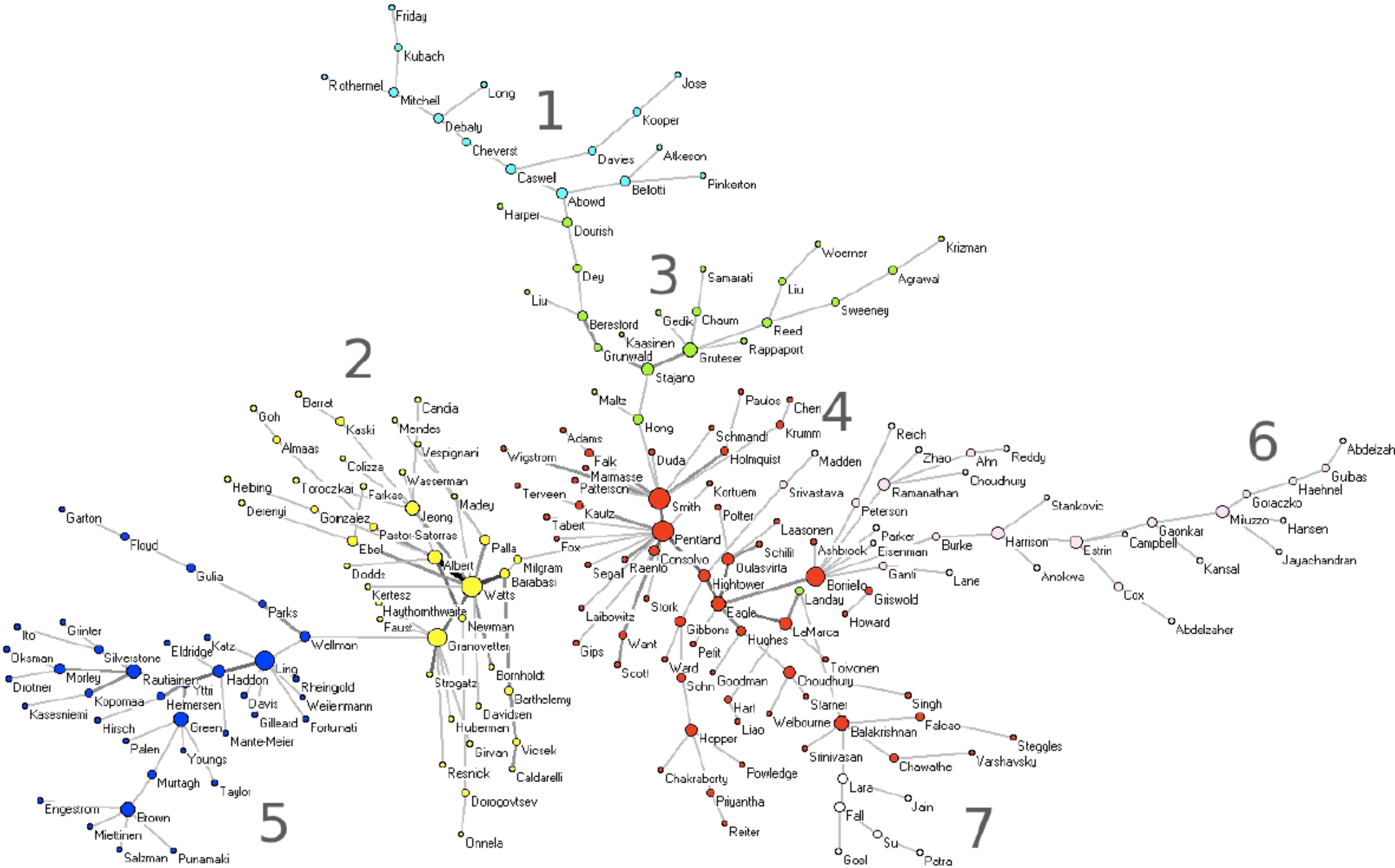
  until i = q

  retain all edges (j,k) for which $W^{(1)}_{jk} = D^{(q)}_{jk}$

  [6]

- this alg: $O(n^4)$ for q = n-1 and $r = \infty$: $O(n^2 \log n)$ alg known (see [6])

example: co-citation network in the field „mobile social networking“([6b])



Drawing by Using Physical Analogies [5]

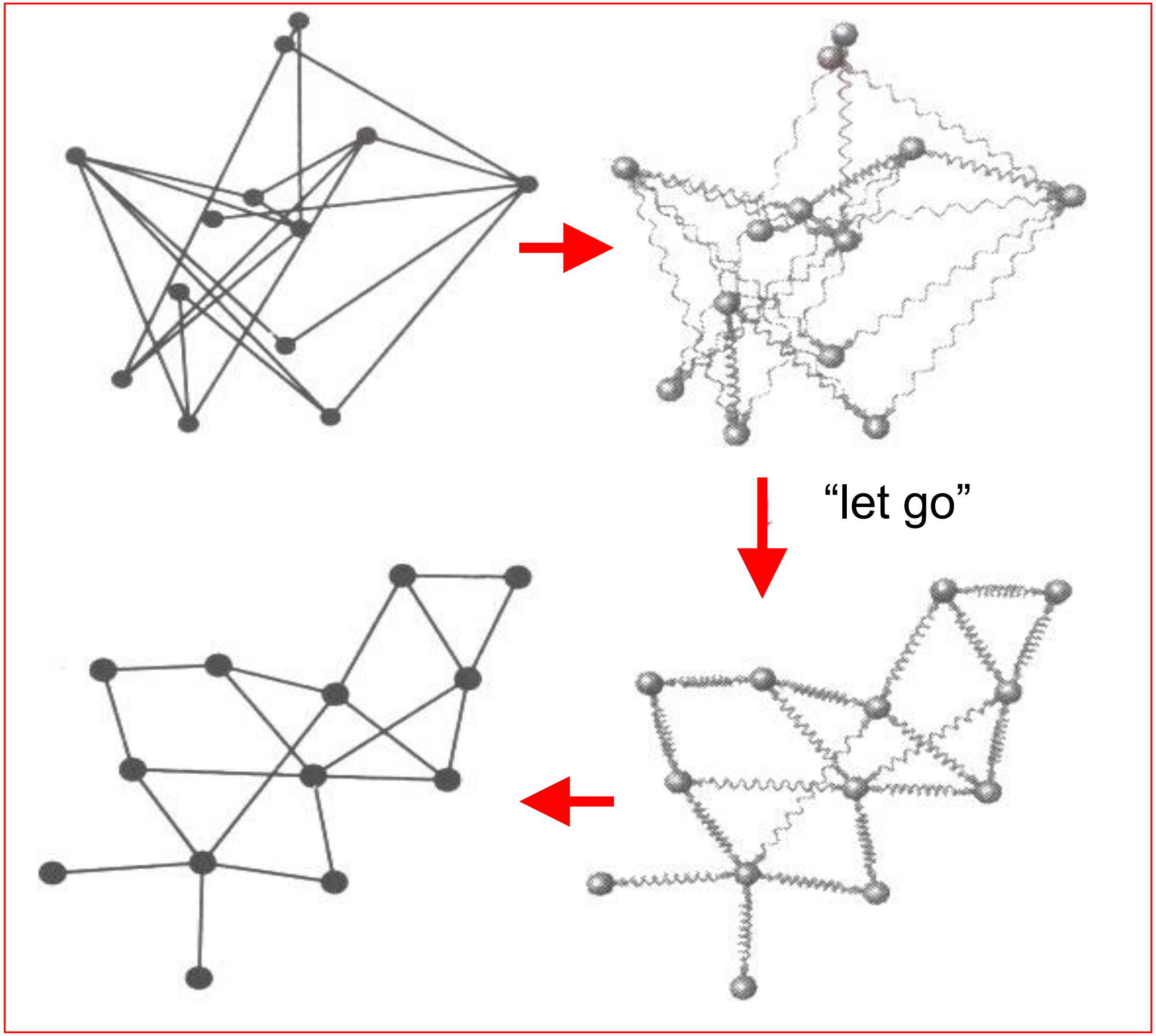
- Many graph drawing algorithms make formal structural assumptions about the graph → In social networks wie often cannot guarantee such formal properties
- → Desirable to have a framework for graph drawing which is customizable (e.g. to incorporate edge weights), easy to use, and yields good results for a large class of graphs
- → General Idea: We associate edges between nodes with physical forces acting upon the nodes and compute an energy minimum by “setting off the dynamics that is induced by the forces”
- → Advantages: Intuitive, easy to compute, yield “satisfactory results on medium sized graphs (~50 nodes)” [5]

Drawing by Using Physical Analogies [5]

- Quality criteria used here:
  - nodes should spread well on the canvas
  - adjacent vertices should be close to each other

- Realized by:
  - Repelling forces between nodes: Charged nodes
  - Attractive + Repelling forces between nodes: Springs between nodes
  - → small deviations from uniform length edges

Drawing by Using Physical Analogies [5]



[5]

Drawing by Using Physical Analogies [5]

- → Edges will have different length in equilibrium. (Deciding whether a graph has an embedding in $\mathbb{R}^n$ with equal length edges (regardless of n) is NP-hard.)[5]

- Intuition: System will “move” into a state of minimal energy → Either minimize energy function or incrementally “move” nodes until “stable state” is reached.

The function of the following physics excursus is just to explain the background and to make it easier for those interested in a deep understanding to follow the models presented in [5]. It will NOT ITSELF be part of the exam.

Forces, Classical Mechanics, Classical Electrodynamics

- 2 Particles v_i and v_j with positive charges q each at coordinates $\mathbf{v_i}=(\mathrm{v_{ix}}, \mathrm{v_{iy}})$ and $\mathbf{v_j}=(\mathrm{v_{jx}}, \mathrm{v_{jy}})$ exert electrostatic force on each other:

$$\mathbf{F}_e(\mathbf{v}_i, \mathbf{v}_j) = -\frac{q^2}{\|\mathbf{v}_i - \mathbf{v}_j\|^2} \frac{(\mathbf{v}_i - \mathbf{v}_j)}{\|\mathbf{v}_i - \mathbf{v}_j\|}$$

- 2 Particles v_i and v_j at coordinates $\mathbf{v_i}=(\mathrm{v_{ix}}, \mathrm{v_{iy}})$ and $\mathbf{v_j}=(\mathrm{v_{jx}}, \mathrm{v_{jy}})$ connected through a spring with spring constant k and natural length l exert elastic force on each other (Hooke's law):

$$\mathbf{F}_s(\mathbf{v}_i, \mathbf{v}_j) = k(\|\mathbf{v}_i - \mathbf{v}_j\| - l) \frac{\mathbf{v}_i - \mathbf{v}_j}{\|\mathbf{v}_i - \mathbf{v}_j\|}$$

Classical Mechanics

- Dynamics of any system of physics: Action principle: Minimize the Action (System's state at t1 und t2 fixed):

$$\dot{q} = \frac{d}{dt} q$$

Action S:
L is Lagrange Function of system
(in simple cases: L=T-U
where T=kinetic Energy; U=pot.Energy

$$S = \int_{t1}^{t2} L(q, \dot{q}) dt$$

"Minimize" S →
Variational derivative:

$$\delta S = 0$$

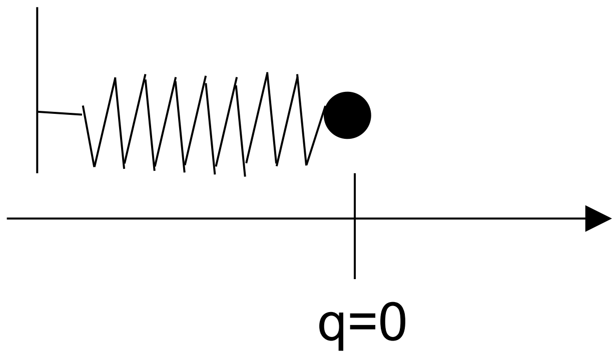
→ Euler-Lagrange-Equations
(Eq. of Motion)

$$\frac{\partial}{\partial x} L - \frac{d}{dt} \frac{\partial}{\partial \dot{x}} L = 0$$

Classical Mechanics

- Example 1: 1-dim. Harmonic oscillator:

$$L = \frac{1}{2} m\dot{q}^2 - \frac{1}{2} k q^2$$



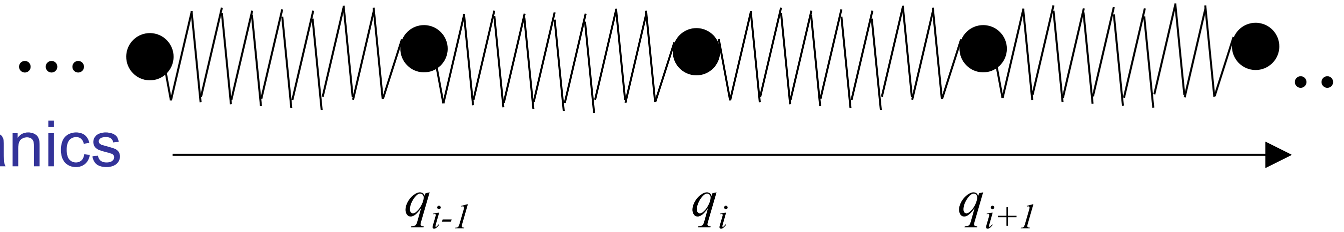
$$\Rightarrow \quad m\ddot{q} + kq = 0$$

$$\Rightarrow \quad q(t) = A\cos\left(\sqrt{\frac{k}{m}}\,t\right)$$

possible solution

(Physics Excursus)

Classical Mechanics



- Example 2: Many coupled 1-dim. harmonic oscillators

$$L = \sum_i \frac{1}{2} m_i \dot{q}_i^{\,2} - \frac{1}{2} \sum_{ij} K_{ij} q_i q_j$$

$$\Rightarrow \quad \ddot{q}_i + \sum_j \frac{1}{m_i} K_{ij} q_j = 0$$

$$\ddot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{0}$$

K is symmetric matrix

$\Rightarrow$ diagonalize K: $\mathbf{K} = \mathbf{D}^{-1} diag(\{k_i\})\mathbf{D}$

introduce new Coordinates: $\mathbf{x} = \mathbf{D}\mathbf{q}$

$$\Rightarrow \quad \ddot{x}_i + \frac{k_i}{m_i} x_i = 0$$

Classical Mechanics

- Example 2: Many coupled 1-dim. harmonic oscillators

$$\ddot{x}_i + \frac{k_i}{m_i} x_i = 0$$

$$\Longrightarrow \quad x_i(t) = A_i \cos\left(\sqrt{\frac{k_i}{m_i}}\, t\right)$$

possible solution

$$\Longrightarrow \quad \mathbf{q}(t) = \mathbf{D}^{-1}\mathbf{x}(t)$$

Drawing by Using Physical Analogies [5]

- → Problem: The System is undamped → It will "oscillate forever" (in a coupled way)

- → introduce damping (friction) term $\sim r\dot{q}$ into Equations of motion

- → System will still oscillate forever but oscillation amplitudes will converge to zero → make a cut-off in time when movements have become small

- But: (1) Equation will become a bit harder to solve; (2) While the "spring constants" K_{ij} are interpretable in a social (graph theoretical) way (corresp. to "tie strength" / edge weights) the friction term is not as easily interpretable socially (graph theoretically)

Drawing by Using Physical Analogies [5]

- → we have not yet transformed the system to 2 dimensions → solution becomes harder
- → we have not yet incorporated the electromagnetic forces (dynamic case: not only electrostatic!) → exact solution becomes even harder
- → drop all the exact physics (after all we just want to draw a graph) and make algorithmic approach
- ☺

Drawing by Using Physical Analogies [5]

- Orginal approach by Eades (see [5]): Change repelling and spring forces a bit:

$$\mathbf{F}_{repell}(\mathbf{v}_i, \mathbf{v}_j) = -\frac{c_\rho}{\| \mathbf{v}_i - \mathbf{v}_j \|^2} \frac{(\mathbf{v}_i - \mathbf{v}_j)}{\| \mathbf{v}_i - \mathbf{v}_j \|}$$

applied only to nodes not connected by edge of graph

$$\mathbf{F}_{spring}(\mathbf{v}_i, \mathbf{v}_j) = c_\sigma \log \frac{\| \mathbf{v}_i - \mathbf{v}_j \|}{l} \frac{(\mathbf{v}_i - \mathbf{v}_j)}{\| \mathbf{v}_i - \mathbf{v}_j \|}$$

applied only to nodes connected by edge of graph

log a/b instead of a-b exerts less force on far apart nodes

Drawing by Using Physical Analogies [5]

small constant

- Move node under total force F as: $\mathbf{v}_i \leftarrow \mathbf{v}_i + \delta \cdot F_{v_i}(t)$

whole spring embedder algoritm (see [5])

Input: Graph G=(V,E); inital placements $\{\mathbf{v}_i\}$ of nodes

Output: "Relaxed" new positions $\{\mathbf{v}_i\}$

for (t = 0; t < numberOfIterations; t++){

forall (v_i ∈ V){

$$F_{v_i}(t) \leftarrow \sum_{\{u \mid (u,v_i) \notin E\}} F_{repell}(u, v_i) + \sum_{\{u \mid (u,v_i) \in E\}} F_{spring}(u, v_i)$$

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \delta \cdot F_{v_i}(t)$$

}

}

Drawing by Using Physical Analogies [5]

- Modified approach by Fruchterman & Reingold (see [5]):

  1) Change repelling and spring forces a bit:

$$\left.\begin{aligned}
\mathbf{F}_{repell}(\mathbf{v}_i,\mathbf{v}_j) &= -\frac{l^2}{\|\mathbf{v}_i-\mathbf{v}_j\|^2}\,\frac{(\mathbf{v}_i-\mathbf{v}_j)}{\|\mathbf{v}_i-\mathbf{v}_j\|} \\[2em]
\mathbf{F}_{spring}(\mathbf{v}_i,\mathbf{v}_j) &= \frac{\|\mathbf{v}_i-\mathbf{v}_j\|^2}{l}\,\frac{(\mathbf{v}_i-\mathbf{v}_j)}{\|\mathbf{v}_i-\mathbf{v}_j\|}
\end{aligned}\right\}
\quad
\begin{gathered}
\mathbf{F}_{repell}-\mathbf{F}_{spring}=0 \\
\text{if} \\
\|\mathbf{v}_i-\mathbf{v}_j\|=l
\end{gathered}$$

Drawing by Using Physical Analogies [5]

- Modified approach by Fruchterman & Reingold (see [5]):

2) Speed up computation by restricting contributions to repulsive force on node v_i to “nearby” nodes. Use grid to determine which nodes are “nearby”. Among these use only those repelling force contributions above a threshold

3) Clip net displacement $\delta(t)$ on node at iteration t to prevent excessive displacements in late stages → better convergence

4) Clip coordinates at canvas boundaries

Drawing by Using Physical Analogies [5]

- Further Modified approach by Frick et al. (see [5]):

  1) change forces a bit further in order to
     - slow down nodes with high degree
     - speed up computation further by dropping square-root computation

$$\mathbf{F}_{repell}(\mathbf{v}_i, \mathbf{v}_j) = -\frac{l^2}{\| \mathbf{v}_i - \mathbf{v}_j \|^2} (\mathbf{v}_i - \mathbf{v}_j)$$

$$\mathbf{F}_{spring}(\mathbf{v}_i, \mathbf{v}_j) = \frac{\| \mathbf{v}_i - \mathbf{v}_j \|^2}{l \;\; \Phi(v_j)} (\mathbf{v}_i - \mathbf{v}_j)$$

$$\Phi(v_j) = 1 + \frac{\deg(v_j)}{2}$$

Drawing by Using Physical Analogies [5]

- Further Modified approach by Frick et al. (see [5]):

  2) introduce additional “gravitational” force towards the “barycenter” to prevent disconnected components of graph to drift too far to the boundary of the canvas

$$\mathbf{F}_{grav}(\mathbf{v}_i, \mathbf{v}_j) = \gamma\, \Phi(v_j)\, \left(\frac{\zeta}{|V|} - \mathbf{v}_j\right)$$

Barycenter:

$$\zeta = \sum_{v_i \in V} \mathbf{v}_i$$

- Even simpler possibility: Drop Barycenter and direct gravitation towards canvas center ☺

$$\Phi(v_j) = 1 + \frac{\deg(v_j)}{2}$$

Drawing by Using Physical Analogies [5]

- Further Modified approach by Frick et al. (see [5]):

  3) Detect and suppress rotational and oscillatorial moves by nodes (“ineffective” movements) (see [5] for details)

What we see: Spring embedder type of graph drawing algorithms are

- Robust towards adaptations
- Allow for an intuitive interpretation of change options

Drawing by Using Physical Analogies [5]

- Different approach (Kamada & Kawai): Minimize potential energy directly

- Potential energy of spring of natural length l and actual length d is

$$U_{spring}(d) = \frac{1}{2} c_\sigma (d - l)^2$$

To compare forces:
Remember that

$$\mathbf{F} = \nabla U$$

- Further assumption: Ideal distance between two nodes u and v should be proportional to the length of the shortest path between u and v in G

Drawing by Using Physical Analogies [5]

- Undirected graph: Compute for all pairs of nodes the shortest paths between them: ASSP problem → Floyd Warshall algorithm
- Denoting the shortest path between u and v in G by $dist_G(u,v)$ and the ideal length of an edge by l, we have to minimize the overall potential

$$U_{KK} = \sum_{i<j} \frac{c}{dist_G(v_i, v_j)^2} \left(\| \mathbf{v}_i - \mathbf{v}_j \| - l \; dist_G(v_i, v_j)^2 \right)$$

Drawing by Using Physical Analogies [5]

$$U_{KK}(\{\mathbf{v}_i\}) = \sum_{i<j} \frac{c}{dist_G(v_i,v_j)^2} (\| \mathbf{v}_i - \mathbf{v}_j \| - l\ dist_G(v_i,v_j)^2)$$

- Minimize by modified Newton Raphson method: compute

$$\max_i \| \nabla_i U_{KK}(\{\mathbf{v}_i\}) \|$$

and move v_i until gradient length drops below threshold

Drawing by Using Physical Analogies [5]

- KK is rather complicated computationally and the paradigm "spring strength (inverse) proportional to shortest path" is not always suitable for every application (Social networks ?)

- → Use very simple model of springs with natural length zero:

$$U_{center} = \sum_{(i,j)\in E} \| \mathbf{v}_i - \mathbf{v}_j \|^2$$

- Direct optimization (∇U=0) leads to set of linear equations: (A is the adjacency matrix of G; $\deg_G(v)$ denotes the degree of node v; x is the vector of x-coordinates of all nodes; y is the vector of y-coordinates of all nodes)

$$(diag(\deg_G(v_i)) - \mathbf{A})\,\mathbf{x} = 0$$
$$(diag(\deg_G(v_i)) - \mathbf{A})\,\mathbf{y} = 0$$

- If for each connected component of G one node's coordinates are fixed, the equations are guranteed to yield a solution (see [5])

- Other approach that is often used for these family of techniques (direct optimization of potential ("objective function")): Simulated Annealing

simulated annealing (see [5])

Input: Graph G=(V,E); inital placement $p=\{\mathbf{v}_i\}$ of nodes
Output: new positions $p=\{\mathbf{v}_i\}$ with minimal $U(p)$

```
while (T > THRESHOLD){
  forall (v_i ∈ V){
      p^old ← p
      v_i ← v_i + δ_random
      if( U(p^old) < U(p) ){
         with probability 1 - e^(-(u(p^old)-U(p))/T) reset p ← p^old
      }
  }
  reduce T;
}
```

$$p^{old} \leftarrow p$$

$$\mathbf{v}_i \leftarrow \mathbf{v}_i + \boldsymbol{\delta}_{random}$$

$$\text{if}(\, U(p^{old}) < U(p)\,)\{ \quad \text{with probability } 1 - e^{-\frac{u(p^{old})-U(p)}{T}} \text{ reset } p \leftarrow p^{old}$$

Recommended Readings

- Read the sections containing the material we covered in [1], [4], [5] and ([3] or [6])

- Interested students: read rest of [1], [4], [5] as well, or, alternatively, read survey [10].
Read [6].

Possible Exam Questions

- Define “2-dimensional Graph Drawing”!

- Define “planar representation of a graph”! Explain, why planar graphs are especially convenient to draw! (1 sentences)

- For a social network visualization: are `splitting nodes‘ and `inserting nodes‘ in order to make a network planar, good alternatives? Give an informal counter-argument! (1 sentence)

- Explain the role of the parameters q and r for Pathfinder networks! (1 sentence each)

- Explain the meaning of the following elements in Eades‘ expression for the spring force $\mathbf{F}_{spring}(\mathbf{v}_i, \mathbf{v}_j) = c_\sigma \log \frac{\|\mathbf{v}_i - \mathbf{v}_j\|}{l} \frac{(\mathbf{v}_i - \mathbf{v}_j)}{\|\mathbf{v}_i - \mathbf{v}_j\|}$:
    - $\mathbf{v}_i, \mathbf{v}_j$
    - l
    - $\log \frac{\|\mathbf{v}_i - \mathbf{v}_j\|}{l}$

- What is the motivation behind the gravitational force $\mathbf{F}_{grav}(\mathbf{v}_i, \mathbf{v}_j) = \gamma\, \Phi(v_j)\left(\frac{\sum_{\mathbf{v}_i \in V} \mathbf{v}_i}{|V|} - \mathbf{v}_j\right)$ with $\Phi(v_j) = 1 + \frac{\deg(v_j)}{2}$ in Frick‘s approach?

- Name two differences between the approach of Kamada and Kawai and the original spring embedder approach of Eades!

- Explain the basic idea of Simulated Annealing!

Citations

(1) R. Fleischer, C. Hirsch: Graph Drawing and Its Applications in M. Kaufmann, D. Wagner (Eds.): Drawing Graphs – Methods and Models; Springer LNCS 2025, 2001

(2) http://arts.anu.edu.au/sss/CV_Klovdahl_USL.pdf (URL, 2011)

(3) Schvaneveldt, R. W., Durso, F. T., & Dearholt, D. W. (1989). Network structures in proximity data. In G. Bower (Ed.), The *psychology of learning and motivation: Advances in research and theory*, Vol. 24 (pp. 249–284). New York: Academic Press

(4) R. Weiskircher "Drawing Planar Graphs" in M. Kaufmann, D. Wagner (Eds.): Drawing Graphs – Methods and Models; Springer 2001

(5) U. Brandes; Drawing on Physical Analogies" in M. Kaufmann, D. Wagner (Eds.): Drawing Graphs – Methods and Models; Springer LNCS 2025, 2001

(6) Quirin, A. and Cordon, O. and Guerrero-Bote, V.P. and Vargas-Quesada, B. and Moya-Anegon, F.: A quick MST-based algorithm to obtain Pathfinder networks (∞, n- 1), journal={Journal of the American Society for Information Science and Technology}, volume={59}, number={12}, pages={1912--1924}, year={2008}

(6b) Georg Groh and Christoph Fuchs: Multi-modal Social Networks for Modeling Scientific Fields", Scientometrics 80(2), pp. 569-590, 2011

Text-Books on Graph Visualization

7) Giuseppe Di Battista, Peter Eades, Roberto Tamassia, Ioannis G. Tollis Graph Drawing
Algorithms for the Visualization of Graphs Prentice Hall, 1999

8) Michael Kaufmann, Dorothea Wagner (Eds.) Drawing Graphs:
Methods and Models Lecture Notes in Computer Science Tutorial 2025 Springer, 2001

9) Michael Jünger, Petra Mutzel (Eds.) Graph Drawing Software Springer, 2003

Survey Articles:

10) Ivan Herman, Guy Melancon, and M. Scott Marshall: Graph Visualization and Navigation in Information Visualization: A Survey; IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, VOL. 6, NO. 1, JANUARY-MARCH 2000

11) Ellson, J. and Gansner, E.R. and Koutsofios, E. and North, S.C. and Woodhull, G.: Graphviz and dynagraph--static and dynamic graph drawing tools; journal==Graph Drawing Software, volume={127}, pages={148}, year={2003}, publisher={Springer}