Kap. 6: Kräftebasierte Verfahren

Prof. Dr. Petra Mutzel

Lehrstuhl für
Algorithm Engineering LS11
Universität Dortmund

25./26. VO WS07/08 28./29. Januar 2008

Literatur

- M. Kaufmann, D. Wagner (Eds.): Drawing Graphs: Methods and Models, Lecture Notes in Computer Science, Tutorial, Vol. 2025, 2001, ISBN 3-540-42062-2.
- M. Jünger, P. Mutzel (Eds.): Kapitel 2: Technical Foundations aus: Graph Drawing Software, Mathematics and Visualization, Springer Verlag, 2004, ISBN 3-540-00881-0.
- Brandenburg, Himsolt, Rohrer: An experimental comparison of force-directed and randomized graph drawing algorithms, Proc. Int. Symp. Graph Drawing, GD´95, LNCS 1027, Springer, 76-87, 1995.

Original-Literatur für diese VO

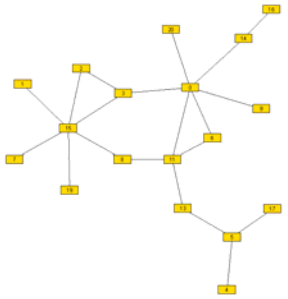
- Originalartikel: Peter Eades: A heuristic for graph drawing, Congressus Numerantium, vol. 42, 149-160, 1984.
- Fruchterman & Reingold: Graph-drawing by force-directed placement, Software Practice and Experience, 21 (11), 1129-1164, 1991.
- Frick, Ludwig, Mehldau: a fast adaptive layout algorithm for ndirected graphs. Proc. of the DIMACS Int. Workshop on Graph Drawing, GD 1994, LNCS, vol. 894, 388-403, Springer, 1995.
- Kamada & Kawai: An algorithm for drawing general undirected graphs, Inform. Processing Letters 31, 7-15, 1989.
- W.T. Tutte: How to draw a graph, Proc. Mathematical Society, no. 3, 743-768, 1963.

Original-Literatur für diese VO

- Davidson & Harel: Drawing graphs nicely using simulated annealing, ACM Trans. Graph, vol. 15, no. 4, 301-331, 1996.
- Sugiyama & Misue: Graph drawing by magnetic-spring model, Journal Visual Language Computing, vol. 6, (3), 876-892, 1995.
- Harel & Sardas: Randomized graph drawing with heavy-duty preprocessing, TR CS93-16, Weizmann Inst. Science, 1993.
- Tunkelang: A practical approach to drawing undirected graphs, TR CMU-CS-94-161, Carnegie Mellon Univ., 1994.
- Ryall, Marks & Shieber: An interactive constraint-based system for drawing graphs, ACM Symp. on User Interface Software and Technology, 97-104, 1997

Kap. 6: Kräftebasierte Verfahren

- Netzwerke
- Soziale Netzwerke
- Beziehungsstrukturen
- Rechnernetze
- ...



Eades 1984

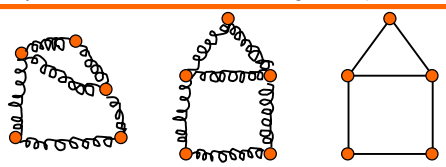
Überblick zu Kapitel 6

6.1 Grundidee

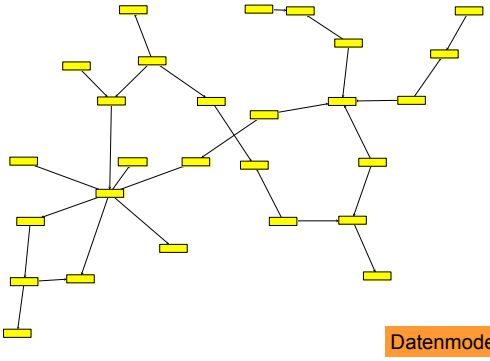
Erstellung eines Kräftesystems

- **Knoten:** elektrisch geladene Partikel, die sich gegenseitig abstoßen
- **Kanten:** Federn (durch Kanten verbundene Knoten ziehen sich also durch Federn an)

Gesucht: Geradlinige Zeichnung, die dem Systemzustand minimaler Energie entspricht



Beispiel: Kräftebasiertes Verfahren



Datenmodell

Kräftebasierte Verfahren werden charakterisiert durch

Modell:

- ein durch die Knoten und Kanten definiertes **Kräftesystem**, d.h. ein physikalisches Modell oder
- Modellierung der **potentiellen Energie** eines Systems.

Algorithmus:

- berechnet eine Approximation eines Gleichgewichts-Zustands des Kräftesystems, d.h. an jedem Knoten herrscht die Kraft =0, oder
- berechnet einen Systemzustand mit minimaler Energie.

Es gibt eine Vielzahl an Variationen
HIER: nur die wichtigsten

Eigenschaften: Kräfteverfahren sind:

- sehr beliebt und weit verbreitet
- intuitiv leicht verständlich
- leicht zu programmieren
- relativ gute Resultate für baumartige Graphen
- leicht erweiterbar um viele Typen von Nebenbedingungen
- existieren relativ lange (Tutte 1963, danach Eades 1984)
- es gibt viele publizierte Varianten
- und noch mehr Software (z.B. auf dem Web)
- Animationen sind schön anzusehen ☺

6.2 Kräfteverfahren / Spring Embedder

Kraft an einem Knoten $v \in V$:

- $F(v) = \sum_{u \in \delta(v)} f_{uv} + \sum_{(u,v) \in V \times V} g_{uv}$ wobei
- f_{uv}: **Federkraft:** Kraft durch Feder zwischen u und v
- g_{uv}: **Abstoßungskraft:** elektrische Abstoßung zwischen u und v

- f_{uv} folgt dem Hookeschen Gesetz, ist also proportional zur Differenz zwischen der Distanz von u und v und der natürlichen Länge der Feder
- g_{uv} ist umgekehrt proportional zum Quadrat der Distanz von u und v

6.2.1 Spring Embedder (Eades)

Für $p,q \in R^n$ sei $d(p,q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$

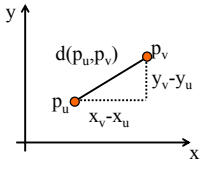
der Euklidische Abstand von p und q

ab jetzt: 2D (n=2), 3D analog

Abstoßende Kraft an v: $p_u - p_v$
Anziehende Kraft an v: $p_v - p_u$

Position von Knoten v: $p_v = (x_v, y_v)$

normierte Richtung



Einheitsvektor: $p_v - p_u \,/\, d(p_v,p_u)$

Koordinaten des Einheitsvektors:

$\left((x_v - x_u)/d(p_u,p_v), (y_v - y_u)/d(p_u,p_v) \right)$

Spring Embedder (Basic)

ab jetzt: 2D (n=2), 3D analog

Position von Knoten v: $p_v=(x_v,y_v)$

Betrag / normierte Richtung

x-Komponente von F(v) (y analog):

$$\sum_{u\in\delta(v)} S_{uv}\,(d(p_u,p_v)-l_{uv})\,(x_v-x_u)/d(p_u,p_v) + \sum_{(u,v)\in V\times V} R/(d(p_u,p_v)^2\,(x_u-x_v)/d(p_u,p_v)$$

Federkraft | Abstoßungskraft

mit:
- l_{uv} : natürliche Länge der Feder zwischen u und v (keine Kraft falls $d(p_u,p_v)=l_{uv}$)
- S_{uv} = Steife der Feder zwischen u und v (Je steifer die Feder, desto größere Tendenz zur Ideallänge)
- R = Stärke der elektrischen Abstoßung zwischen den Knoten

Modellierte Aesthetikkriterien

- Tendenz zur Ideallänge von Kanten
- Knoten nicht zu nahe zusammen
- (indirekt:) Tendenz zur Symmetrie
- Beeinflussbarkeit durch Wahl der Parameter l_{uv}, S_{uv}, R

Spring Embedder von Eades

Federkraft nicht linear, sondern logarithmisch:
→ schwächere Kraft auf weit entfernten Knoten

Betrag / normierte Richtung

x-Komponente von F(v) (y analog):

$$\sum_{u\in\delta(v)} S_{uv}\log(d(p_u,p_v)/l_{uv})\,(x_v-x_u)/d(p_u,p_v) + \sum_{(u,v)\in V\times V} R/(d(p_u,p_v)^2\,(x_u-x_v)/d(p_u,p_v)$$

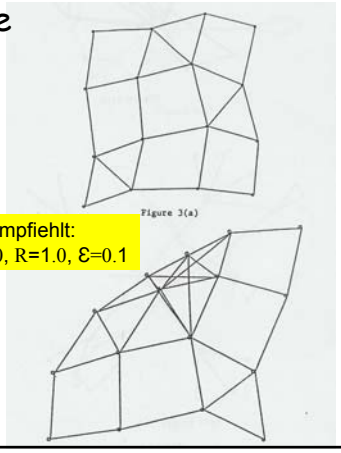
Federkraft | Abstoßungskraft

mit:
- l_{uv} : natürliche Länge der Feder zwischen u und v (keine Kraft falls $d(p_u,p_v)=l_{uv}$)
- S_{uv} = Steife der Feder zwischen u und v (Je steifer die Feder, desto größere Tendenz zur Ideallänge)
- R = Stärke der elektrischen Abstoßung zwischen den Knoten

Algorithmus zur Approximation eines Kräftegleichgewichts

- Starte mit zufälligen Positionen p_v für alle $v\in V$
- **Iteriere:**
  - Berechne Kräfte F(v) für alle $v\in V$
  - Modifiziere für alle $v\in V$ die Position p_v leicht in Richtung F(v): $p_v=p_v+\varepsilon F(v)$ (für kleines ε)
- bis die Summe über alle v wirkenden Kräfte klein genug ist.

Beispiele



Eades empfiehlt:
S_{uv}=0.2, l_{uv}=1.0, R=1.0, ε=0.1

aus: Eades 1984

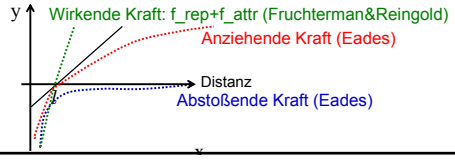
6.2.2 Fruchterman und Reingold 1991

Idee: Modifikation der Kräfte für schnellere Konvergenz

Abstoßende Kräfte für alle Knotenpaare:
$f\_rep(p_u,p_v) = l^2/(d(p_u,p_v)\ (p_u-p_v)/d(p_u,p_v)$ korrigiert

Anziehende Kräfte für adjazente Knotenpaare (Federkraft):
$f\_attr(p_u,p_v) = d(p_u,p_v)^2 / l\ \ (p_v-p_u)/d(p_u,p_v)$

Effekt: Kraft nimmt überproportional zu mit Distanz
Hoffnung: schnellere Konvergenz



Weitere Änderungen FR

- Keine Auswertung der abstoßenden Kräfte für weit entfernte Knotenpaare
  - Dazu: Gittertechnik (s. Kap. 6.5 für große Graphen)

Displacement Vektor Ɛ:

- Im Laufe der Iterationen wird Ɛ nach und nach verkleinert, um große Änderungen zu vermeiden (“springt”, “oszilliert”)
- Falls neue Position außerhalb eines vorher definierten Rechtecks ist, dann: setze diese Koordinate auf den Rand (*clipping*)

Beispiele: s. GDE, AGD, LEDA

6.2.3 Frick, Ludwig und Mehldau 1995

Idee: Modifikation der Kräfte für schnellere Konvergenz und bessere Qualität, „GEM“ (für Graph Embedder)

- Definition der Kräfte, so dass keine Quadratwurzel berechnet werden muss → ganzzahlige Arithmetik

Abstoßende Kräfte für alle Knotenpaare:
$f\_rep(p_u,p_v) = l^2/(d(p_u,p_v)^2 \ (p_u-p_v)$

Anziehende Kräfte für adjazente Knotenpaare (Federkraft):
$f\_attr(p_u,p_v) = d(p_u,p_v)^2 / (l^2\Phi(v)) \ (p_v-p_u)$
wobei $\Phi(v)=1+\Delta(v)/2$.

- $\Phi(v)$ sorgt dafür, dass Knoten mit hohem Grad weniger Anziehungskraft besitzen.

Weitere Änderungen FLM

- Einführung einer Gravitationskraft, die alle Knoten zum Schwerpunkt aller Knoten hinzieht:

Gravitationskraft:
$f\_grav(p_u,p_v) = \Phi(v)\ \gamma\ (\ \zeta\ /\ |V|-p_v)$ wobei
$\zeta = \sum_{w\in V} p_w$
wobei $\Phi(v)=1+\Delta(v)/2$ und γ eine Gravitationskonstante.

- f_grav ist für Knoten mit hohem Grad größer
- f_grav ist wichtig für nicht-zusammenhängende Graphen

- Einführung einer zusätzlichen kleinen Zufallskraft für alle Knoten → Erhöhung der Robustheit gegen schwache Konfigurationen

Änderungen bei Berechnung FLM

- Merke die Kraft $F_v(t-1)$ auf Knoten v aus der vorigen Iteration
- Jeder Knoten hat eigenes $Ɛ \rightarrow Ɛ_v(t)$, die in jeder Iteration angepaßt wird:
  - falls Knoten in ungefähr gleicher Richtung wie in vorheriger Iteration bewegt wird, dann ist $Ɛ_v(t)$ größer
  - sonst (umgekehrte Richtung): kleiner
  - ähnliches, falls “Rotation” des Layouts entdeckt wird
- Knoten werden in zufälliger Reihenfolge berechnet

- → sorgt für schnellere Konvergenz

Algorithmus FLM

1. Für alle $v\in V$: initialisiere v
2. **Solange** $T_{global} > T_{min}$ **und** #Iterationen$<R_{max}$**{**
   1. Wähle einen Knoten v für Update
   2. Berechne Kräfte $F(v)$ an v
   3. Aktualisiere die Position von v um $p_v=p_v+Ɛ_v(t)F(v)$
   4. Aktualisiere $Ɛ_v(t)$ (abhängig von Oszillation, Rotation,…)
3. **}**

- T_{global} : globale Temperatur: Durchschnitt über alle lokalen Temperaturen $Ɛ_v(t)$ für alle v
- R_{max}: maximale Anzahl an Iterationen
- Auswahl der Knoten: pro Runde: zufällige Permutation

Bewertung

- leicht besser als FR
- für kleine Graphen leicht schlechter als Kamada-Kawai, aber viel besser für große Graphen
- schneller als FR und KK
- experimentelle Laufzeit: $O(|V|^3)$:
  - $|V|$ Iterationen pro Runde,
  - jede Iteration betrachtet $|V|$ Knoten,
  - meist werden experimentell ca. $|V|$ Runden benötigt

Layout [illegible]ahren

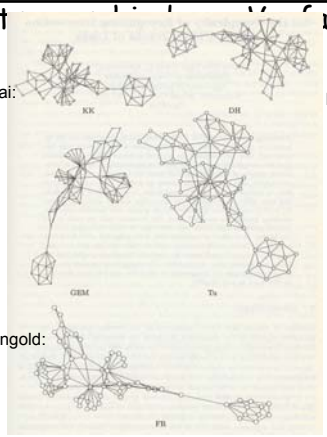
Kamada & Kawai:

Davidson & Harel

Frick et al:

Tunkelang

Fruchterman & Reingold:



Brandenburg et al., 1995

6.3 Energie-basierte Verfahren

Idee: ähnlich wie Kräfteverfahren, aber:
- keine Kräftevektoren mehr im Modell
- stattdessen: Energiefunktionen

- 6.3.1 Kamada & Kawai 1989
- 6.3.2 Tutte 1963
- 6.3.3 Davidson & Harel 1996 (1989)
- 6.3.4 Tunkelang 1994

6.3.1 Simulation graphentheoretischer Distanzen (Kamada & Kawai 1989)

- Sei $G=(V,E)$ zusammenhängend und $u,v \in V$
- $\delta(u,v)$ = Anzahl der Kanten auf einem kürzesten (u,v)-Weg

- **Idee:** Definiere Federn zwischen allen Paaren von Knoten
- mit natürlicher Länge proportional zu $\delta(u,v)$

- Kraft zwischen u und v: $S_{uv}(d(p_u,p_v)-\delta(u,v))$
- Die potentielle Energie der (u,v)-Feder wird beschrieben durch die Funktion: $\frac{1}{2} S_{uv}(d(p_u,p_v)-\delta(u,v))^2$
- Wahl des Steife-Parameters S_{uv}: $S_{uv} = k / \delta(u,v)^2$ mit k const.
- **Interpretation:** stärkere Federn zwischen graphen-theoretisch nahen Knoten

keine Vektoren mehr!

6.3.1 Simulation graphentheoretischer Distanzen (Kamada & Kawai 1989)

- Potentielle Energie der (u,v)-Feder:
- $k/2 \, (\, (d(p_u,p_v) / \delta(u,v)) - 1)^2$
- Potentielle Energie der Zeichnung:
- $\eta = k/2 \sum_{u \neq v \in V} (\, (d(p_u,p_v) / \delta(u,v)) - 1)^2$

- Kraft zwischen u und v: $S_{uv}(d(p_u,p_v)-\delta(u,v))$
- Die potentielle Energie der (u,v)-Feder wird beschrieben durch die Funktion: $\frac{1}{2} S_{uv}(d(p_u,p_v)-\delta(u,v))^2$

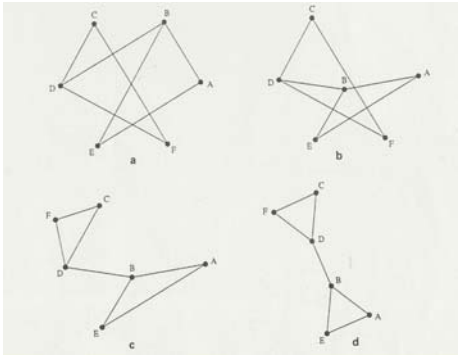
- Notwendige Bedingungen für minimales η:
- $\partial\eta / \partial x_v = 0$ und $\partial\eta / \partial y_v = 0$ für alle $v \in V$
- $\rightarrow 2|V|$ nicht-lineare Gleichungen

Lösungsmethode von KK

- **Problem:** Newton-Raphson Methode nicht geeignet, weil beide Systeme abhängig voneinander sind.
- **Lösung:** je 1 Knoten wird versetzt, und zwar an seine energie-minimale Position: betrachte η als Funktion von (x_m,y_m) und berechne Minimum mit 2-dim. Newton-Raphson Methode

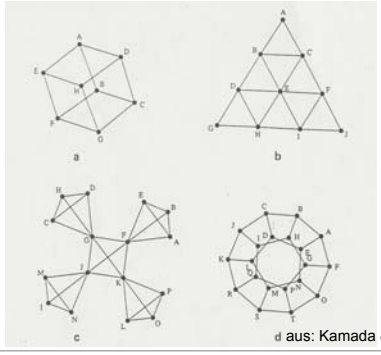
1. **Iteriere**
   1. $v \leftarrow$ Knoten mit größter Kraft $\sqrt{((\partial\eta/\partial x_v)^2 + (\partial\eta/\partial y_v)^2)}$
   2. Verschiebe v in eine **energie-minimale Position** (bei festen Positionen der anderen Knoten)
2. **bis** Verschiebung einen Schwellwert unterschreitet

Animation des Energieminimierung-Prozesses



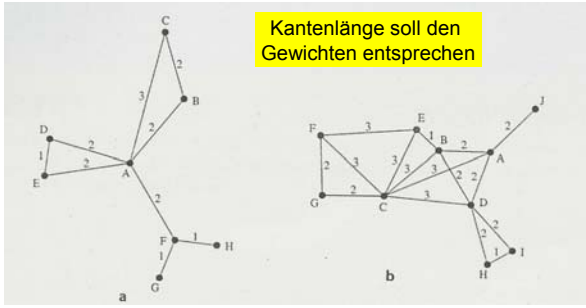
aus: Kamada & Kawai 1989

Layout Beispiele



aus: Kamada & Kawai 1989

Beispiele gewichteter Graphen



aus: Kamada & Kawai 1989

6.3.2 Verfahren von Tutte

Älteste Variante 1960, 1963

- l_{uv}=0, S_{uv}=1, und keine abstoßenden Kräfte
- → $f(v) = \sum_{u \in \delta(v)} (p_u - p_v)$

- **Problem:** p_v=0 für alle $v \in V$ ist optimal
- → keine gute Zeichnung

- **Ausweg:** Fixiere Positionen von wenigstens drei Knoten, etwa als Ecken eines konvexen Polygons:
- Festnageln und bestimme Positionen p_v der anderen (freien) Knoten $v \in V$, so dass $F(v)=0$.

6.3.2 Verfahren von Tutte

Älteste Variante 1960, 1963

- D.h. löse das Gleichungssystem
- $\sum_{u \in \delta(v)} (x_u - x_v) = 0$ und $\sum_{u \in \delta(v)} (y_u - y_v) = 0$ für alle $v \in V$

- **Problem:** p_v=0 für alle $v \in V$ ist optimal
- → keine gute Zeichnung

- **Ausweg:** Fixiere Positionen von wenigstens drei Knoten, etwa als Ecken eines konvexen Polygons:
- Festnageln und bestimme Psitionen p_v der anderen (freien) Knoten $v \in V$, so dass $F(v)=0$.

Verfahren von Tutte

Notationen:

$V = V_0 \dot\cup V_1$ ($V_0 \cap V_1 = \emptyset$) mit
- V_0: Menge der fixierten Knoten
- V_1: Menge der freien Knoten

Für $v \in V$ seien
- $N_0(v) = \{u \in V_0 \mid (u,v) \in E\}$ Menge der fixerten Nachbarn von v
- $N_1(v) = \{u \in V_1 \mid (u,v) \in E\}$ Menge der freien Nachbarn von v
- $\Delta(v) = |\{(u,v) \mid (u,v) \in E\}|$ Grad von v

Für $v \in V_0$ sei (x_v^*, y_v^*) die fixierte Position von v

Wir erhalten das Gleichungssystem
- $\Delta(v)\, x_v - \sum_{u \in N_1(v)} x_u = \sum_{w \in N_0(v)} x_w^*$ für alle $v \in V_1$ und
- $\Delta(v)\, y_v - \sum_{u \in N_1(v)} y_u = \sum_{w \in N_0(v)} y_w^*$ für alle $v \in V_1$

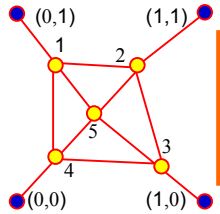
GLS mit je $|V_1|$ Unbekannten und Gleichungen

Eigenschaften der Lösung

- Jeder freie Knoten liegt im Schwerpunkt (Barycenter) seiner Nachbarn
- → „**Schwerpunkt-Methode**“ (Barycenter-Methode)

- Lösung des GLS mittels Linearer Algebra, Numerik, Schnelle Methoden für dünne Matrizen

Beispiel

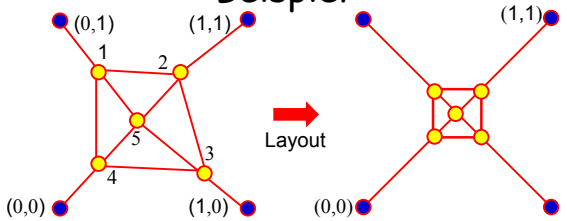


Wir erhalten das Gleichungssystem

- $\Delta(v)\, x_v - \sum_{u \in N_1(v)} x_u = \sum_{w \in N_0(v)} x_w$* für alle $v \in V_1$
- $\Delta(v)\, y_v - \sum_{u \in N_1(v)} y_u = \sum_{w \in N_0(v)} y_w$* für alle $v \in V_1$

$$A=\begin{pmatrix} 4 & -1 & 0 & -1 & -1 \\ -1 & 4 & -1 & 0 & -1 \\ 0 & -1 & 4 & -1 & -1 \\ -1 & 0 & -1 & 4 & -1 \\ -1 & -1 & -1 & -1 & 4 \end{pmatrix} \quad Ax=\begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad Ay=\begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad x=\begin{pmatrix} 0.375 \\ 0.625 \\ 0.625 \\ 0.375 \\ 0.500 \end{pmatrix} \quad y=\begin{pmatrix} 0.625 \\ 0.625 \\ 0.375 \\ 0.375 \\ 0.500 \end{pmatrix}$$

Beispiel



$$A=\begin{pmatrix} 4 & -1 & 0 & -1 & -1 \\ -1 & 4 & -1 & 0 & -1 \\ 0 & -1 & 4 & -1 & -1 \\ -1 & 0 & -1 & 4 & -1 \\ -1 & -1 & -1 & -1 & 4 \end{pmatrix} \quad Ax=\begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad Ay=\begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad x=\begin{pmatrix} 0.375 \\ 0.625 \\ 0.625 \\ 0.375 \\ 0.500 \end{pmatrix} \quad y=\begin{pmatrix} 0.625 \\ 0.625 \\ 0.375 \\ 0.375 \\ 0.500 \end{pmatrix}$$

Eigenschaften der Lösung

Theorem: Sei G ein 3-zusammenhängender planarer Graph, f eine Region in einer planaren Einbettung von G, und P eine streng konvexe planare Zeichnung von f.

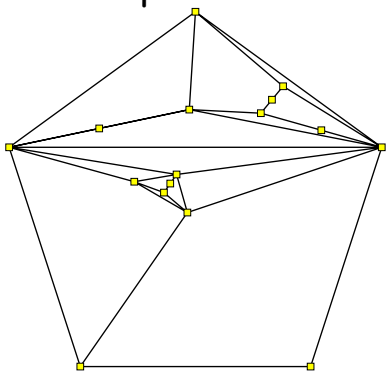
Werden die Knoten von f gemäß P fixiert, so liefert die Schwerpunkt-Methode eine konvexe planare Zeichnung von G, d.h. jede Region ist ein konvexes Polygon.

(ohne Beweis)

→ planares Verfahren, benötigt aber exponentiellen Platz

Beispiele: s. AGD

Beispiel Tutte



AGD-Layout

6.3.3 Davidson & Harel 1996

Erster Technical Report: 1989

- Minimiere $\eta = \lambda_1\eta_1+\lambda_2\eta_2+\lambda_3\eta_3+\lambda_4\eta_4$ mit
- λ_i : Parameter
- η_i : Energiefunktionen als Modelle von Aesthetikkriterien

- $\eta_1 = \sum_{u,v \in V} (1 / d(p_u,p_v)^2)$ (Abstoßung von Knoten)
- $\eta_2 = \sum_{u \in V} (1/r_u^2 + 1/l_u^2 + 1/t_u^2 + 1/b_u^2)$ mit

  r_u, l_u, t_u, b_u Abstände zu dem rechten, linken, oberen, unteren Rand der Zeichenfläche
- $\eta_3 = \sum_{(u,v) \in E} d(p_u,p_v)^2$ (kurze Kanten)
- η_4 = Zahl der Kreuzungen

Approximation eines Energieminimums durch Simulated Annealing

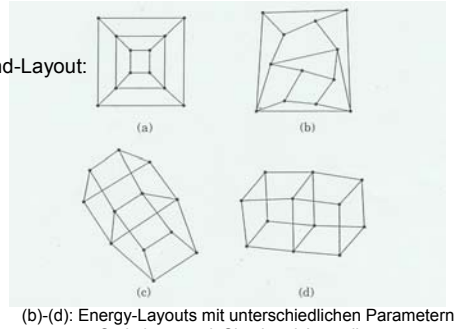
Bemerkungen DH

- In „fine tuning“ Version (nicht default): Einführung von Knoten-Kanten Separation; hierzu:
  - Verallgemeinerung der Kreuzungen:
  - **Idee:** Kreuzungen sind Extremfall von Knoten-Kanten Distanz 0.
  - Für Knoten v und Kante k mit Distanz g_{vk}: Beitrag $\eta_5 = \lambda_5 / g_{vk}^2$
  - Zusätzlich definiere minimum Distanz g_{min} und bei Distanzen kleiner als g_{min} ist der zusätzliche Beitrag $\lambda_4 = \lambda_5 / g_{min}^2$

- im Vergleich zu anderen Verfahren langsamer wegen Simulated Annealing
- meist sehr gute Qualität

Layout-Beispiele

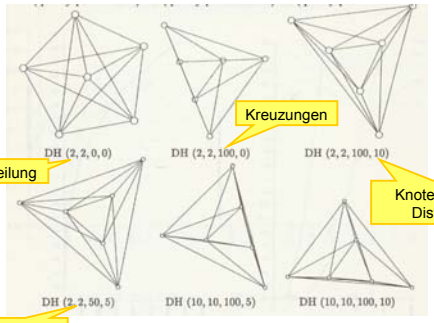
(a) Hand-Layout:



(b)-(d): Energy-Layouts mit unterschiedlichen Parametern
Optimierung mit Simulated Annealing

aus: Davidson & Harel 1996

K_6 mit unterschiedlichen Parametern



Kreuzungen

Knotenverteilung

Knoten/Kanten Distanzen

Kantenlänge

aus: Brandenburg et al. 1995

6.3.4 Tunkelang 1994

- Minimiere $\eta = \lambda_1\eta_1+\lambda_2\eta_2+\lambda_3\eta_3$ mit
- λ_i : Parameter
- η_i : Energiefunktionen als Modelle von Aesthetikkriterien

- $\eta_1 = \sum_{u,v\in V} (1 / d(p_u,p_v)^2)$ (Abstoßung von Knoten)
- $\eta_2 = \sum_{(u,v)\in E} d(p_u,p_v)^2$ (kurze Kanten)
- η_3 = Zahl der Kreuzungen

Initiale Knotenplatzierung

- Berechne Knotenordnung, so dass die Knoten nach ihrem Center-Rank (min max Distanz) absteigend sortiert sind.
- Platziere den ersten Knoten in die Mitte der Zeichenfläche
- Platziere den nächsten Knoten v durch „sampling“ und wähle die beste Position aus (bzgl. Energiefunktion)
  - dazu teile die Zeichenfläche in Zellen ein und teste jeweils l Zellen von Nachbarn entfernt (l sei Wunschkantenlänge).
  - versuche zunächst für v die Position in Nähe der Nachbarn (direkt oder in Distanz l), dann den Barycenter der Nachbarn und die Positionen um l entfernt, dann die 4 Ecken der Zeichenfläche;

Lokale Optimierung

- Nach der initialen Platzierung jedes einzelnen Knoten (!) erfolgen jeweils lokale Optimierungsschritte:
  - Iterativ wird versucht, bessere Positionen für bereits platzierte Knoten zu finden; dies geschieht wieder durch Sampling derselben Positionen wie bei Initialisierung
  - War für Knoten v eine Umplatzierung erfolgreich, dann werden rekursiv alle seine Nachbarn aufgerufen
- Nachdem alle Knoten platziert wird, wird für jeden Knoten einmal eine lokale Optimierung gestartet

Bewertung

- Laufzeit laut Autor: Laufzeit vergleichbar mit FR, DH ist deutlich langsamer
- Qualität laut Autor:
  - Knotenverteilung und Kantenlänge
    - Für dünne Graphen ist Tu besser als FR, insbesondere für große Graphen, DH ist viel schlechter
    - Fü dichte Graphen schneidet FR am besten ab
  - Kreuzungen
    - Tu deutlich besser als FR und DH

„None of the three algorithms produce satisfactory drawings for most large dense graphs. It is unclear whether these graphs are inherently ugly, or whether the algorithms are simply not up to drawing them well.“ [Tunkelang 1994]

Layouts verschiedener Verfahren

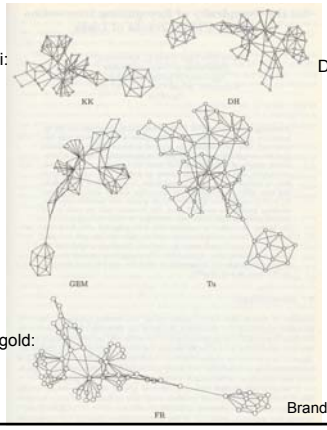
Kamada & Kawai:

Davidson & Harel

Frick et al:

Tunkelang

Fruchterman & Reingold:



Brandenburg et al, 1995

Experimenteller Vergleich von Brandenburg, Himsolt, Rohrer 1995

- FR, KK, FKM und DH ohne Kreuzungsoptimierung generieren oft ähnliche Layouts, zeigen Symmetrien, und sind relativ gut auf Graphen mit regulären Strukturen.
- Tu generiert oft Layouts, die anders sind als alle anderen. „Tu is worth a trial, if the others fail…ist behaviour is hard to predict. Ist quality parameter has an estimated exponential impact on the run time."
- KK generiert Layouts mit Kantenlängen, die ungefähr gleich lang sind.
- DH ist der flexibelste, hat aber auch die längsten Laufzeiten. Es ist schwierig die vielen Parameter einzustellen.
- FKM und KK haben ähnliche Laufzeiten und sind schneller als die anderen.
- FR ist schnell für kleine Graphen, aber für mehr als 60 Knoten und Kanten bricht er ein.

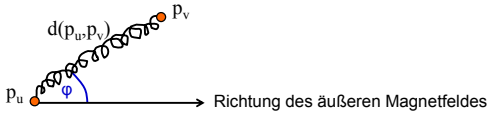
6.4 Erweiterungen

- 6.4.1 Magnetische Felder (Sugiyama & Misue 1985)
- 6.4.2 Preprocessing mit Planarisierung (Harel & Sardas 1993)
- 6.4.3 Nebenbedingungen
- 6.4.4 Constraint-Driven Layout (Marks et al. 1997)

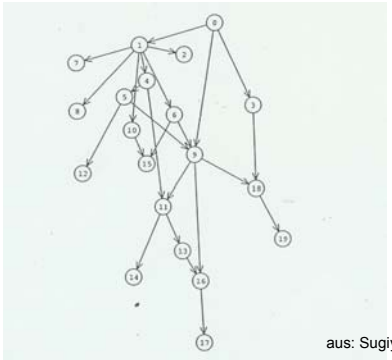
6.4.1 Magnetische Felder (Sugiyama & Misue 1985)

- **Idee:** Sorgt dafür, dass es für gerichtete Kanten eine Vorzugsrichtung gibt
- **Dazu:** Die Federn sind magnetisch und es gibt ein äußeres magnetisches Feld

- **Rotationskraft:** proportional zu $d(p_u,p_v)^{\alpha}\varphi^{\beta}$ (α,β konstant)



Beispiel



aus: Sugiyama & Misue

6.4.2 Preprocessing mit Planarisierung (Harel & Sardas 1993)

- **Idee:** Generiere ein topologisch gutes Layout (wenige Kreuzungen)

- **Phase A:** Planaritätstest
- **Phase B-:** Extrahiere planaren Untergraph
- **Phase B:** Planare Einbettung
- **Phase B+:** Wiedereinfügen der entfernten Kanten
- **Phase C:** Generiere eine planare straightline Zeichnung
- **Phase D:** Erweiterte randomisierte „Beautifikation" durch Davidson & Harel

Bemerkungen

- Wähle möglichst große Region als Außenfläche
- Verbessere dreieckige Layouts durch:
  - Solange keine Kreuzung erzeugt wird: bewege jeden Knoten in den Schwerpunkt seiner Nachbarn
- In Phase D (Davidson & Harel):
  - für planare Graphen sind keine Bewegungen erlaubt, die Kreuzungen einführen, selbst wenn diese zu Kostenreduktion führen würden
  - für nicht-planare Graphen gilt diese Beschränkung nicht

Problem: Knicke an Dummy-Knoten der Planarisierung

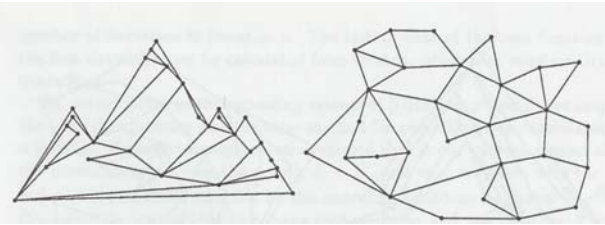
Lösung:

- **Falls** Knicke einfach (kreuzungsfrei) entfernt werden können, dann ersetze Weg durch geradlinige Kante
- **Sonst:** Für jeden Dummyknoten w und jede der zu w adjazenten beiden Kanten**:**
  - Führe zusätzliche Kosten ein: $\lambda_6 \cos(\alpha/2))^2$, wobei α der Winkel des (nicht gewünschten) Kantenknickes ist.
  - Diese Kosten sind größer, falls α nahe an 90° ist

Bewertung

- Alle planaren Graphen werden kreuzungsfrei gezeichnet (laut Aussage von Harel & Sardas)
- „Planar graphs with 50 vertices yield drawings that have a clear close-to-perfect look."
- Laufzeit: viel schneller, weil Davidson&Harel mit guter Lösung startet
- bei „fast planaren" Graphen auch fast immer gute Ergebnisse, während vorher. manchmal gut, manchmal sehr schlecht.
- Graphen mit vielen Kreuzungen → Verbesserungsbedarf
- vorheriger Versuch: Einfügen der Kanten straightline und dann Phase D → das war schlecht

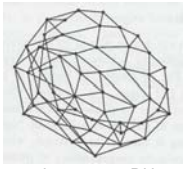
Beispiel



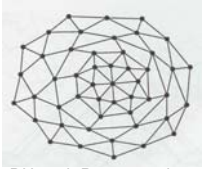
Input: geradlinige Zeichnung (ähnlich FPP)
Output: nach Optimierung

aus: Harel und Sardas 1993

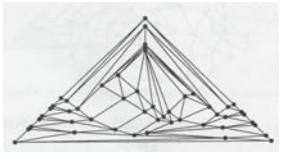
Planarer Graph



Layout von DH



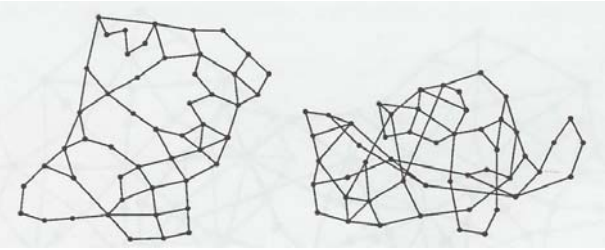
Layout von DH nach Preprocessing mit HS



Input für DH nach Preprocessing mit HS

aus: Harel und Sardas 1993

Beispiel für planaren Graphen

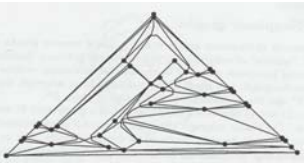


Mit Preprocessing nach DS

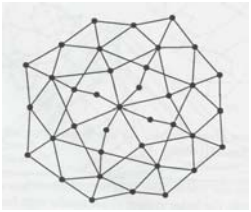
Original DH-Layout

aus: Harel und Sardas 1993

Beispiel für nicht-planaren Graphen



Input für DH nach Preprocessing mit HS



Layout von DH nach Preprocessing mit HS

aus: Harel und Sardas 1993

ENDE 29.1.

6.4.3 Nebenbedingungen

Verschiedene Varianten modellieren z.B.

- **Positionsbedingungen:** Einschränkungen des Bereichs, in dem sich Knoten bewegen dürfen
- **Feste Subgraphen:** vorgegebene Subgraphen werden bis auf Translation und Rotation durch starre Federn immer gleich gezeichnet
- **Geometrisches Clusterung:** vorgegebene Knotenmengen (Cluster) sollen nahe beieinander gezeichnet werden: Modellierung durch Hinzufügen eines künstlichen Knotens pro Cluster und anziehenden Kräfte der dazugehörigen Clusterknoten, abstoßende Kräfte zwischen verschiedenen Clusterknoten.

6.4.4 Constraint-Driven Layout (Marks et al. 1997)

Ryall, Marks und Shieber 1997

- Einführung von VOFs (Visual Organization Features):
  - horizontale Alignierung
  - vertikale Alignierung
  - Achsen- und radiale Symmetrie
  - verschiedene Form Motive, z.B. „T-shape“, „hub-shape“
  - sequentielle Platzierung (von links nach rechts, oben nach unten)

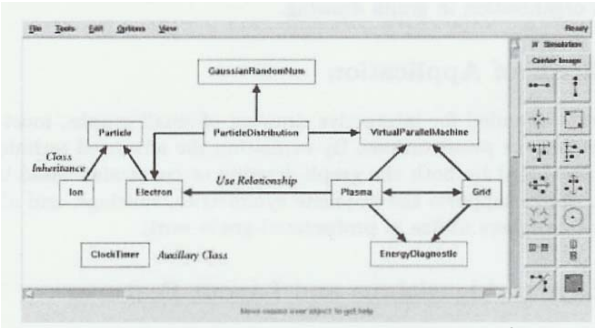
Idee: Benutzer spezifiziert VOFs, Constraints werden modelliert & optimiert

Umsetzung in GLIDE System

Lösungsverfahren (Marks et al. 1993)

- Einführung einer Energiefunktion mit verschiedenen Komponenten
- Optimierung nacheinander mit abnehmender Wichtigkeit:
  - Knotenseparierung
  - sequentielle Platzierung (von links nach rechts, oben nach unten)
  - horizontale und vertikale Alignierung
  - kreiere Motive, z.B. „T-shape“, „hub-shape“
  - forme und separiere Cluster
  - beachte Achsen- und radiale Symmetrie
  - sepaeriere Knoten von Kanten
  - reduziere Kantenkreuzungen

Screenshot aus GLIDE



aus: Ryall, Marks, Shieber 1997