Sugiyama für grosse Graphen

Martin Gronemann
Bernd Zey

Übersicht

- Probleme bei grossen Graphen
- Ideen
- Beispiel
- Experimentelle Ergebnisse

Probleme bei grossen Graphen

Im Worst Case:

- für jede Kante O(|V|) Dummy-Knoten
- O(|V|*|E|*log|E|) Laufzeit
- O(|V|*|E|) Speicherplatz

$\Rightarrow$ Anzahl der Dummy-Knoten reduzieren

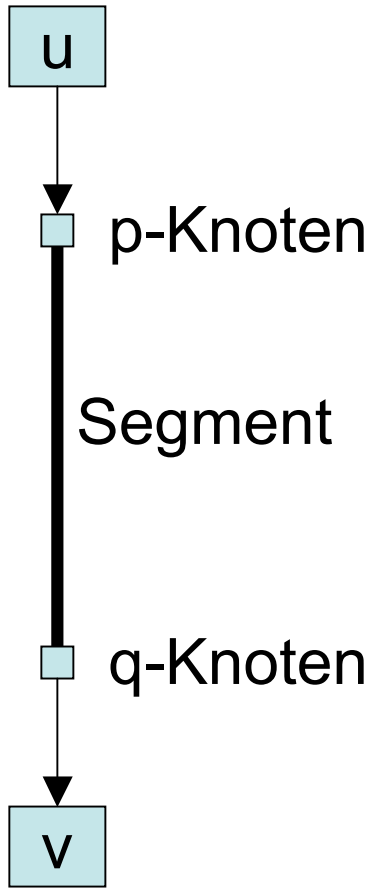
Idee

- Sugiyama-Gerüst beibehalten
- Dummy-Knoten werden nicht auf jeder Schicht benötigt
- Füge für jede Kante nur konstant viele Dummy-Knoten ein

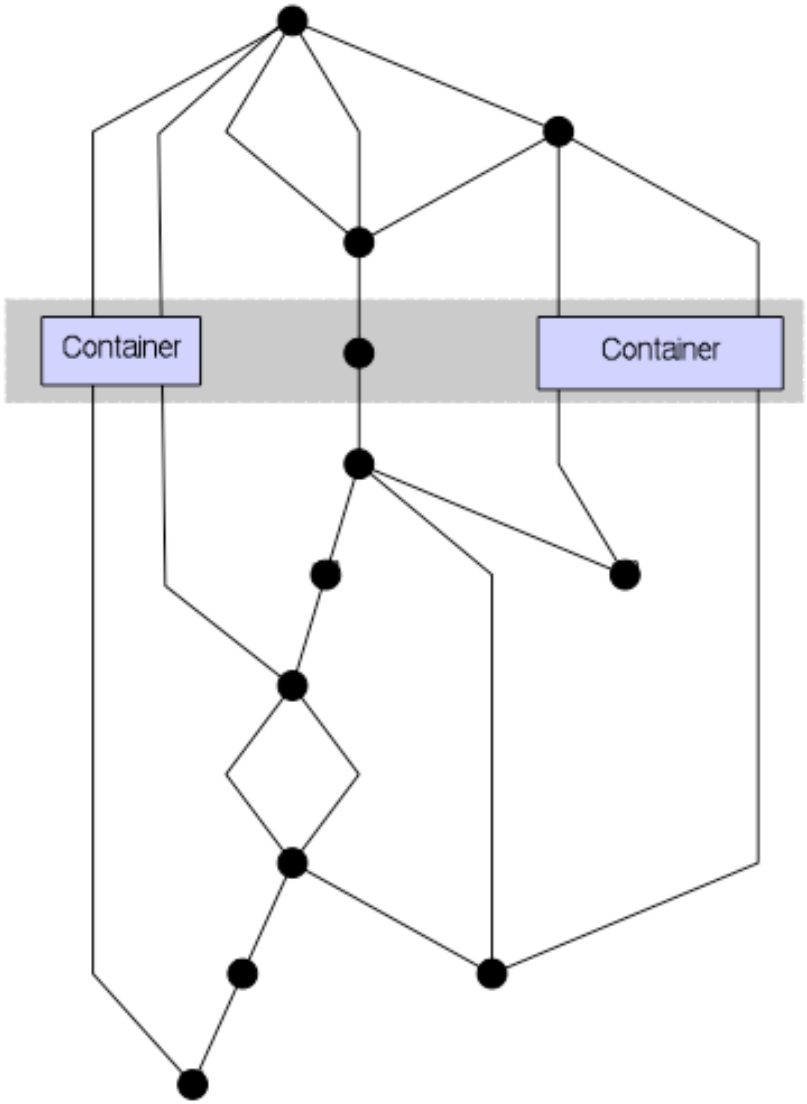
„ O(|V|*|E|) wird zu O(|V|+|E|) “

Lange Kanten

- Lange Kanten werden in 3 Teile zerlegt
  - (u,p) Kante
  - Segment
  - (q,v) Kante
- Pro Kante 2 Dummy-Knoten und 2 neue Kanten



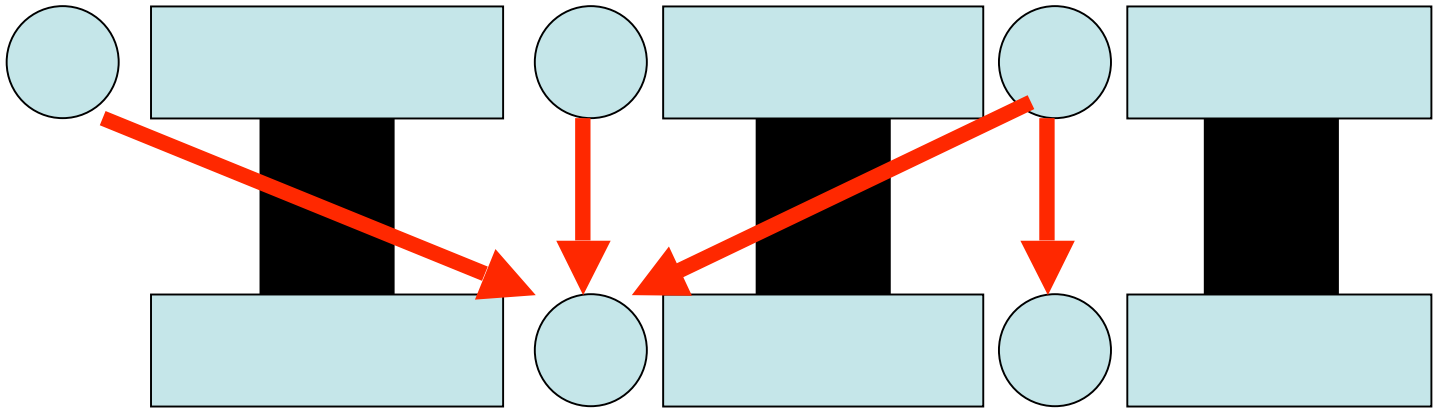
Container



- Zusammenfassen der Segmente auf jeder Schicht zu Bündeln (sog. Container)
- DS mit effizienten Split-/Join-Operationen

Alternierende Schicht

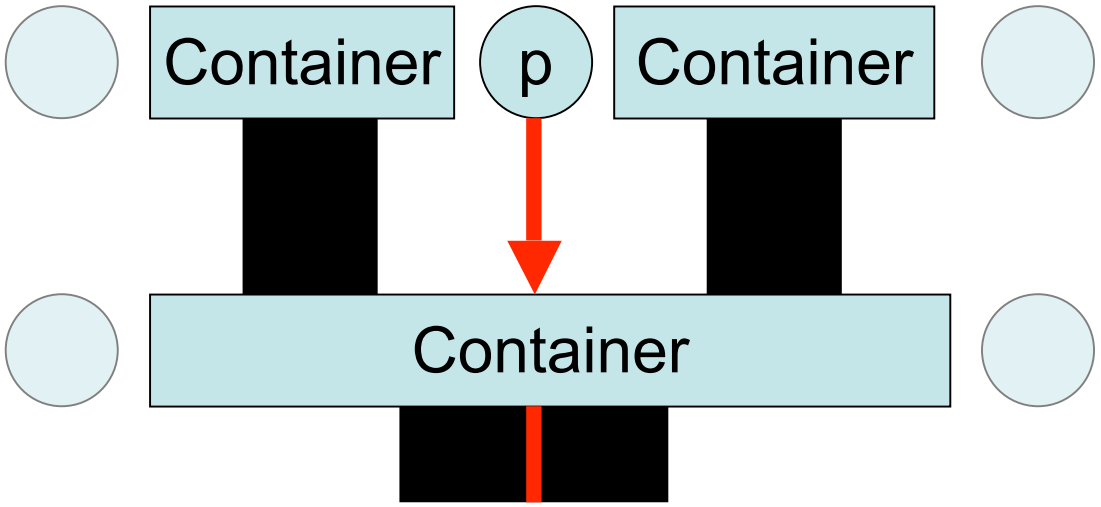
Idee: jede Schicht besteht abwechselnd aus einem Knoten und einem Container



Achtung: ein Container kann auch leer sein

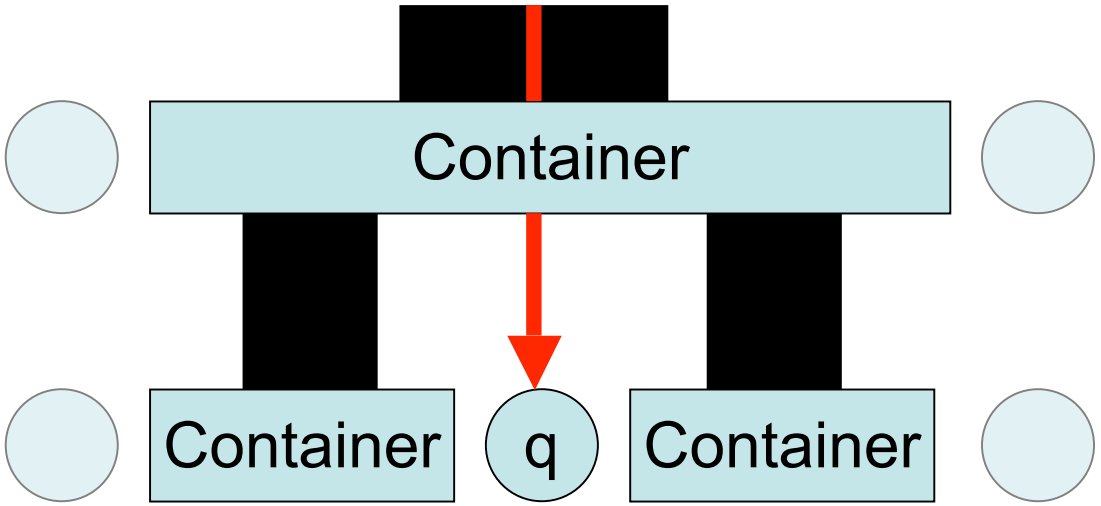
Container (2)

- Innerhalb eines Containers besitzen Segmente eine Ordnung
- 2 Container werden zusammengefasst wenn ein p Knoten auftritt



Container (3)

- Der Container wird geteilt wenn ein q Knoten auftritt

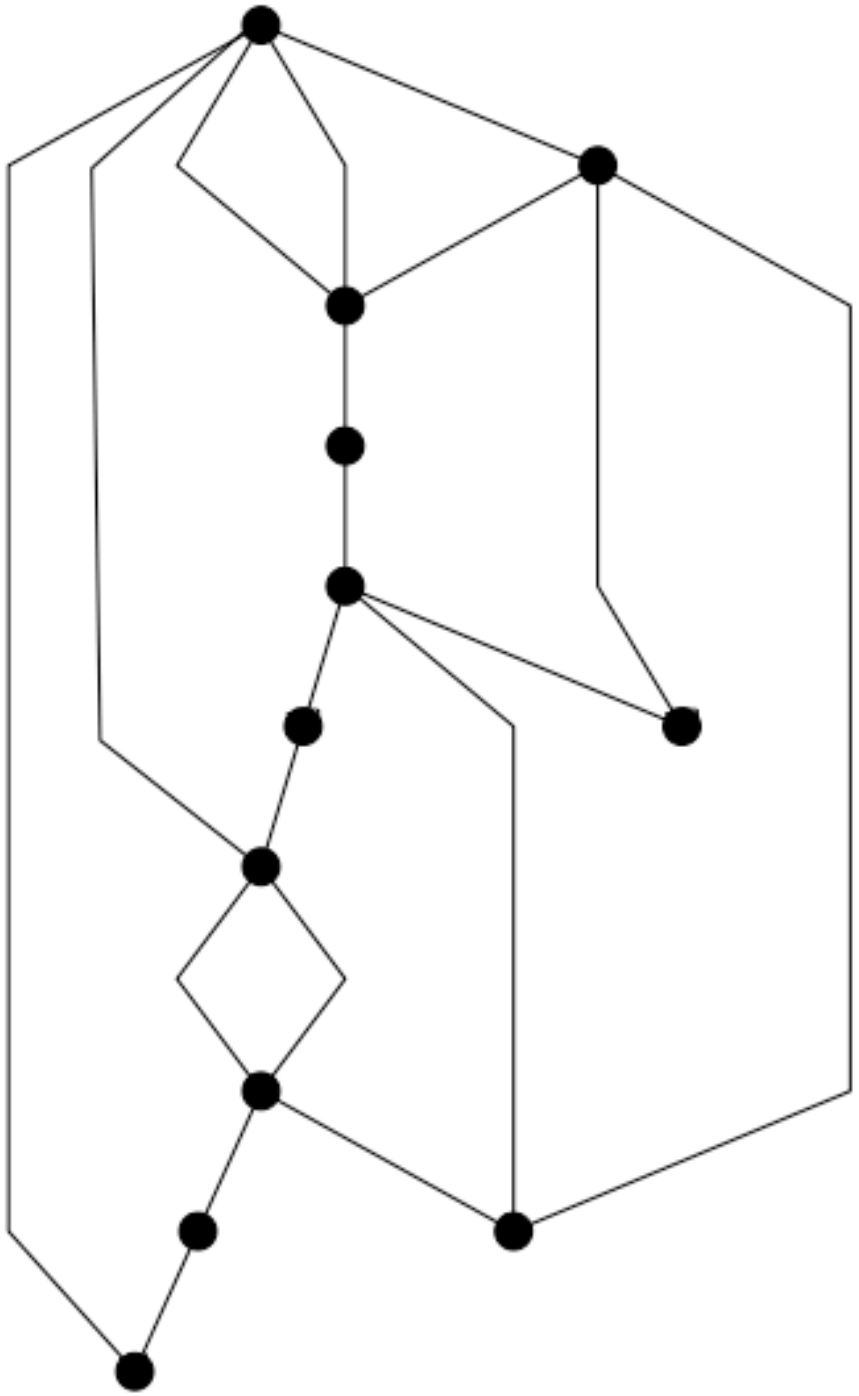


Crossing Minimization

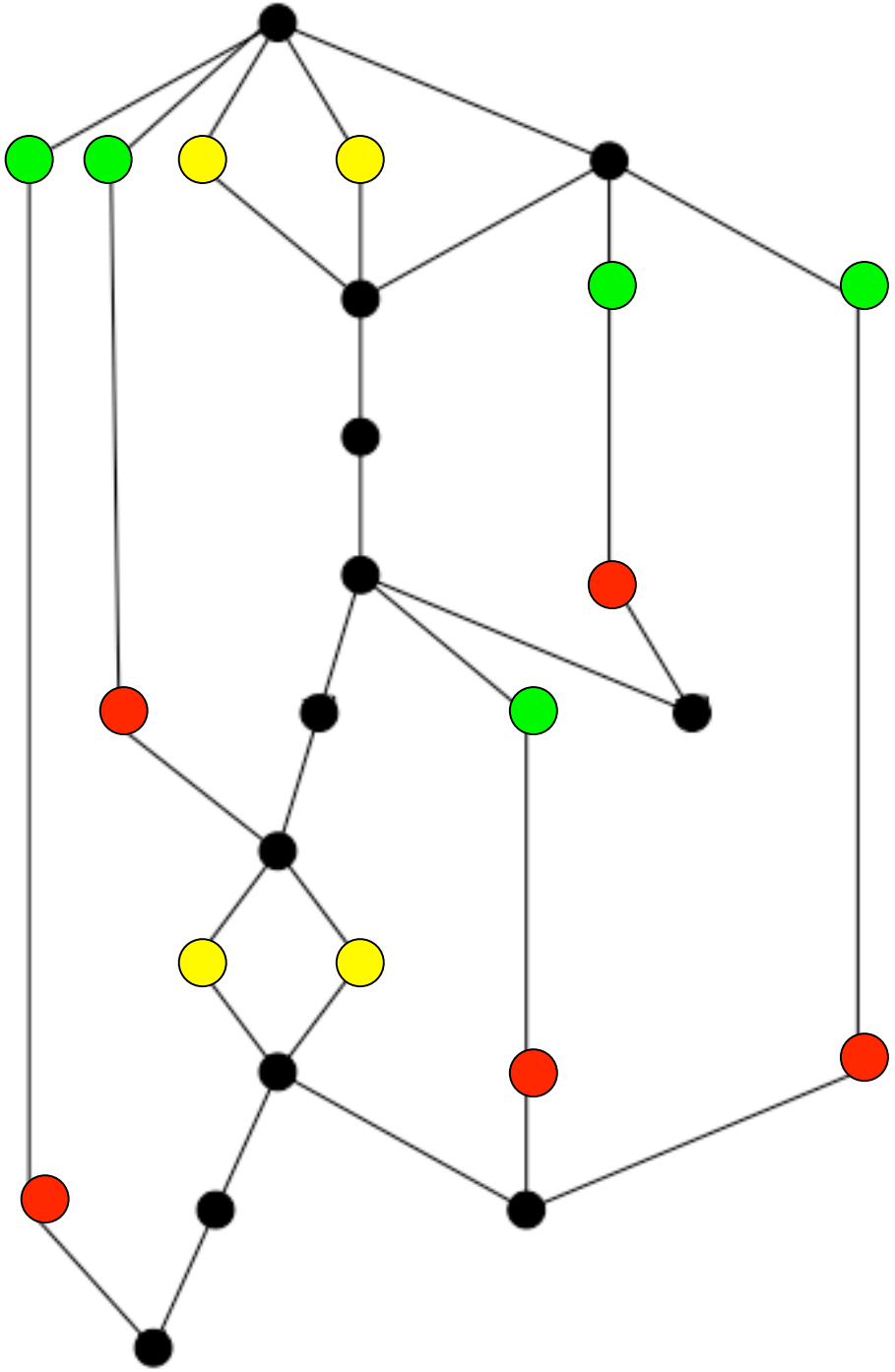
- Behandle bei der 2 Schichten-Kreuzungsminimierung Container wie Knoten
- Ein Container bekommt bei der Barycenter bzw. Medianheuristik den Wert des Vorgängers zugewiesen

$\Rightarrow$Keine Kreuzungen zwischen Segmenten

- Dummy Knoten würden in dem traditionellen Sugiyama auch den Wert ihres Vorgängers bekommen



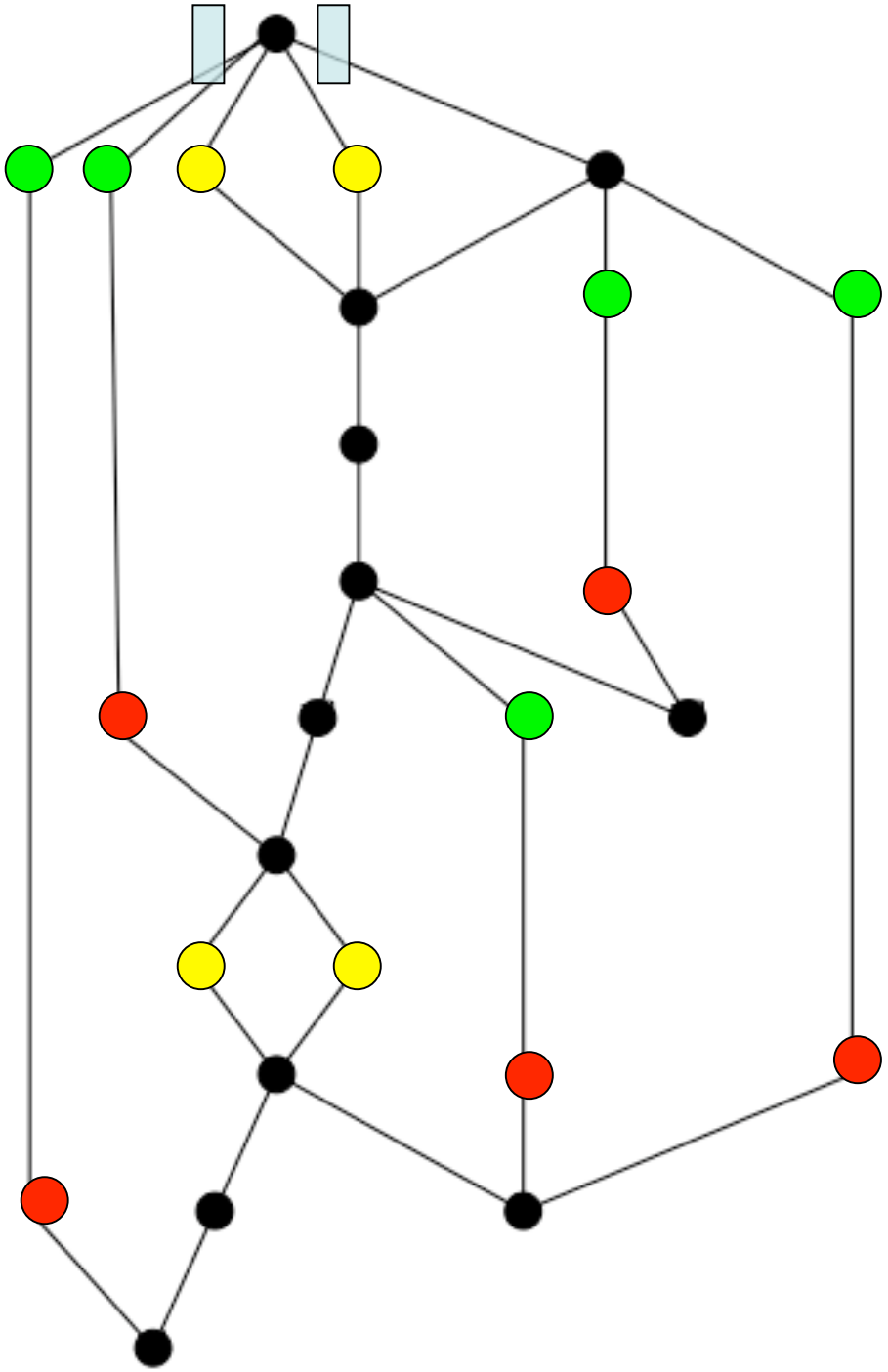
Beispiel



p Knoten

p-q Knoten

q Knoten

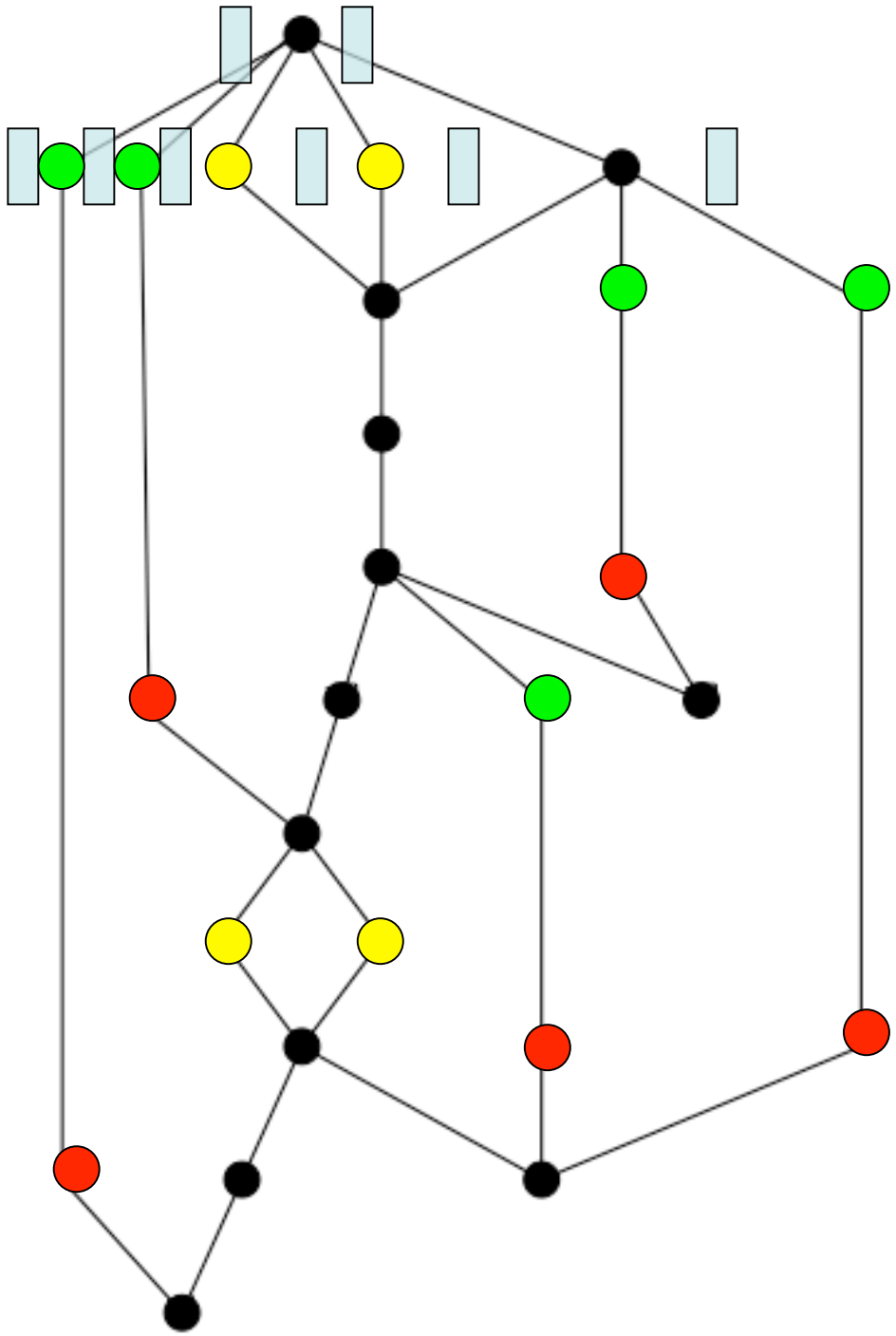


p Knoten

p-q Knoten

q Knoten

Container

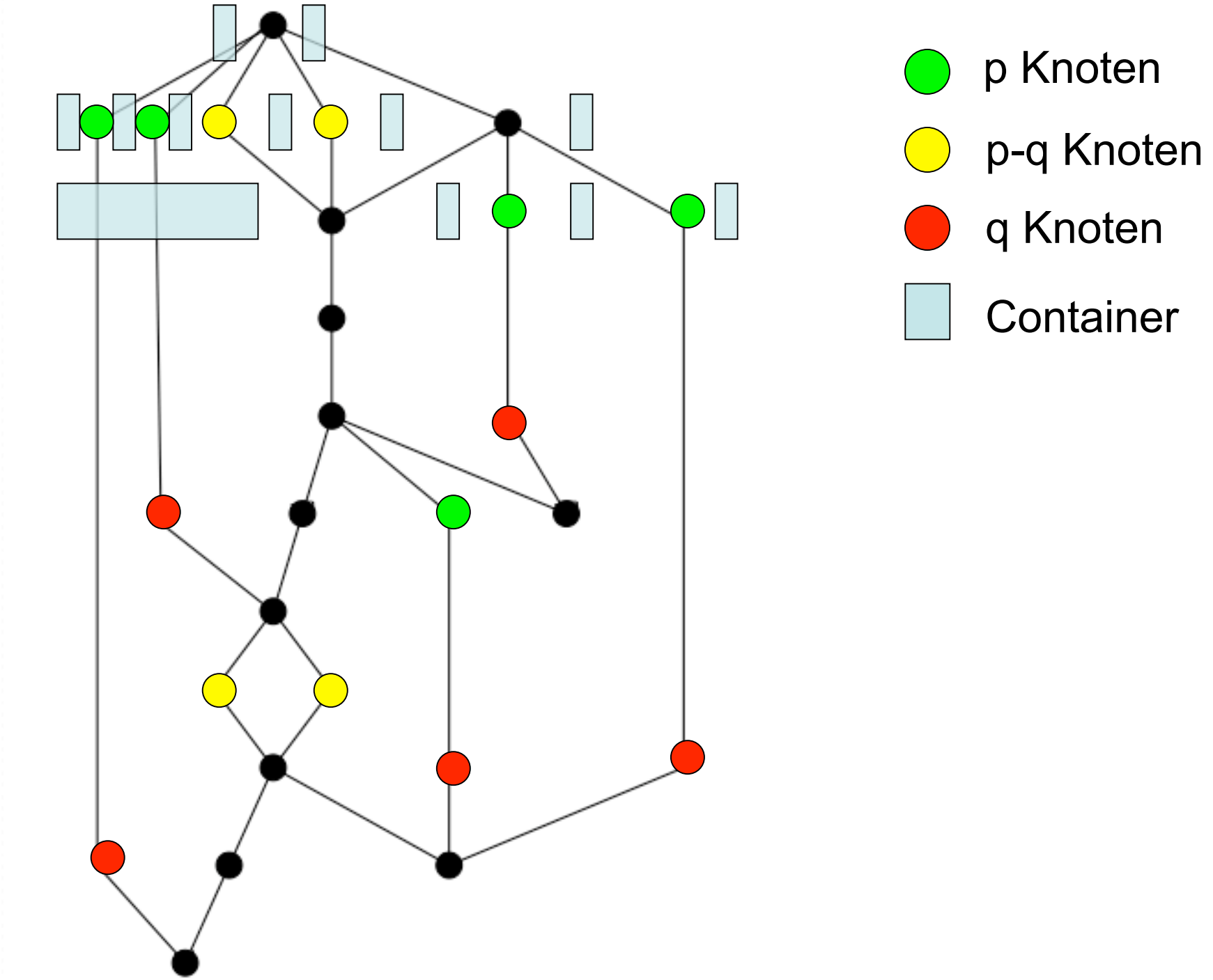


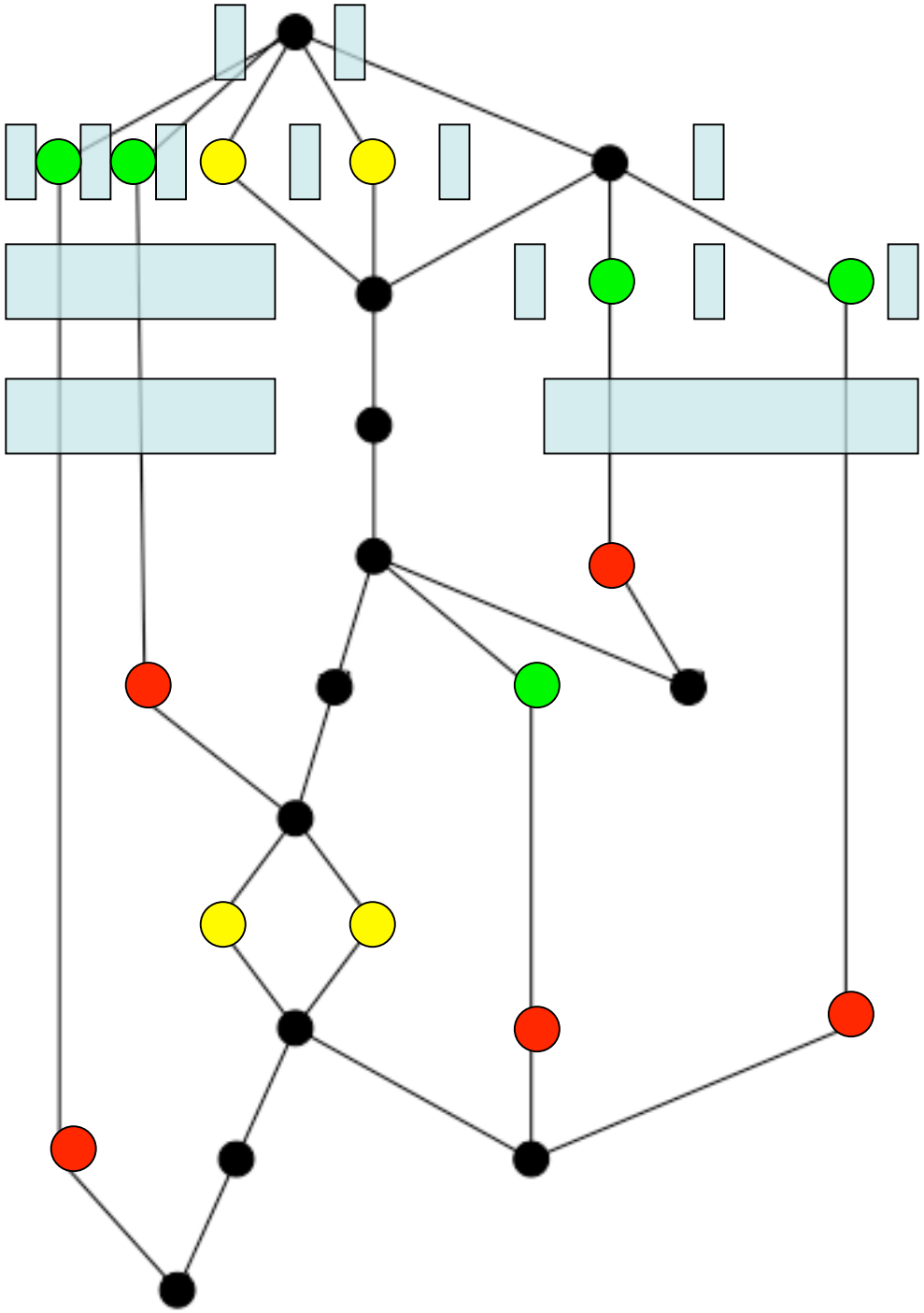
p Knoten

p-q Knoten

q Knoten

Container


p Knoten
p-q Knoten
q Knoten
Container

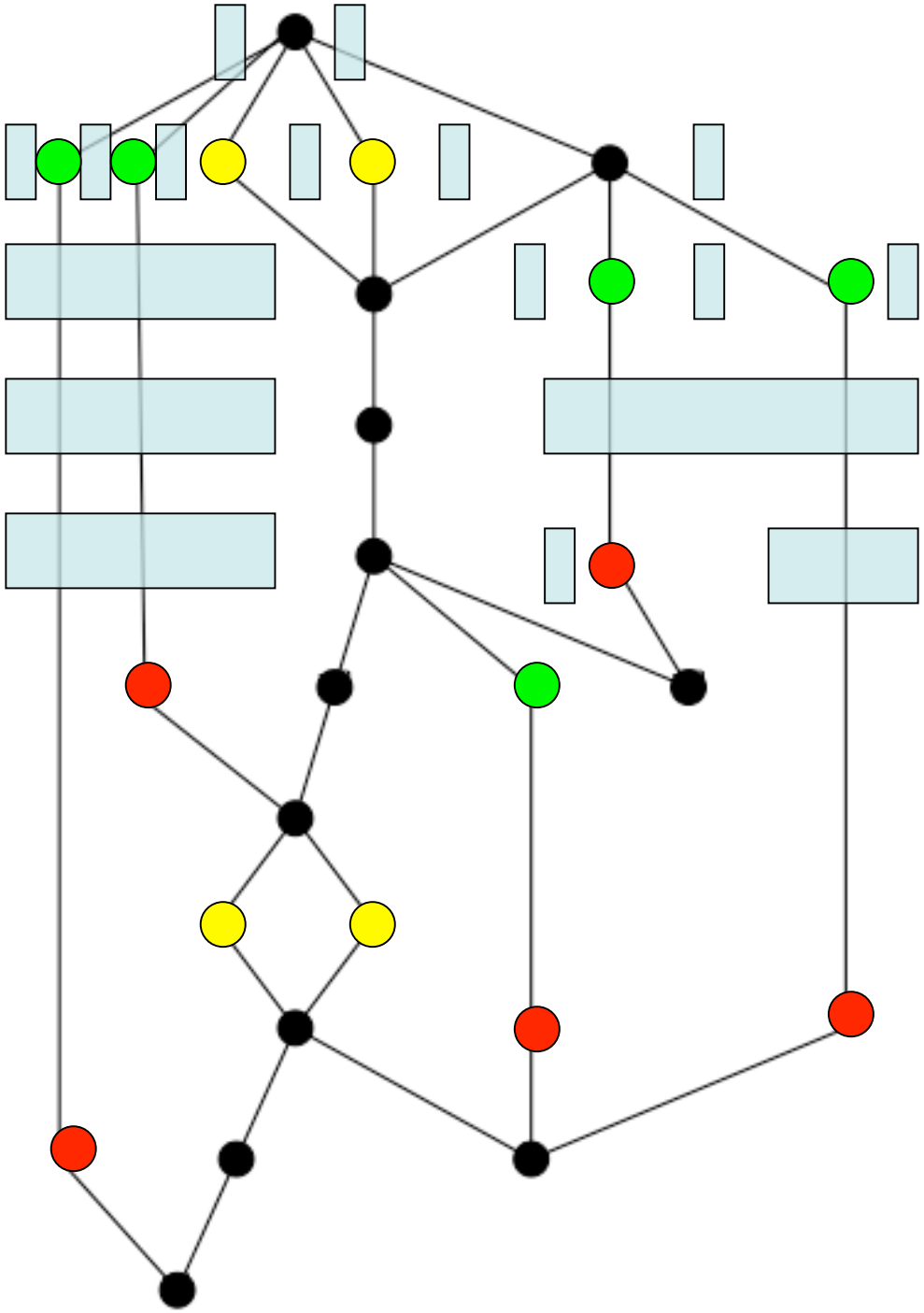


p Knoten

p-q Knoten

q Knoten

Container

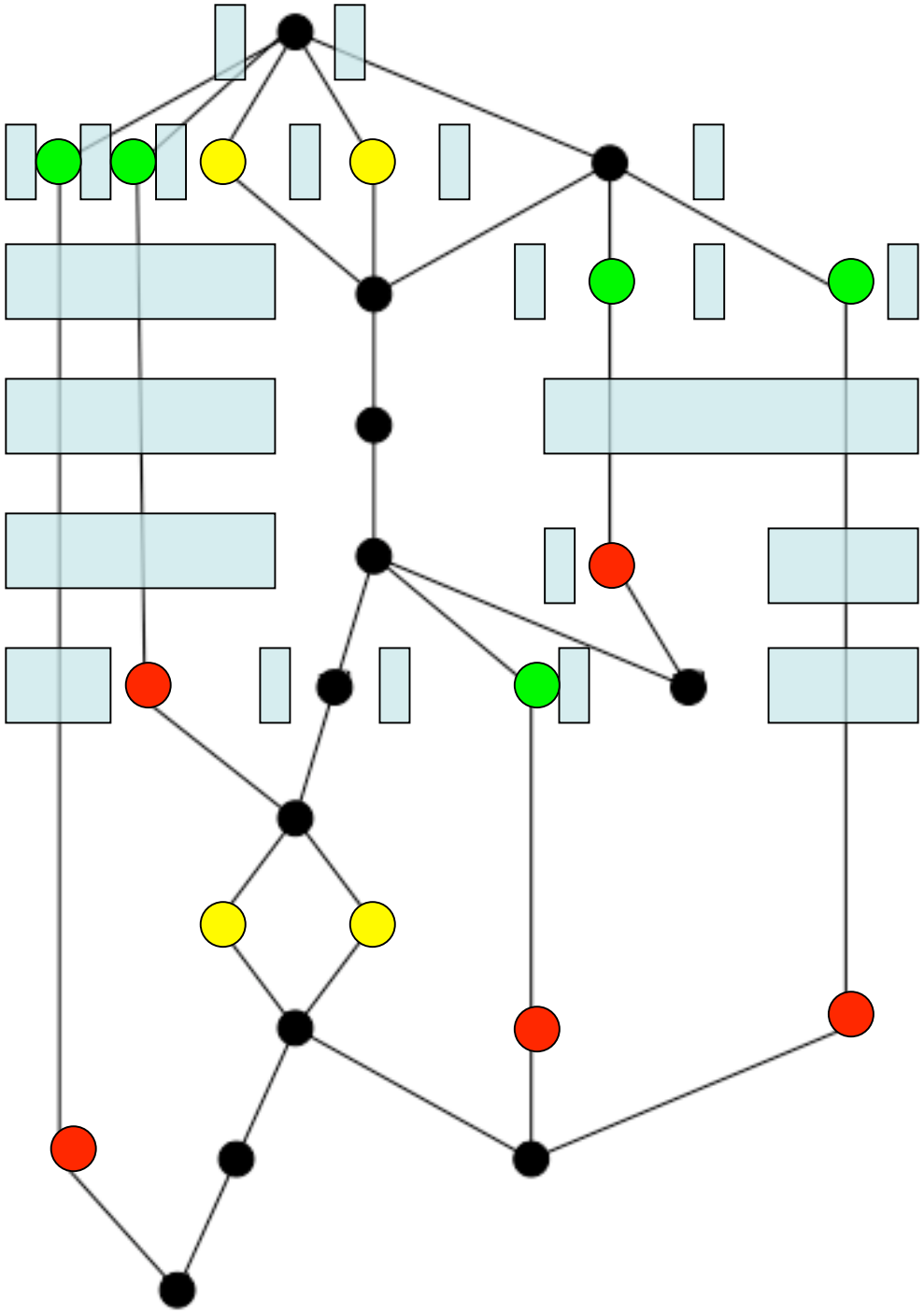


p Knoten

p-q Knoten

q Knoten

Container

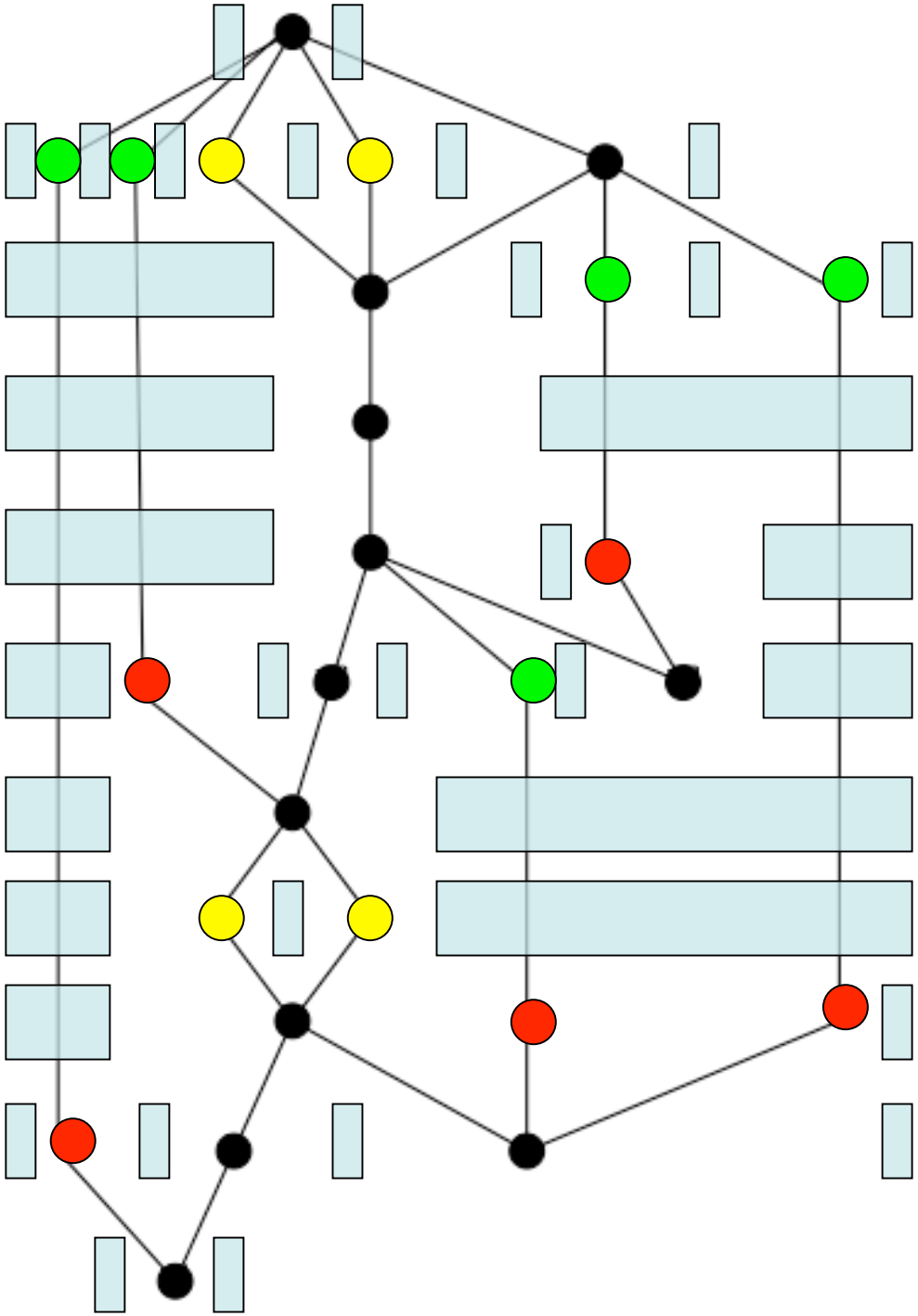


p Knoten

p-q Knoten

q Knoten

Container



p Knoten

p-q Knoten

q Knoten

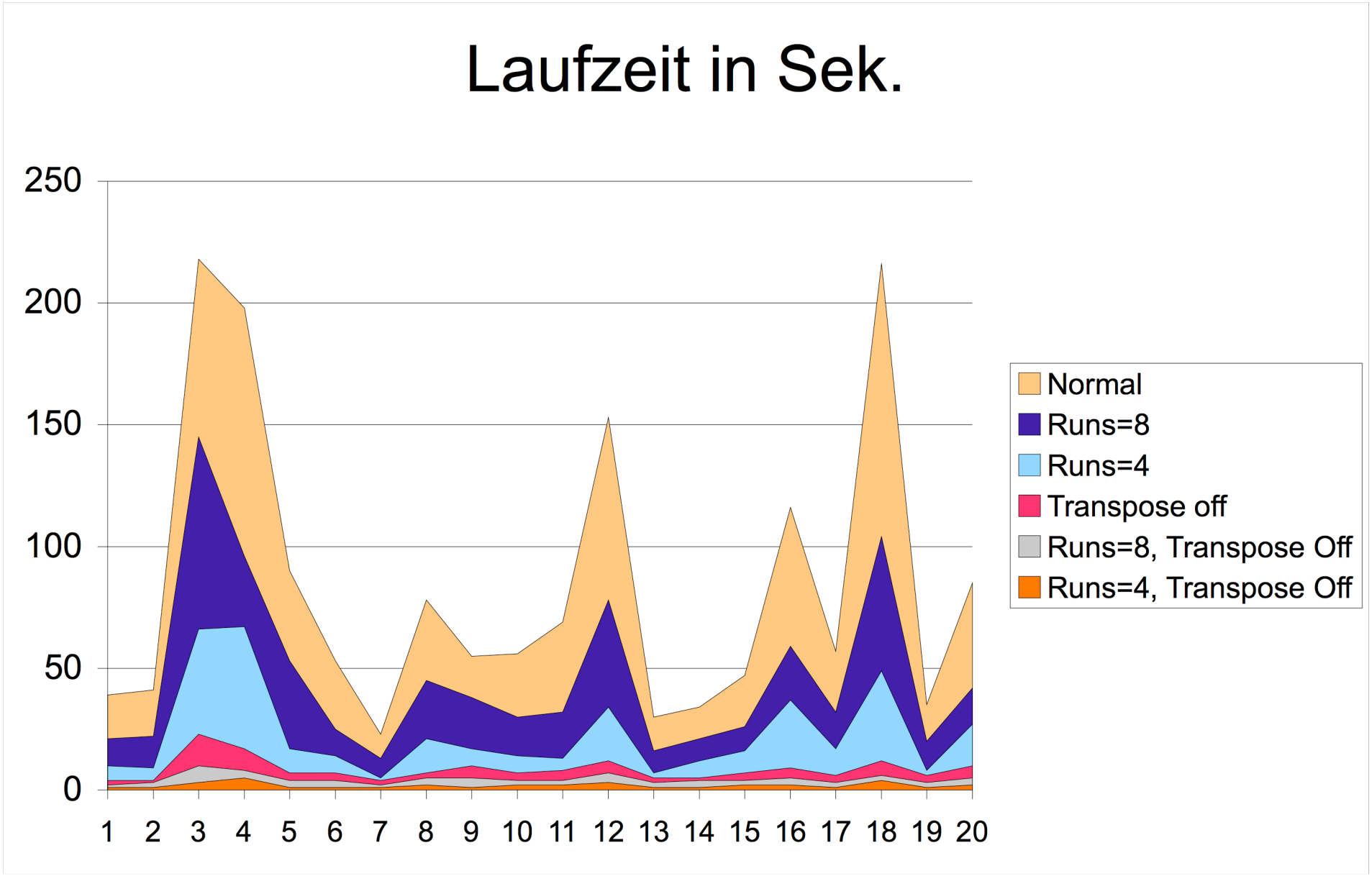
Container

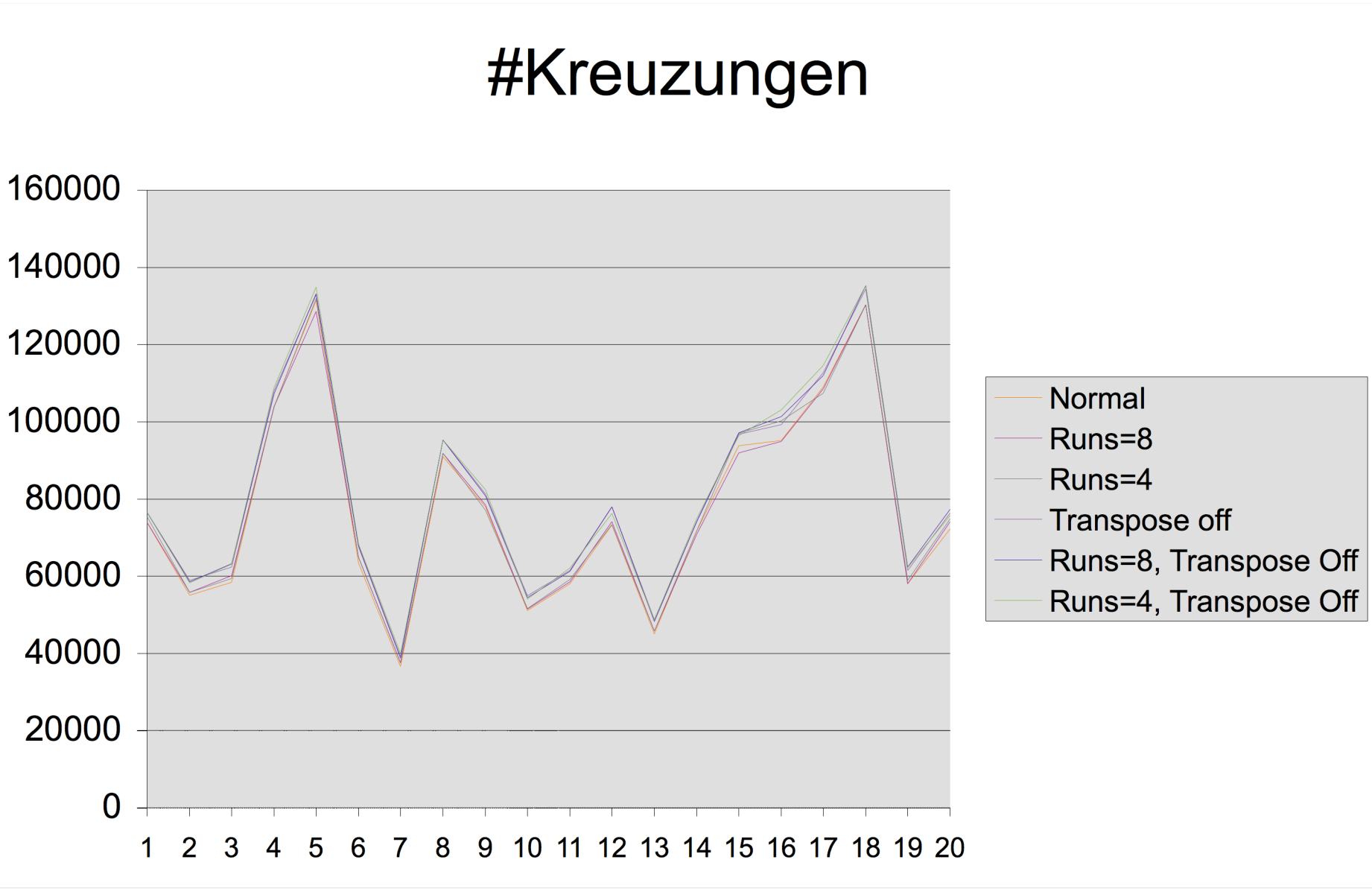
Fazit

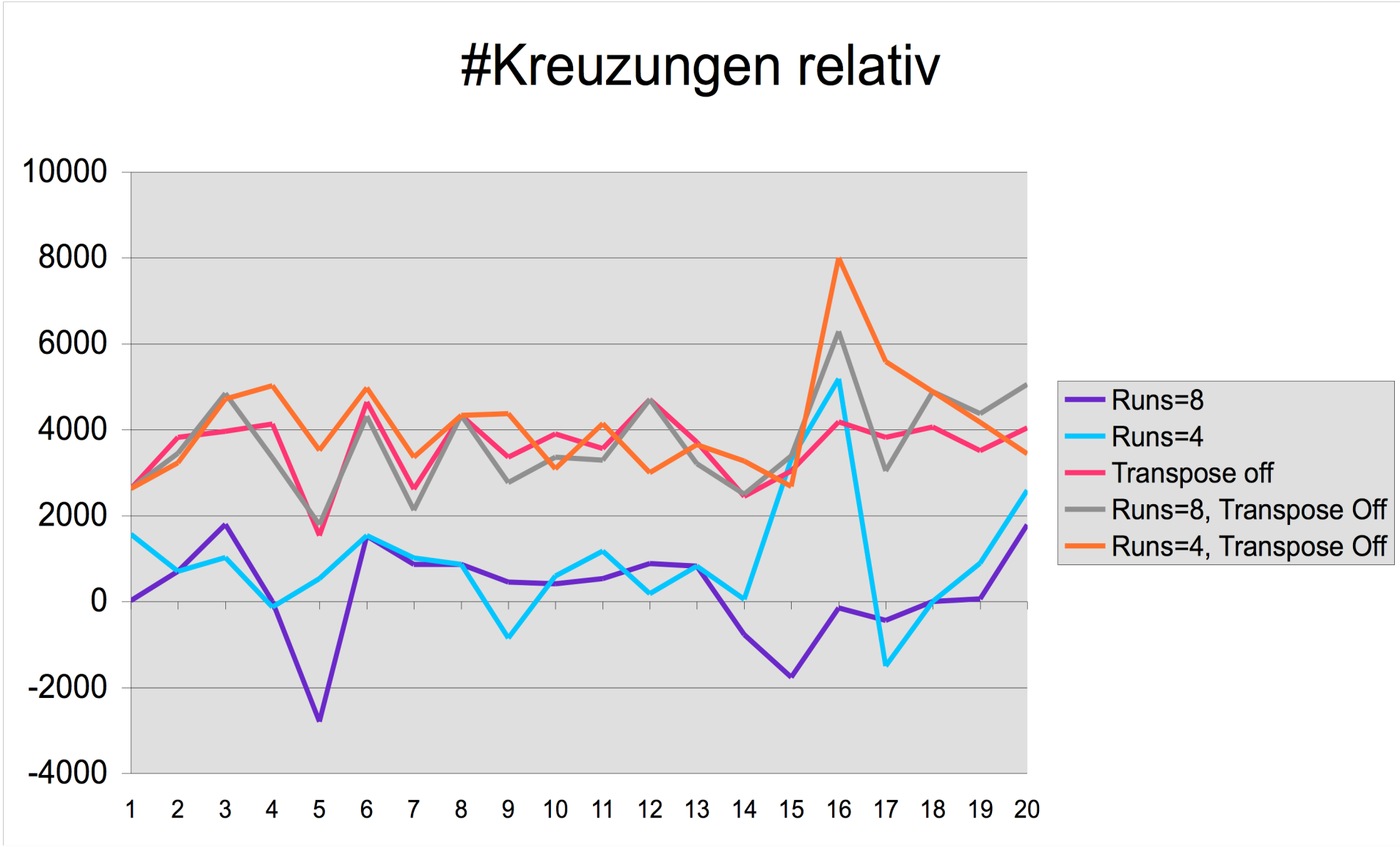
- Die Modifikation verschlechtert das Layout nicht
- Laufzeit: O((|V|+|E|)*log|E|)
- Speicherplatz: O(|V|+|E|)

Experimentelle Ergebnisse

- 20 Graphen mit ~1000 Knoten und ~2000 Kanten
- Generator: ***ogdf::randomHierarchy*** *mit Parametern (singleSource=false, planar=false, longEdges=true)*
- Testläufe mit:
  - Transpose an / aus
  - 4,8,15 Runs


Laufzeit in Sek.
250
200
150
100
50
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Normal
Runs=8
Runs=4
Transpose off
Runs=8, Transpose Off
Runs=4, Transpose Off


#Kreuzungen
160000
140000
120000
100000
80000
60000
40000
20000
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Normal
Runs=8
Runs=4
Transpose off
Runs=8, Transpose Off
Runs=4, Transpose Off


#Kreuzungen relativ
10000
8000
6000
4000
2000
0
-2000
-4000
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Runs=8
Runs=4
Transpose off
Runs=8, Transpose Off
Runs=4, Transpose Off

Literatur

Eiglsperger, Siebenhaller, Kaufmann - An Efficient Implementation of Sugiyama‘s Algorithm for Layered Graph Drawing, JGAA vol9, no 3, pp. 305-325 (2005)