Algorithmen im Chip-Entwurf 11

Floorplanning

Andreas Koch
FG Eingebettete Systeme
und ihre Anwendungen
TU Darmstadt

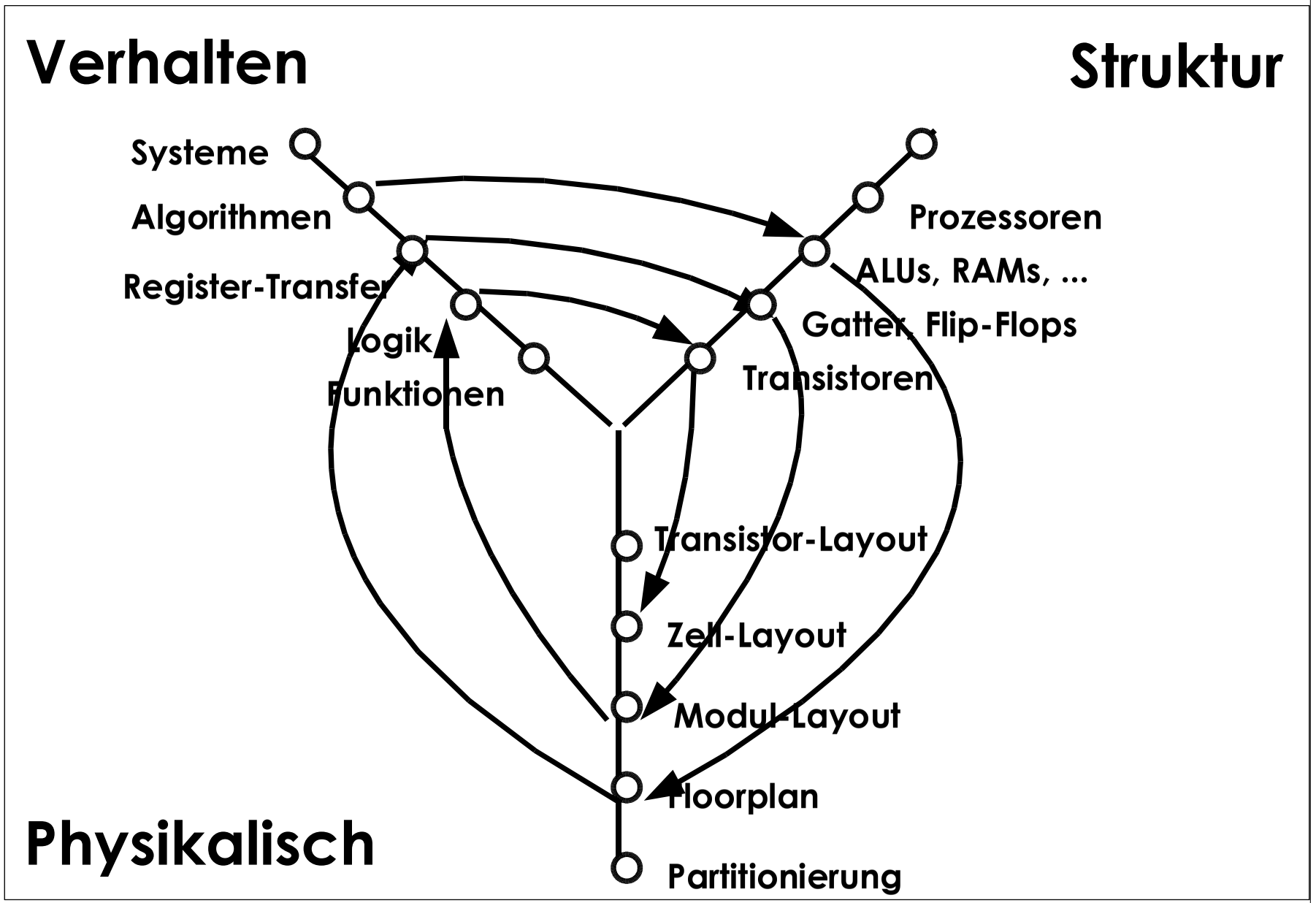
Überblick

- Floorplanning
- Problemstellungen
  - Cell Sizing Problem
- Technische Konzepte
- Modellierung
- Lösungsalgorithmus

Floorplanning 1

- Normale Vorgehensweise im VLSI-Entwurf: Bottom-Up
- Problem: Ineffizient
  - Übergeordnete Aspekte nicht berücksichtigt
    - „Big Picture“ fehlt
  - Führt zu schlechtem Layout
- Alternative: Top-Down
  - Berücksichtige Layout bei allen Schritten
  - Vereinfachungen
    - Relative Anordnungen statt absoluter Position
    - Abschätzungen z.B. für Fläche, Verdrahtungslänge

Floorplanning 2



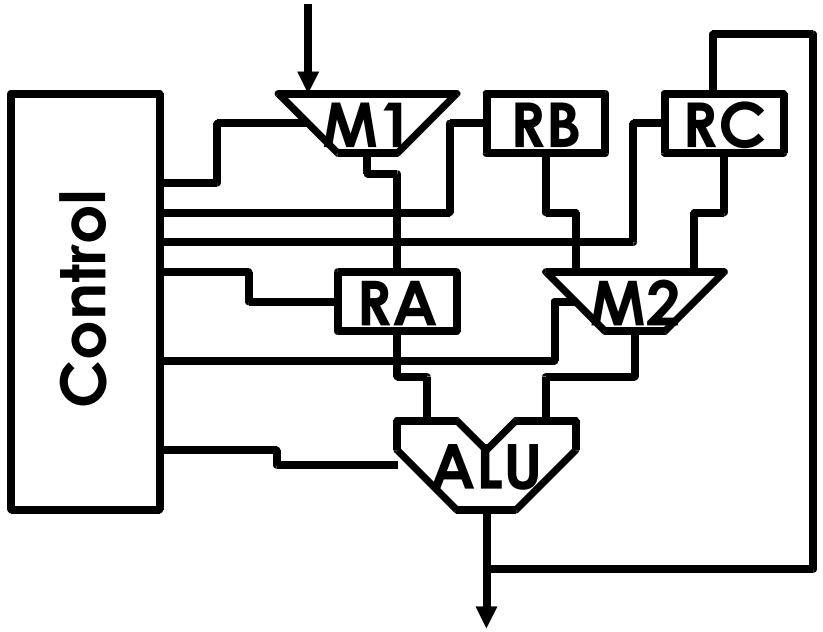
Floorplanning 3

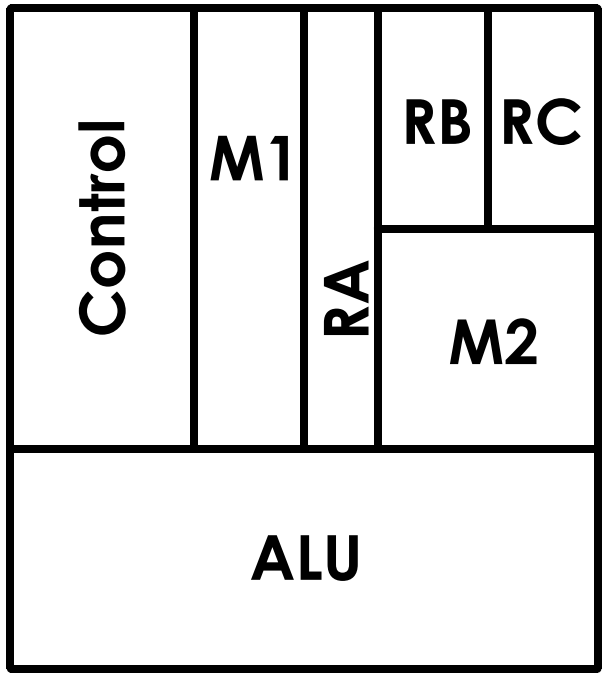
- Design Closure
- Extrem wichtig bei kleiner Strukturbreite
  - Deep Sub-Micron (DSM)
  - Verzögerungen nun überwiegend in Leitungen
  - Kapazitive und induktive Effekte
  - Layout muß berücksichtigt werden
- Aber ähnliche Situation auch bei FPGAs
  - Programmierbare Verbindungen *langsam*

Floorplanning 4

- **Auf unteren Entwurfsebenen**
  - Ausreichend Details vorhanden
    - Fläche
    - Verdrahtung
  - Layout leicht zu berücksichtigen
- **Auf höheren Entwurfsebenen**
  - Details fehlen
  - Abschätzungen erforderlich z.B. für
    - Fläche
    - Verdrahtungsmuster

Floorplanning 5





- *Topologische* Anordnung
- Flexible Blöcke, nach Festlegen bekannt:
  - Abmessungen
  - Lage der Terminals
- Floorplanning
  - Bestimme optimale Form und Anordnung

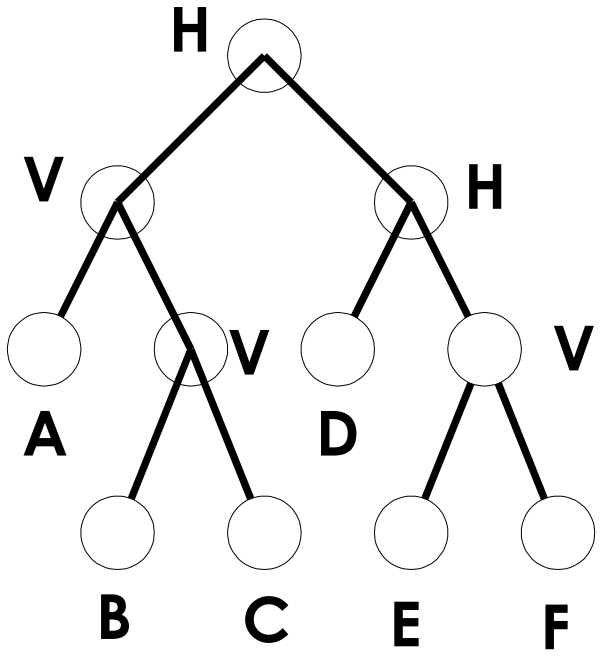
Konzepte

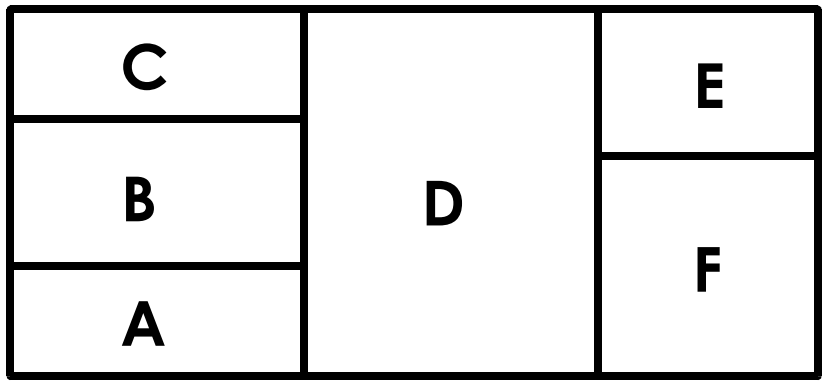
- **Blattzellen (*leaf cells*)**
  - Zellen auf niedrigster Hierarchiestufe
  - Enthalten keine weiteren Zellen mehr
- **Zusammengesetzte Zellen (*composite cells*)**
  - Enthalten weitere zusammengesetzte Zellen und/oder Blattzellen
  - Gesamter Chip als zusammengesetzte Zelle
- **Einschränkung**
  - Nur rechteckige Zell-Layouts

Slicing Floorplans 1

- **Vereinfachung**
  - Fordern weiterer Einschränkungen
- **Niedrigere Hierarchiestufe durch**
  - Durchschneiden (slicing) der aktuellen Zelle
    - Nicht zwangsläufig Halbierung (./. Min-Cut) !
  - Horizontal oder Vertikal
- **Konstruktive Sicht**
  - Setze Zelle durch Anreihen von Unterzellen zusammen
    - Horizontal oder Vertikal

Slicing Floorplans 2

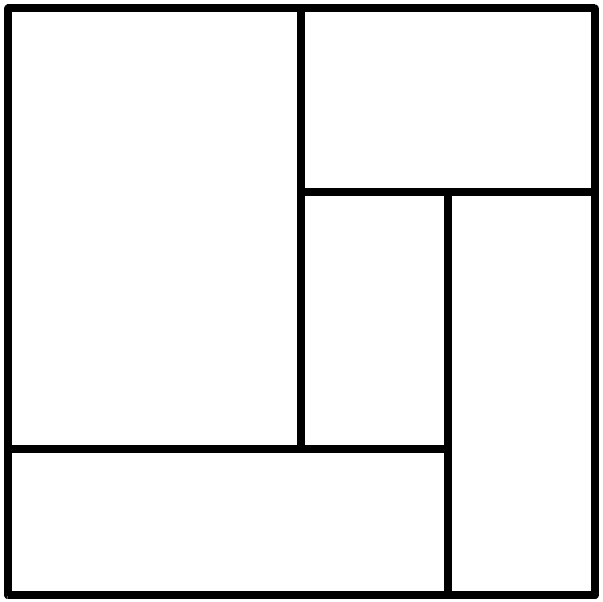




- Darstellung durch Slicing Tree
  - Knoten sind Schnitte oder Blattzellen
  - Schnitte nach Richtung getrennt
    - H: Linker Unterbaum *LINKS* von rechtem
    - V: Linker Unterbaum *UNTER* rechtem
- Ordnung=2

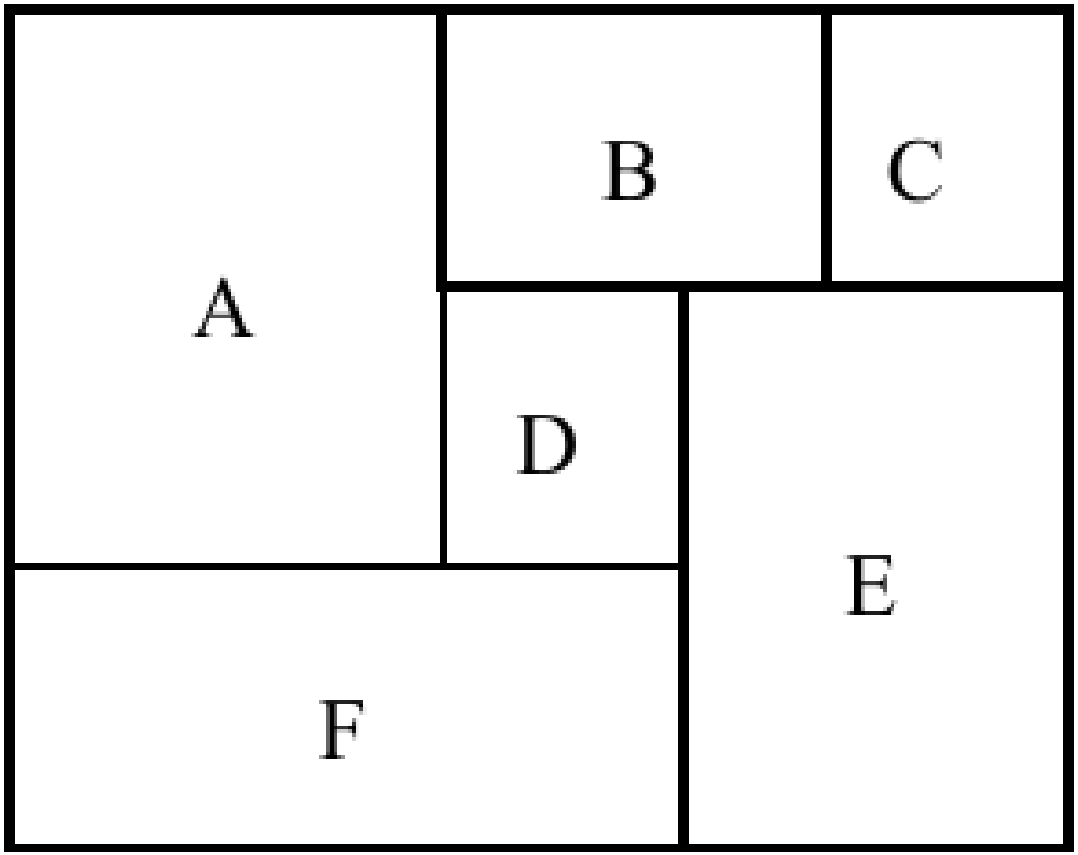
Spiral Floorplans 1

- Nicht alle Floorplans sind slicing!
  - Kann ab 5 Zellen auftreten
- Z.B. Rad oder Spirale



- Aber erstmal eingeschränkter
  - Flache statt hierarchischer Darstellung

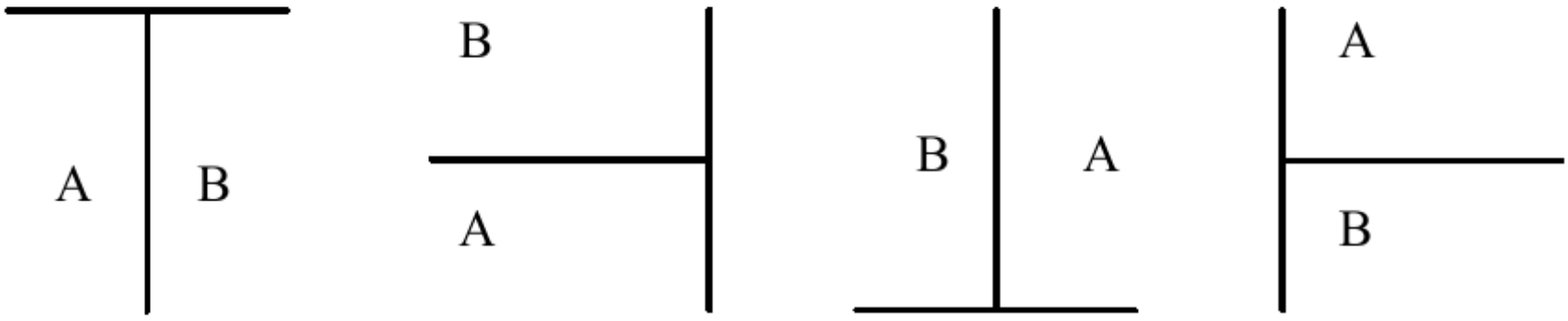
Mosaic Floorplans



- Haben im Inneren nur „T“-Kreuzungen

Ab hier aus Yao et al., ISPD '01

Darstellung mit Twin Binary Trees (TBT)



a. 0° T-junction | b. 270° T-junction | c. 180° T-junction | d. 90° T-junction

- **Idee: Charakterisiere Arten von T-Kreuzungen**
  - **Rechte obere Ecke von A: 0° und 270°**
    - **C^+ Nachbarschaft**
  - **Linke untere Ecke von A: 180° und 90°**
    - **C^- Nachbarschaft**
- **Beschreibung durch (V,E^+) und (V,E^-) Bäume**

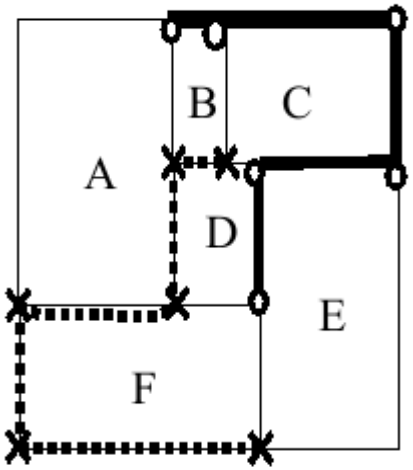
Floorplan zu TBT-Darstellung

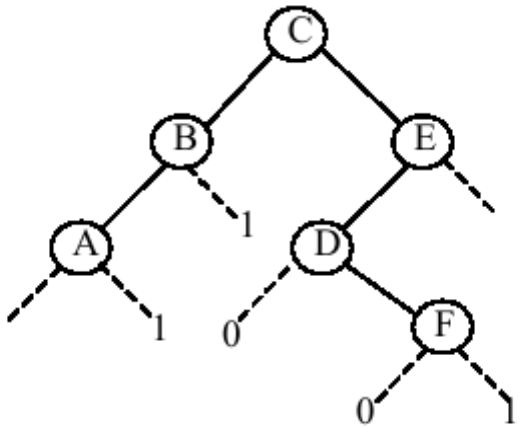
```
func MFTBT(Blocks) {
        foreach i in Blocks {
                if !i.isTopRightCorner() {
                        j = i.getCplusNeighbor(V+);
                        if j.RightOf(i)
                                j.leftChild = i
                        else
                                j.rightChild = i
                }
                if !i.isBottomLeftCorner() {
                        j = i.getCminusNeighbor(V-);
                        if j.LeftOf(i)
                                j.rightChild = i
                        else
                                j.leftChild = i
                }
        }
}
```

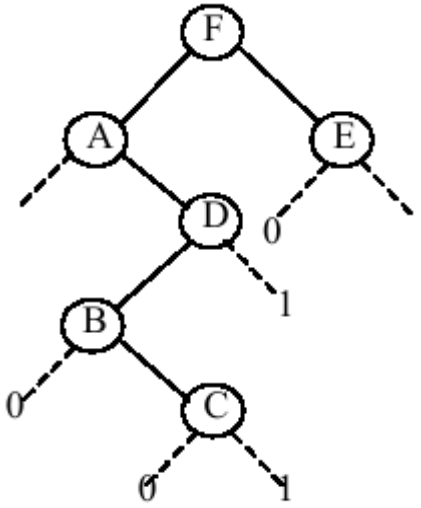
Initialisierung:

$V^+ = V^- = \text{Blocks}$

Beispiel MFTBT





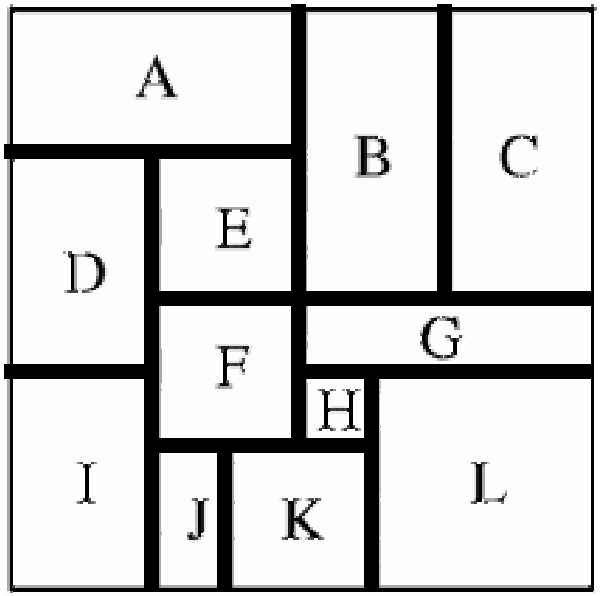


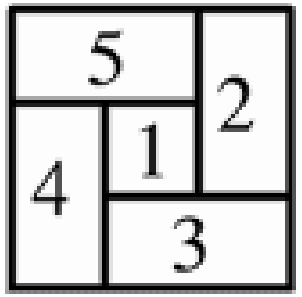
Floorplan (V^+,E^+) (V^-,E^-)

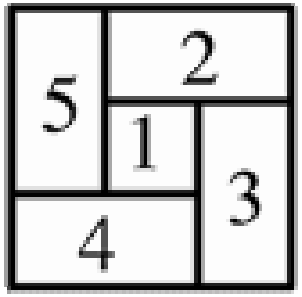
Hierarchische Spiral Floorplans

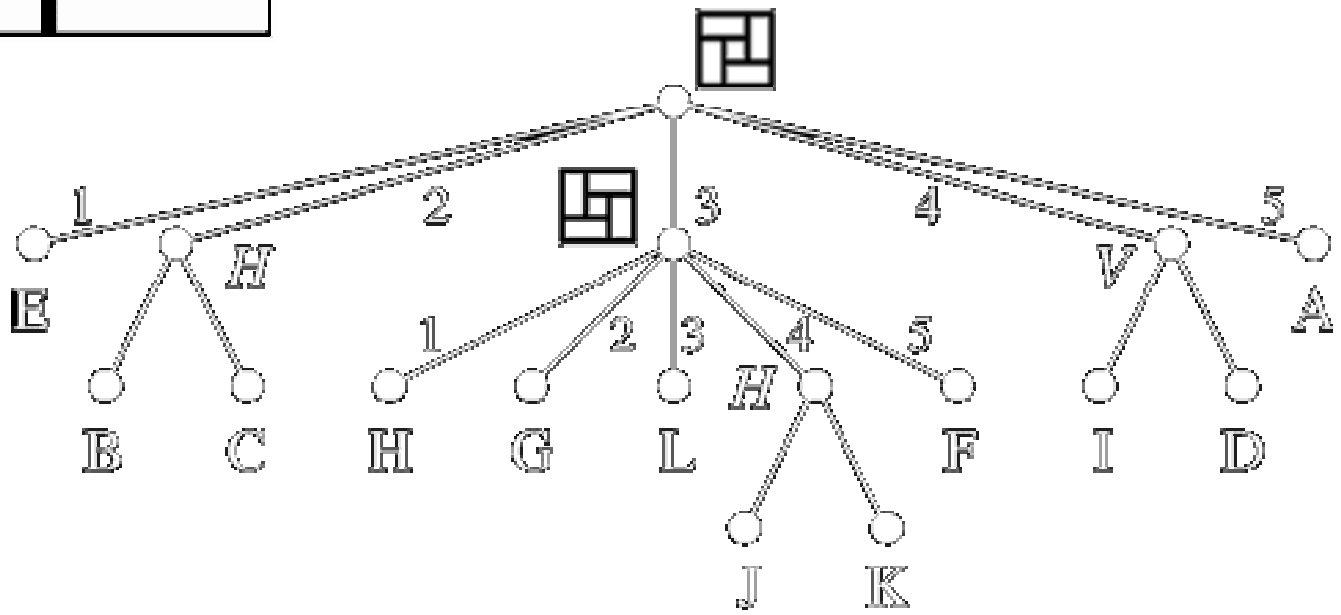
- Enthalten weitere Spiral Floorplans
  - Nun können +-Kreuzungen auftreten
  - Kein Mosaic-Floorplan mehr
- Modellierbar durch erweiterten Slicing-Tree
  - Neue Operatorknoten im Baum
  - Operator hat 5 Operanden (Ordnung 5)

Spiral Floorplan 2







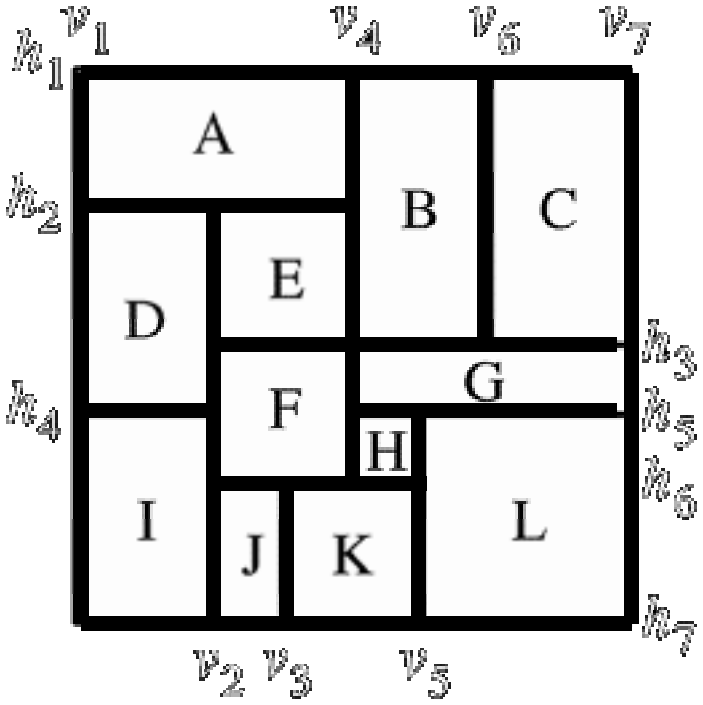


Polare Graphen 1

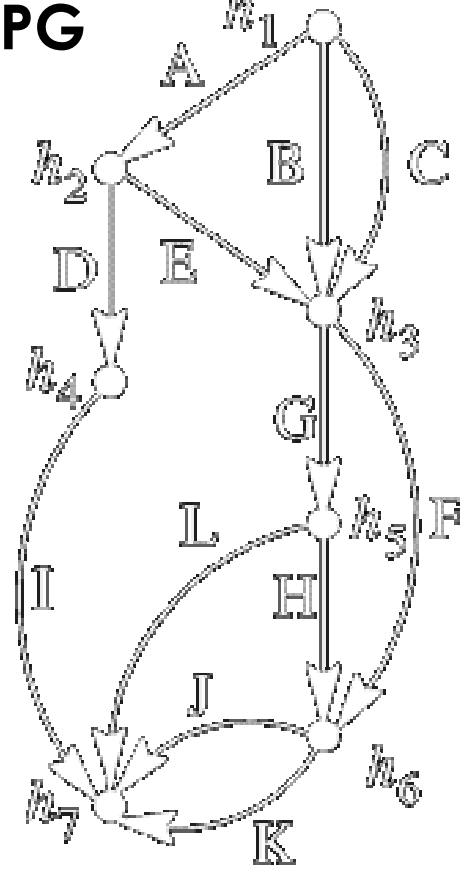
- **Höhere Ordnungen als 5 existieren**
  - **Extremfall: n Zellen → Ordnung n**
  - **Hier keine hierarchische Darstellung mehr**

- **Alternative: Polarer Graph**
  - **Finde längste Trennlinien zwischen Zellen**
    - **Horizontale Linien → Knoten im Horizontalen PG**
    - **Vertikale Linien → Knoten im Vertikalen PG**
  - **Zellen → Kanten in beiden PGs**
    - **Im H-PG: Von der oberen Begrenzung zur unteren**
    - **Im V-PG: Von der linken Begrenzung zur rechten**

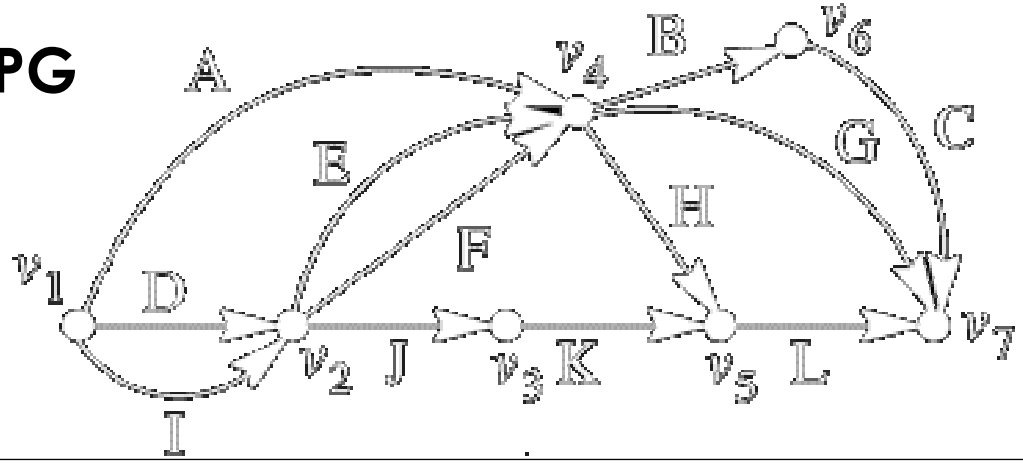
Polare Graphen 2



H-PG

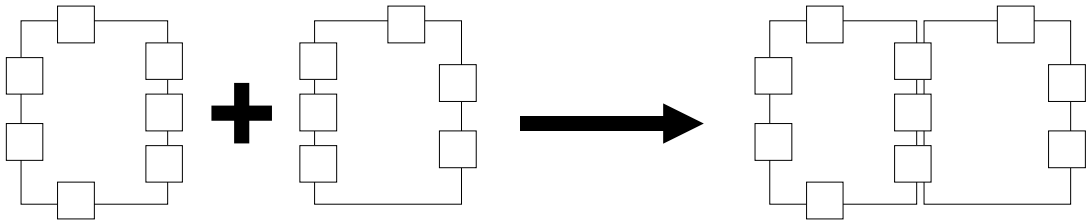


V-PG



Anreihung

- Anreihbarkeit (abutment)



- Optimal für Verdrahtung
  - Kleinste Fläche
    - Keine Kanäle
  - Kürzeste Verbindung
- Setzt flexible Zell-Generatoren voraus
  - Akzeptieren vorgegebene Pin-Positionen
  - Zell-Umgebung muß berücksichtigt werden

Bessere Darstellungen

- Polare Graphen
  - Aufwendig!
- Alternativen
  - Erster Durchbruch
    - Sequence Pair (1995)
  - Derzeit bestes Verfahren
    - O-Tree (1999)
    - Greift neben Baumstruktur auch auf Blockabmessungen zurück
- Übersicht in
  - Yao, Chen, Cheng, Graham
    „Revisiting Floorplan Representations“
    ISPD 2001 (auf Web-Seite)

Optimierungsprobleme

- **Floorplan-Erzeugung**
  - Bestimme 2D Anordnung aus Struktur
  - Nicht unbedingt hierarchisch
    - Alternativ z.B. durch Verwendung von Min-Cut, Clustering
  - Ähnlich Platzierungsproblem
    - Komplizierter: Keine festen Formen
- **Größenanpassung (sizing)**
  - Finde optimale Form für jede Blatt-Zelle
    - Eingabeparameter für Zell-Generatoren
- **Zell-Generierung**
  - Erzeuge Layout nach Eingabeparametern
    - Form, Pin-Anordnung, Breite, Datentypen, ...

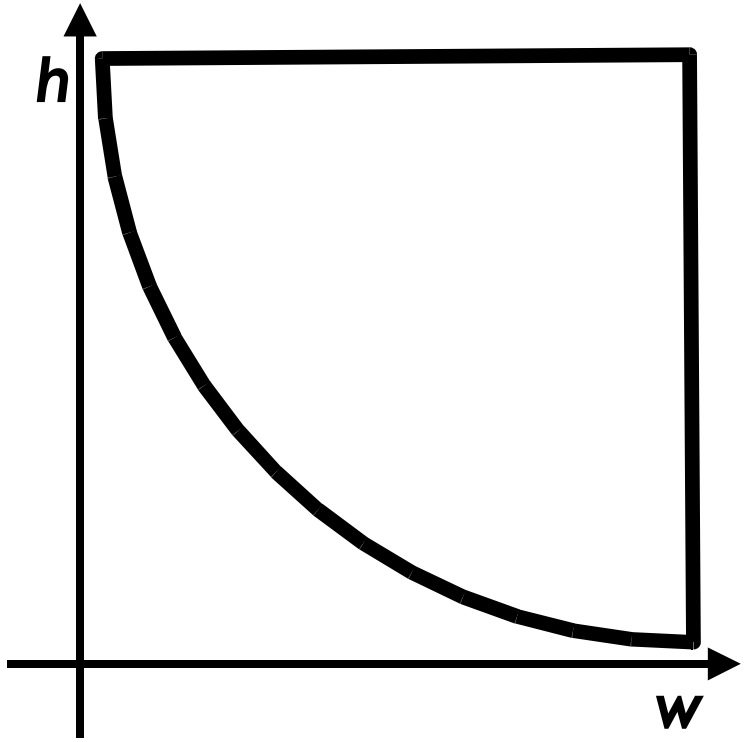
Größenanpassung

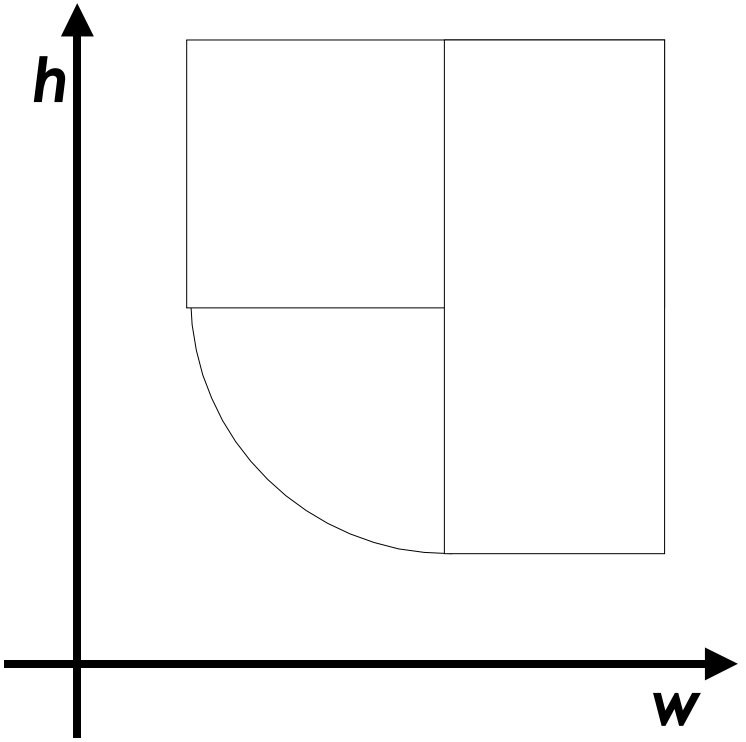
- **Idee:**
  - Fläche zur Realisierung einer Operation: A
  - Flexible Zellen: $h\,w \geq A$
    - Variiere Seitenverhältnis
  - Minimale Zellhöhe als Funktion der Breite: Formfunktion

$$h(w) = \frac{A}{w}$$

  - Praktisch nicht alle Werte möglich
    - Sehr schmal oder sehr breit ausgeschlossen
      - Aufgrund von Design-Rules
    - Fordere Untergrenzen für Ausmaße

Formfunktionen 1

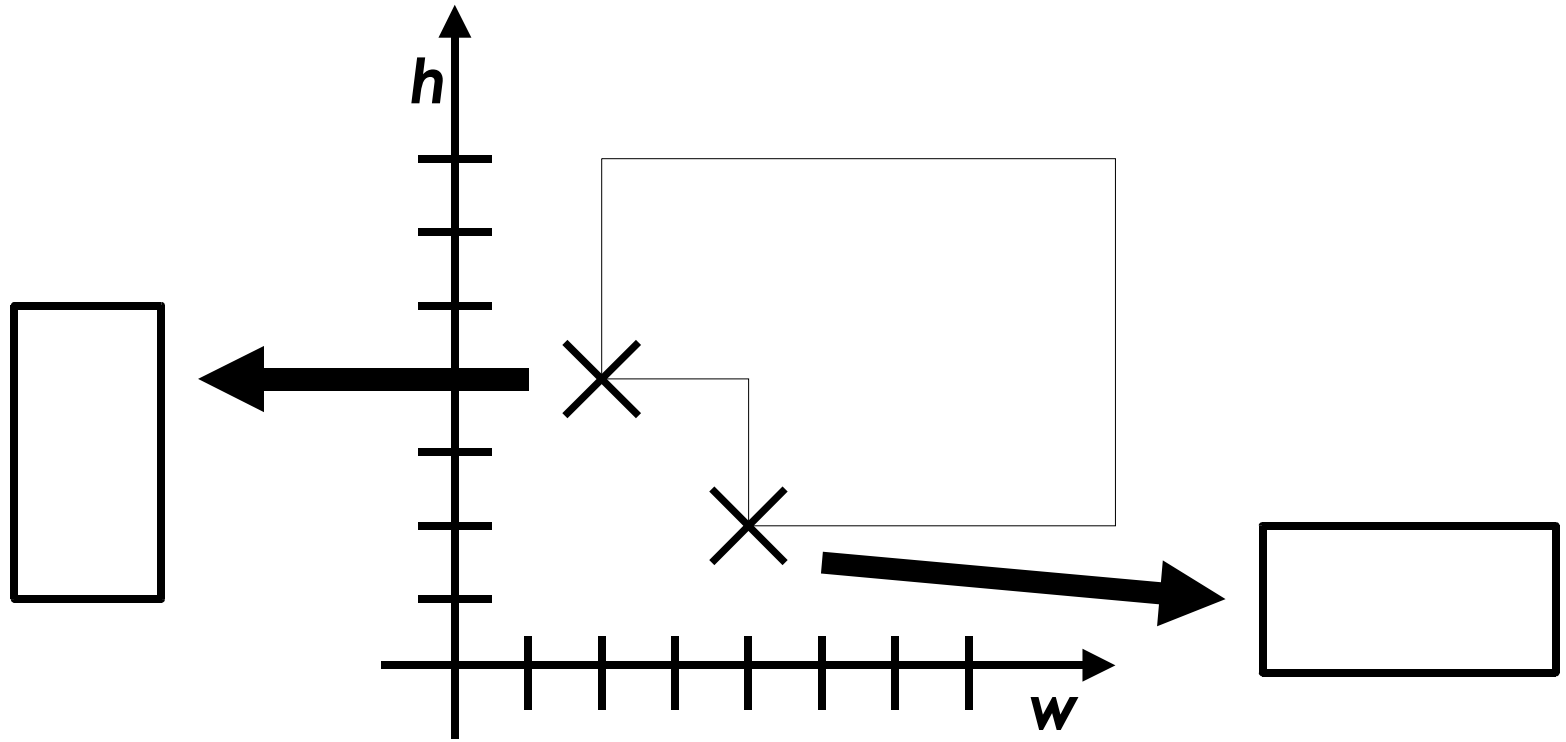




- Formfunktion ohne Einschränkungen
- Form„funktion“ mit Einschränkungen
  - Keine Funktion mehr!

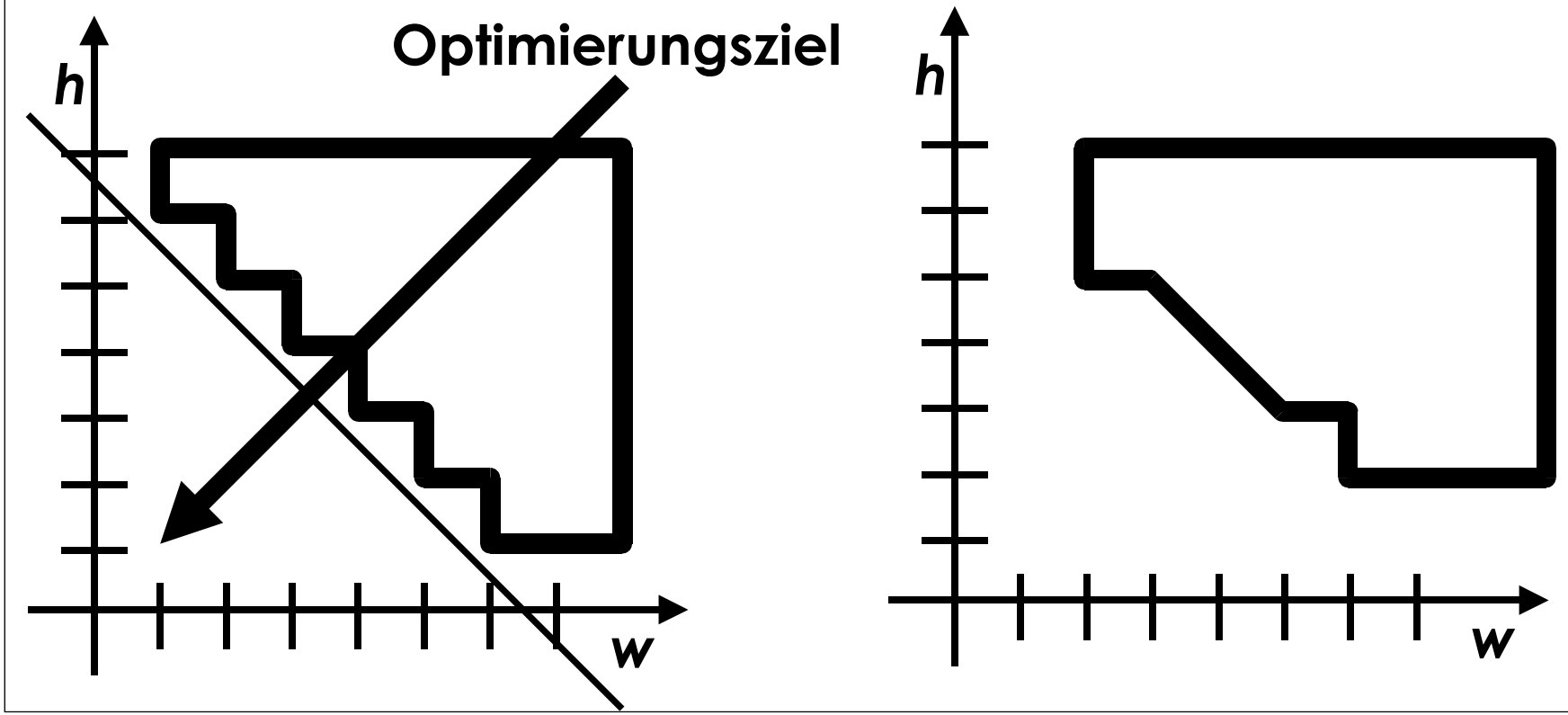
Formfunktionen 2

- Bisher: Stetige Funktionen
- Real: Nur diskrete Werte möglich
- Beispiel: Harte Zelle, *h* x *w* = 4 x 2



Formfunktionen 3

- **Mehrere Seitenverhältnisse möglich**
  - 1 Horizontales Segment je Realisierung
- **Auch stückweise lineare Funktionen möglich**
  - Teilintervalle durch Teilpunkte begrenzt



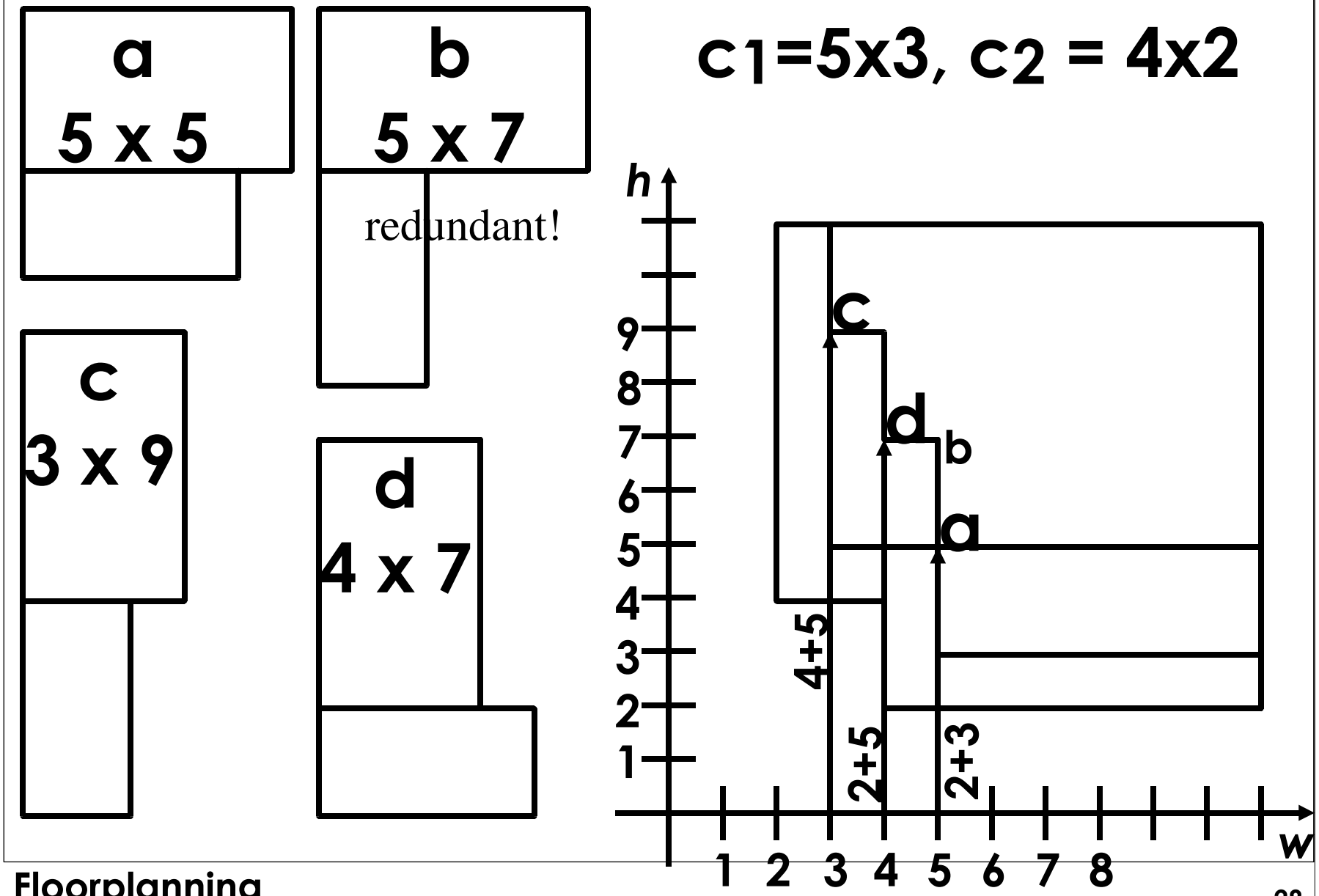
Formfunktionen 4

- Hierarchische Vorgehensweise
- Bestimme Form einer zusammengesetzten Zelle
  - Aus Formen der Unterzellen
  - Verknüpfe Formfunktionen der Unterzellen
  - Bottom-Up
  - Anhand von Schnittrichtungen im Slicing Tree
- Zunächst: Vertikale Anreihung
  - Zelle c_1 über Zelle c_2 angereiht
  - Formfunktionen $h_1(w)$, $h_2(w)$
  - Zusammengesetzte Formfunktion

$$h_3(w) = h_1(w) + h_2(w)$$

  - Berechnung nur an Intervallgrenzen erforderlich

H-Schnitt: Formfunktionen 5



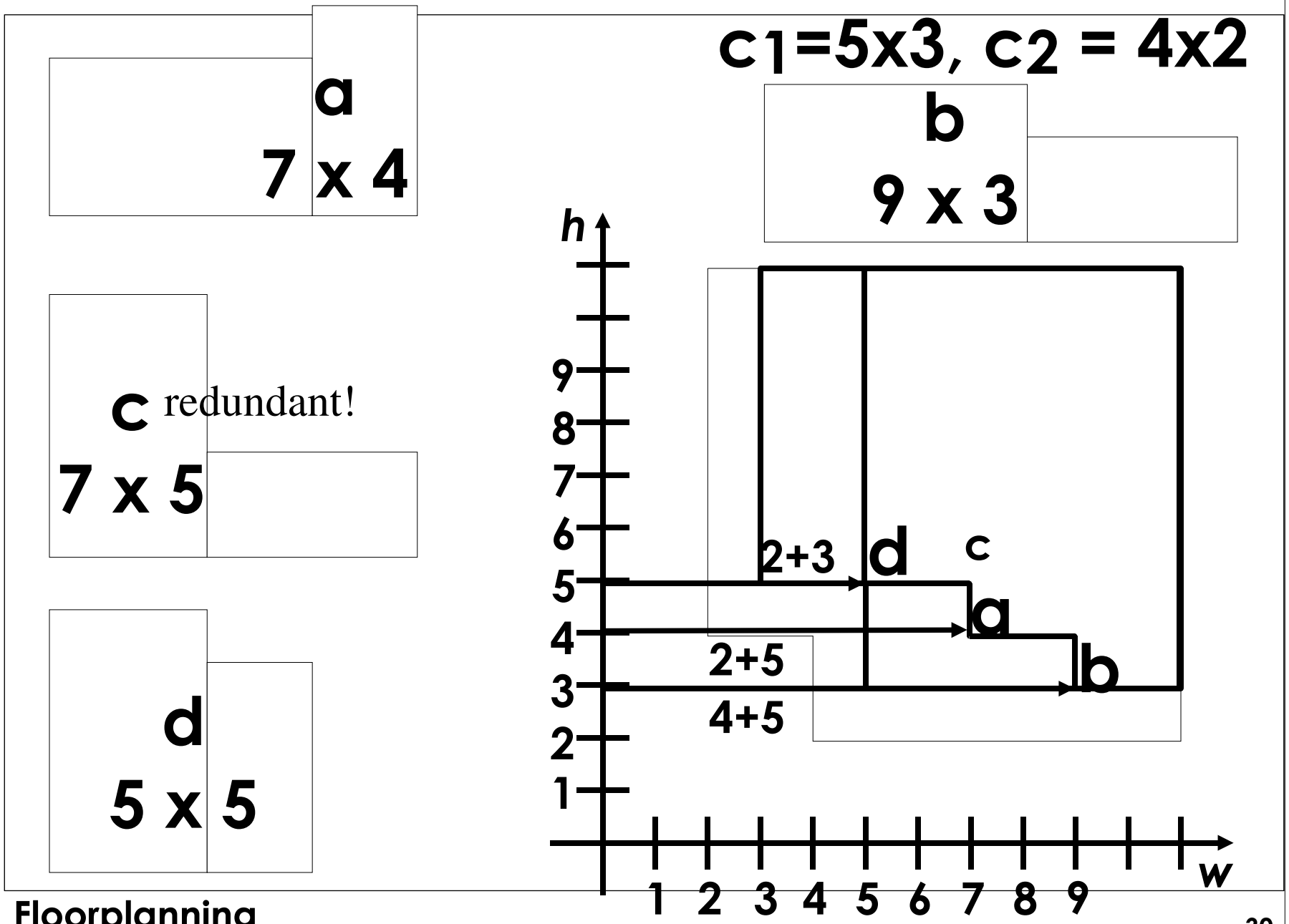
Formfunktionen 6

- **Horizontale Anreihung**
  - Zelle c_1 links neben Zelle c_2 angereiht
  - Formfunktionen $h_1(w)$, $h_2(w)$
  - Jetzt werden aber Breiten aufsummiert
    - Gebraucht „$w_1(h)$“ und „$w_2(h)$“
    - Inverse verwenden!
      - Hier vereinfacht, tatsächlich keine echten „Funktionen“
  - Zusammengesetzte Formfunktion

$$h_3^{-1}(h) = h_1^{-1}(h) + h_2^{-1}(h)$$

  - Auch hier nur an Intervallgrenzen berechnen

V-Schnitt: Formfunktionen 7



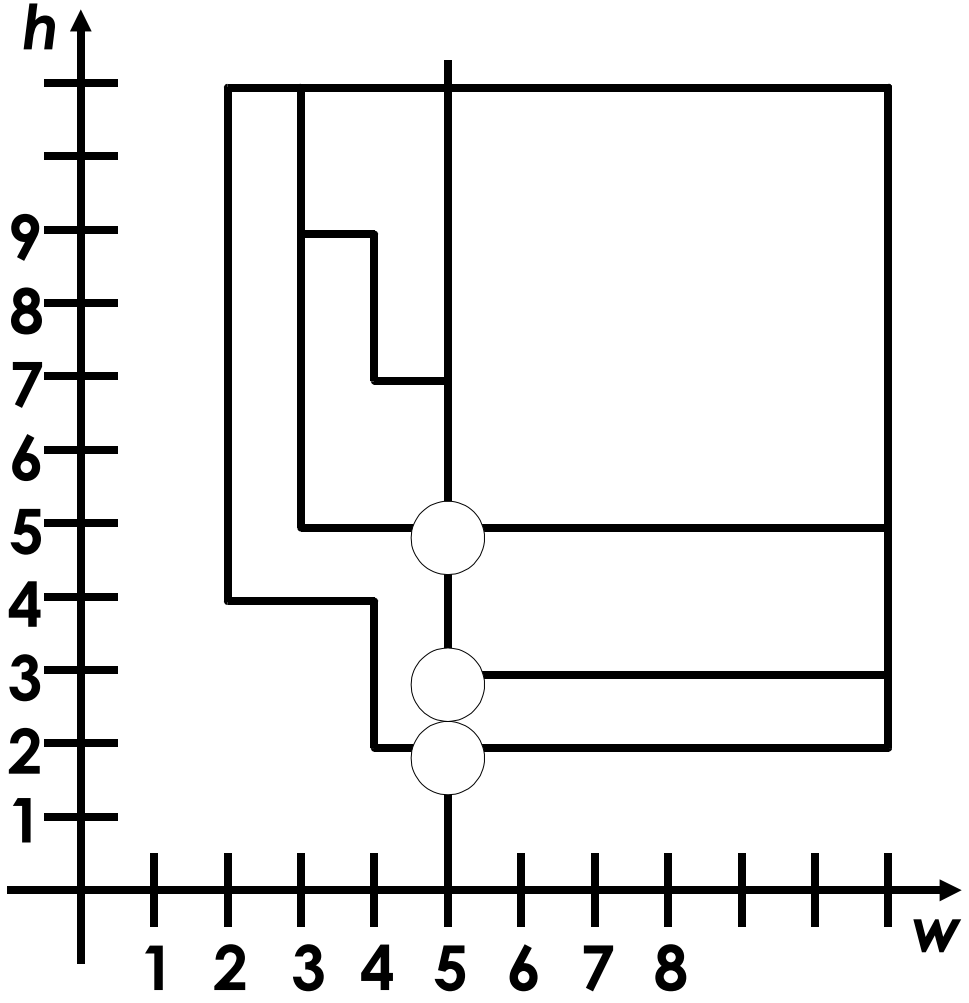
Auswahl der besten Form 1

- **Bestimmen der Formfunktionen über *alle* zusammengesetzten Zellen**
  - **Bottom-Up**
  - **= Formfunktion für gesamten Chip**
- **Gesucht: Formauswahl je Zelle**
  - **Mit bestem Gesamtergebnis**
  - **Häufigstes Optimierungsziel: Min. Fläche**
- **Übergeordnete Zell-Form bestimmt die Formen aller untergeordneten Zellen**
  - **Eindeutige Zuordnung!**
  - **Top-Down**

Auswahl der besten Form 2

Optimale Fläche
5 x 5 = 25

Annahme: Wurzel ist H-Schnitt



- c_1: 5x3
- c_2: 5x2 → 4x2

- H/V aus Knoten im Slicing Tree
  Bei V-Schnitt via $h^{-1}(h)$

Größenanpassungsalgorithmus

- Gegeben: Slicing Tree
  - Erstellt z.B. mittels Min-Cut
- Bestimme Formfunktion der Wurzelzelle
  - Bottom-Up Vorgehen beginnend bei Blatt-Zellen
  - Kombiniere Formfunktionen entsprechende Slice-Richtung
- Wähle optimale Form für Wurzelzelle
- Propagiere Effekte der Auswahl in Slicing Tree
  - Top-Down Vorgehen beginnend bei Wurzel

Komplexität 1

- In polynomialer Zeit möglich
- Annahmen
  - n Zellen → n Formfunktionen
  - q Teilpunkte in allen Formfunktionen ($q \geq n$)
  - d Ebenen im Slicing Tree
- Dann
  - Auf jeder Ebene $O(q)$ Formfunktionen berechnen
    - Anzahlen Teilpunkte summieren sich bei Bottom-Up
  - Also $O(dq)$ Berechnungen
    - Ausbalancierter Baum: $d = \log n \rightarrow O(q \log n)$

Komplexität 2

- Gilt nur für Slicing Floorplans!
  - Sonst NP-vollständig
- Floorplans der Ordnung 5 (mit Spiralen)
  - Problem: Berechnung der Formfunktion
    - Bei k Alternativen pro Zelle
      - Brute Force: k^5 Möglichkeiten
      - Schlauer: $O(k^2 \log k)$
    - Beim Bottom-Up Durchlauf durch d Ebenen dann $\Omega(k^{2d})$

Zusammenfassung

- Floorplanning
- Grundlagen
- Probleme
- Genauer
  - Darstellungen
  - Größenanpassungsproblem
  - Algorithmus

Weitere Veranstaltungen des FG

- Optimierende Compiler (SS07)
  - Grundlagen (Lexing, Parsing, AST)
  - Code-Generierung
  - Optimierung
  - Erweiterung eines einfachen Compilers (Java)
- Praktikum Adaptive Computer (SS07)
  - Praktische Verwendung von FPGAs
  - Effizienter Rechnen als mit Standardprozessoren
  - Einfache Bildbearbeitung mit Verilog und C
  - Tests an realer Hardware
- Prozessorarchitekturen für rechenstarke eingebettete Systeme (WS 07/08)
  - Bandbreite verschiedenster Ansätze
  - Zu jeder Technologie praktische Übungen
  - Arbeit mit verschiedenen CAD-Werkzeuflüssen

Vorbereitung für Di

- Verbesserung von Platzierung und Verdrahtung
  - Neue Ideen
  - Gehen hinaus über
    - Tuning des Simulated Annealing
    - Reine Fehlersuche

☺Seien Sie kreativ!