Algorithmen zur Visualisierung von Graphen

Teile & Herrsche-Algorithmen: Bäume und serien-parallele Graphen

Vorlesung im Wintersemester 2011/2012

Ignaz Rutter

11.1.2012

Institute for Theoretical Informatics
Algorithmics Group I

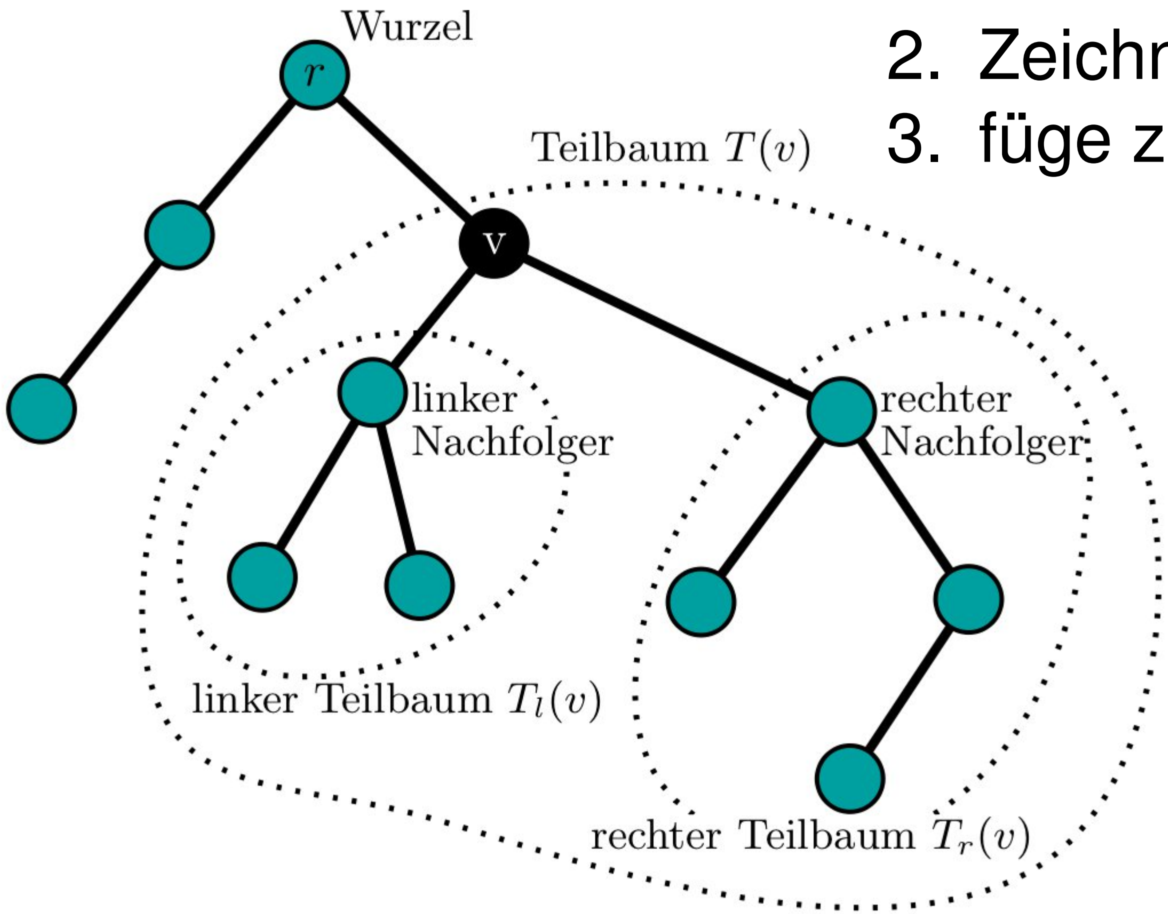
Algorithmen zum Zeichnen von Bäumen

Institute for Theoretical Informatics
Algorithmics Group I

Anwendbarkeit

Gut bei induktiv oder rekursiv definierten Familien von Graphen

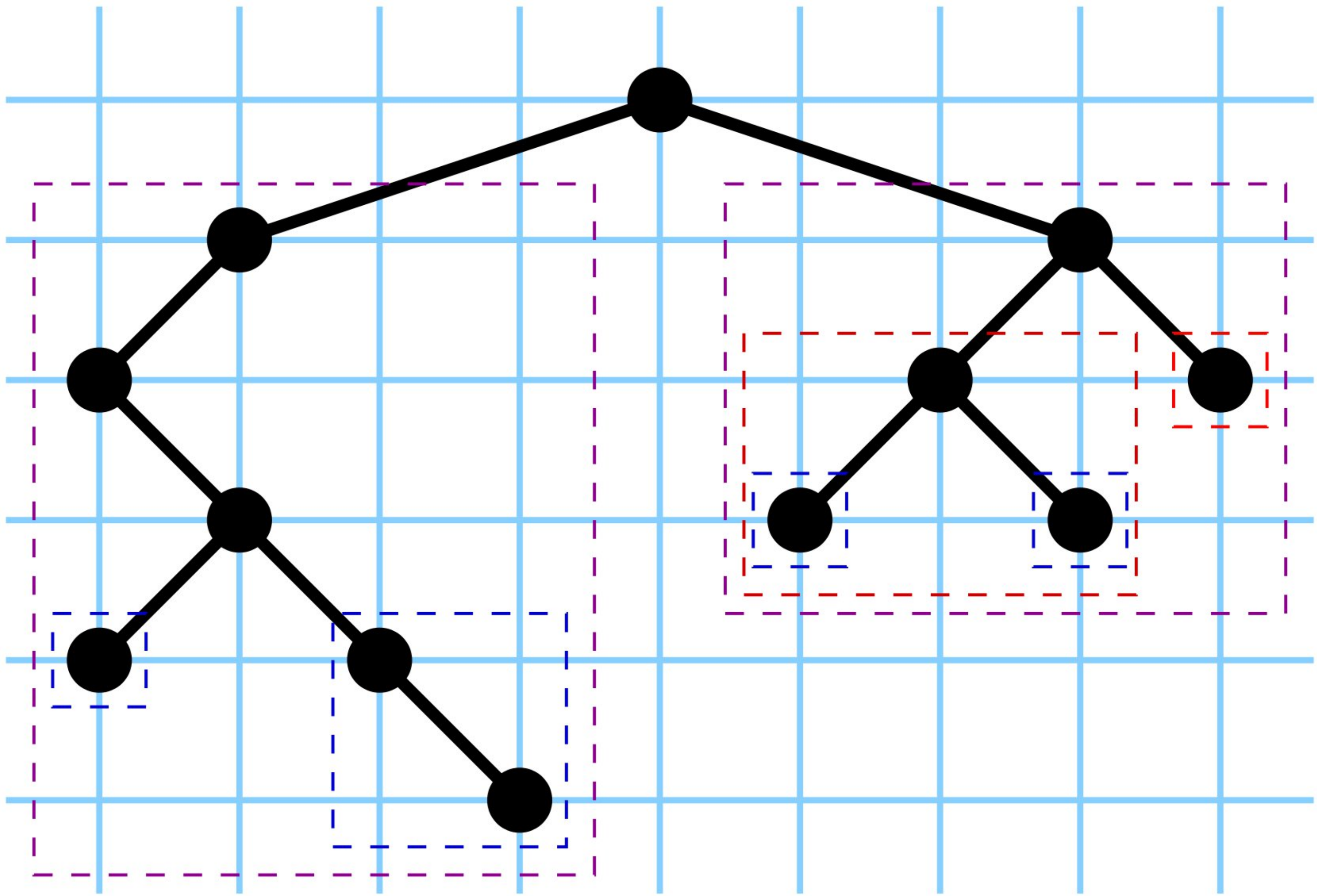
Binärbaum mit Wurzel:



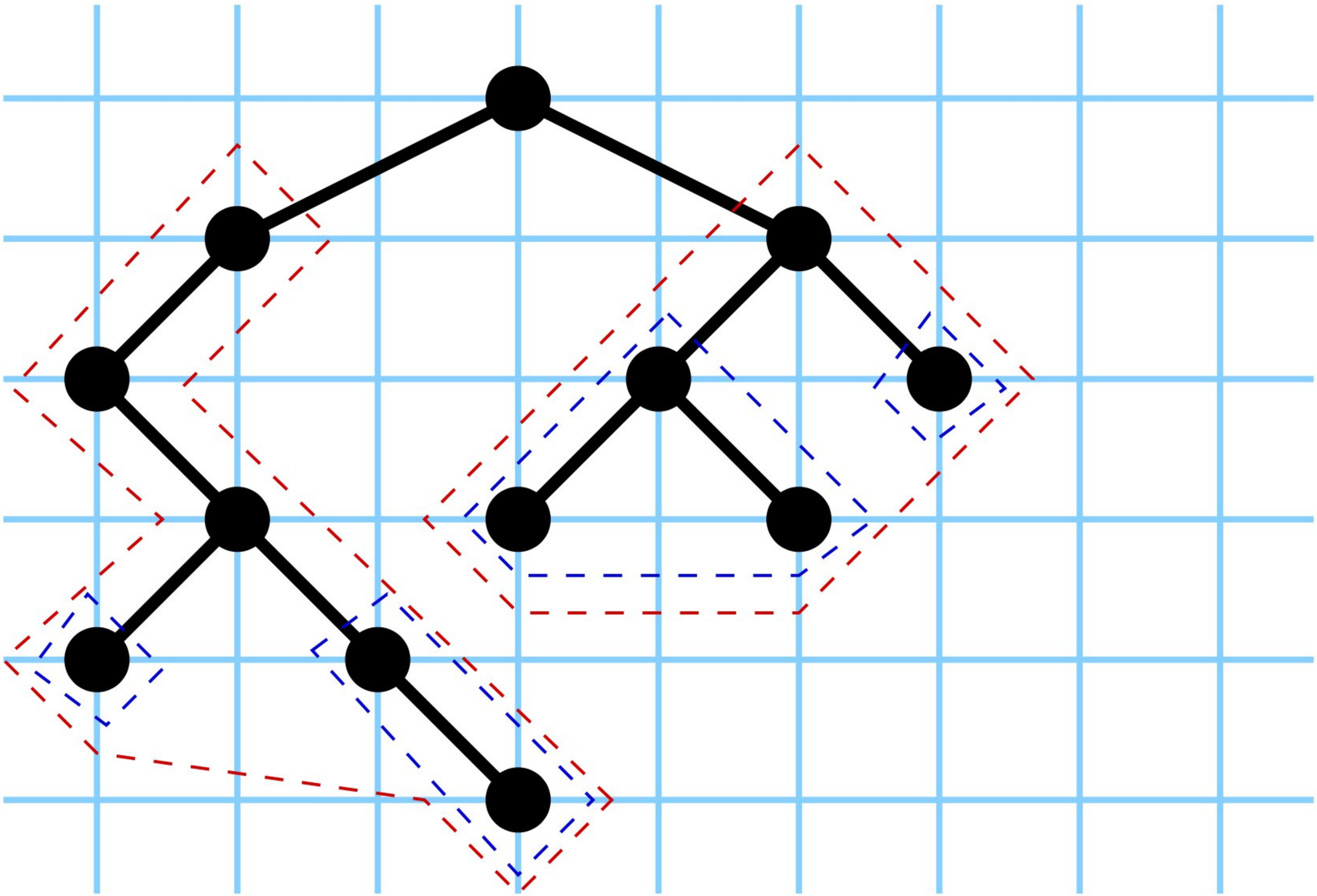
1. Zeichne linken Teilbaum
2. Zeichne rechten Teilbaum
3. füge zusammen + Wurzel

- tiefe(v): Abstand zur Wurzel
- Durchlauf-reihenfolgen
  - preorder
  - inorder
  - postorder

Algorithmus von Reingold und Tilford ('81)



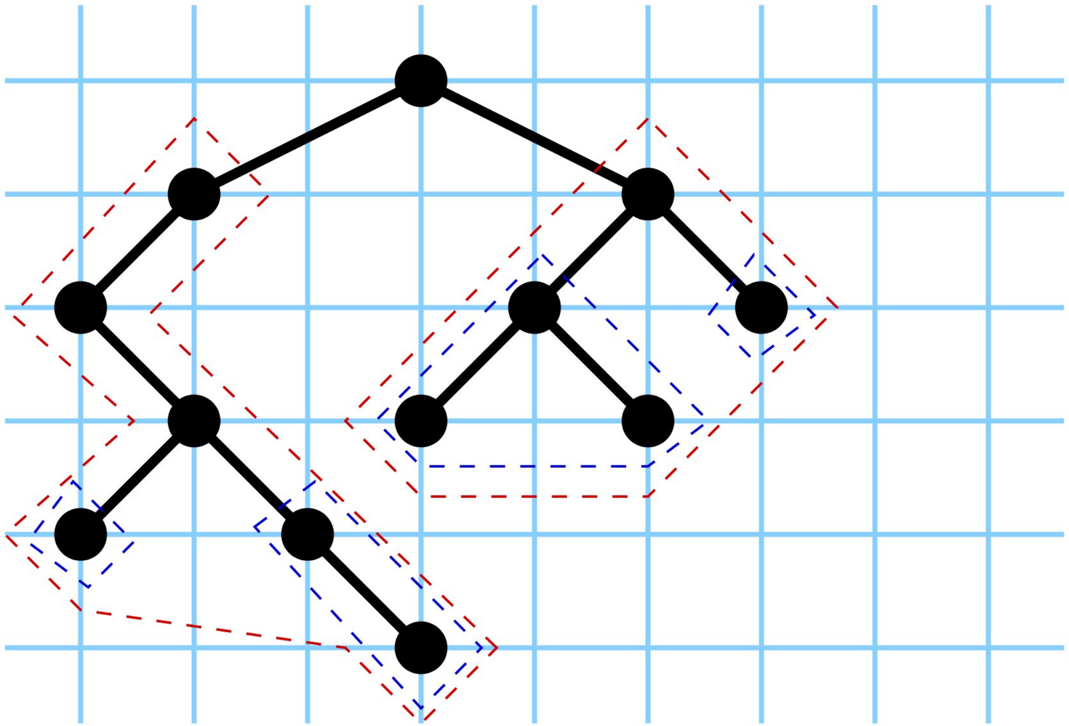
Algorithmus von Reingold und Tilford ('81)



Algorithmus von Reingold und Tilford ('81)

2 Phasen:

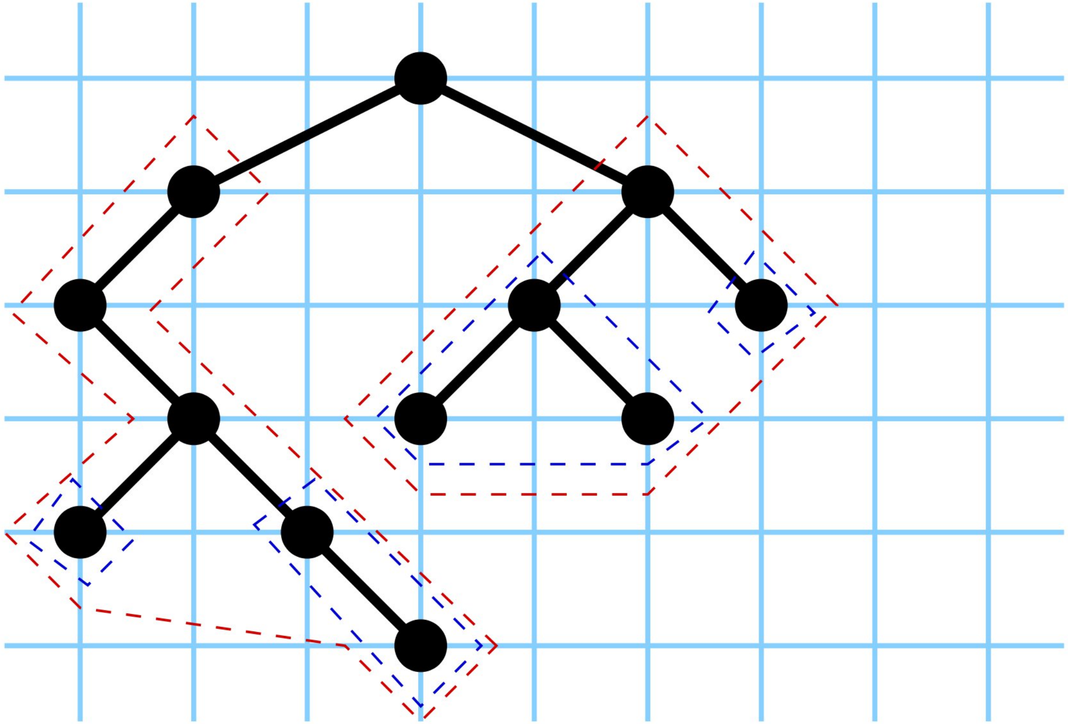
1. postorder (bottom-up): Konturen und x-Offsets zum Vorgänger einsammeln
2. preorder (top-down): absolute Koordinaten ausrechnen



Algorithmus von Reingold und Tilford ('81)

2 Phasen:

1. postorder (bottom-up): Konturen und x-Offsets zum Vorgänger einsammeln
2. preorder (top-down): absolute Koordinaten ausrechnen



Kontur: verkettet Liste von Knoten (-Koordinaten)

Institute for Theoretical Informatics
Algorithmics Group I

Algorithmus von Reingold und Tilford ('81)

Phase 1:

1. Bearbeite $T_\ell(v)$ und $T_r(v)$
2. Laufe parallel linke Kontur von $T_r(v)$ und rechte Kontur von $T_\ell(v)$ ab
3. Bestimmt daraus d_v, den horizontalen Minimalabstand von v_ℓ und v_r
4. x-Offset(v_ℓ) = $-\lceil \frac{d_v}{2} \rceil$, x-Offset($v_r$) = $\lceil \frac{d_v}{2} \rceil$
5. Baue linke Kontur von T_v aus: v, linke Kontur von $T_\ell(v)$ und evtl. überhängendes Teilstück von linker Kontur von $T_r(v)$
6. Rechte Kontur analog

Institute for Theoretical Informatics
Algorithmics Group I

Algorithmus von Reingold und Tilford (’81)

Phase 2

1. Setze y-Koordinate $y(v) = -\text{tiefe}(v)$
2. Setze $x(v) = 0$ für Wurzel und rekursiv die x-Koordinate $x(v_\ell)$ und $x(v_r)$ der Nachfolger von v auf $x(v)+$ x-Offset($x(v_\ell)$) bzw. $x(v)+$ x-Offset($x(v_r)$)

Zusammenfassung:
Algorithmus berechnet Binärbaumlayout:

- geradliniges Gitterlayout
- tiefengeschichtet, kreuzungsfrei
- Knoten derselben Tiefe haben Abstand ≥ 2
- Knoten sind über Nachfolgern zentriert
- linke/rechte Nachfolger sind strikt links/rechts
- identische Teilbäume gleich gezeichnet

Institute for Theoretical Informatics
Algorithmics Group I

Breitenminimierung von Binärbaumlayouts

Satz (Supowit, Reingold)

Die Breitenminimierung von Binärbaumlayouts ist NP-schwer

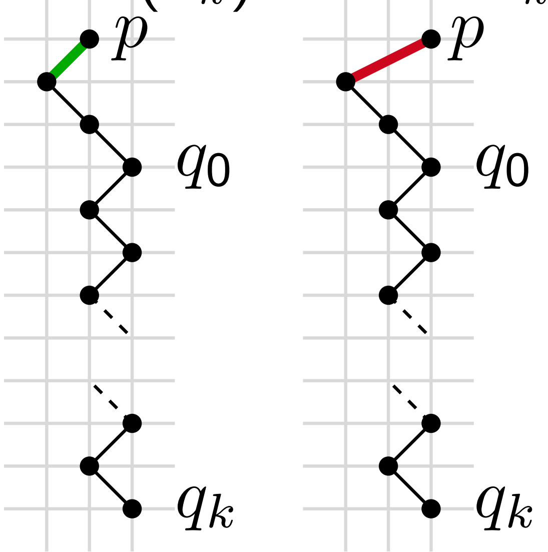
Beweis: Reduktion von 3SAT

$F = C_1 \wedge \cdots \wedge C_m,$

$C_i = y_{i,1} \vee y_{i,2} \vee y_{i,3}, \quad y_{i,j} \in \{x_1, \ldots, x_n, \overline{x_1}, \ldots, \overline{x_n}\}$

Konstruiere Baum $T(F)$, der genau dann Layout mit Breite $W \leq 24$ hat, wenn F erfüllbar ist.

Baum $T(x_k)$ für Var. x_k :



Breitenminimierung von Binärbaumlayouts

Karlsruhe Institute of Technology

Satz (Supowit, Reingold)

Die Breitenminimierung von Binärbaumlayouts ist NP-schwer

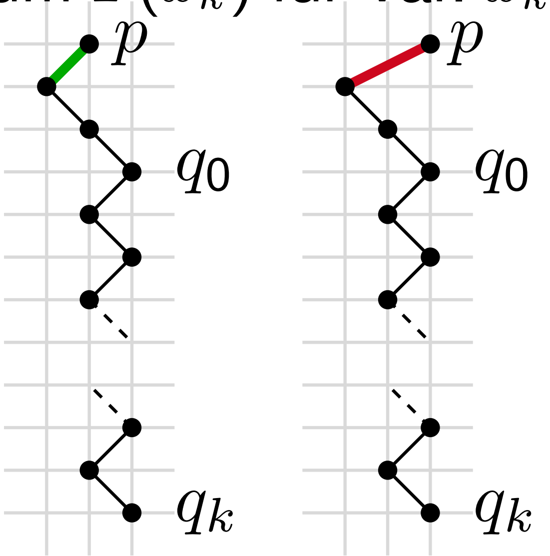
Beweis: Reduktion von 3SAT

$F = C_1 \wedge \cdots \wedge C_m,$

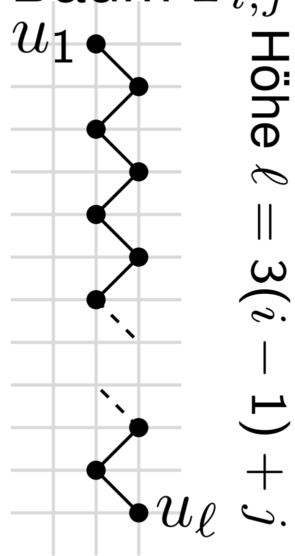
$C_i = y_{i,1} \vee y_{i,2} \vee y_{i,3}, \quad y_{i,j} \in \{x_1, \ldots, x_n, \overline{x_1}, \ldots, \overline{x_n}\}$

Konstruiere Baum $T(F)$, der genau dann Layout mit Breite $W \leq 24$ hat, wenn F erfüllbar ist.

Baum $T(x_k)$ für Var. x_k :



Baum $T_{i,j}$



Breitenminimierung von Binärbaumlayouts

Satz (Supowit, Reingold)

Die Breitenminimierung von Binärbaumlayouts ist NP-schwer

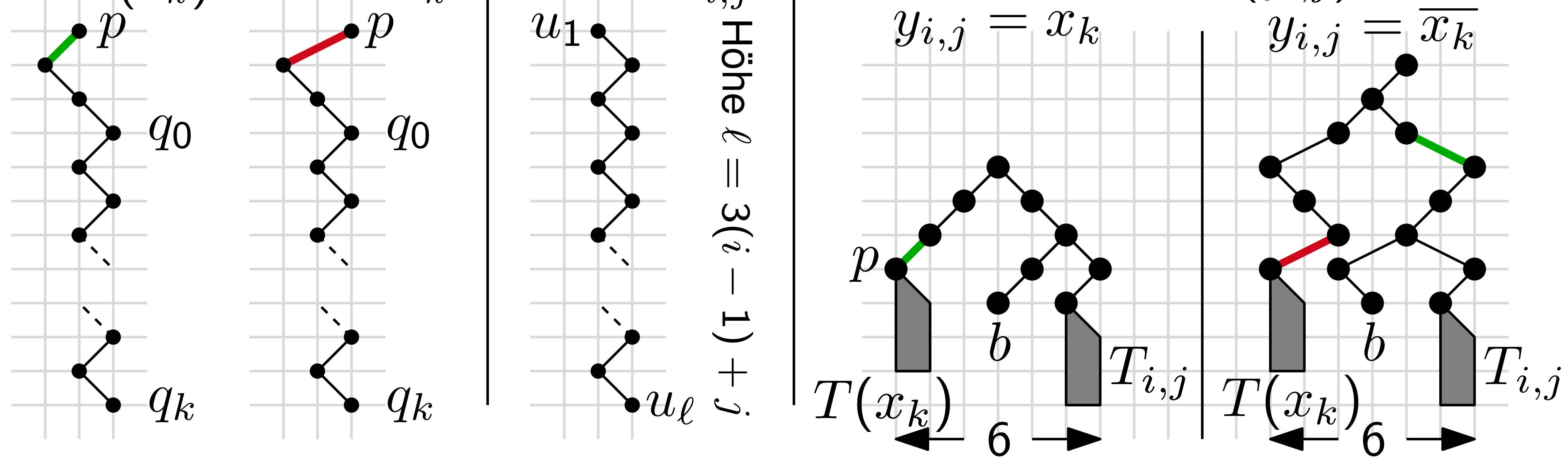
Beweis: Reduktion von 3SAT

$F = C_1 \wedge \dots \wedge C_m,$

$C_i = y_{i,1} \vee y_{i,2} \vee y_{i,3}, \quad y_{i,j} \in \{x_1, \dots, x_n, \overline{x_1}, \dots, \overline{x_n}\}$

Konstruiere Baum $T(F)$, der genau dann Layout mit Breite $W \leq 24$ hat, wenn F erfüllbar ist.

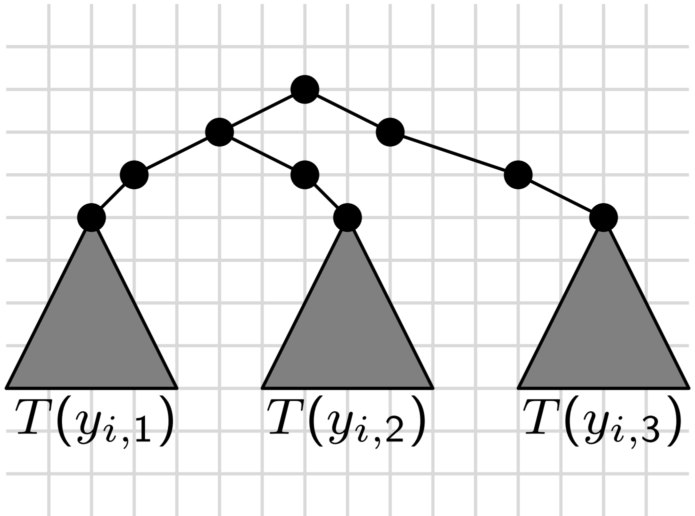
Baum $T(x_k)$ für Var. x_k : | Baum $T_{i,j}$ | Literalbaum: $T(y_{i,j})$ für



Institute for Theoretical Informatics
Algorithmics Group I

Breitenminierung II

Klauselbaum $T(C_i)$:



Erfüllte Klausel hat Breite höchstens

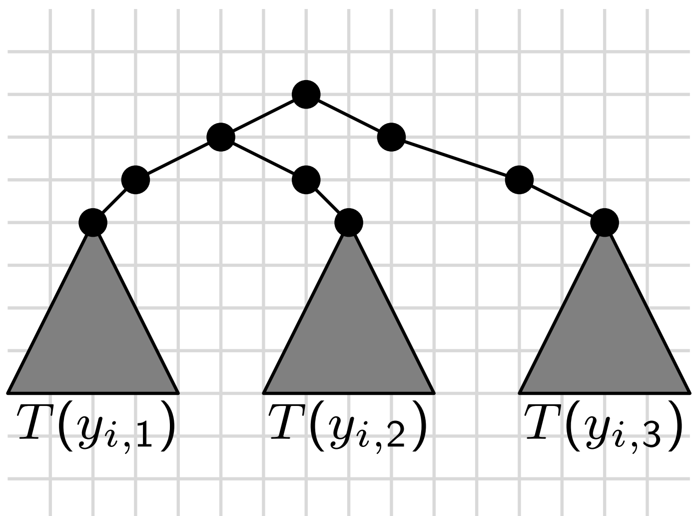
$$6 + 2 + 7 + 2 + 7 = 24$$

Beachte: alle Literalbäume haben volle Breite auf vierter Ebene von oben

Nicht erfüllte Klausel: 7+2+7+2+7 = 25

Breitenminierung II

Klauselbaum $T(C_i)$:



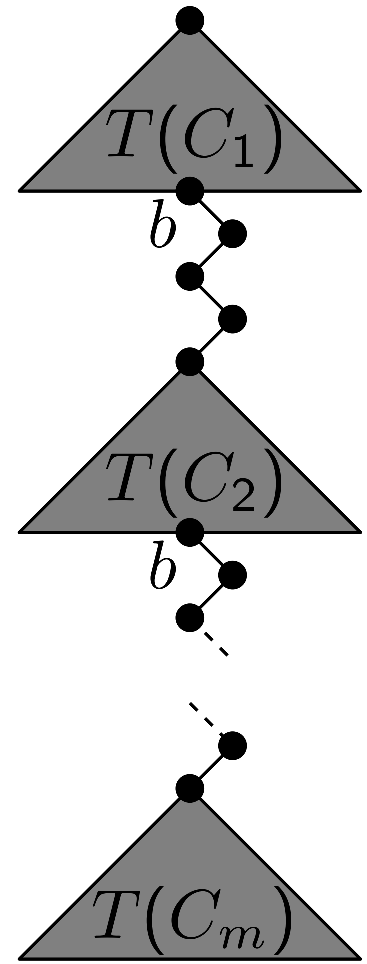
Erfüllte Klausel hat Breite höchstens

$$6 + 2 + 7 + 2 + 7 = 24$$

Beachte: alle Literalbäume haben volle Breite auf vierter Ebene von oben

Nicht erfüllte Klausel: 7+2+7+2+7 = 25

Formelbaum $T(F)$



Breite $\leq 24 \Leftrightarrow F$ erfüllbar

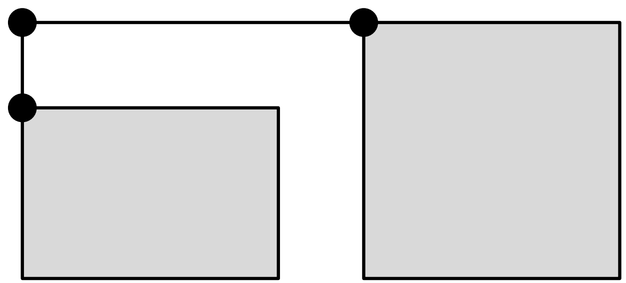
HV-Bäume

Idee:

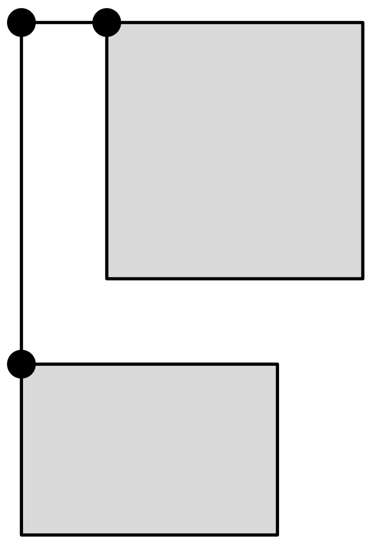
- Zeichne Teilbäume in Rechtecke, Wurzel liegt in linker oberer Ecke
- Nachfolger liegen vertikal unterhalb bzw. horizontal rechts

Induktionsanfang:

Induktionsschritt: kombiniere Layouts



horizontale Kombination
(Fläche: 3×7)



vertikale Kombination
(Fläche: 4×6)

Berechne optimale Zeichnung mit dynamischer Programmierung

Rechtslastige hv-Layouts

Rechtslastiges hv-Layout:

- Wähle in jedem Schritt Horizontal-Kombination
- Platziere größeren Teilbaum rechts

Lemma

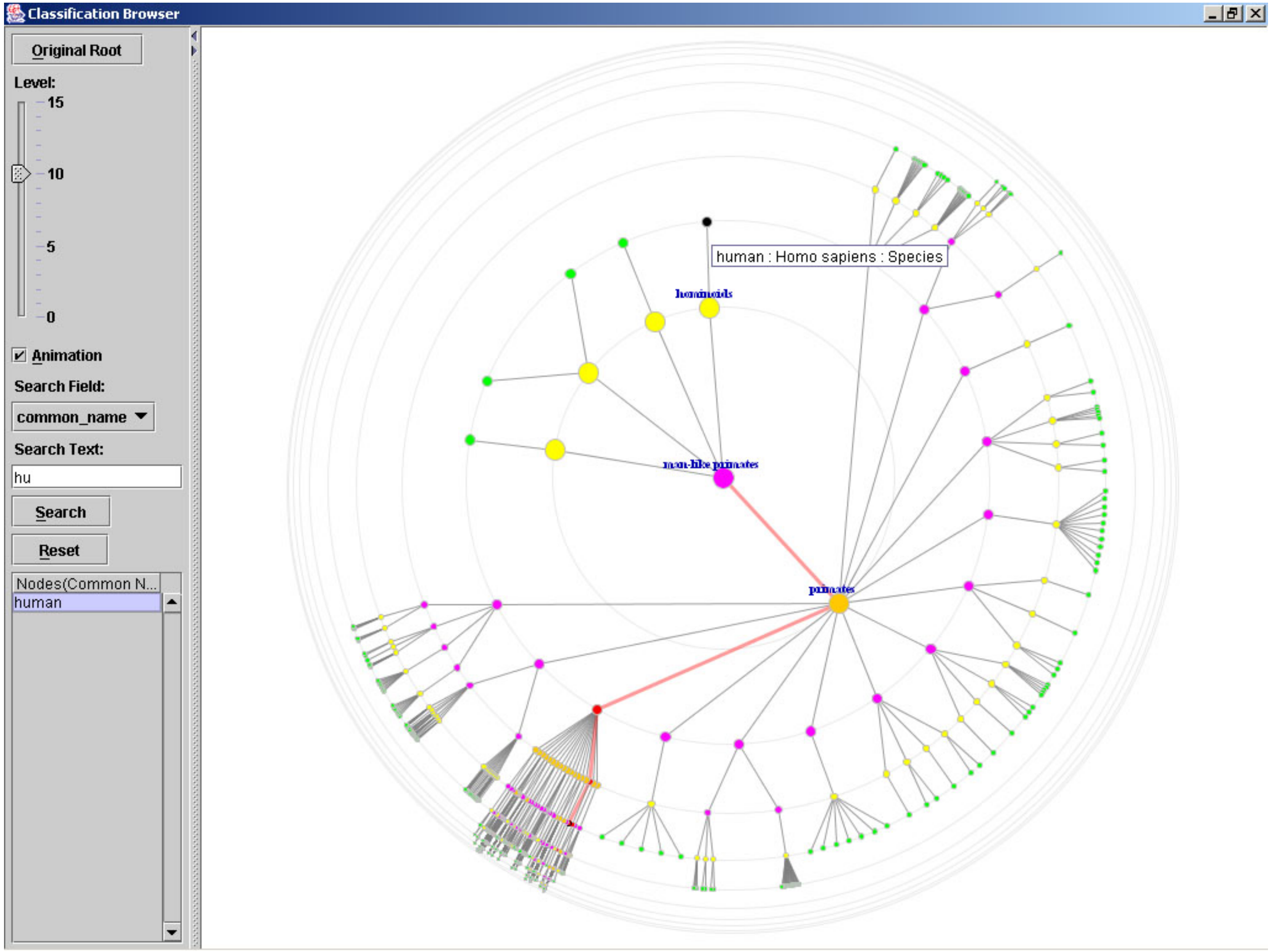
Höhe eines rechtslastigen hv-Layouts für Baum mit n Knoten ist höchstens $\log n$.

Beweis:

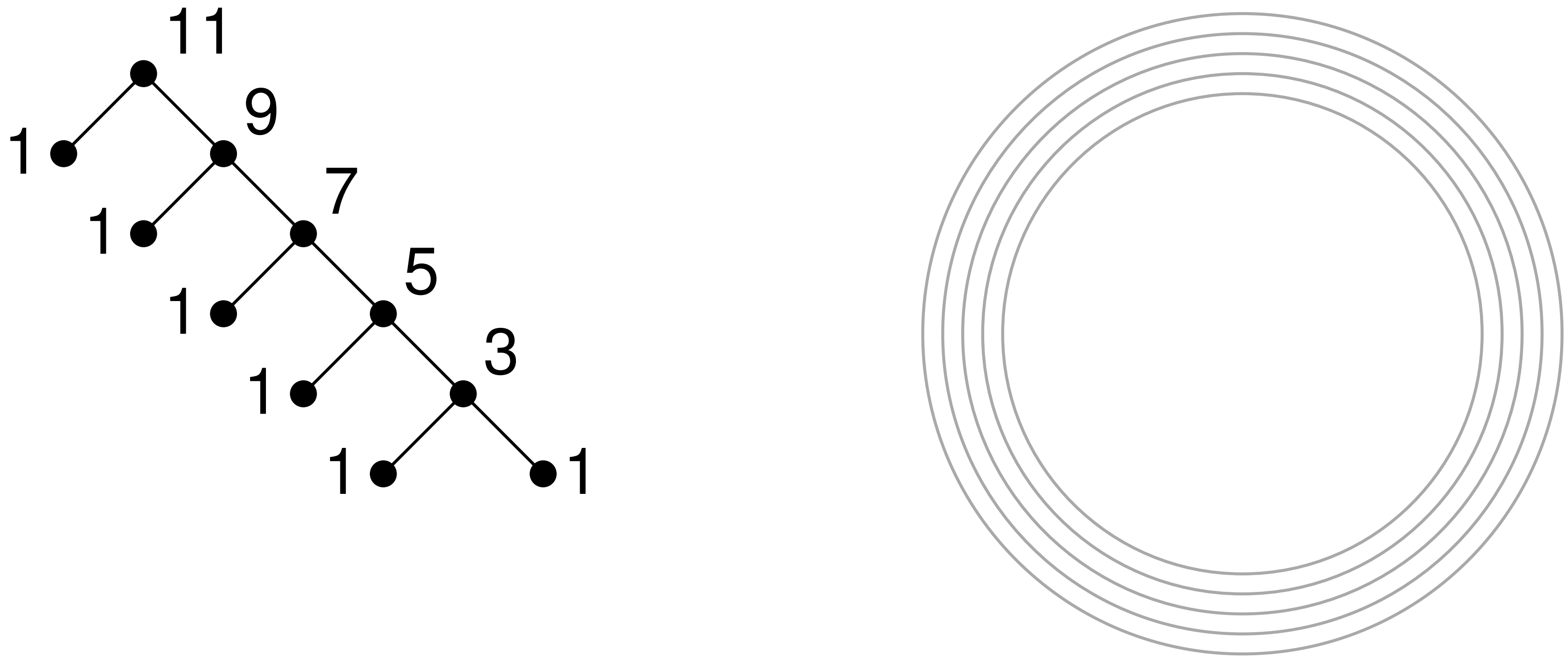
- Vertikale Kanten haben Länge 1
- w Knoten mit minimaler y-Koordinate
- betrachte eindeutigen Pfad P zur Wurzel
- für jede vertikale Kante (u, v) auf P: $|T(v)| > |2T(u)|$
- $\Rightarrow P$ enthält höchstens $\log n$ solcher Kanten

Platzbedarf: $O(n \log n)$

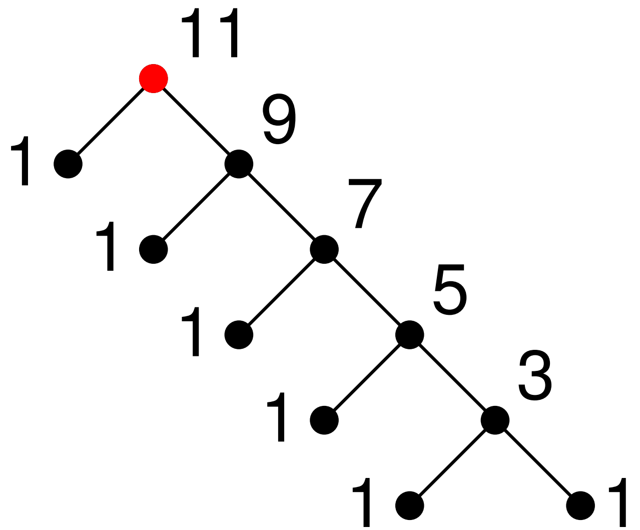
Radiale Baumlayouts

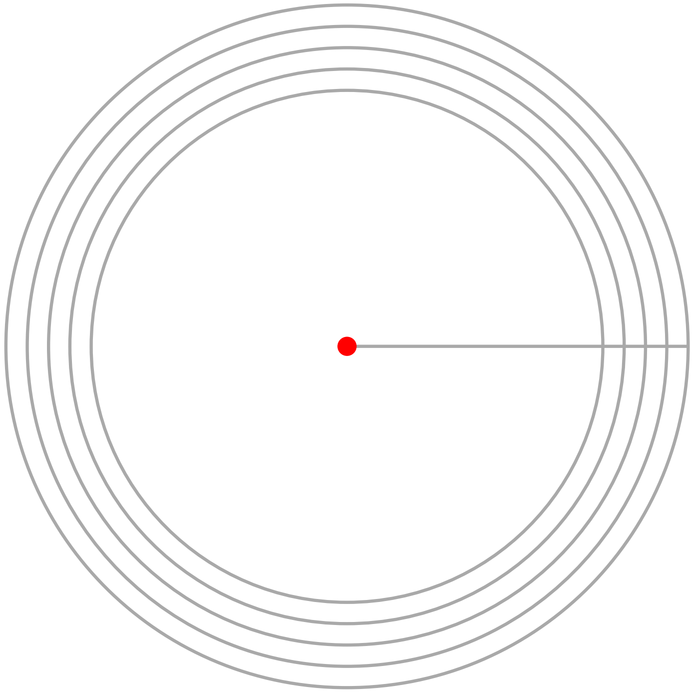


Beispiel Radiallayout

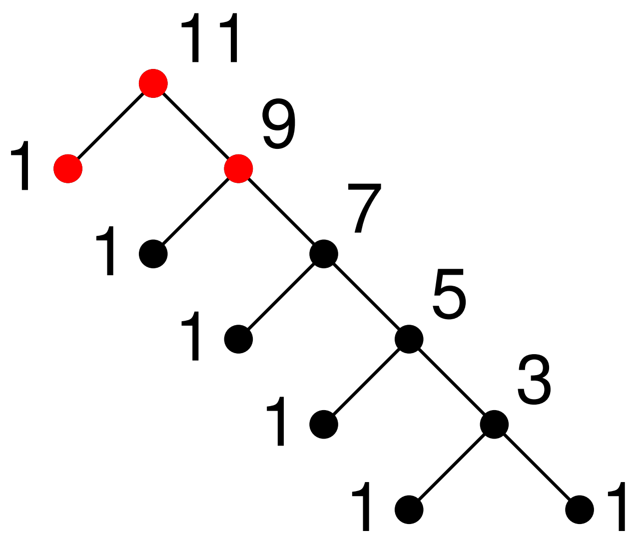


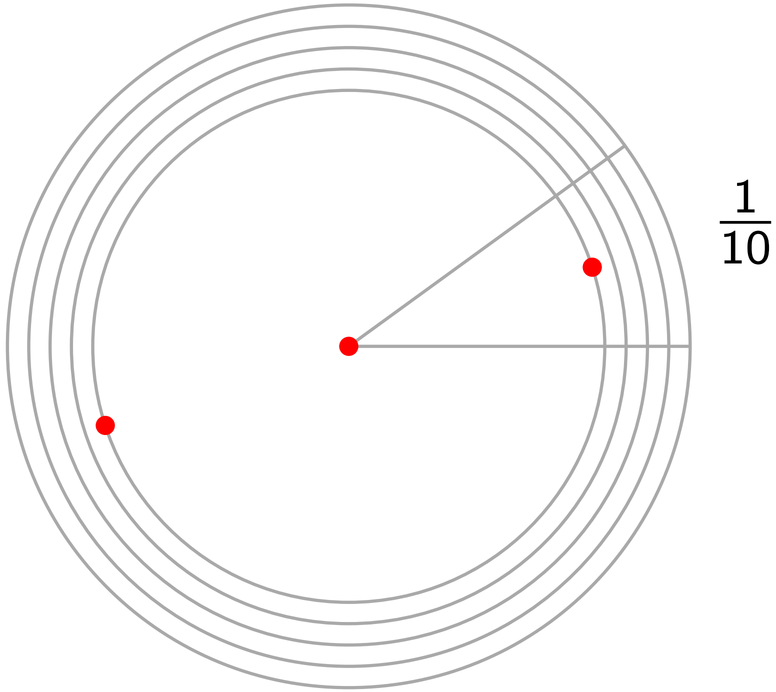
Beispiel Radiallayout



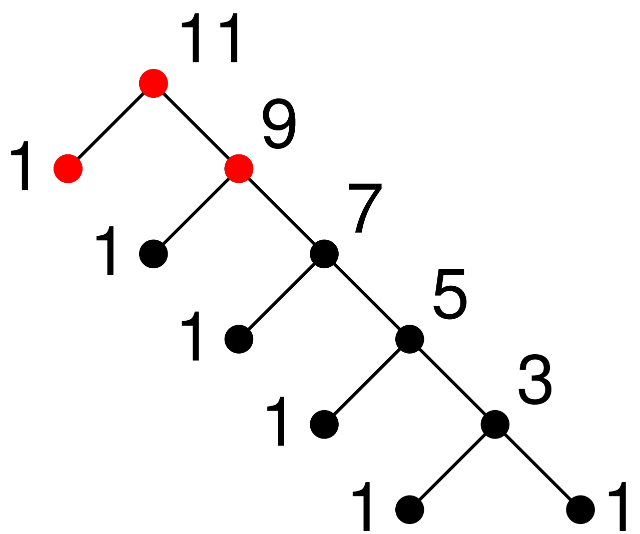


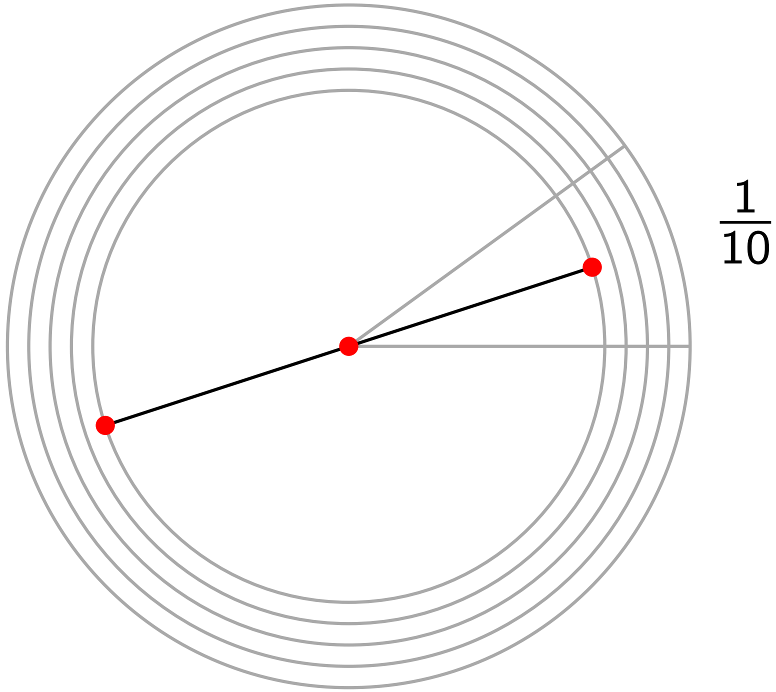
Beispiel Radiallayout





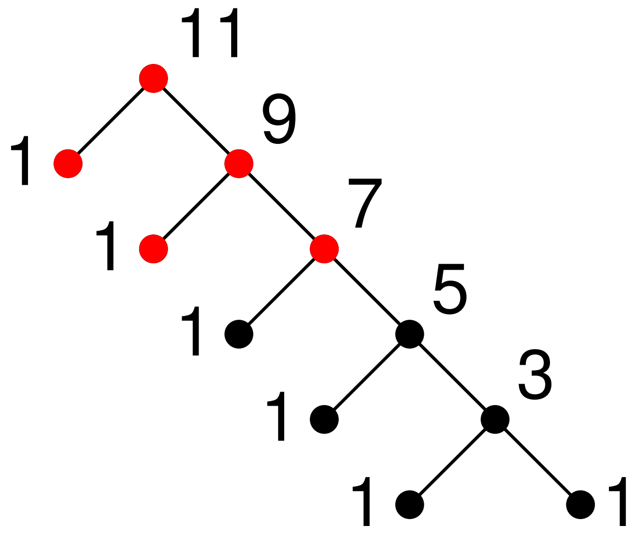
Beispiel Radiallayout

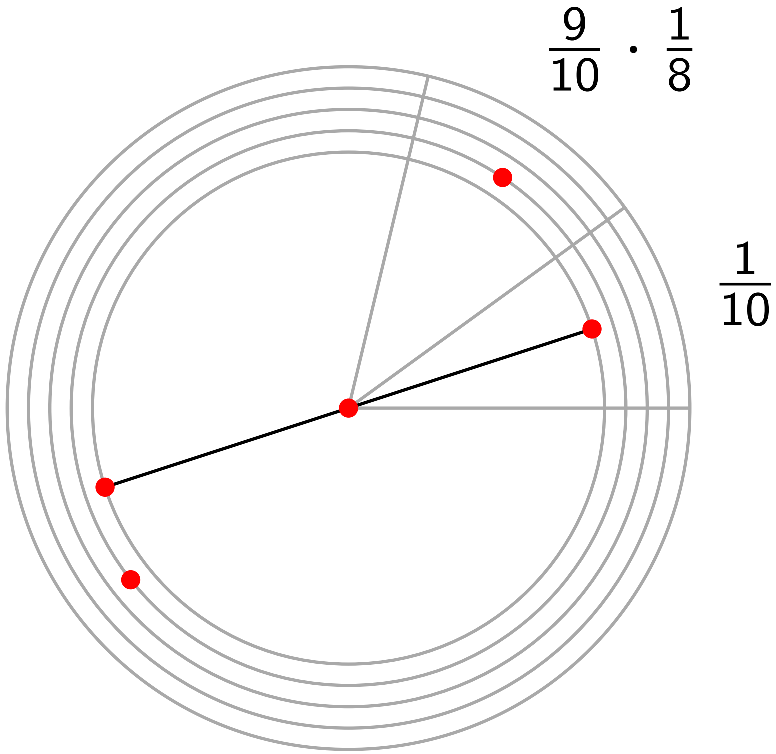




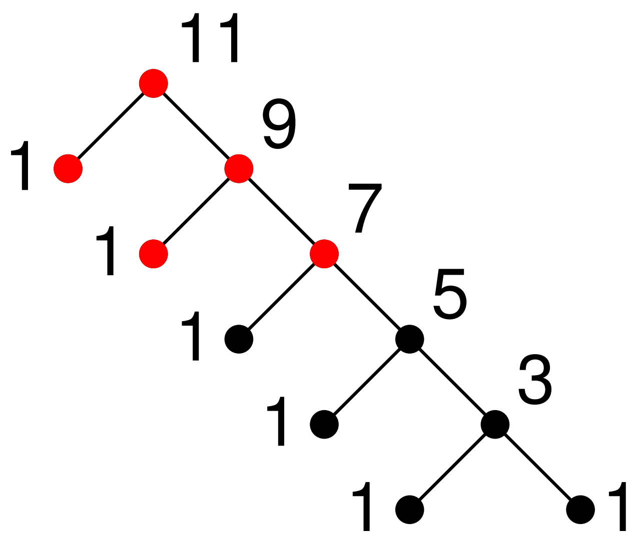
Institute for Theoretical Informatics
Algorithmics Group I

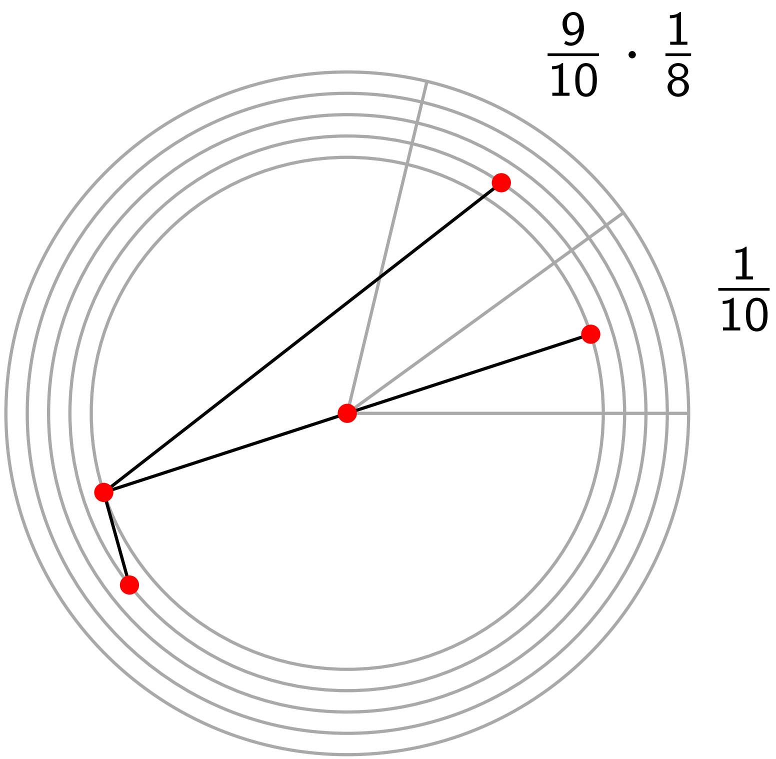
Beispiel Radiallayout



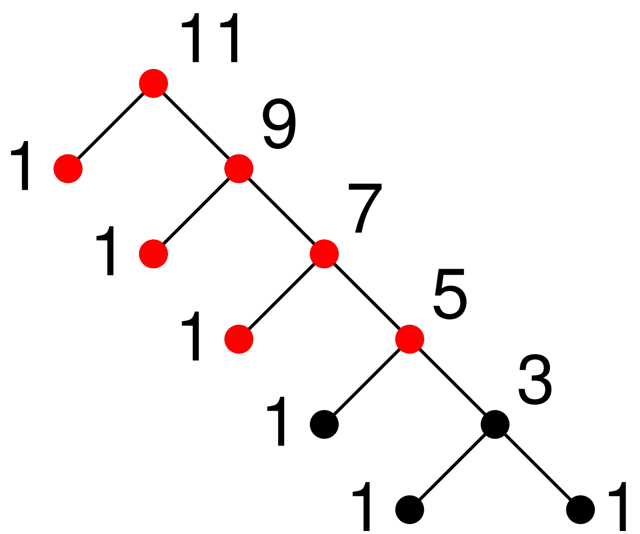


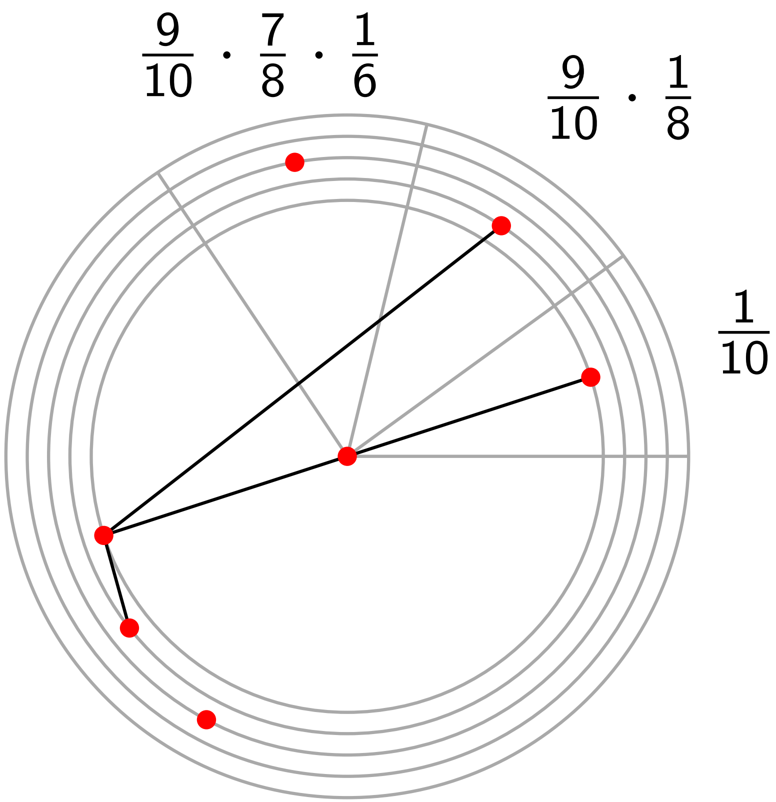
Beispiel Radiallayout



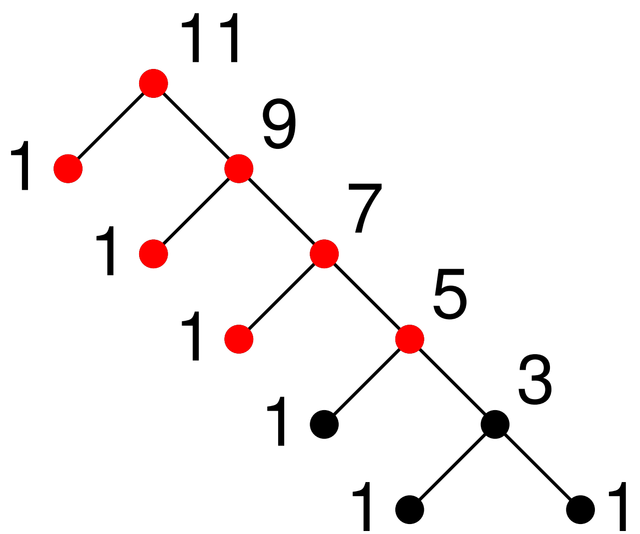


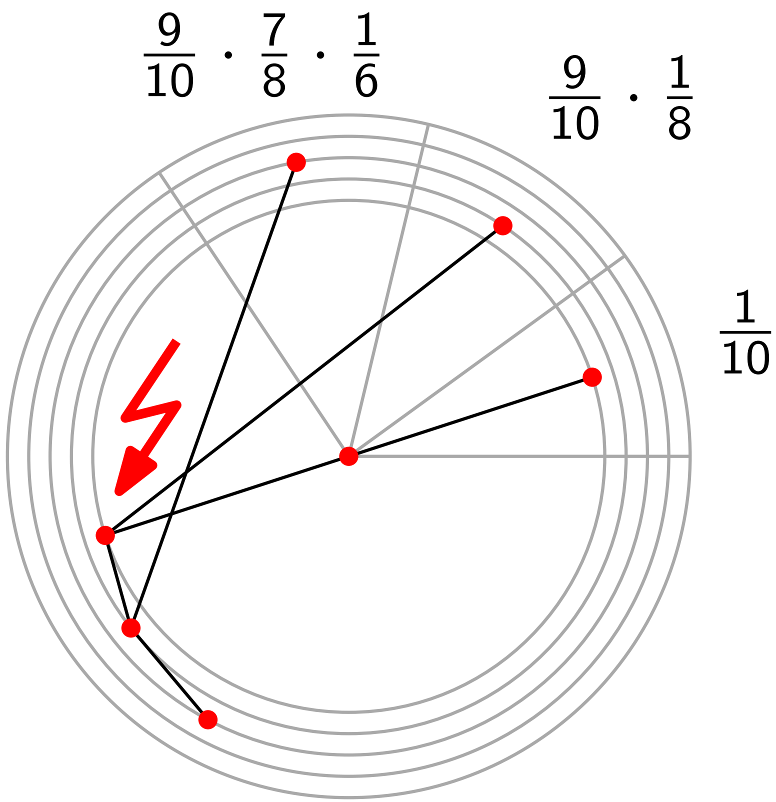
Beispiel Radiallayout



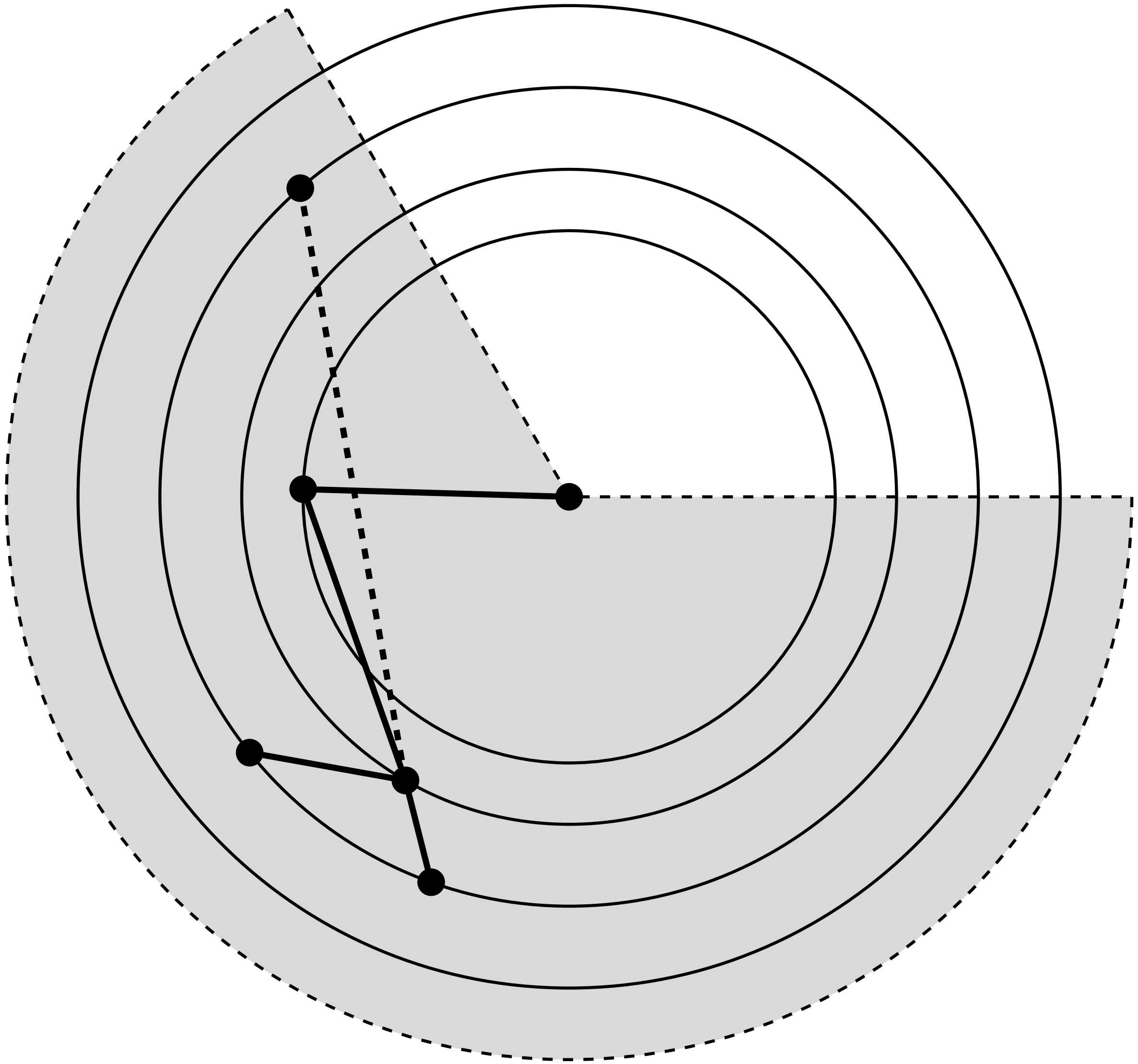


Beispiel Radiallayout



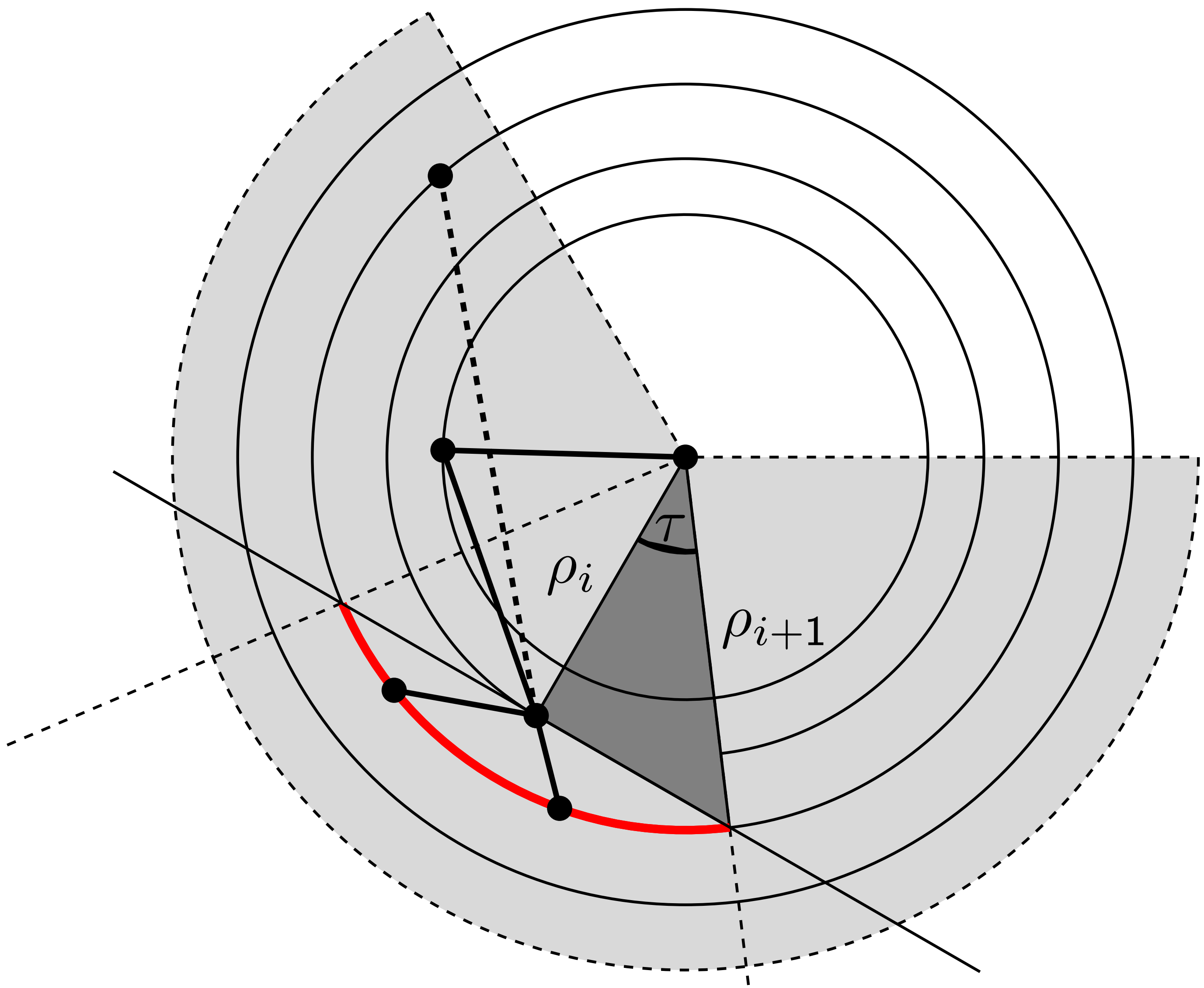


Verlassen des Kreisringsektors



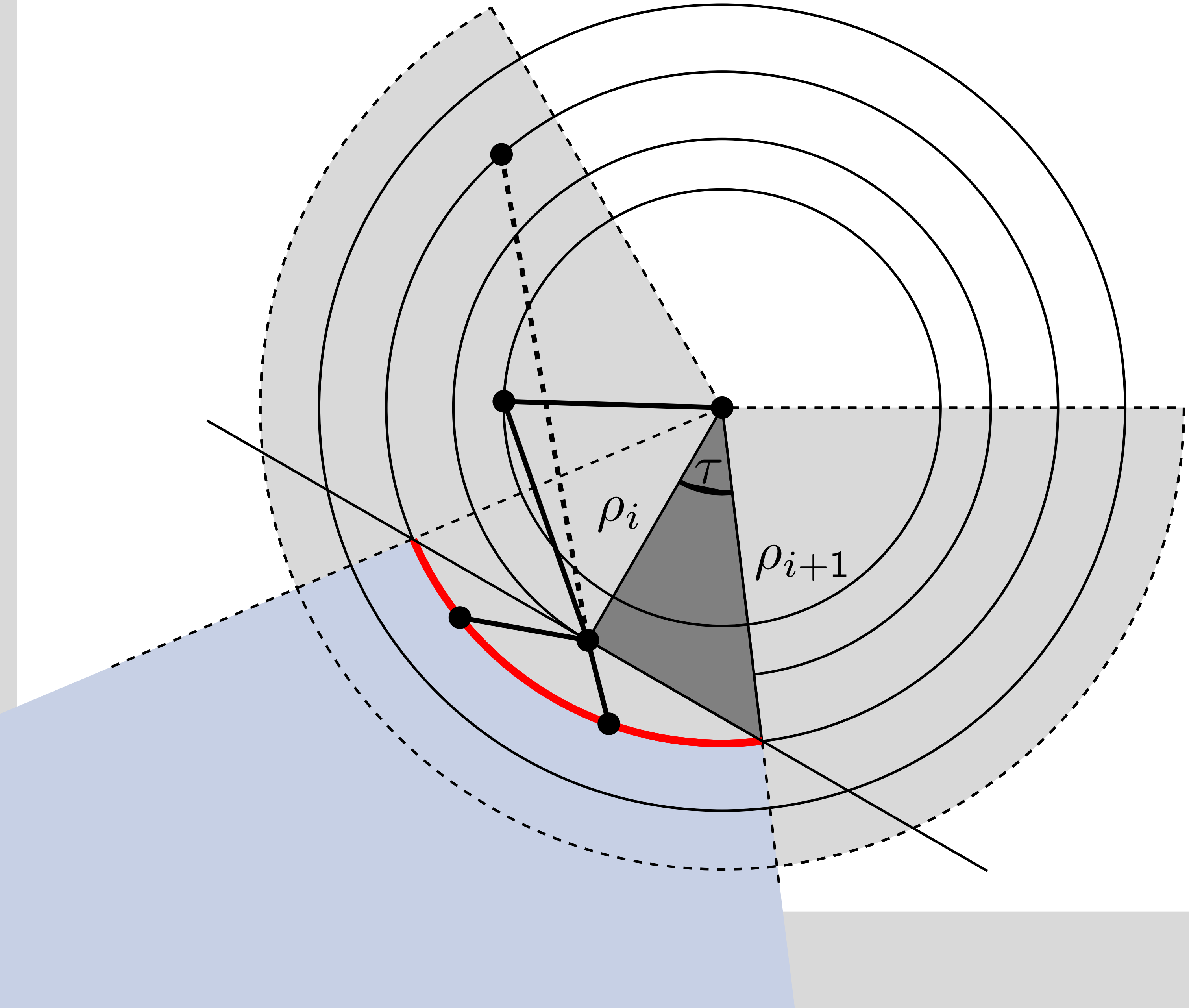
Institute for Theoretical Informatics
Algorithmics Group I

Verlassen des Kreisringsektors



$$\cos \tau = \frac{\rho_i}{\rho_{i+1}}$$

Verlassen des Kreisringsektors



$$\cos\tau = \frac{\rho_i}{\rho_{i+1}}$$

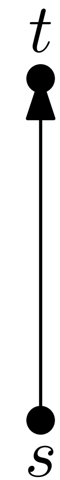
Serien-parallele Graphen

Institute for Theoretical Informatics
Algorithmics Group I

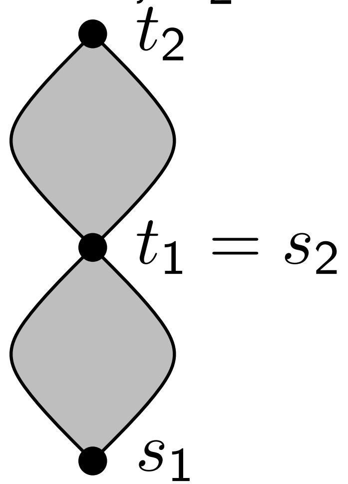
Serien-parallele Graphen

Graph G heißt serien-parallel, wenn er

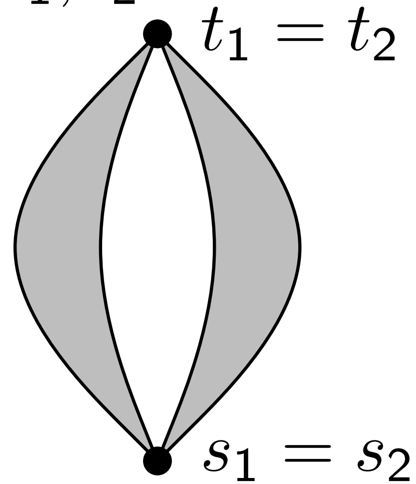
- aus genau zwei Knoten (Quelle s, Senke t sowie der Kante (s, t) besteht oder
- aus zwei serien-parallelen Graphen G_1, G_2 mit Quellen s_1, s_2 und Senken t_1, t_2 durch eine der folgenden Kombinationen hervorgeht



serielle Komposition:
Identifiziere t_1 und s_2,
s_1 neue Quelle, T_2 neue Senke



parallele Komposition:
Identifiziere s_1, s_2 als neue Quelle
Identifiziere t_1, t_2 als neue Senke



Serien-parallele Graphen II

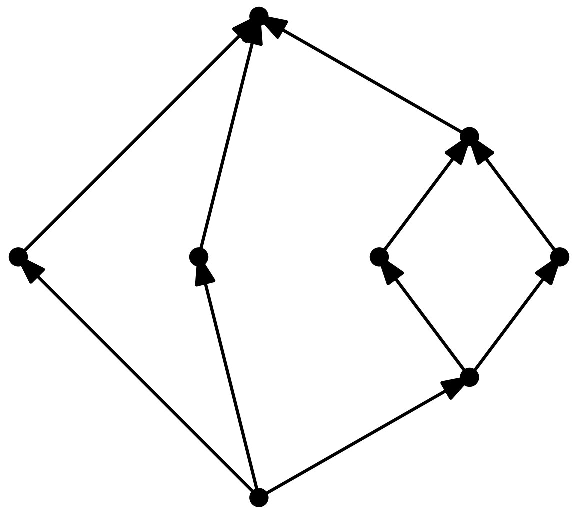
Lemma

Serien-parallele Graphen sind azyklisch und planar.

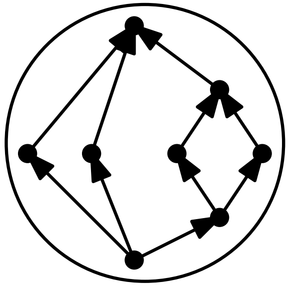
Beschreibung des rekursiven Aufbaus durch binären Baum:

- Blätter sind Kanten (Q-Knoten)
- Innere Knoten sind S- oder P-Knoten

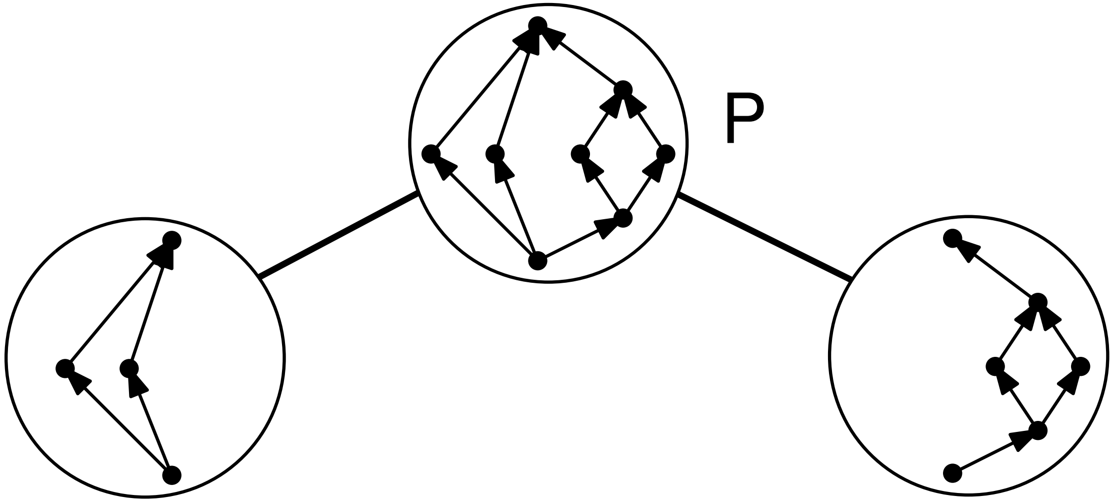
(vgl. SPQR-Baum)



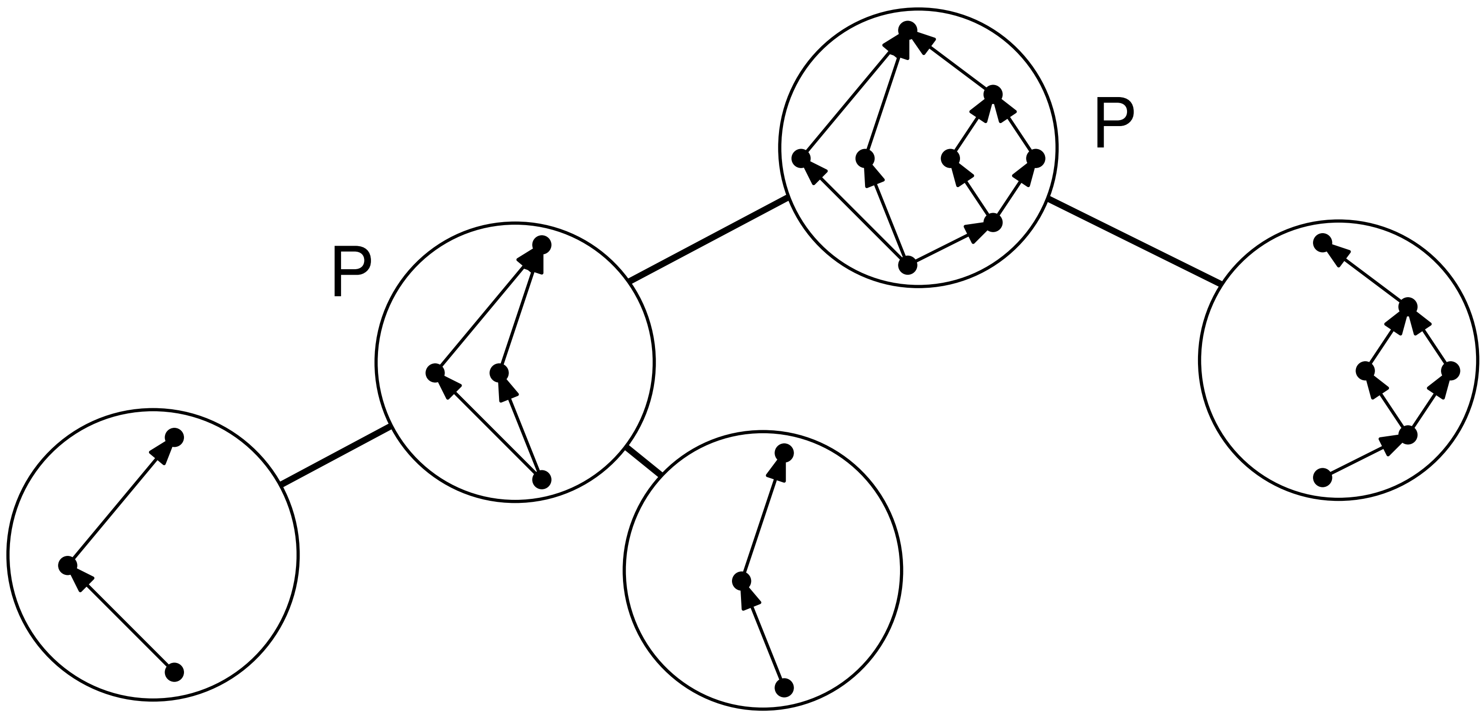
Serienparallele Graphen: Dekompositionsbaum



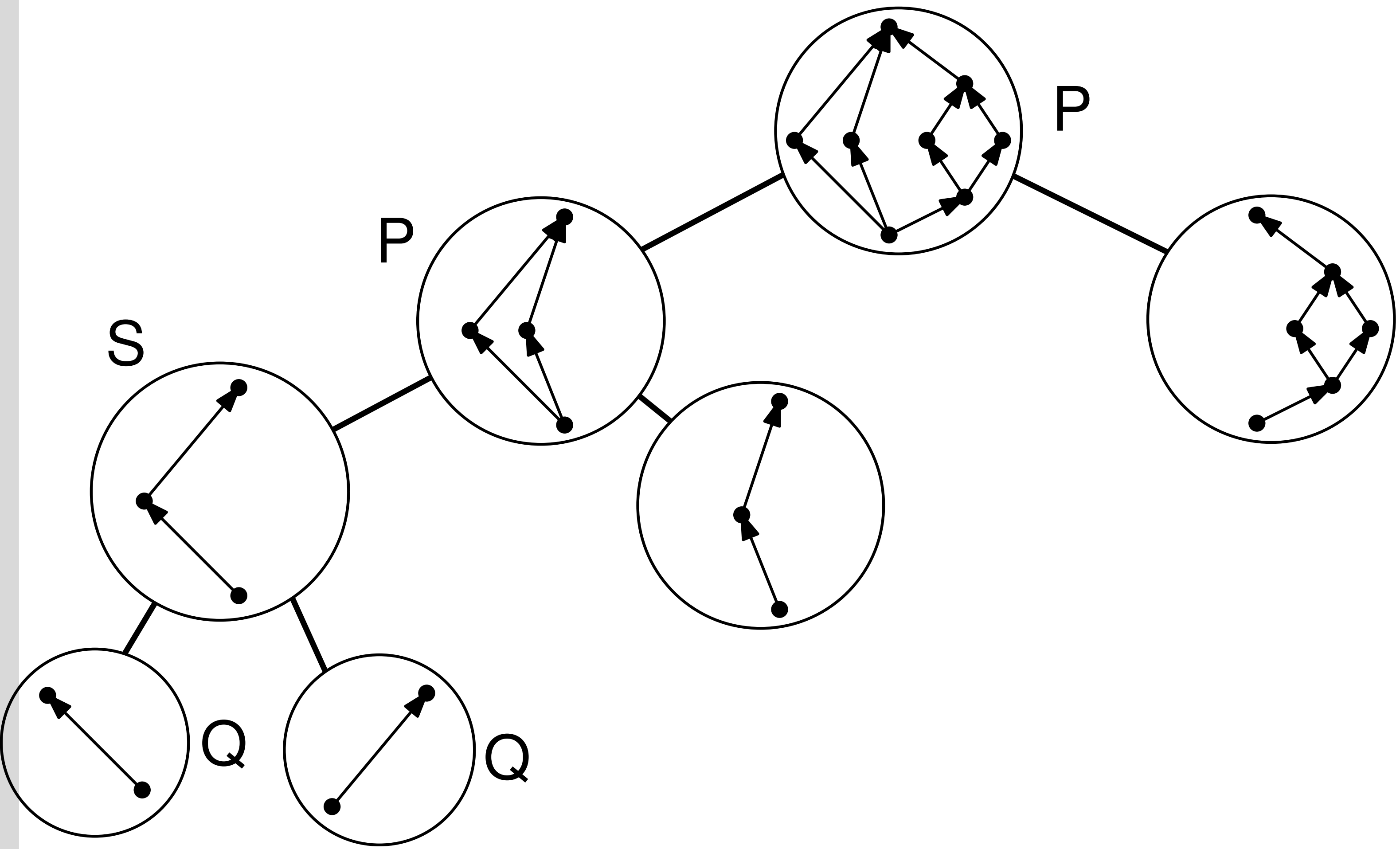
Serienparallele Graphen: Dekompositionsbaum



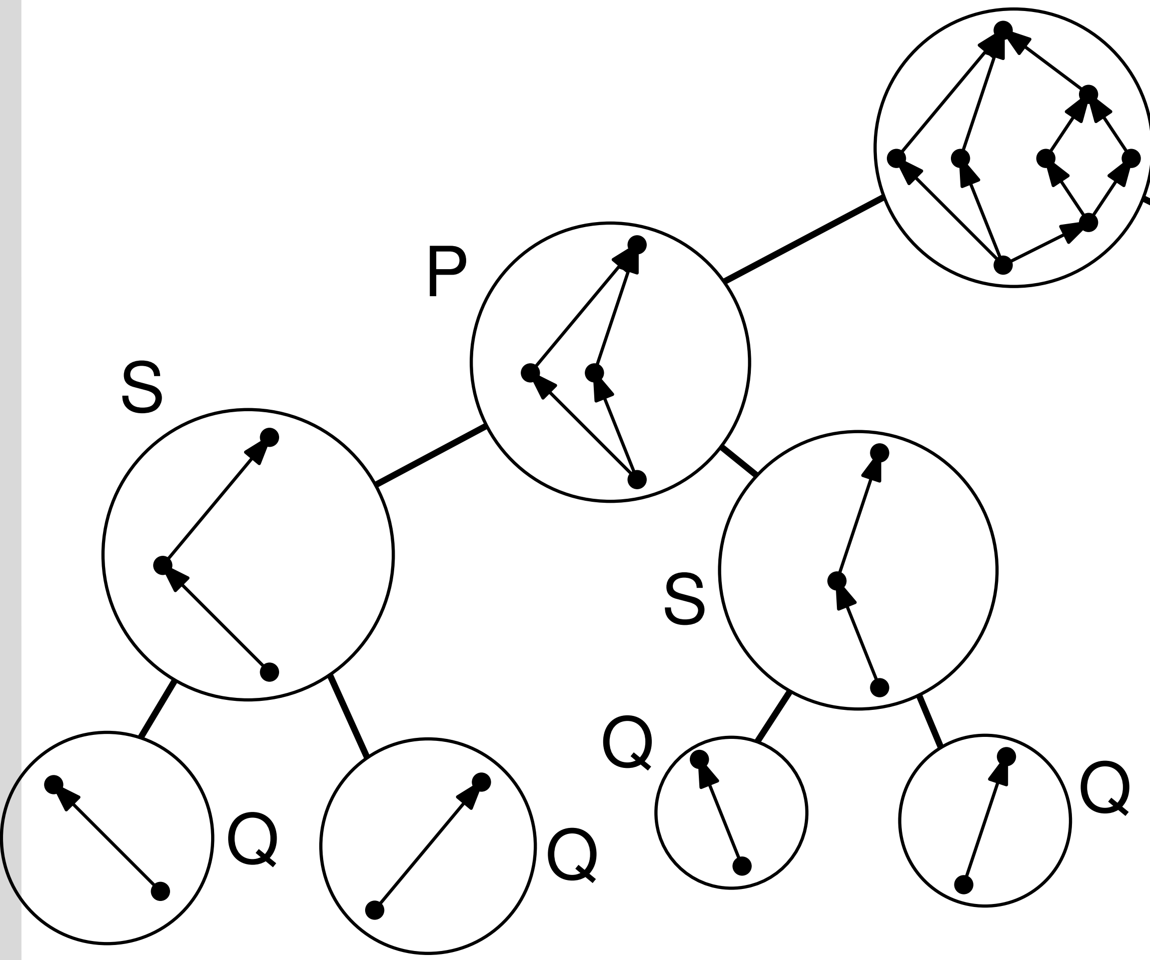
Serienparallele Graphen: Dekompositionsbaum



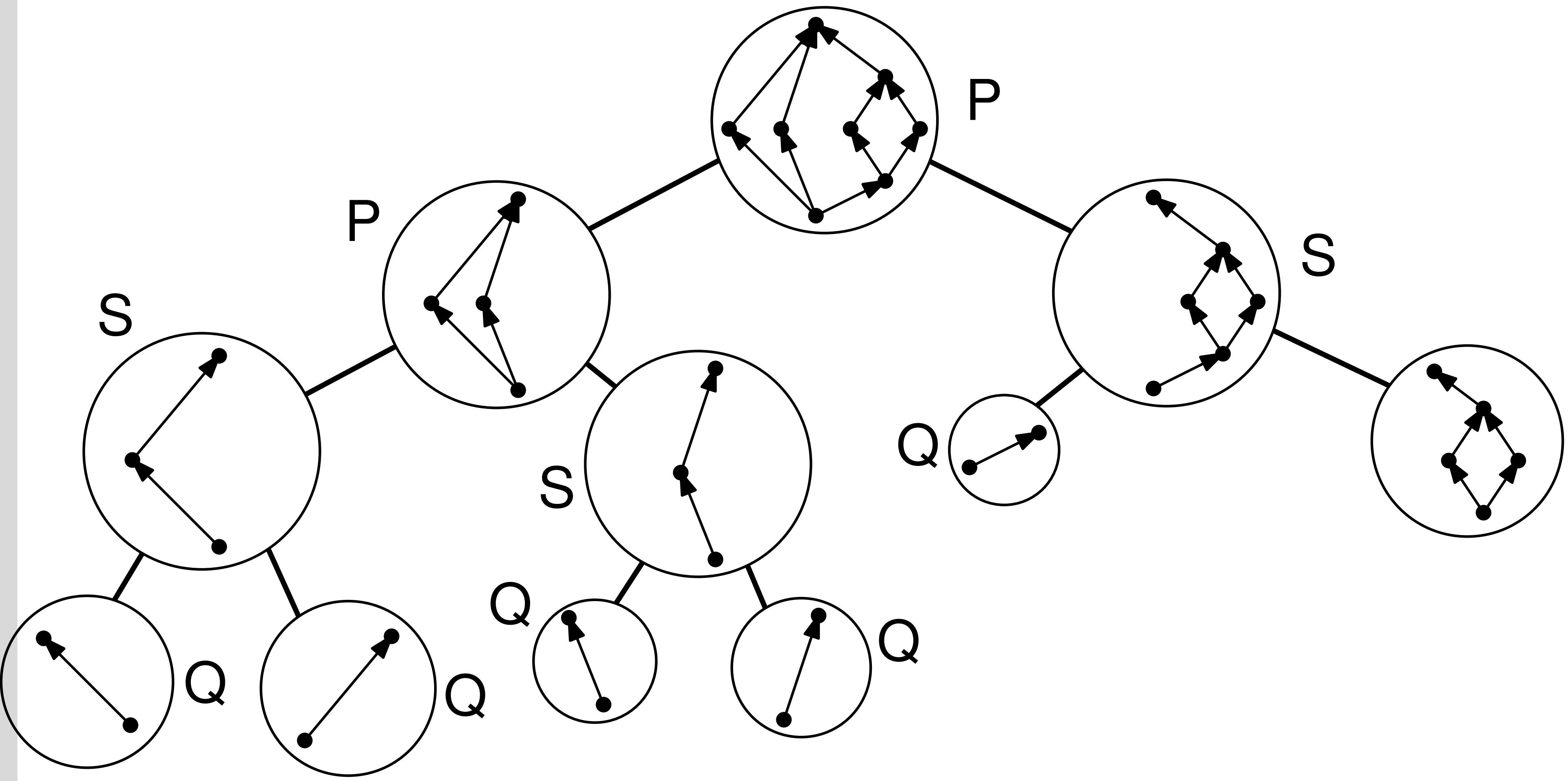
Serienparallele Graphen: Dekompositionsbaum



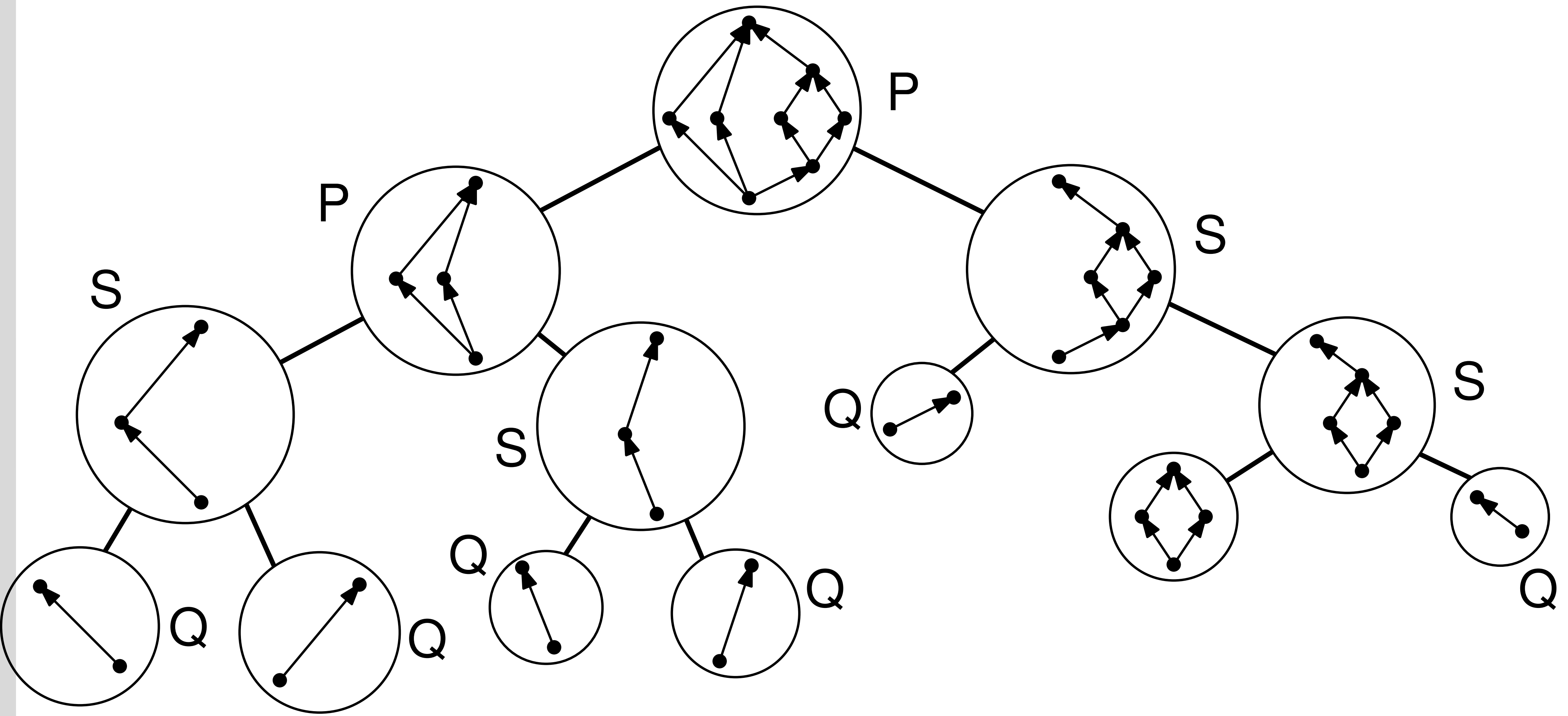
Serienparallele Graphen: Dekompositionsbaum



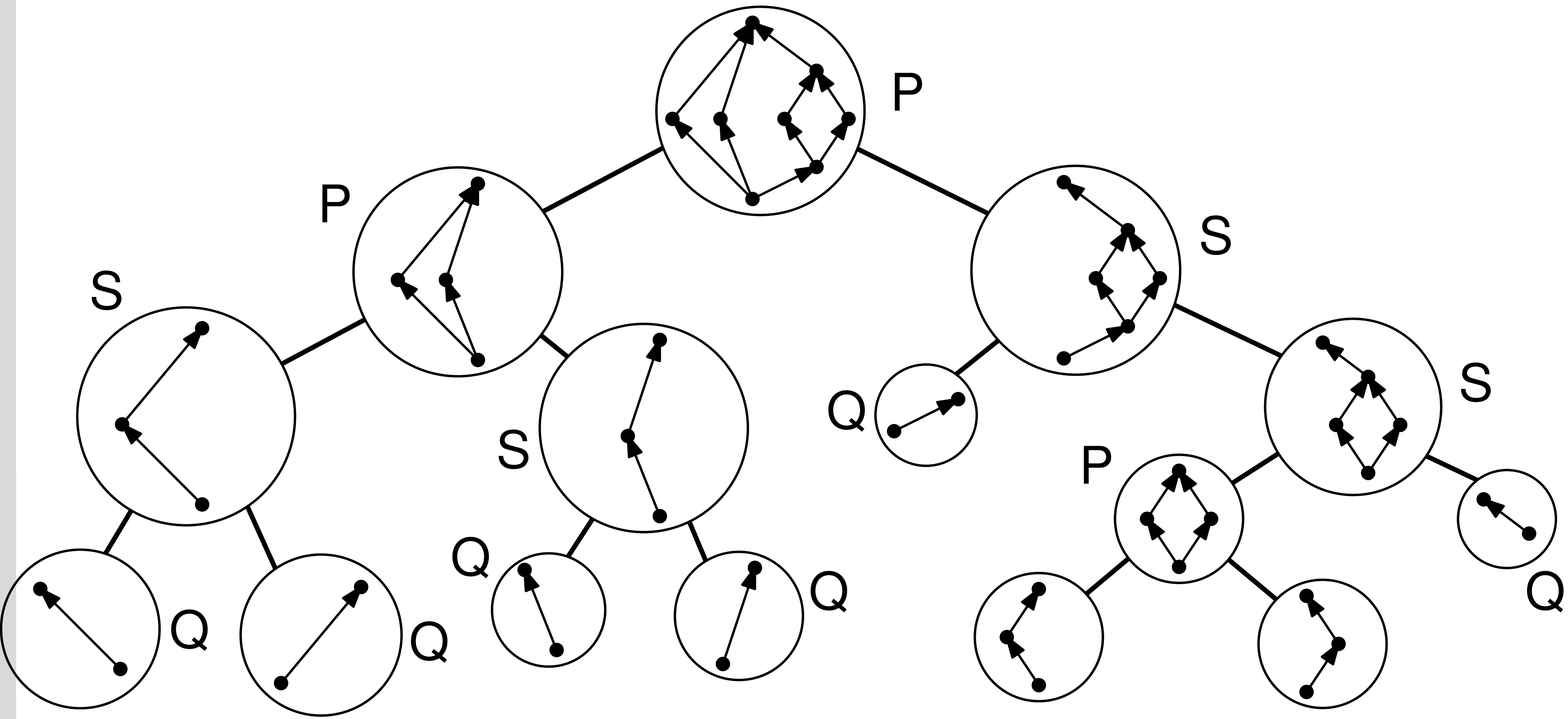
Serienparallele Graphen: Dekompositionsbaum



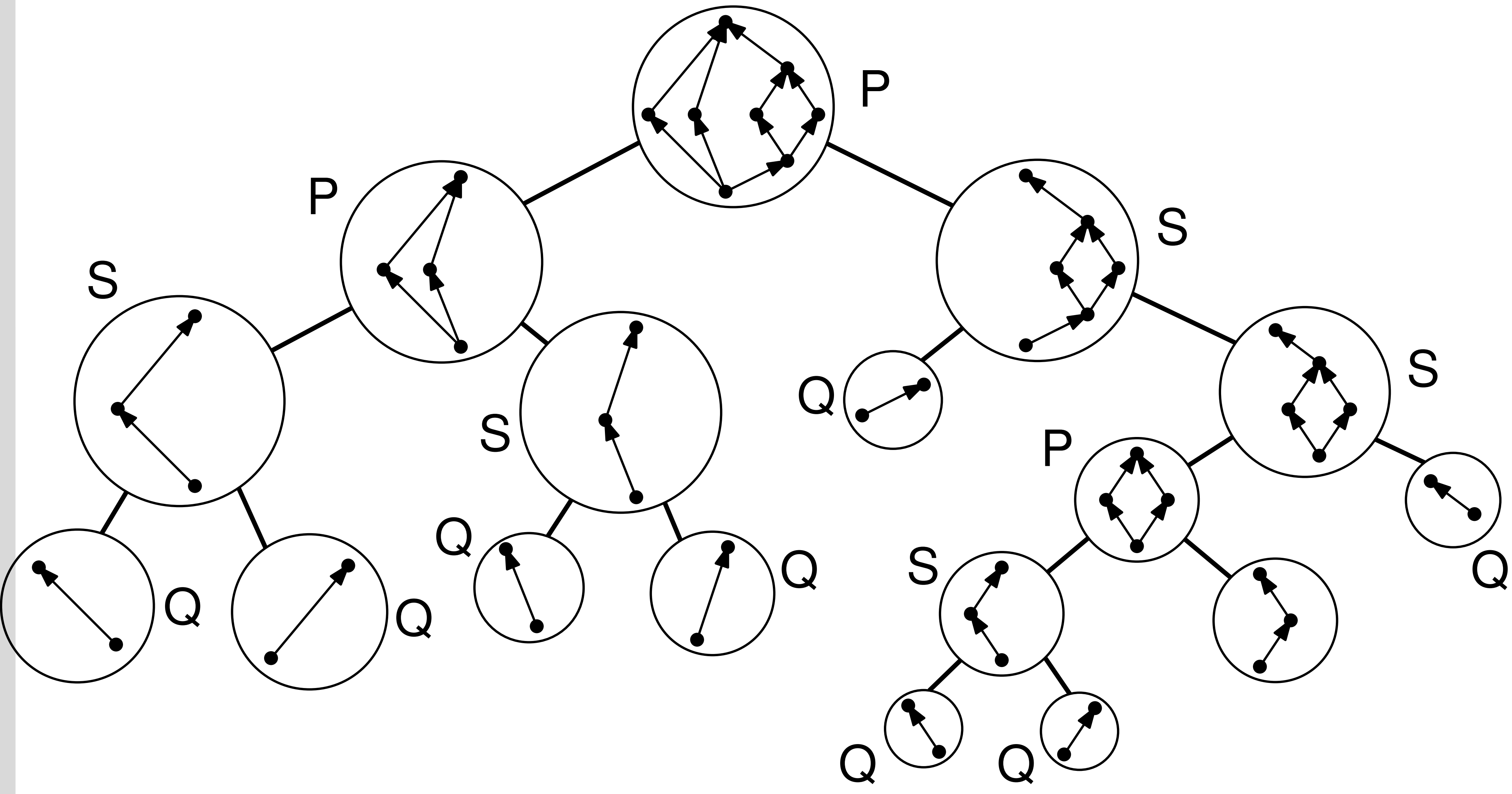
Serienparallele Graphen: Dekompositionsbaum



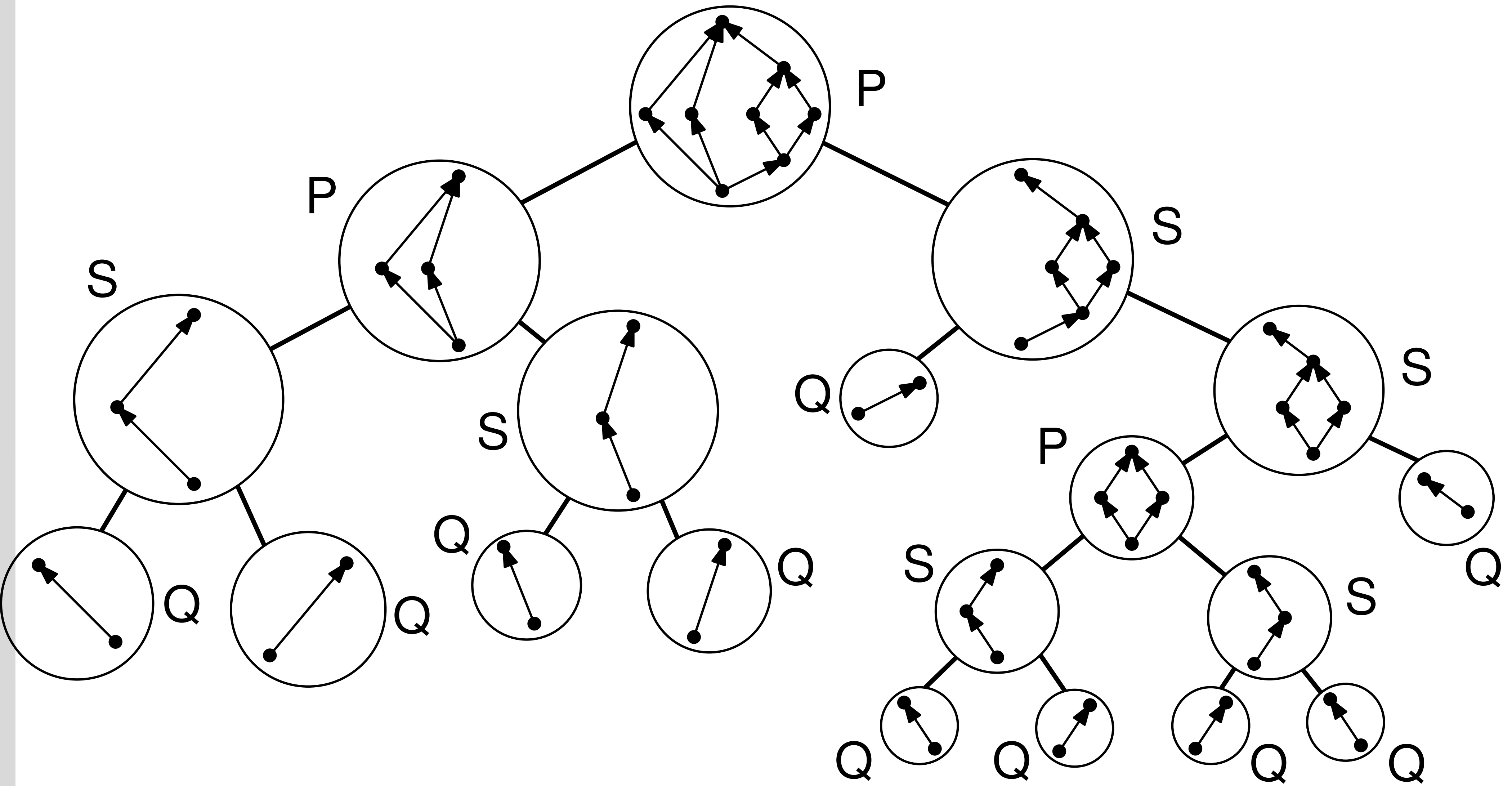
Serienparallele Graphen: Dekompositionsbaum



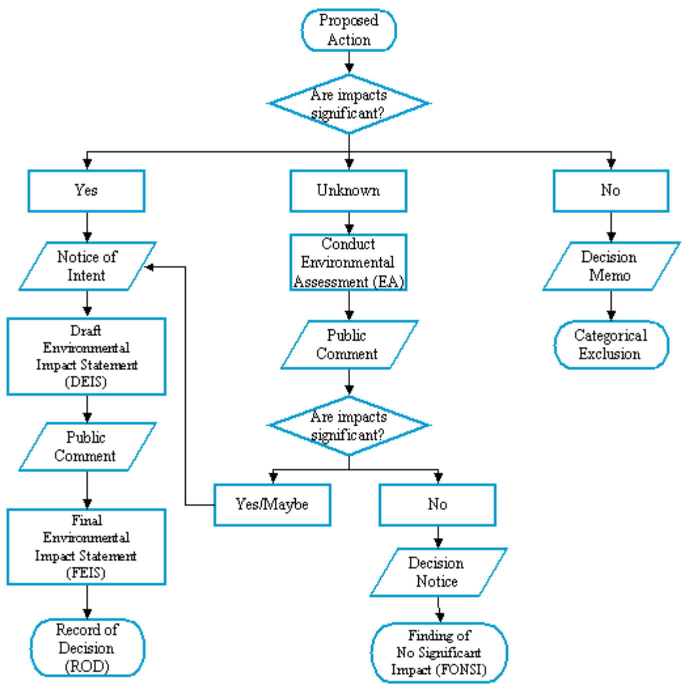
Serienparallele Graphen: Dekompositionsbaum



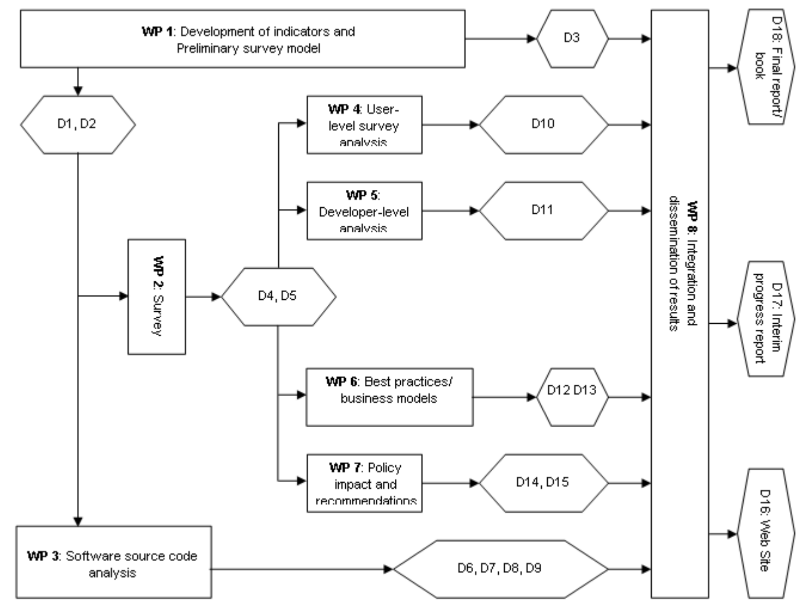
Serienparallele Graphen: Dekompositionsbaum



SP-Graphen in Anwendungen



Ablaufdiagramme



PERT-Diagramme

(Program Evaluation and Review Technique)

Außerdem: Linearzeitalgorithmen für sonst NP-vollständige Probleme (z.B. Maximum Independent Set)

Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

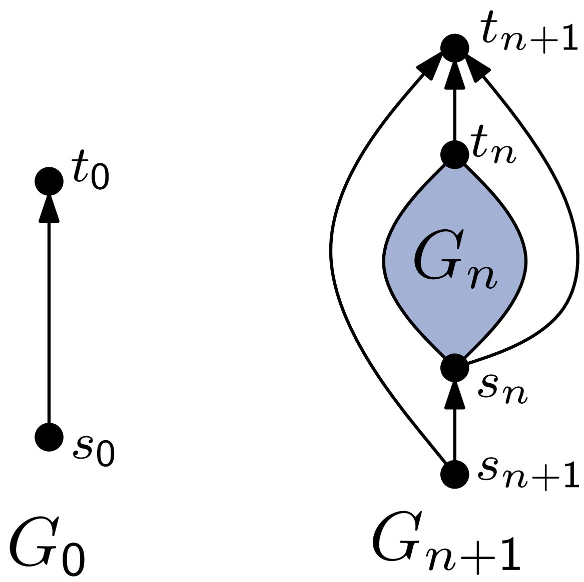
Institute for Theoretical Informatics
Algorithmics Group I

Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:

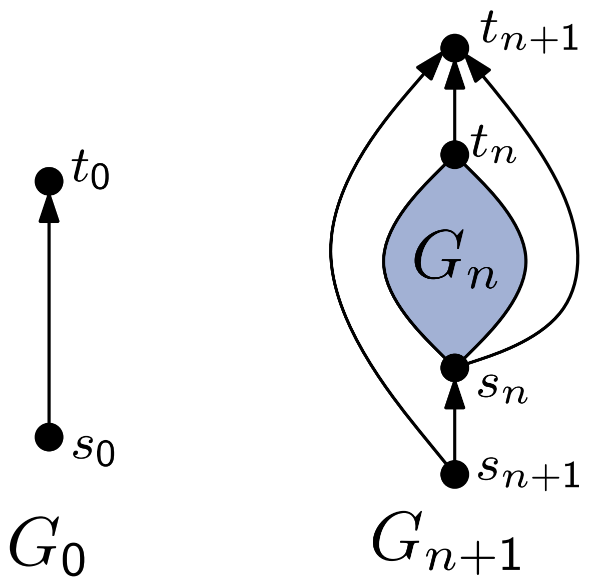


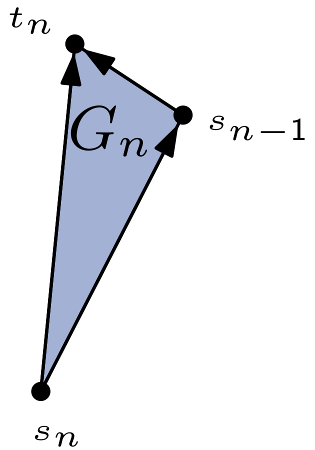
Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:



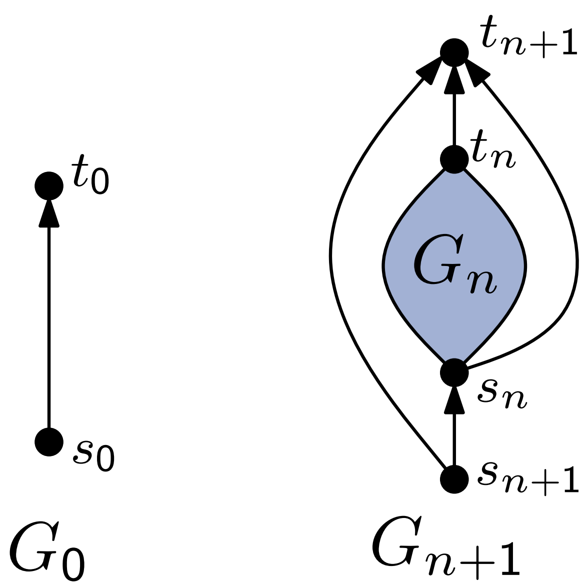


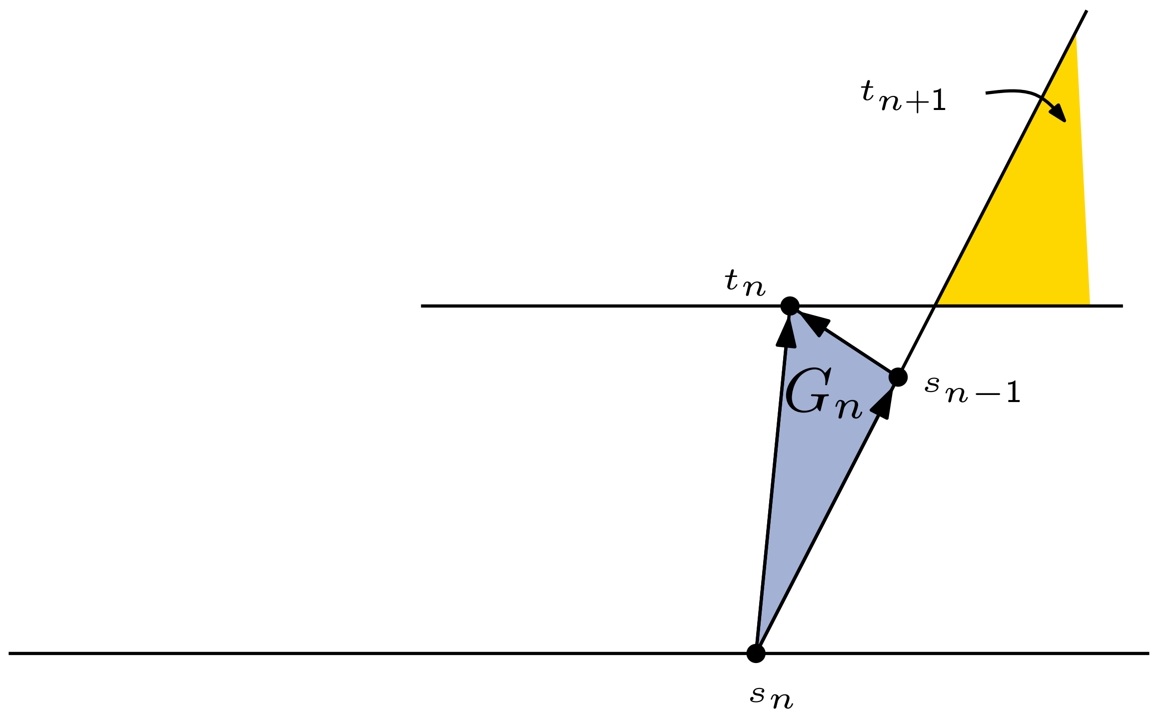
Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:



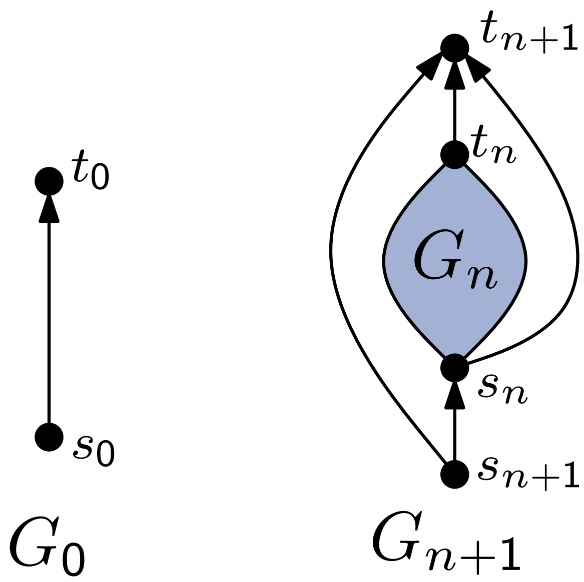


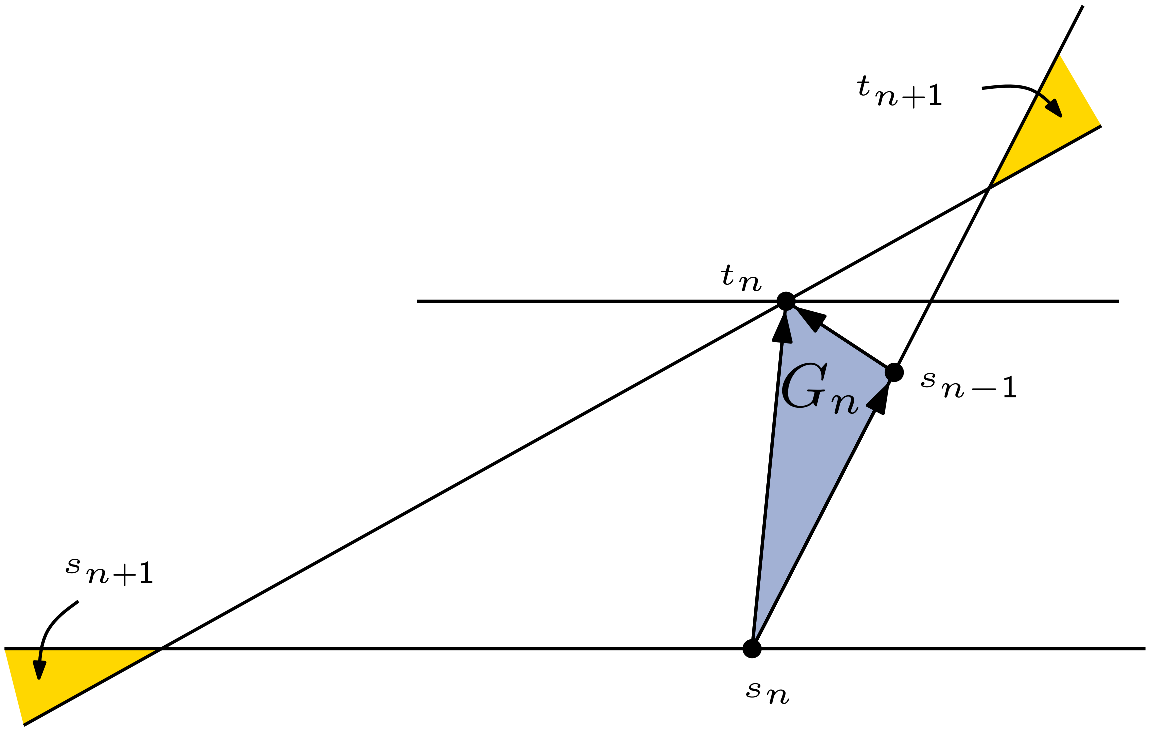
Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:





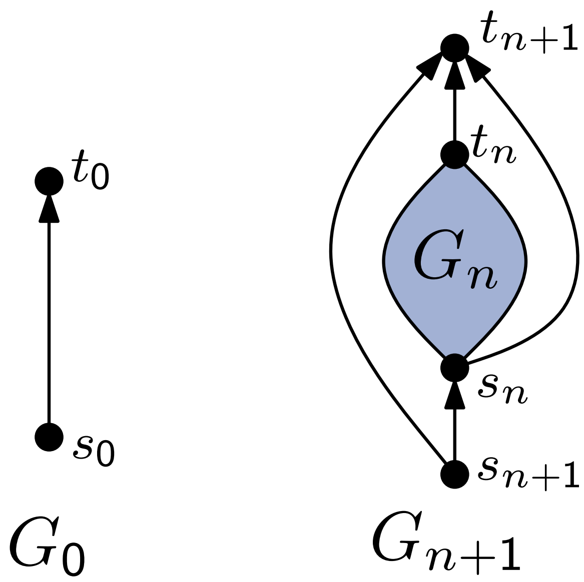
Institute for Theoretical Informatics
Algorithmics Group I

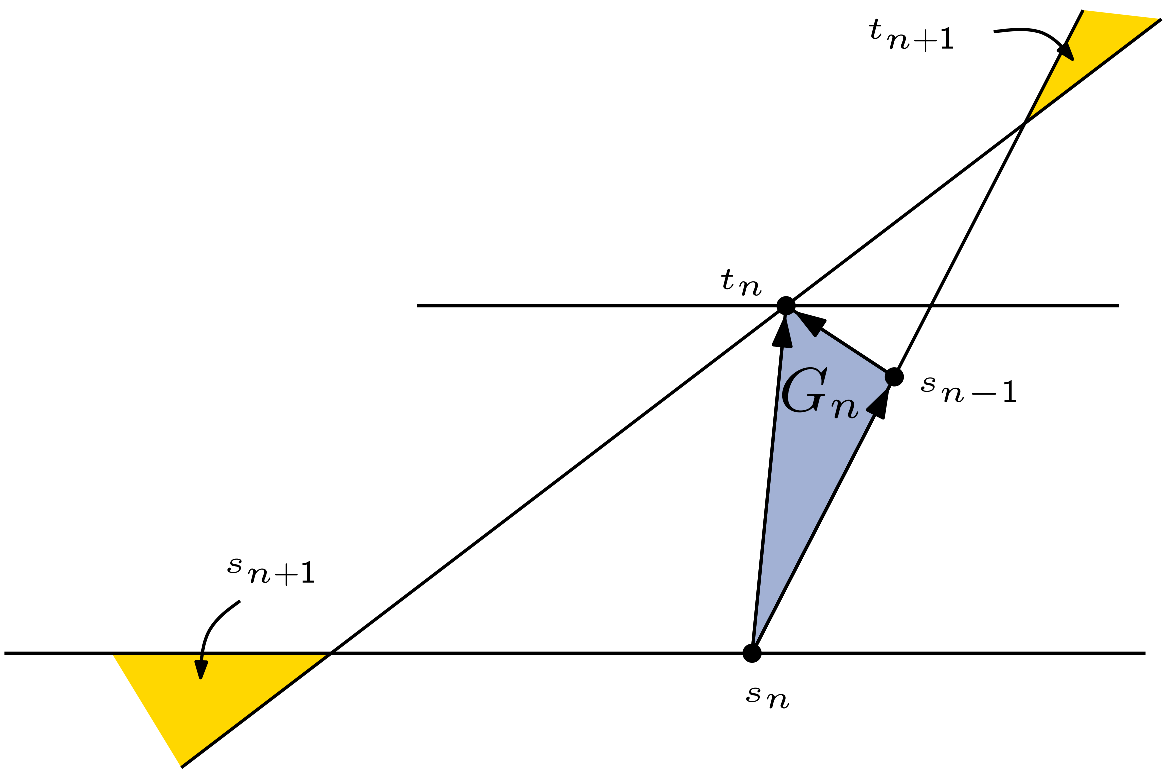
Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:





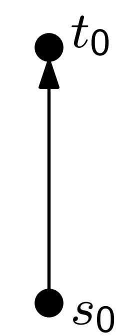
Institute for Theoretical Informatics
Algorithmics Group I

Untere Schranke für die Fläche

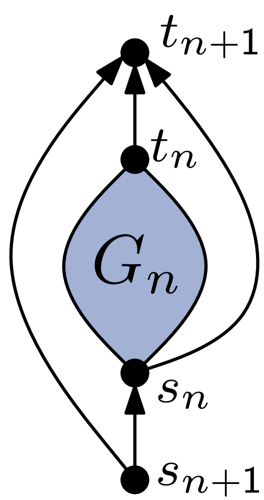
Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

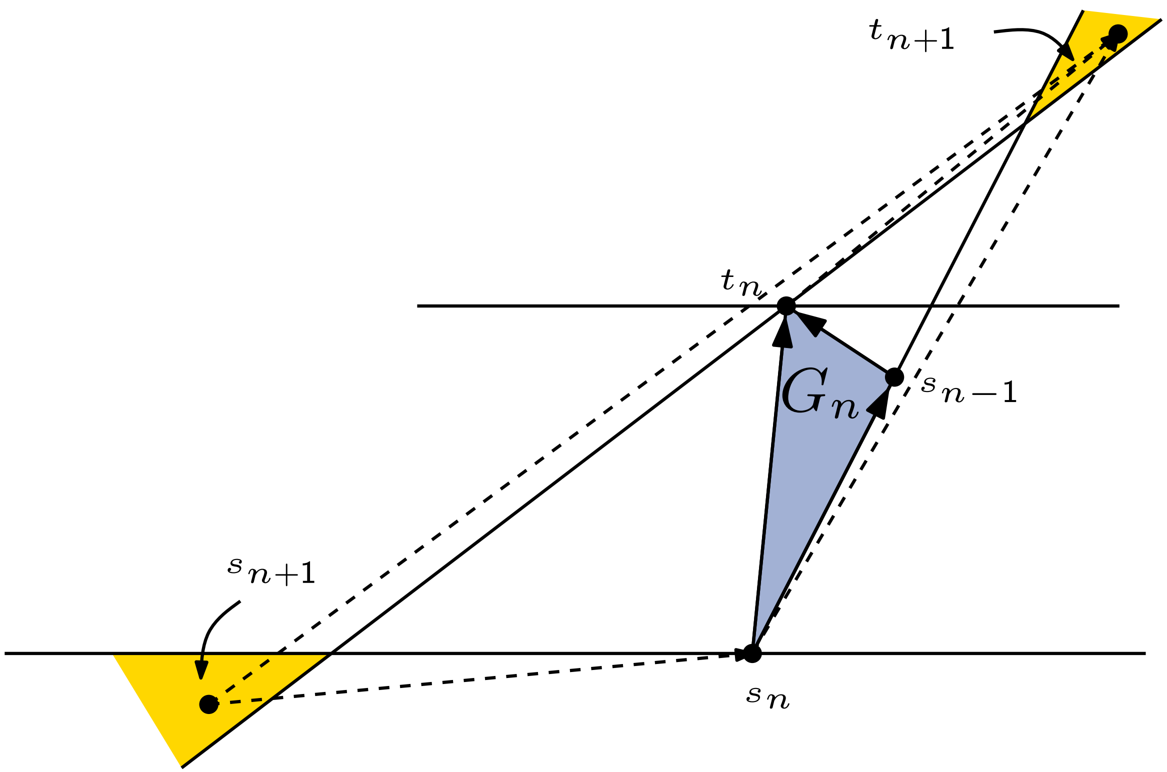
Beweis:



G_0



G_{n+1}

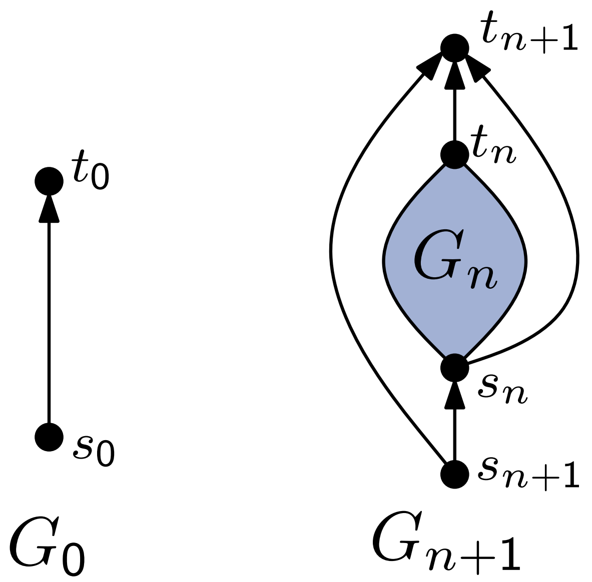


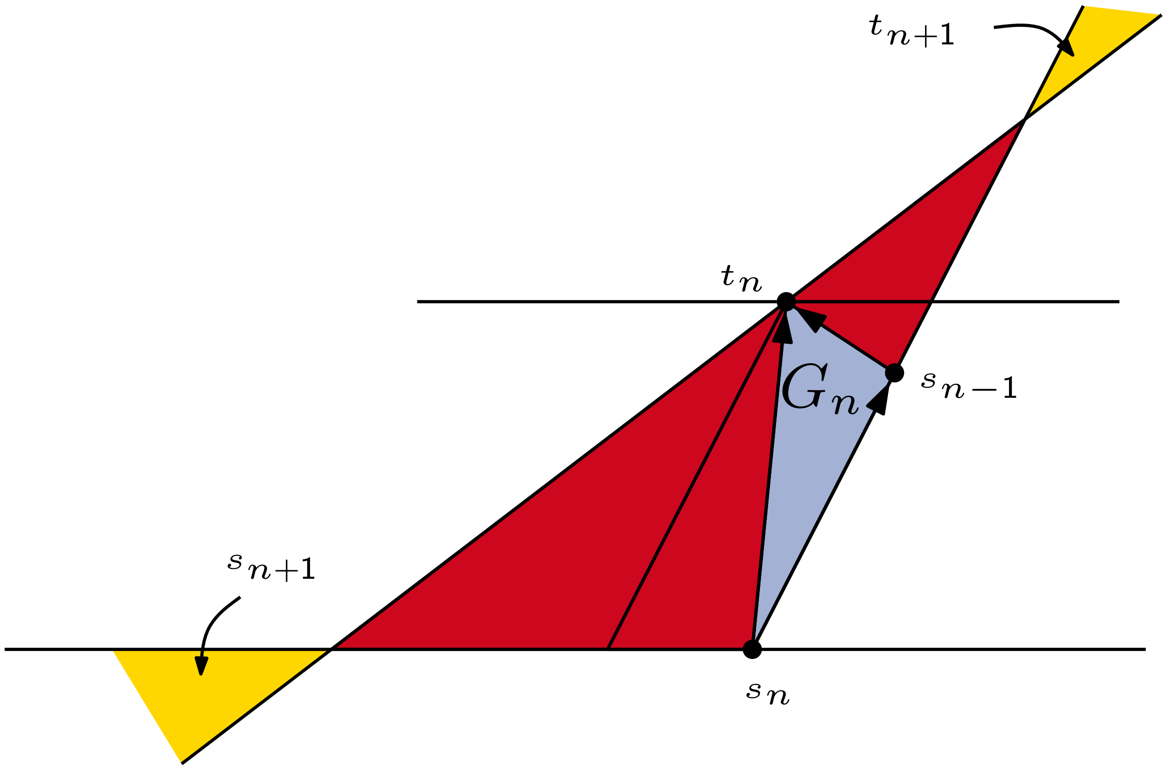
Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:



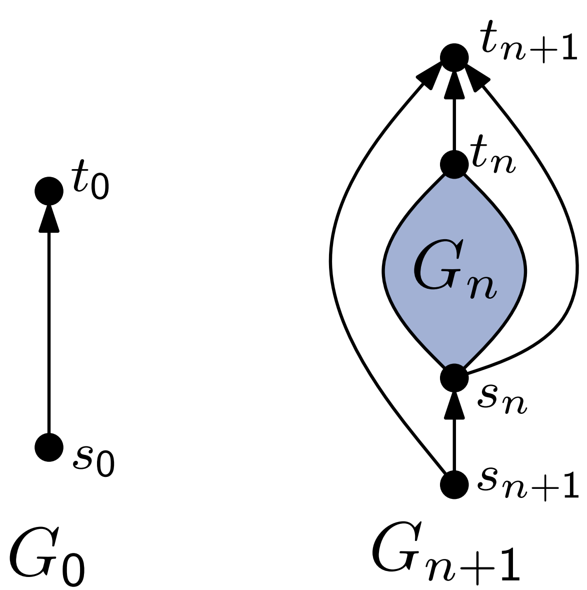


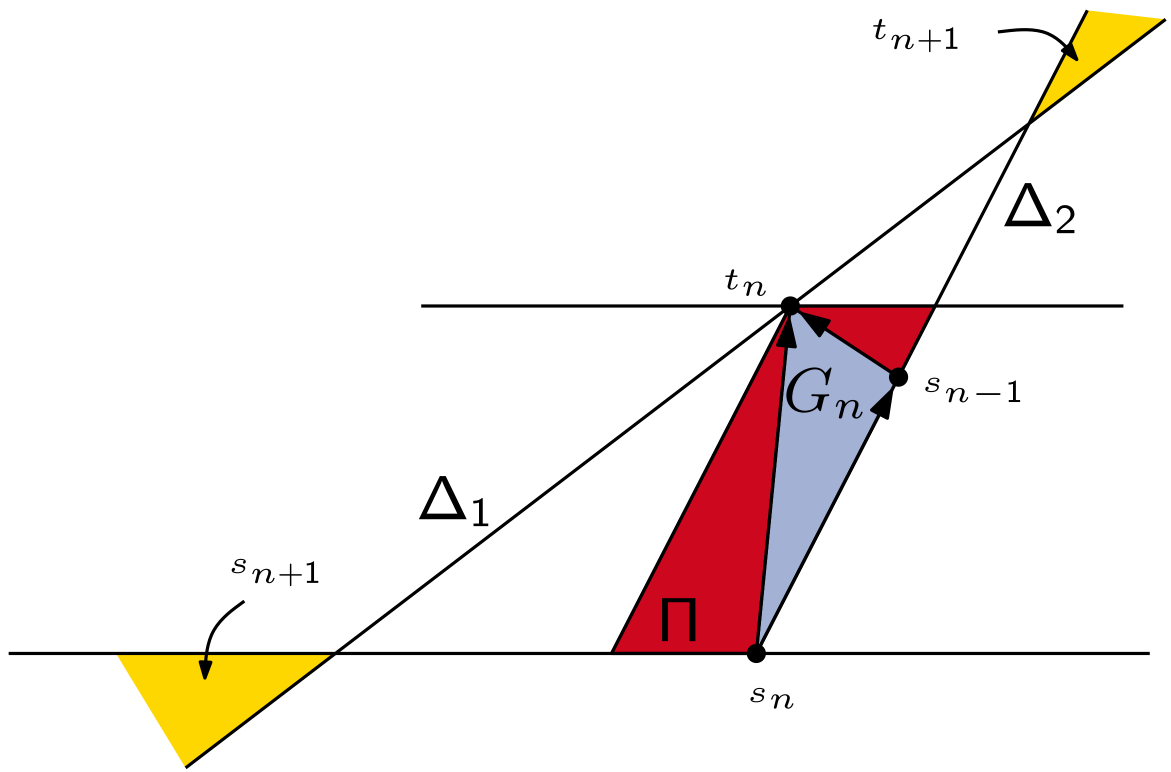
Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:



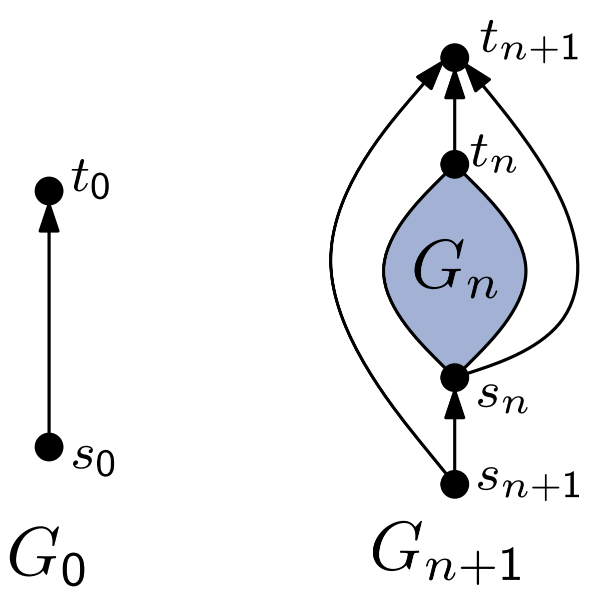


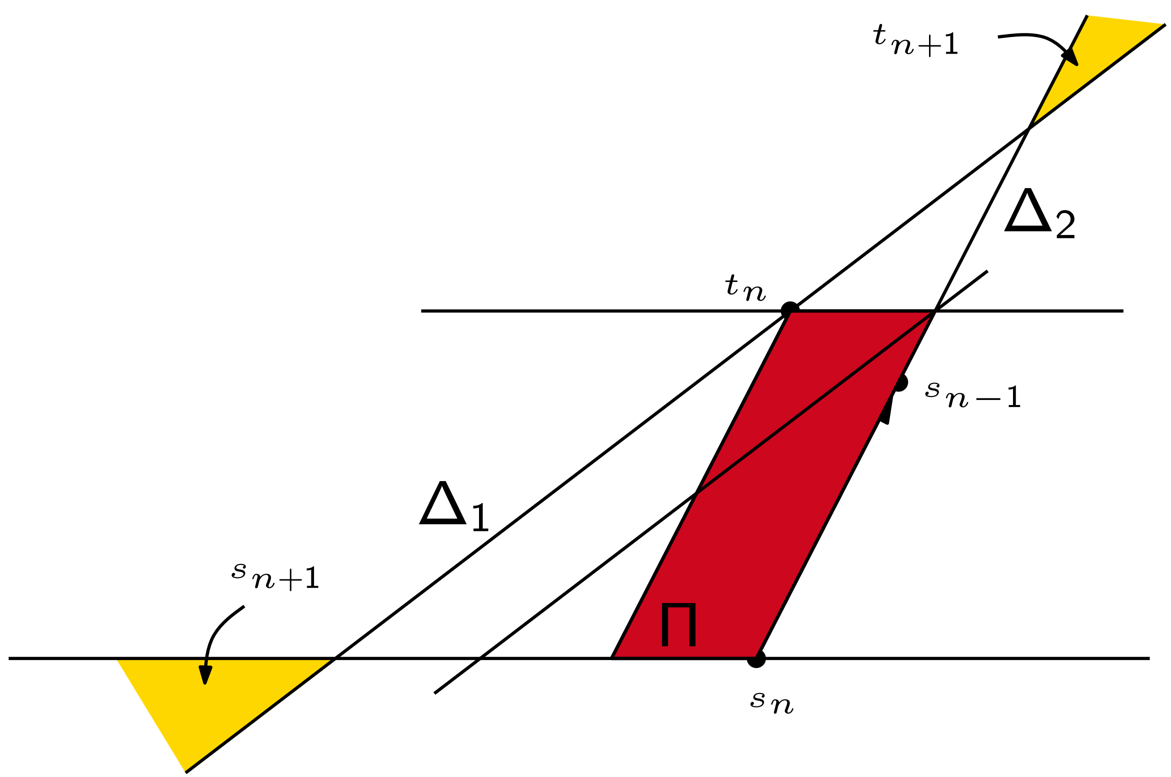
Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:





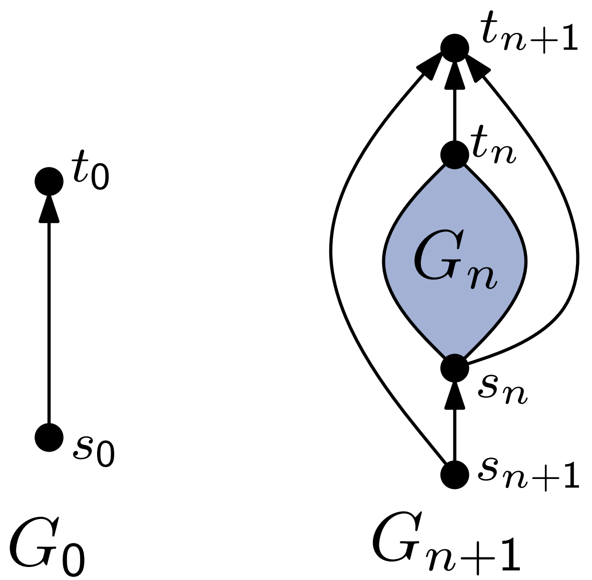
Institute for Theoretical Informatics
Algorithmics Group I

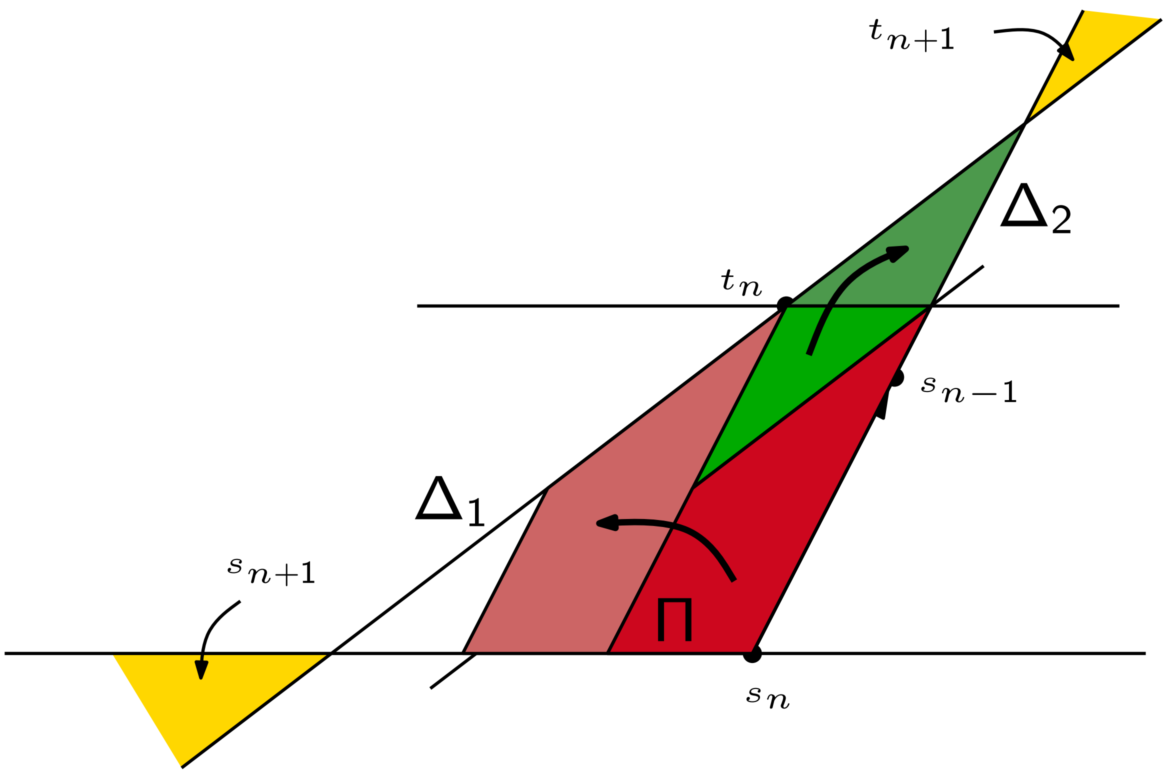
Untere Schranke für die Fläche

Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

Beweis:





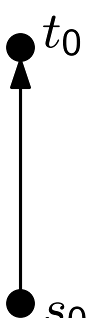
Institute for Theoretical Informatics
Algorithmics Group I

Untere Schranke für die Fläche

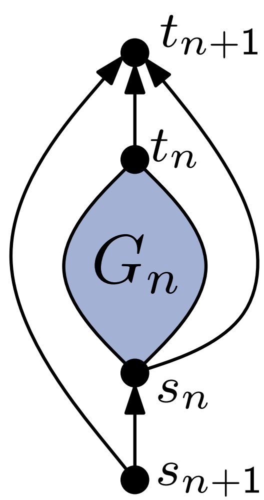
Satz

Jedes kreuzungsfreie Aufwärtslayout für geordnete einfache serien-parallele Graphen mit n Knoten benbötigt im worst case ein Gitter der Größe $\Omega(4^n)$.

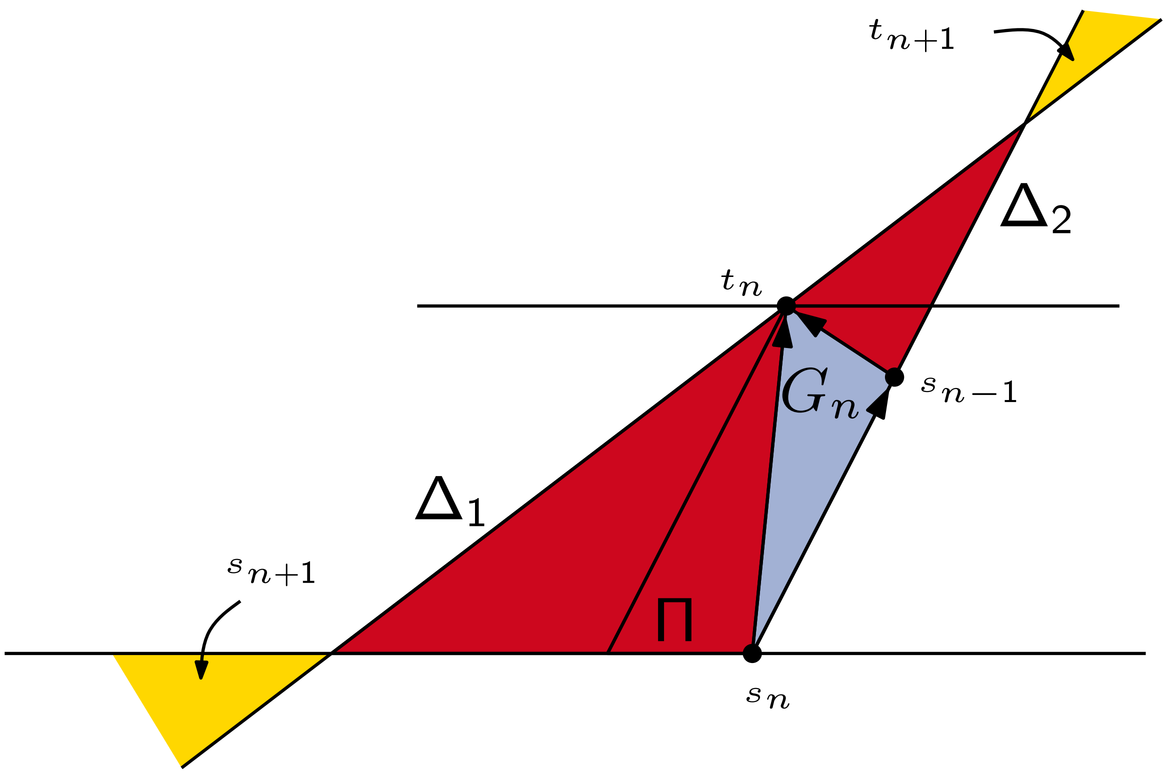
Beweis:



G_0



G_{n+1}



Linkslastige Ordnungen

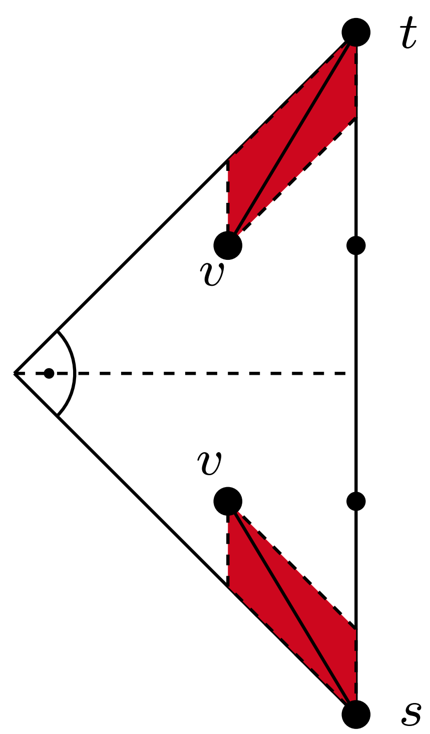
Ordnung heißt linkslastig, wenn Q-Knoten nur als rechte Nachfolger von P-Knoten vorkommen.

Satz

Wenn G serien-parallel, einfach und linkslastig geordnet, so besitzt G Zeichnung der Größe $O(n^2)$.

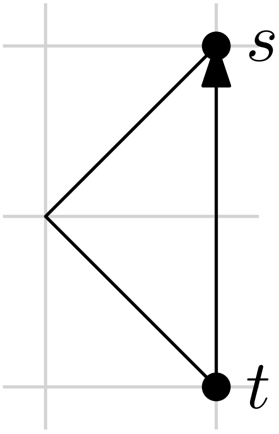
Komponenten des Dekompositionsbaums:

- Layout von G passt in rechtwinkliges, gleichschenkliges Dreieck mit vertikaler Basis, Schenkel nach links.
- Quelle in unterer Ecke, Senke in oberer Ecke, linke Ecke frei
- rechtester Nachbar von $s \neq t$ ($t \neq s$) liegt unterhalb (oberhalb) der Mitte
- v Nachbar der Quelle (Senke) $\Rightarrow$ kein Knoten liegt im Parallelogramm von v und s (t).



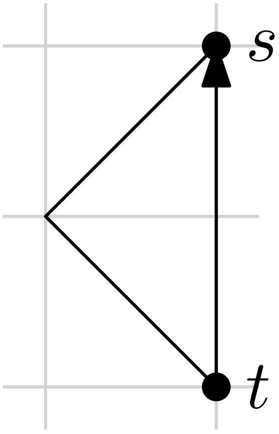
Konstruktion

Q-Knoten (Induktionsanfang):

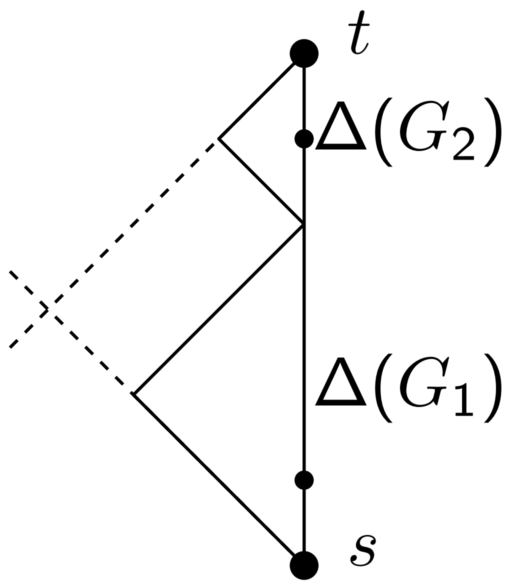


Konstruktion

Q-Knoten (Induktionsanfang):

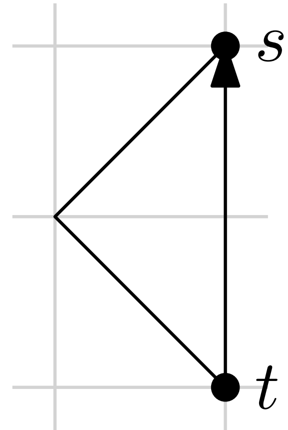


S-Knoten (serielle Komposition):

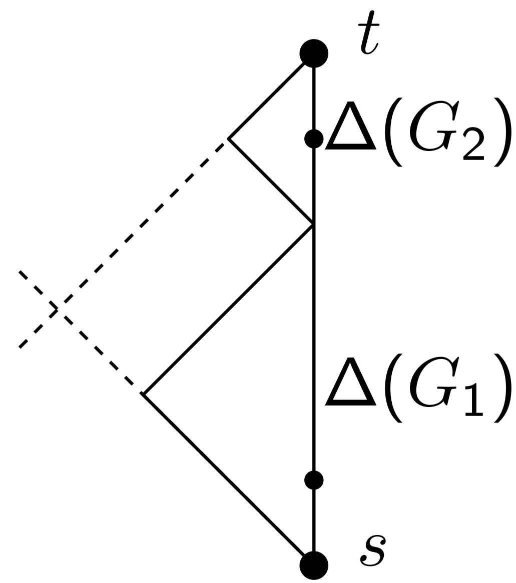


Konstruktion

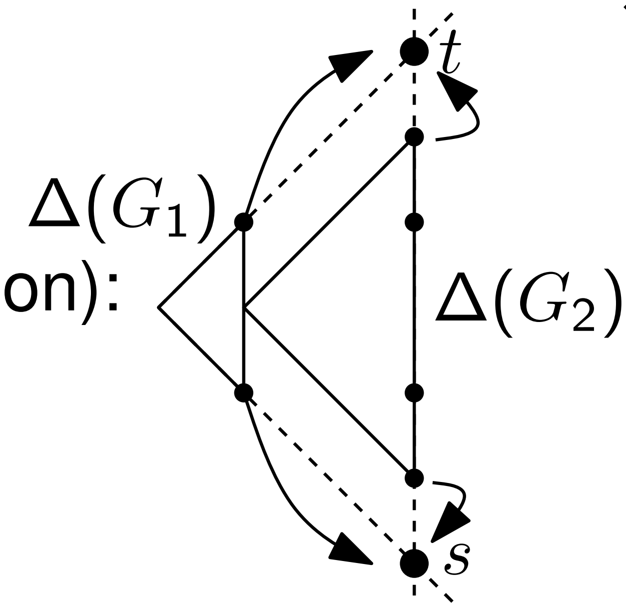
Q-Knoten (Induktionsanfang):



S-Knoten (serielle Komposition):

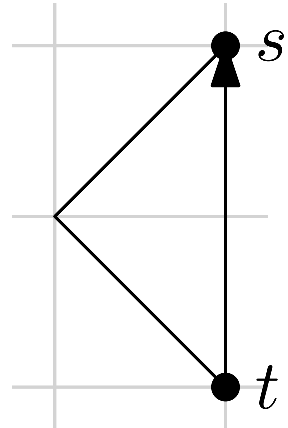


P-Knoten (parallele Komposition):

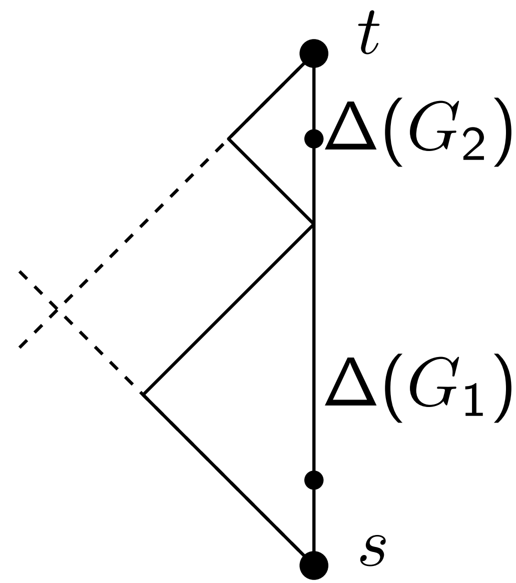


Konstruktion

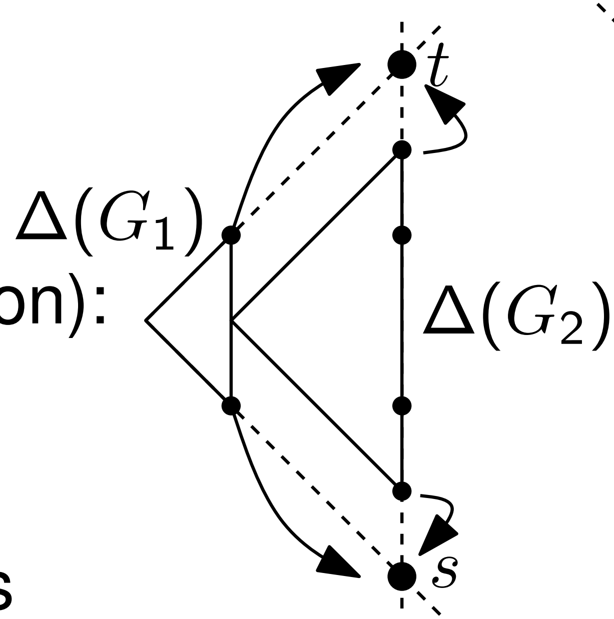
Q-Knoten (Induktionsanfang):



S-Knoten (serielle Komposition):



P-Knoten (parallele Komposition):



Eigenschaften:

- kreuzungsfrei und aufwärts
- höchstens quadratische Fläche