Knotenpositionierung

Vorstellung eines neuen Moduls für die Knotenpositionierung für die Dritte Phase des Sugiyama Algorithmus

von: Andre Lobitz , Alina Sola , Björn Stuhrmann

Inhalt

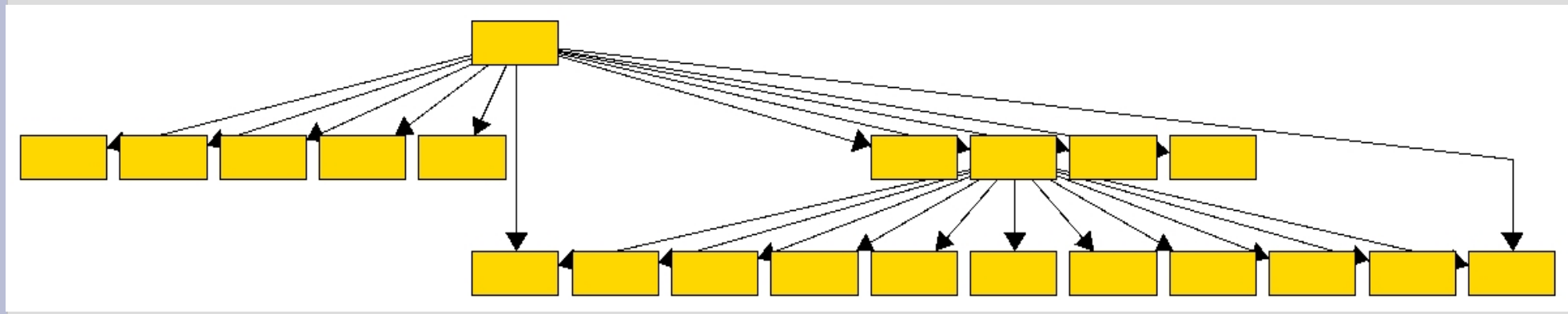
- Aufgabenstellung
- Nachteile von **`FastHierarchyLayout`**
- Vorstellung von **`SmoothedLayout`**
- Evaluierungskriterien
- Testergebnisse (incl. Beispielgraphen)

Aufgabenstellung

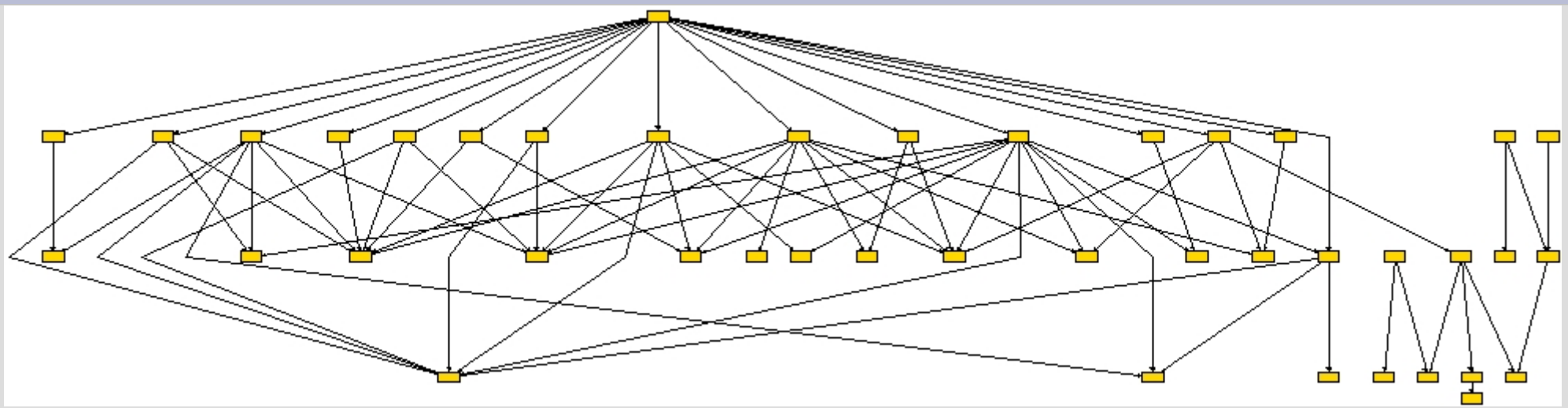
- Entwerfen und Implementieren eines Algorithmus zur Knotenpositionierung
- Evaluation des Verfahrens im Vergleich zu einem in OGDF implementierten Verfahrens (**`FastHierarchyLayout`**) anhand der AT&T-Graphen und selbst erzeugter (zufälliger) Graphen

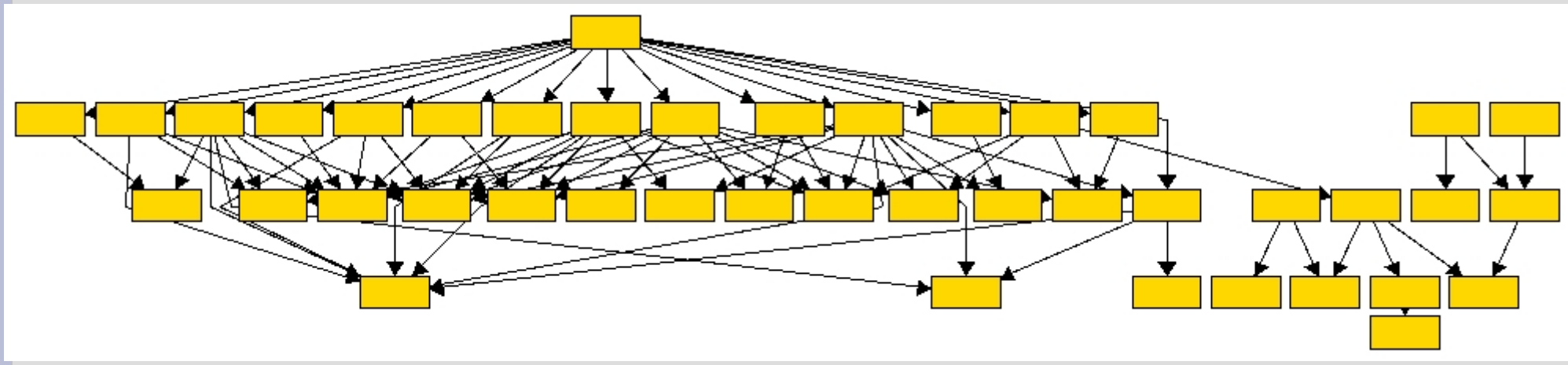
Nachteile von bisherigen Verfahren wie FastHierarchyLayout

- Bei falscher Wahl der Parameter **`m_minNodeDist`** und **`m_minLayerDist`** können Kanten durch Knoten überdeckt werden.

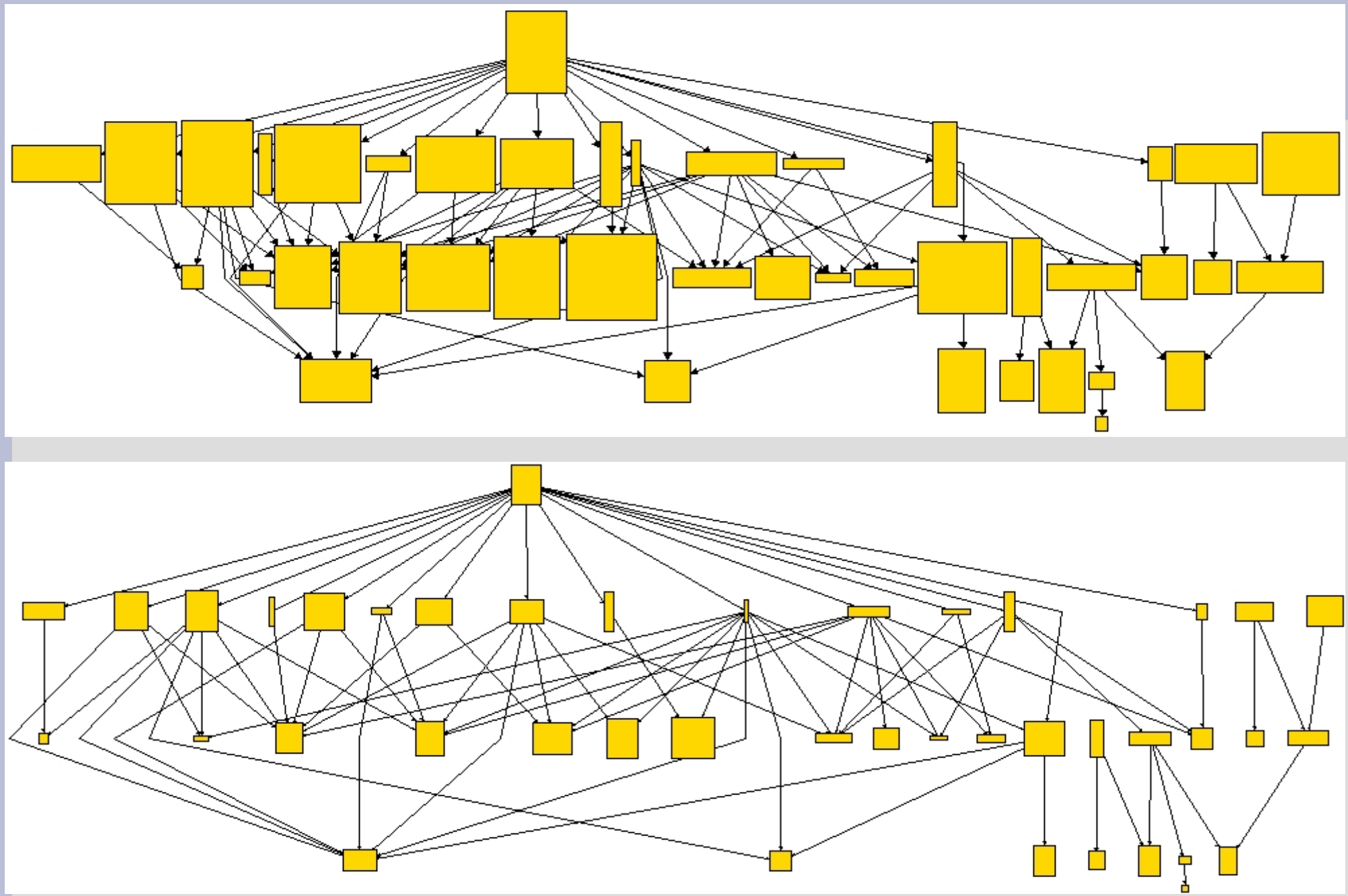


Beispiel (1)

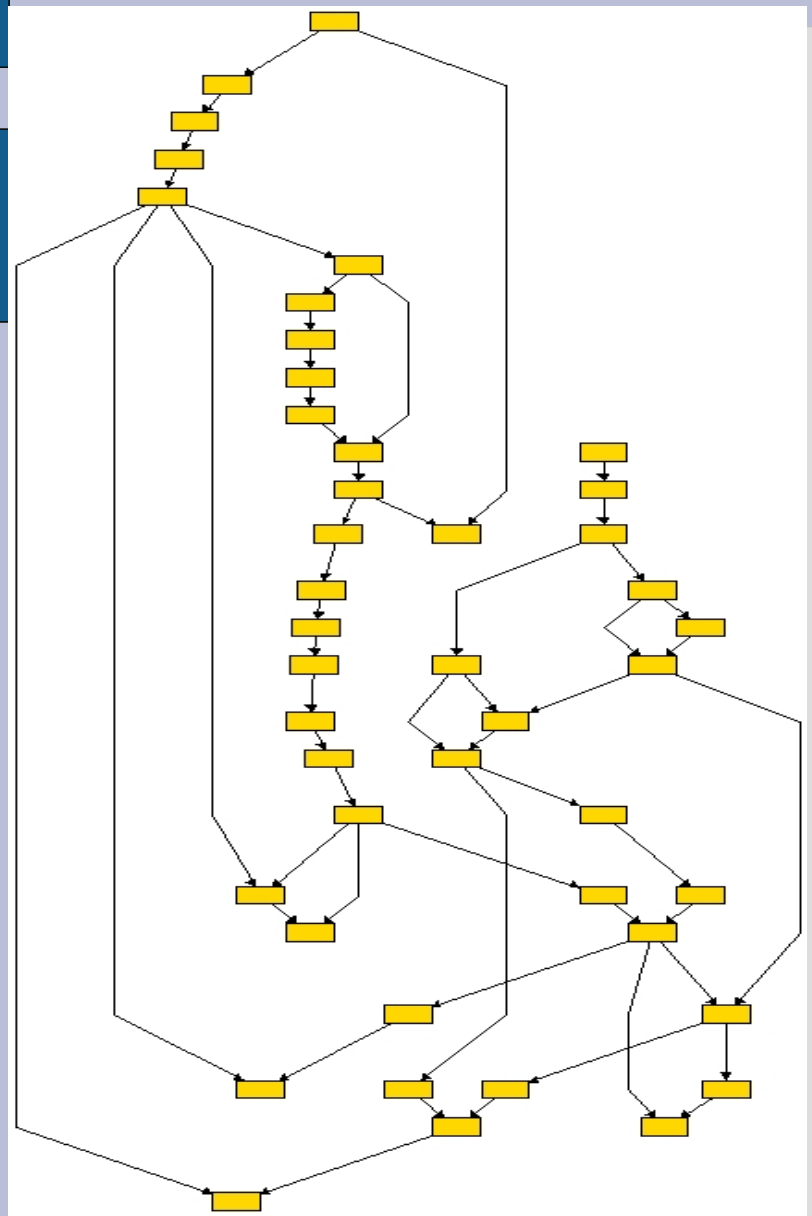




Beispiel (2)



Beispiel (3)



- Manchmal sehen „abgerundete“ Kanten besser aus als „eckige“ Kanten.
- Häufig können „lange Kanten“ begradigt werden.

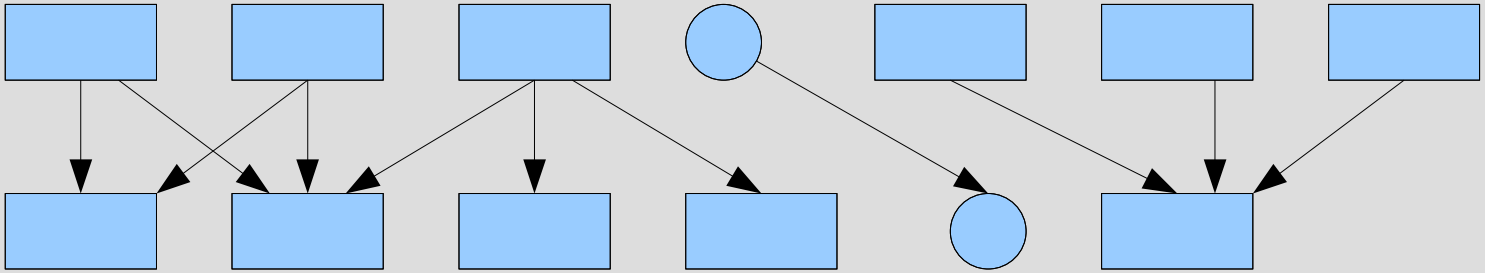
SmoothedLayout

1. Berechnung der minimalen Breite (mB) für jede Schicht.
2. Positionierung der Knoten von links nach rechts, angefangen mit der Schicht mit maximalen mB-Wert. Wahl der nächsten Nachbarschicht mit höherem mB-Wert.
3. Positionierung der Knoten von rechts nach links (analog zu 2.).
4. Berechnung des Mittelwertes von x-Koordinate der beiden Positionierungen.
5. Zuweisung der y-Koordinate (durch Berechnung von Winkeln) zur Vermeidung der Kantenverdeckung .
6. Begradigung von Kanten und
7. Einfügen zusätzlicher Punkte (nach Beendigung von Sugiyama – da unmöglich neue virtuelle Knoten in Dritter Phase hinzuzufügen werden können).

Beispiel (1)

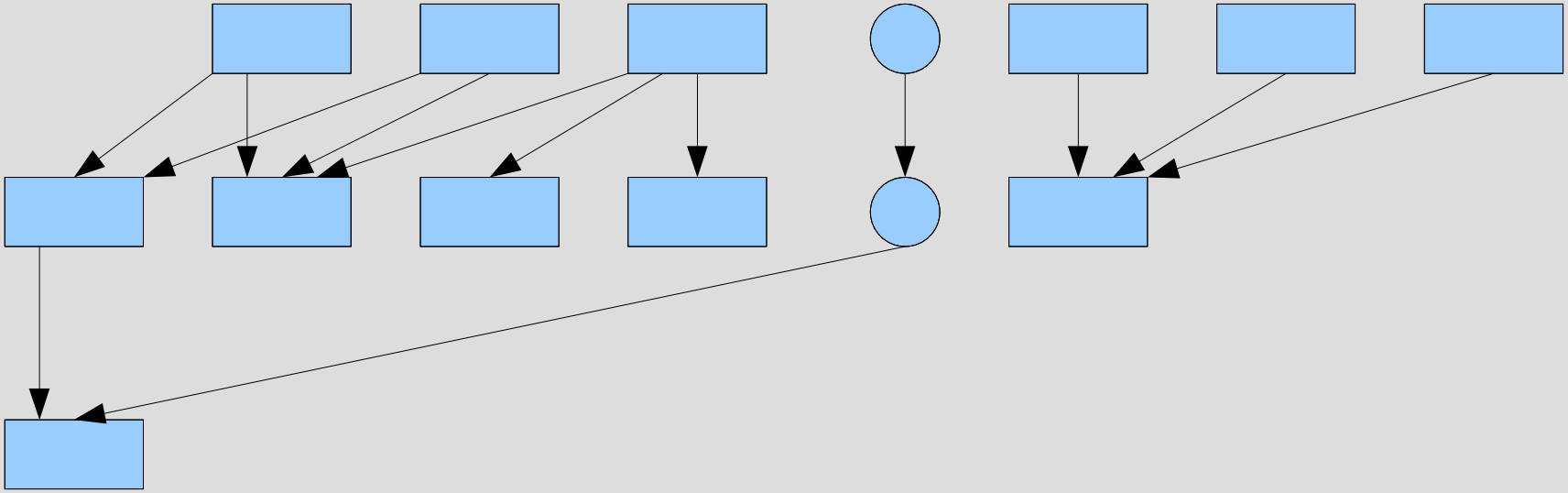
- Platzieren der Knoten mit maximalem mB-Wert

Beispiel (2)



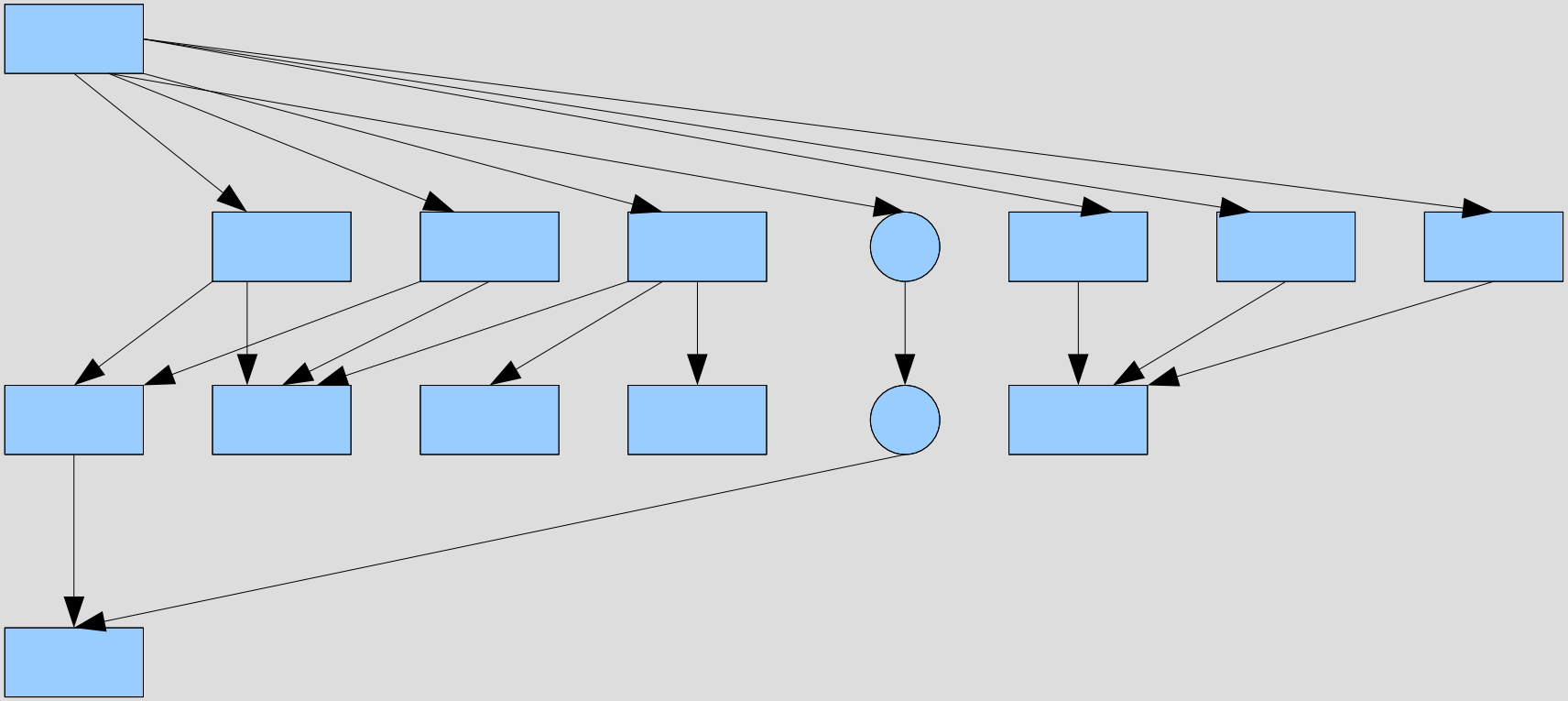
- Platzieren der Knoten der benachbarten Schicht mit größerem mB-Wert

Beispiel (3)



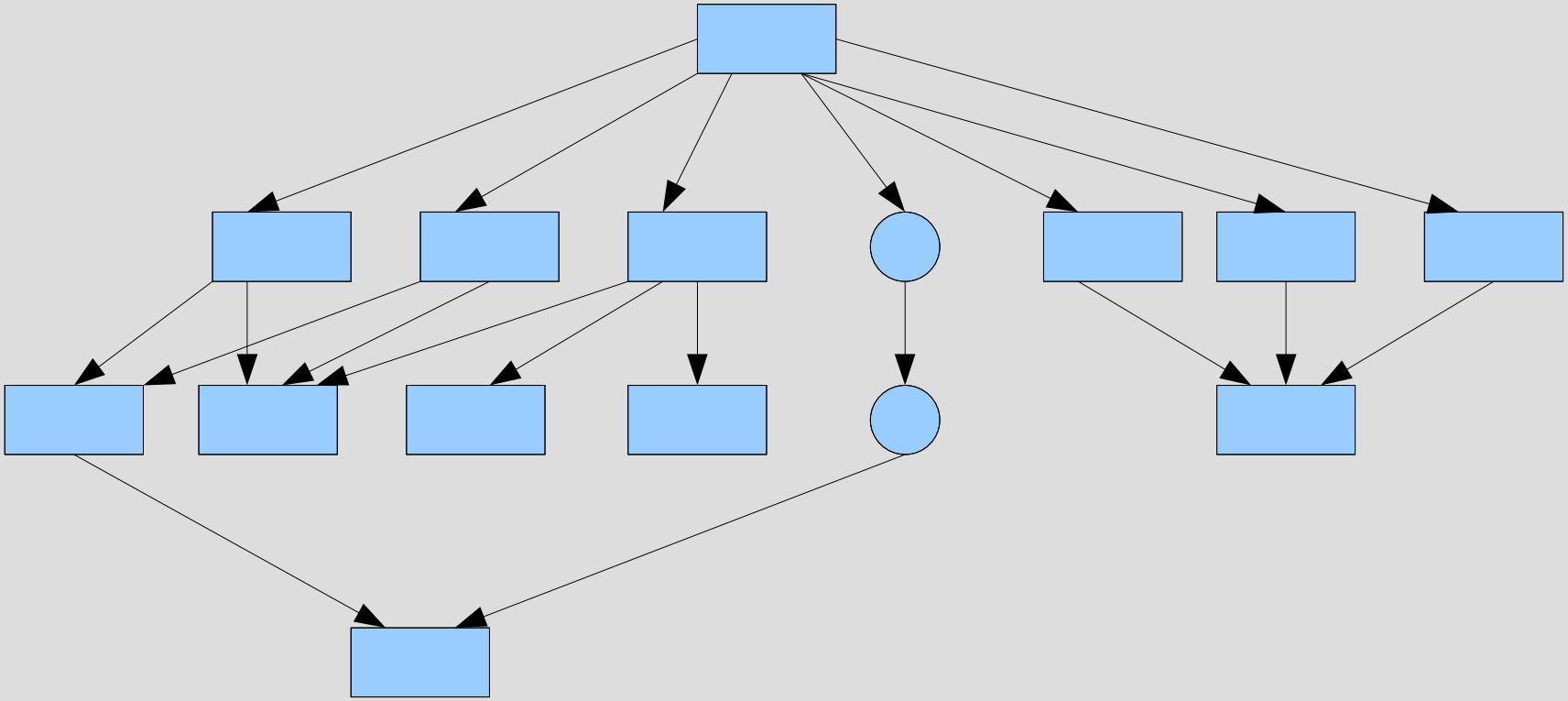
- Verschiebung der oberen Schicht,um virtuelle Knoten untereinander zu bekomen
- Platzierung der Knoten der nächsten Schicht.

Beispiel (4)



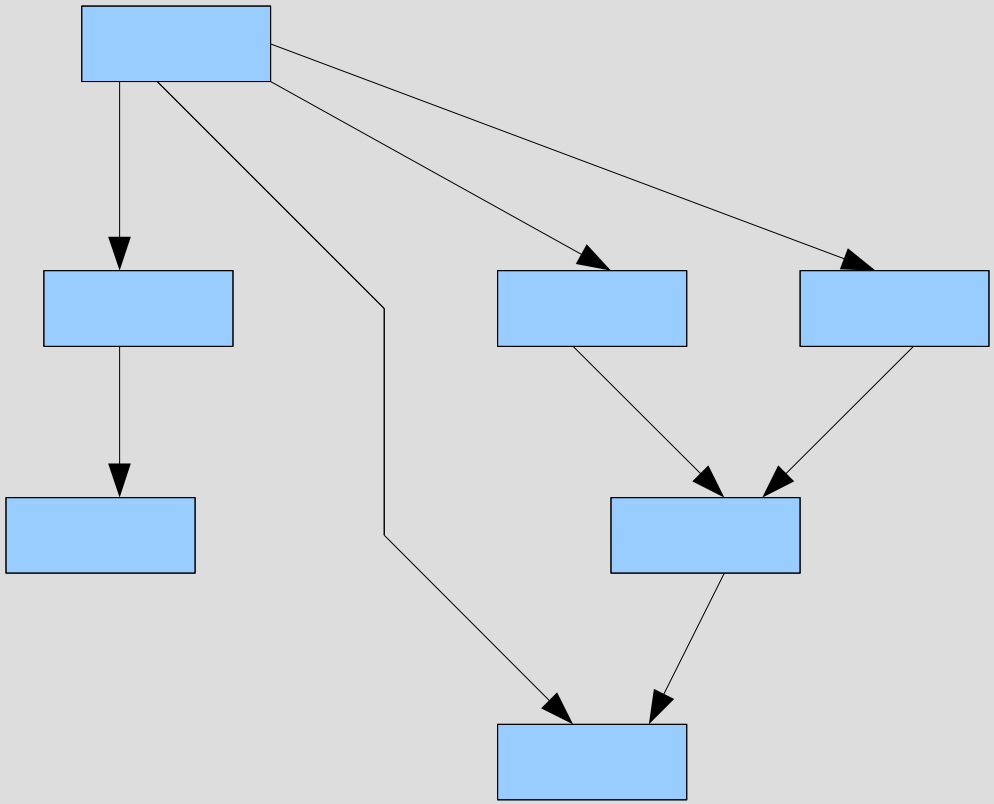
- Einfügen der letzten Schicht

Beispiel (5)

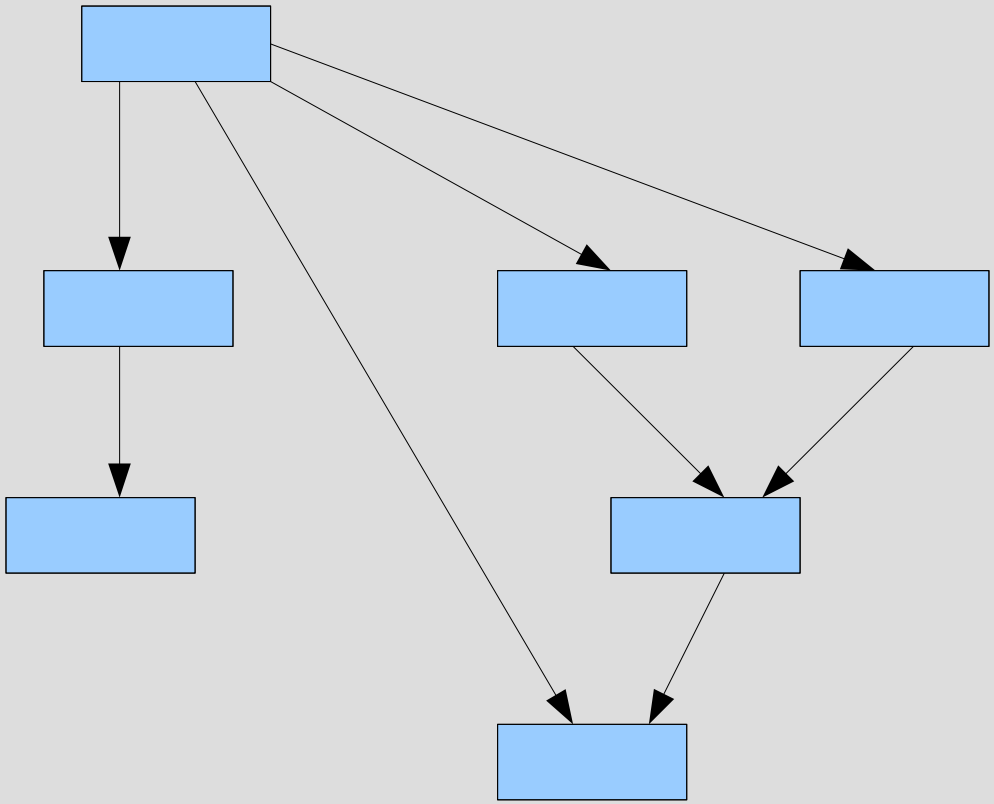


- Nach Platzierung der Knoten von rechts nach links folgt die Zuweisung des Mittelwertes.

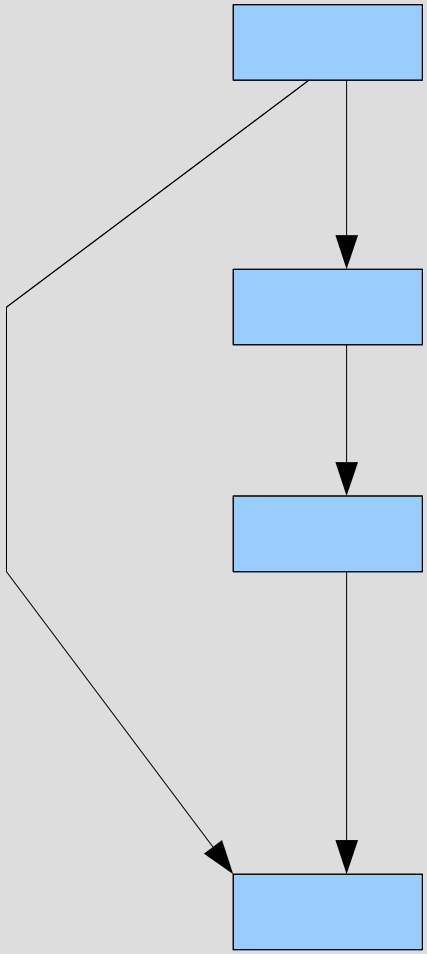
Glättung der Ecken (1)



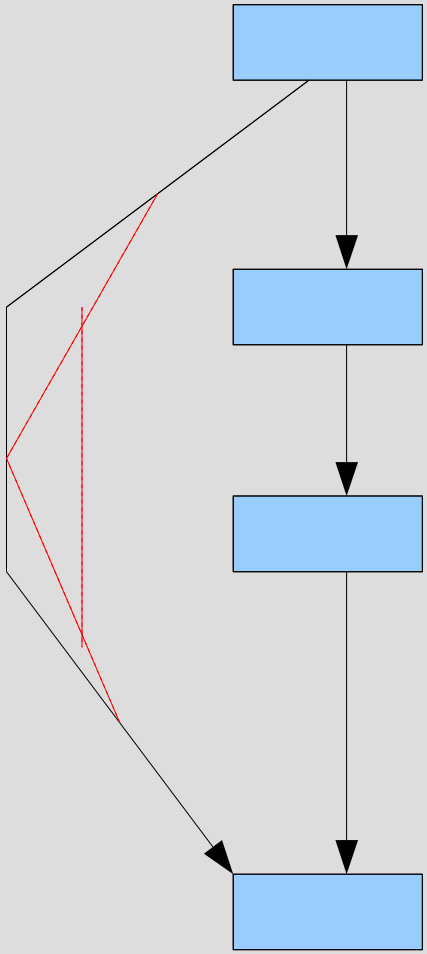
Glättung der Ecken (2)



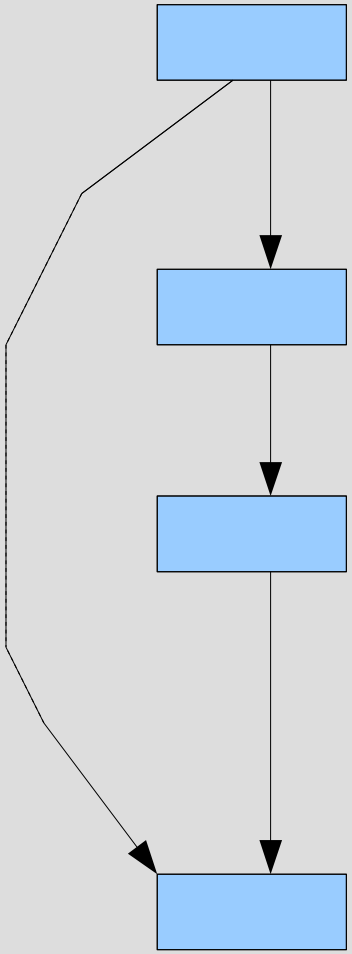
Weitere Kantenknicke/ optisches Abrunden (1)



Weitere Kantenknicke/ optisches Abrunden (2)



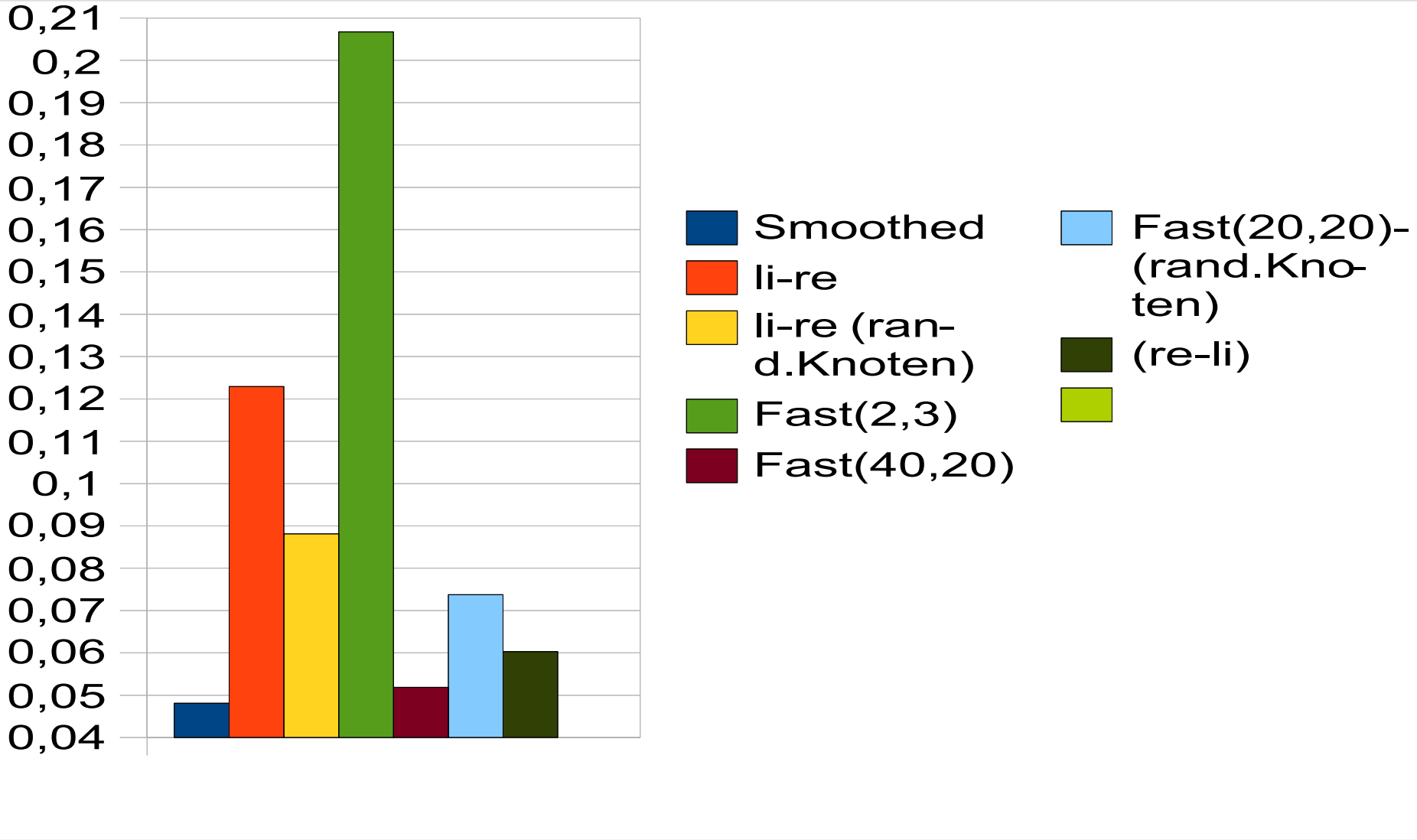
Weitere Kantenknicke/ optisches Abrunden (3)



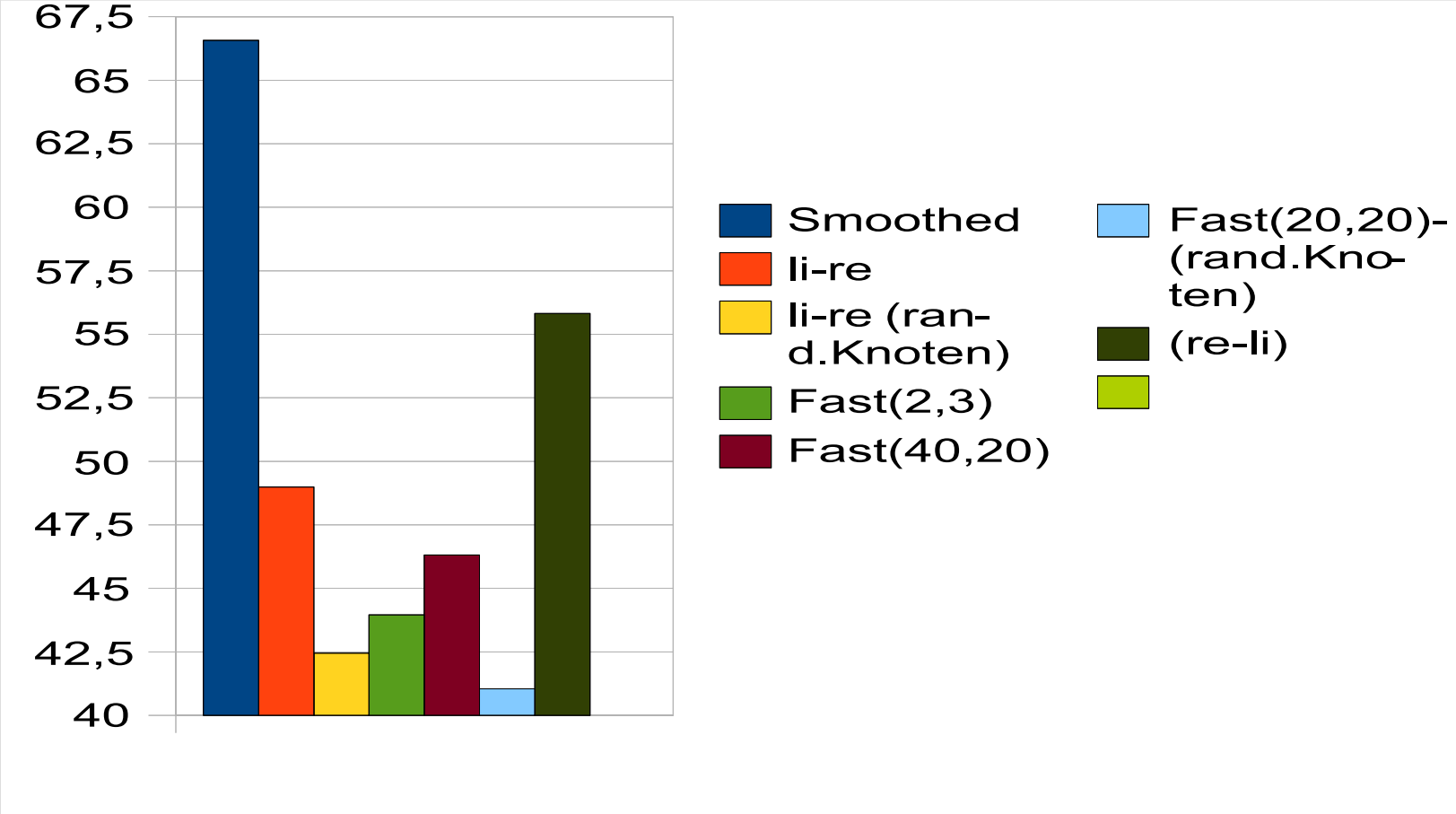
Evaluation

- Kreuzungsminimierungsheuristik: Barycenter
- Testgraphen: Vor allem AT&T-Graphen
- Qualitätsmerkmale:
  - Verhältnis Knotenfläche/Zeichenfläche
  - Gesamtlänge der Kanten
  - Anzahl der teilweise verdeckter Kanten
  - Bewertung der Winkel zwischen y-Achse und Kante (mult. mit Anzahl konsektuiver Richtungsänderungen).

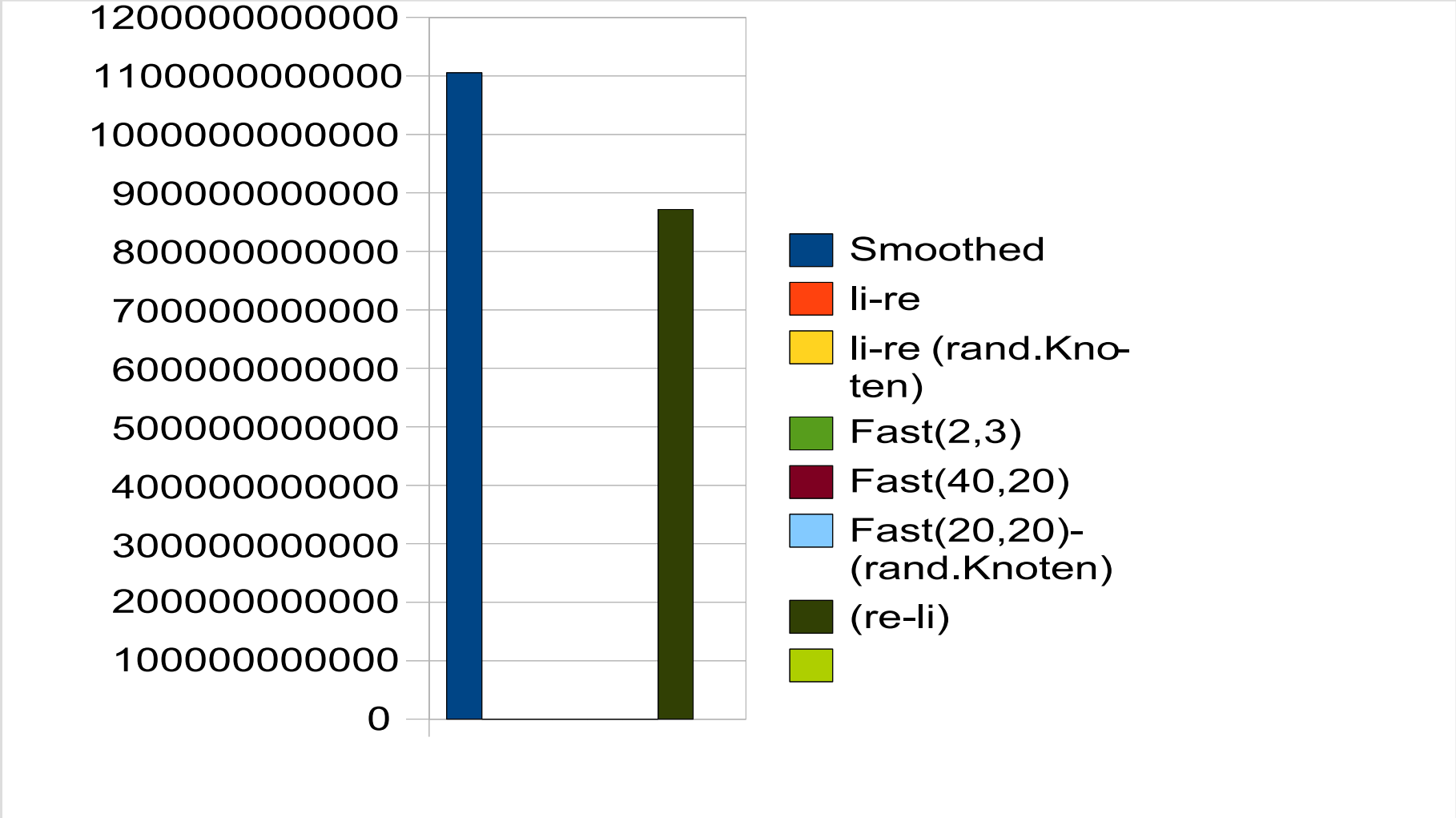
Flächenverhältnis (Knoten/gesamt)



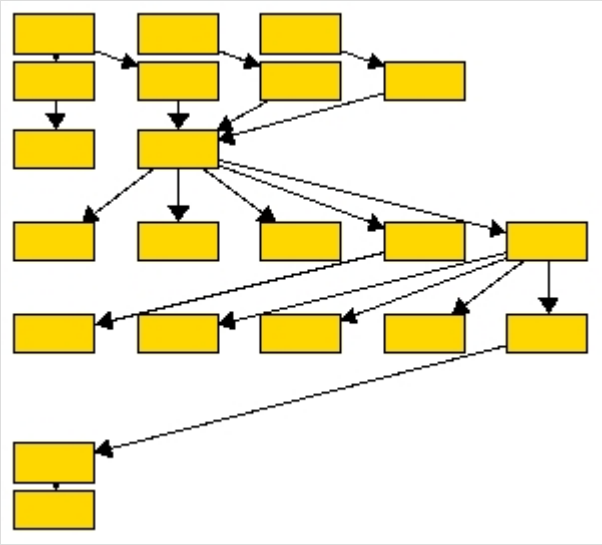
Penalty


67,5
65
62,5
60
57,5
55
52,5
50
47,5
45
42,5
40
Smoothed
li-re
li-re (ran-
d.Knoten)
Fast(2,3)
Fast(40,20)
Fast(20,20)-
(rand.Kno-
ten)
(re-li)

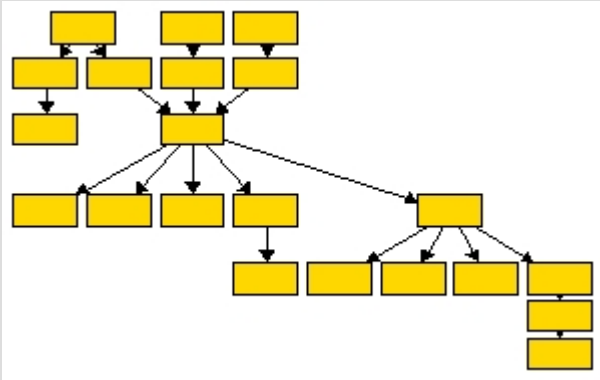
Kantenlänge



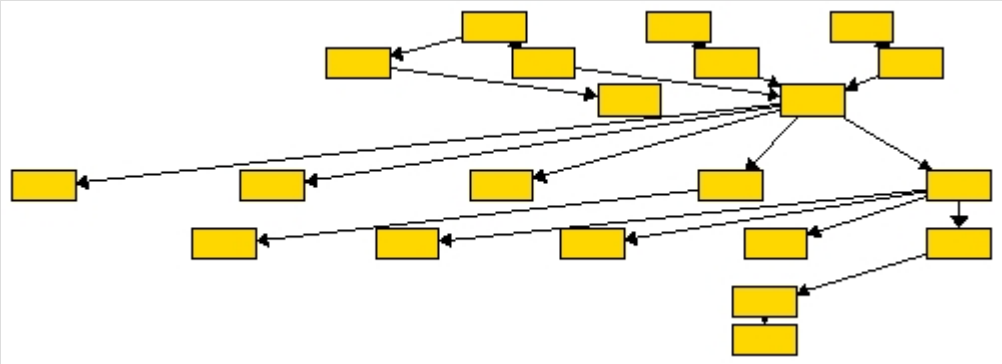
Anzahl teilweise überdeckter Kanten



links/rechts (Smoothed
ohne rechts/links Phase)

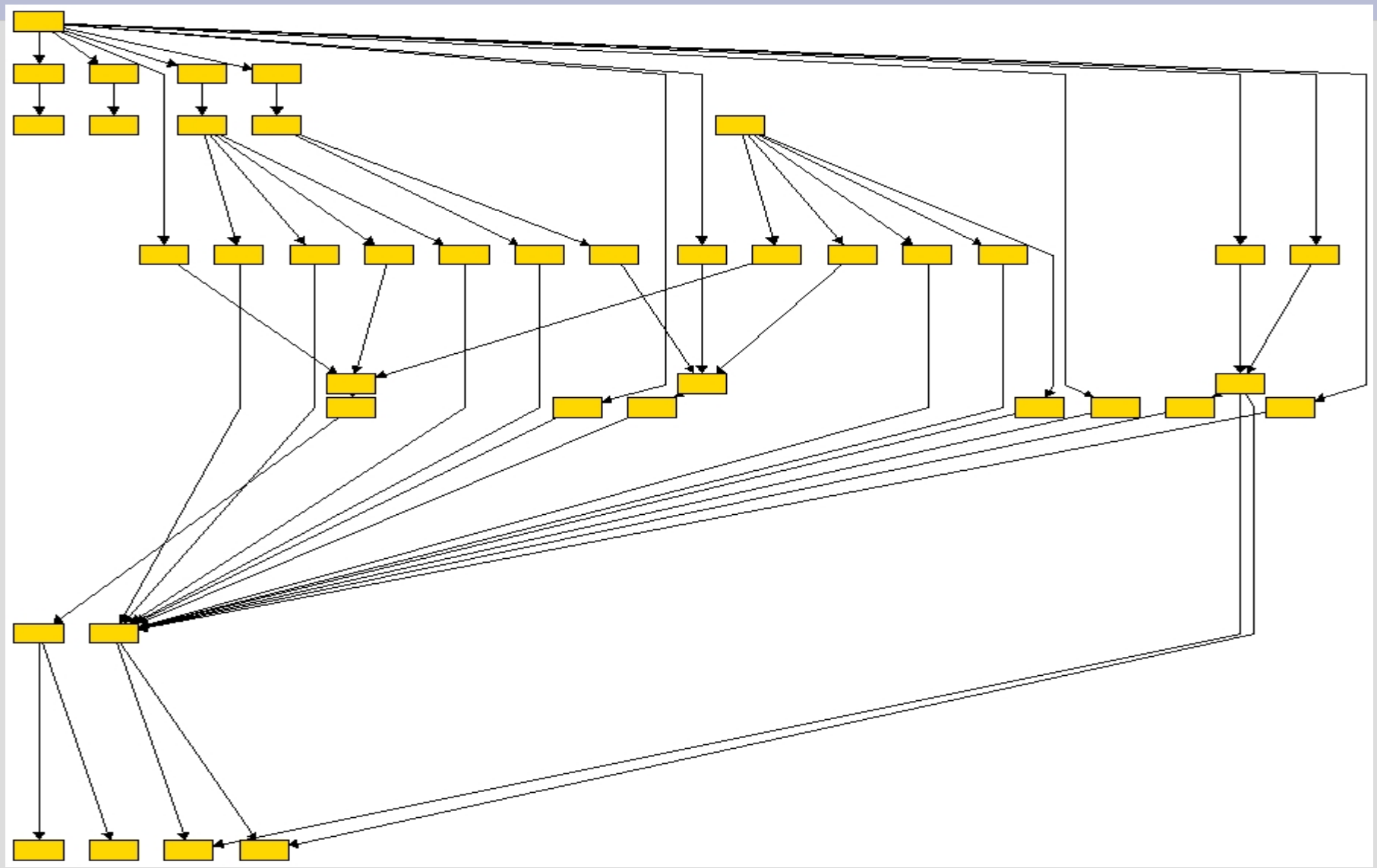


FastHierarchy

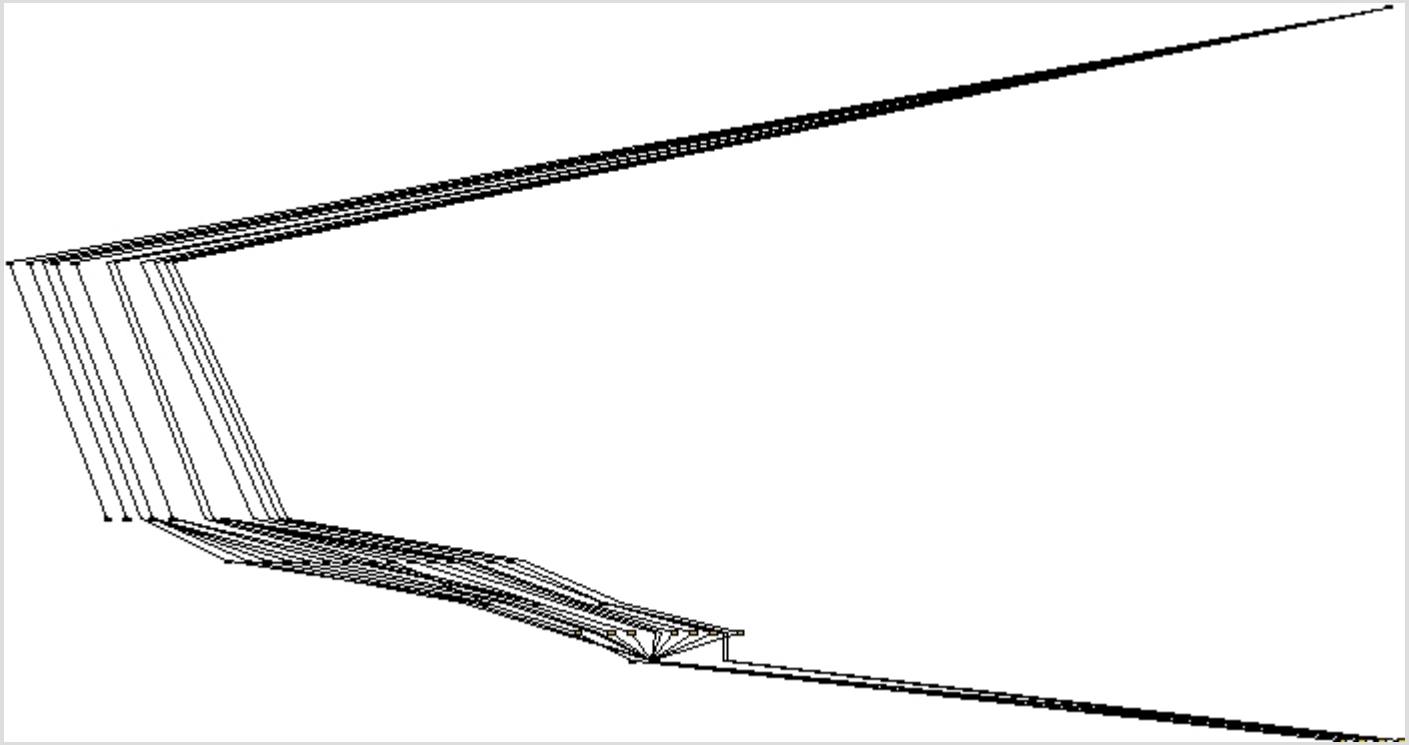


Smoothed

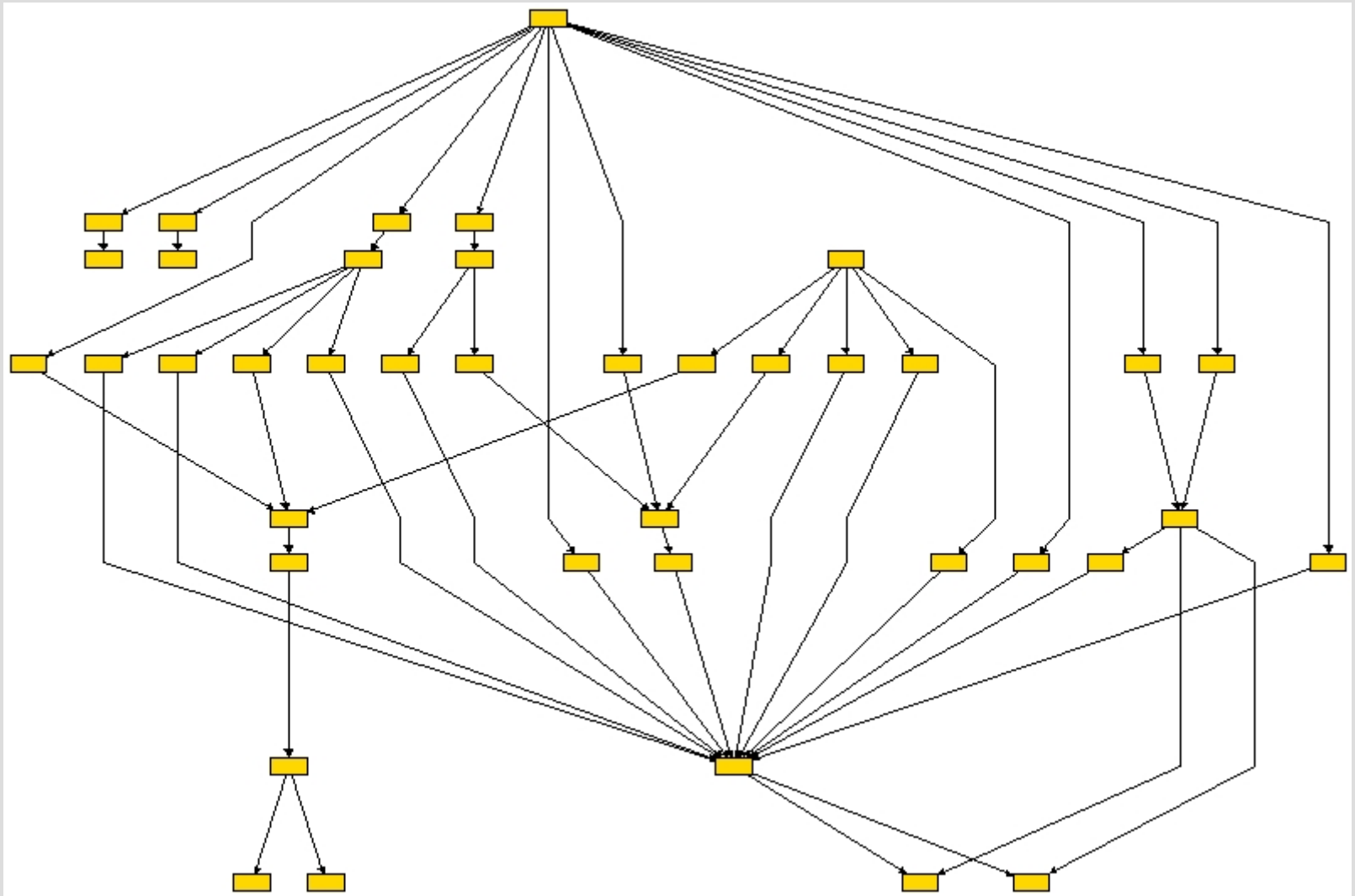
g-40-5 (links/rechts Platzierung)



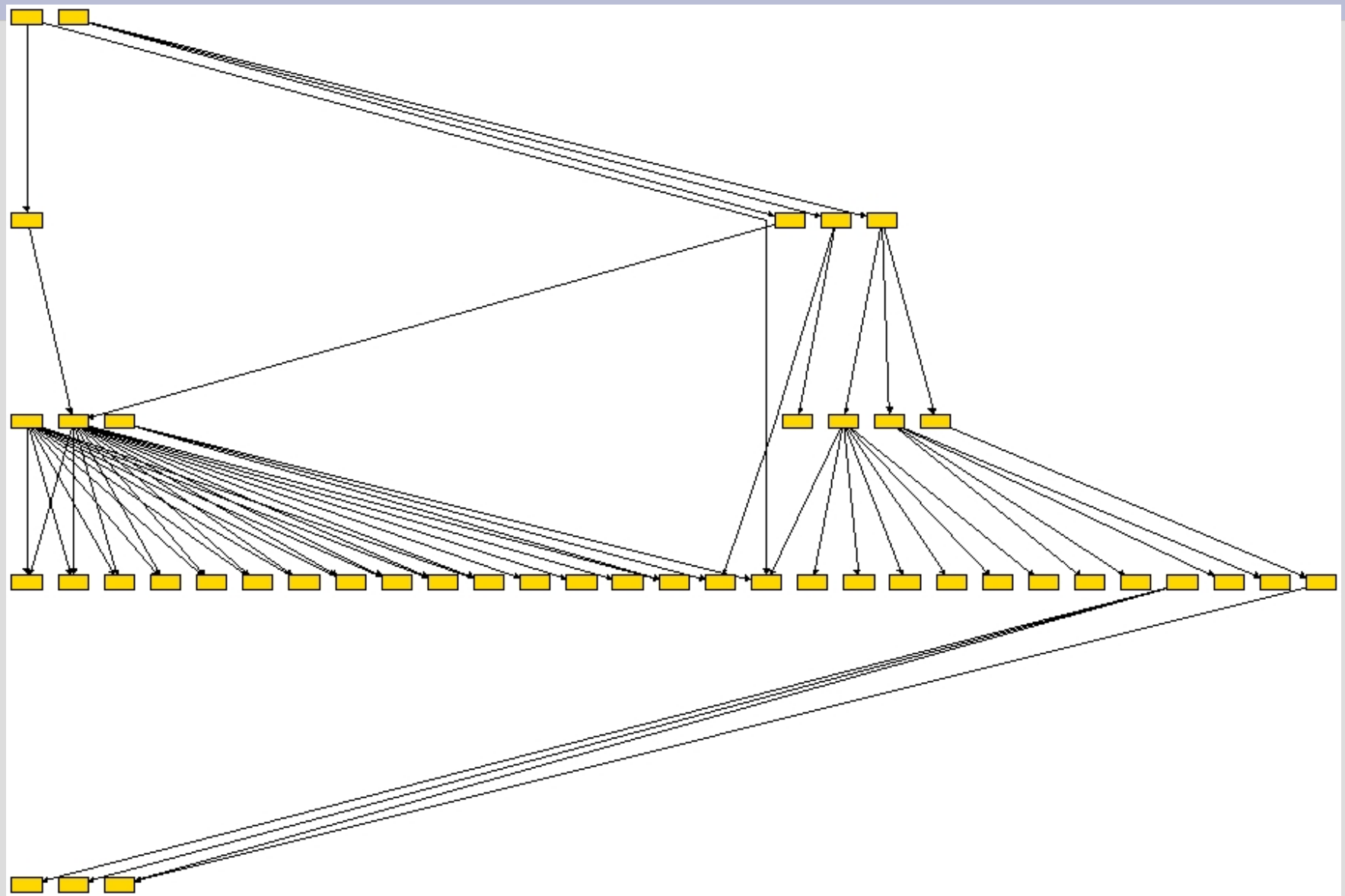
g-40-5 (Smoothed)



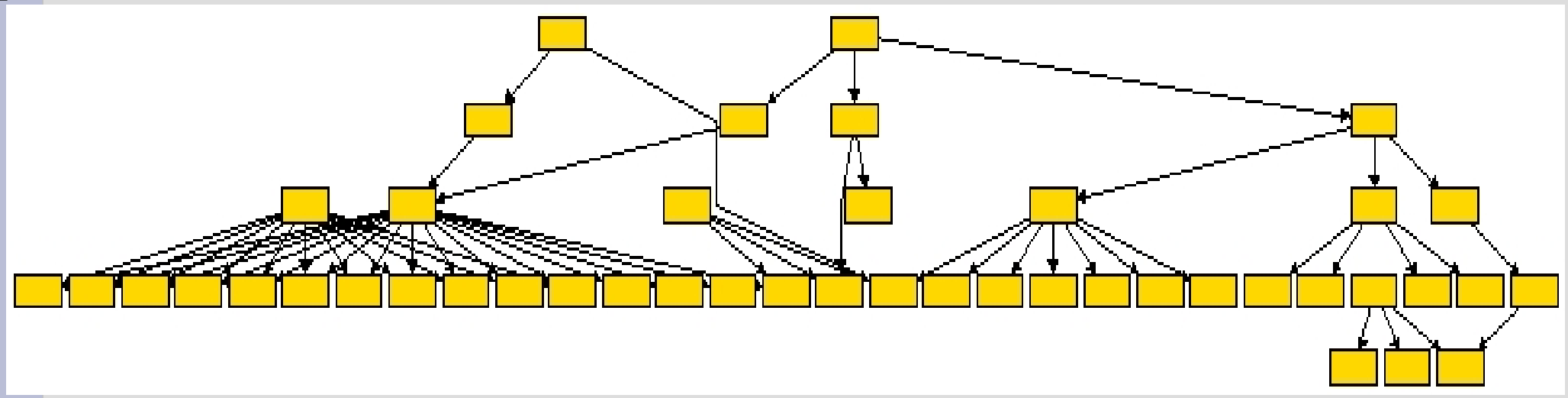
g-40-5 (FastHierarchy)

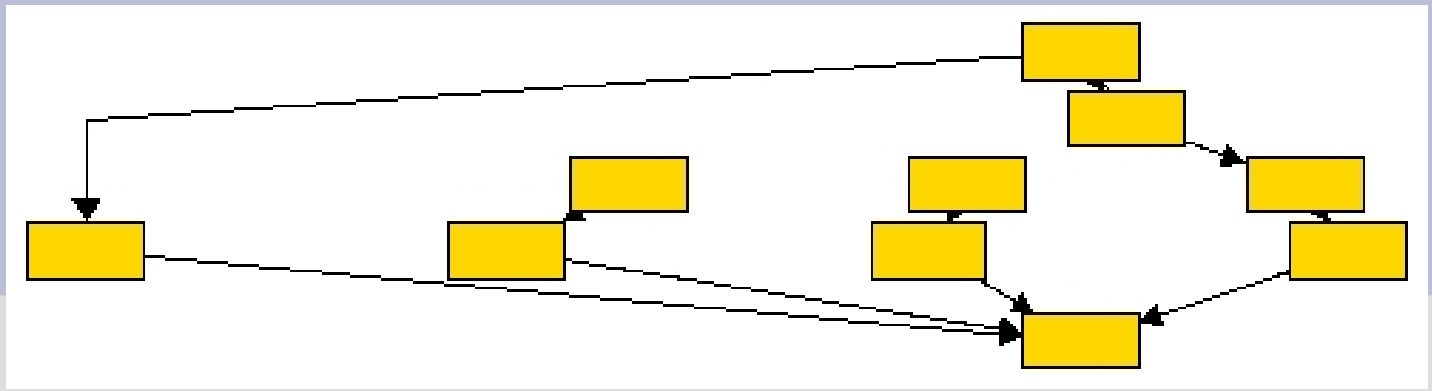


links/rechts Positionierung



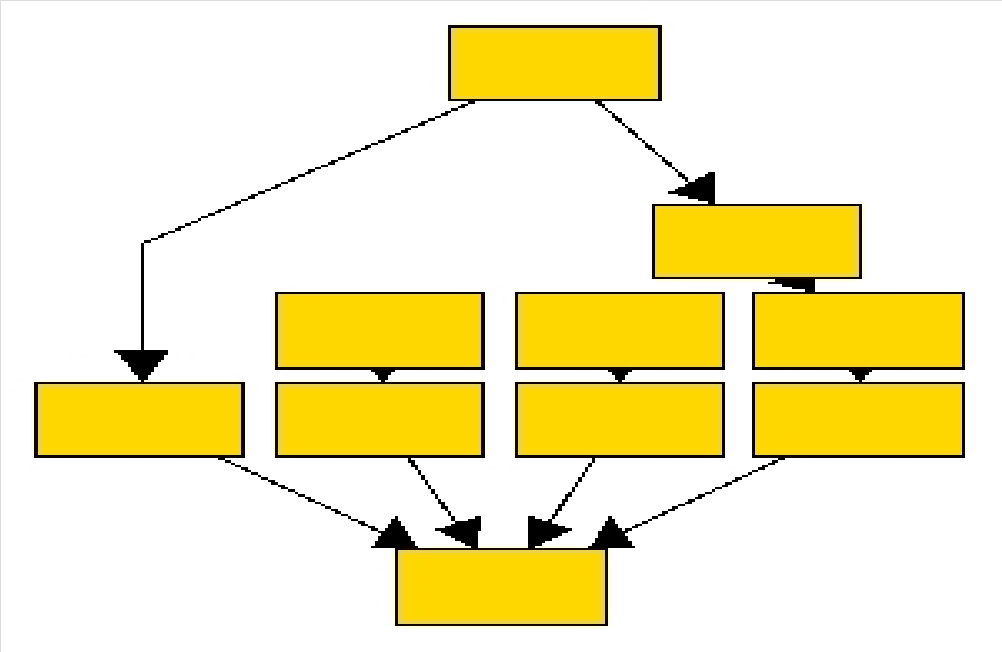
FastHierarchy



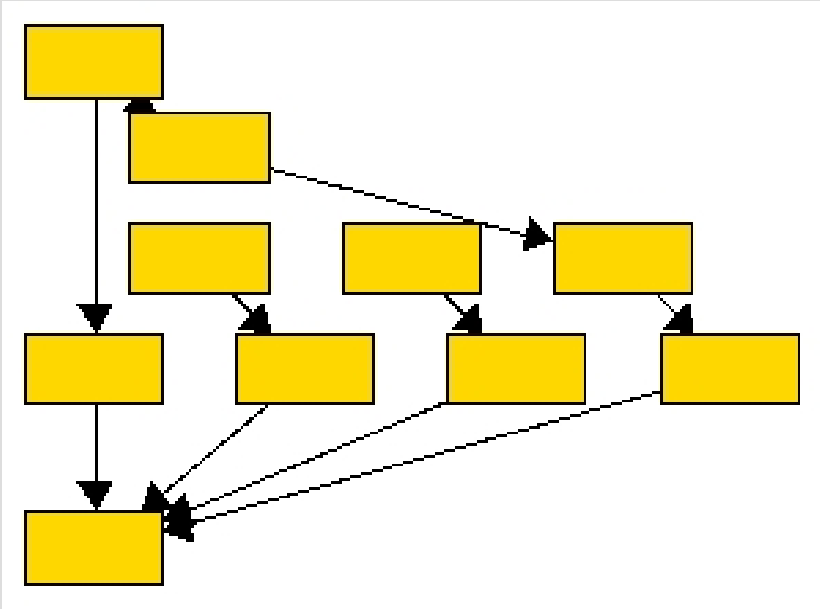


Smoothed

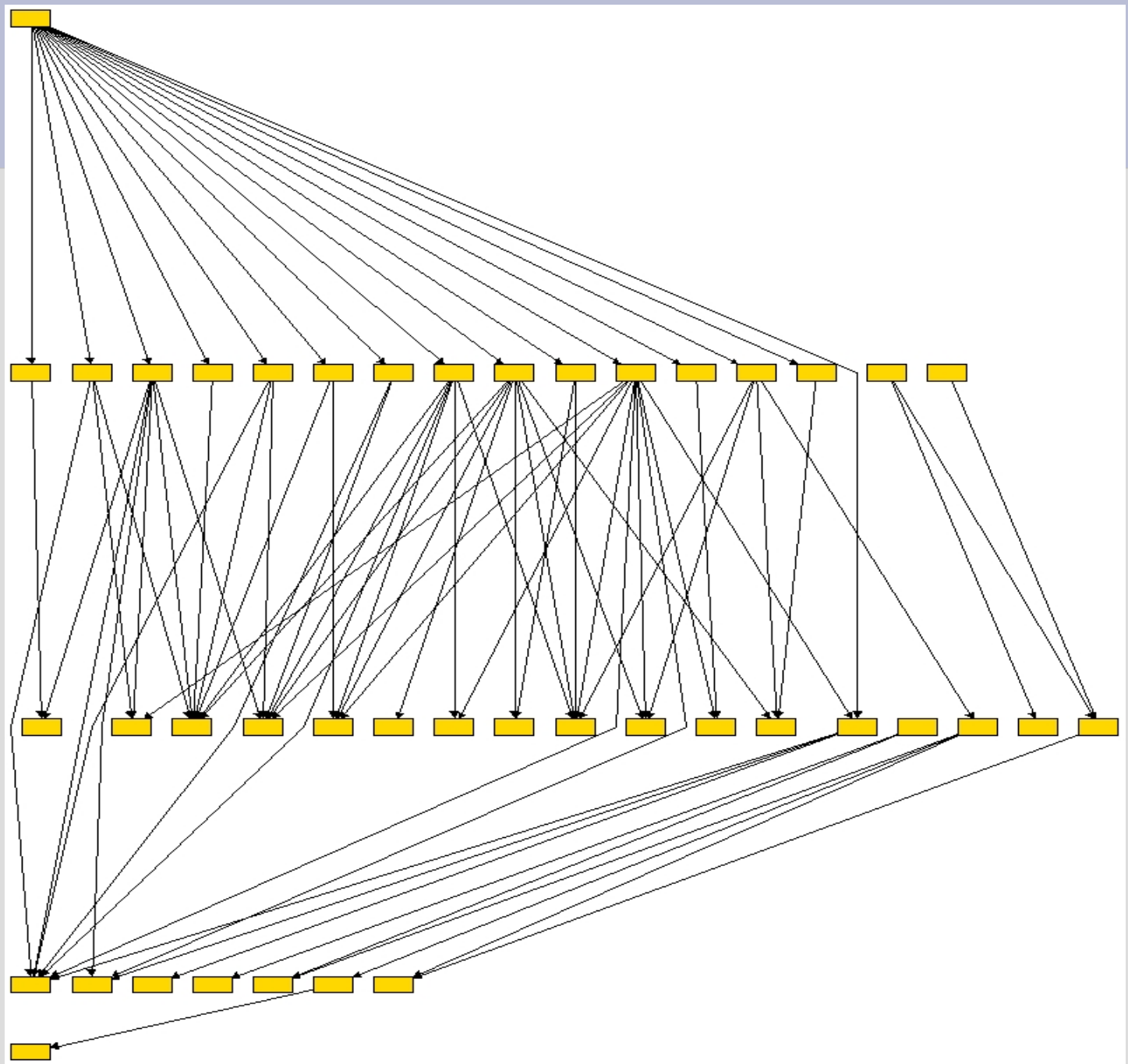
FastHierarchy



links/rechts



llinks/rechts



Fazit:

- **`FastHierarchyLayout`** erstellt viel bessere Layouts als unser Algorithmus
- **`SmoothedLayout`** ist noch Fehlerhaft:
  - Bugs im Code
  - Kanten allerdings immer gut sichtbar
  - Probleme beim platzieren von Folgen virtueller Knoten wenn diese sich auf einer Schicht Kreuzen.