Algorithmen im Chip-Entwurf 4

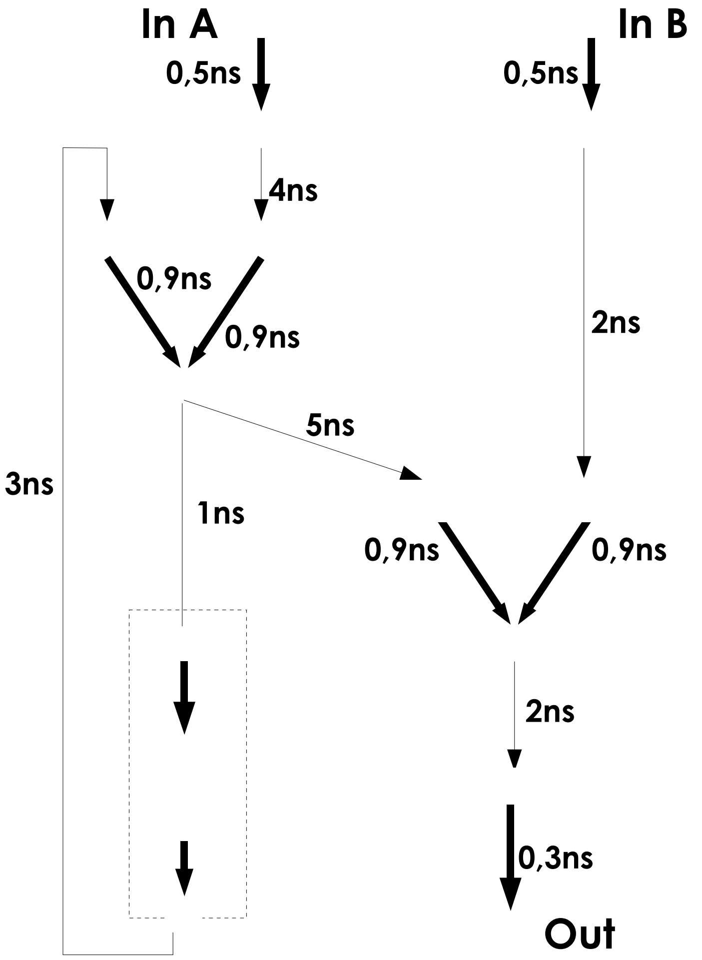
Längenmaße und Platzierung

Andreas Koch
FG Eingebettete Systeme
und ihre Anwendungen
TU Darmstadt

Überblick

- Übung Timing-Analyse
- Längenmaße
- Arten von Platzierungsproblemen
- Platzierungsverfahren
- Partitionierung
  - Kernighan-Lin
- Zusammenfassung

Übung Timing-Analyse



$$T_a(v) = \max_{(u,v)\in E}\left(T_a(u) + w(u,v)\right)$$

$$T_r(u) = \min_{(u,v)\in E}\left(T_r(v) - w(u,v)\right)$$

$$k(u,v) = T_r(v) - T_a(u) - w(u,v)$$

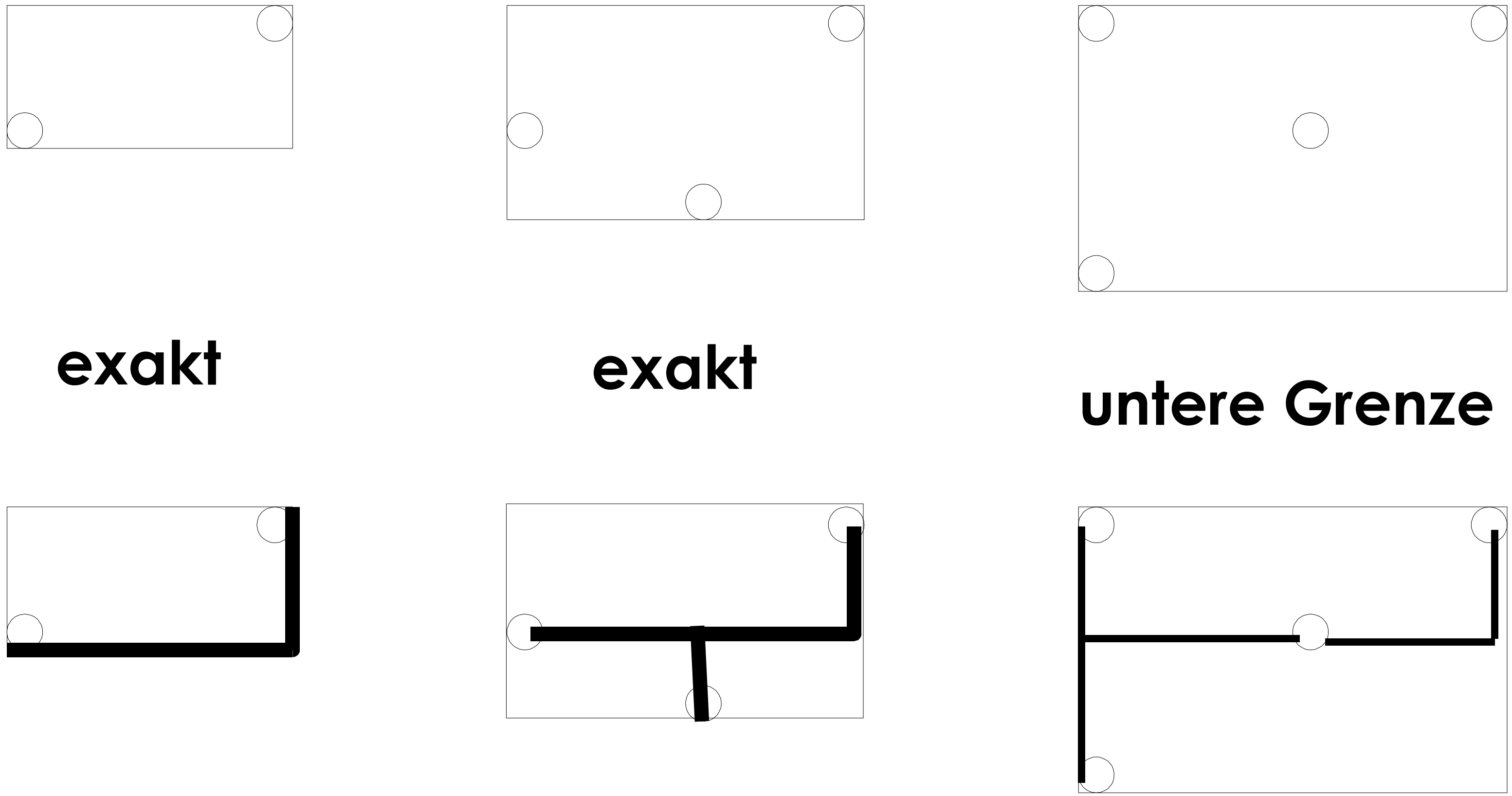
- $D_{max} = 13{,}6ns$

Verdrahtungsfläche

- Mögliches Platzierungs-Qualitätskriterium
  - Gesamtfläche für Verdrahtung
    - Nur bei ASIC
    - Bei FPGA: Feste Breite der Leitungen, Länge reicht
- Aber: Vollständiges Routing zu komplex
  - NP
- Abschätzen der Länge durch Metrik
  - Einzeln pro Netz
  - Aufsummieren der Teillängen
  - Multiplizieren mit angenommener
    - Leitungsbreite plus
    - Leitungsabstand

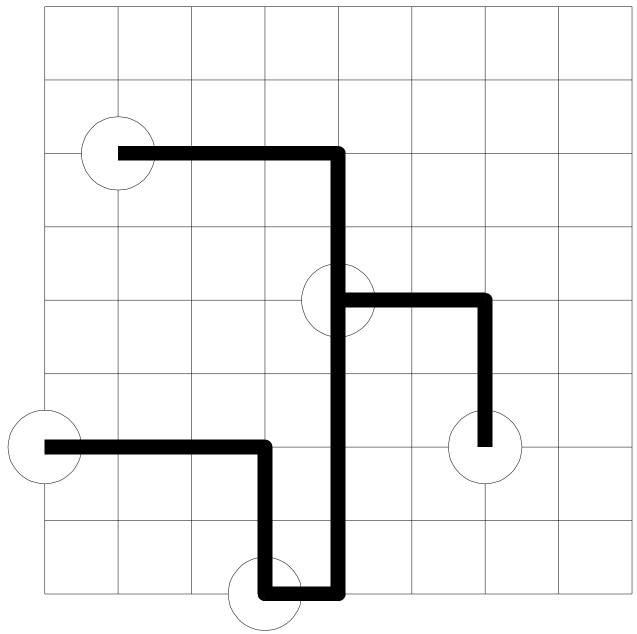
Längenmetriken 1

- Halber Umfang (half perimeter)
  - Rechteck um alle Terminals des Netzes



Längenmetriken 2

- **Minimaler Rechtwinkliger Überspannender Baum (MRST)**

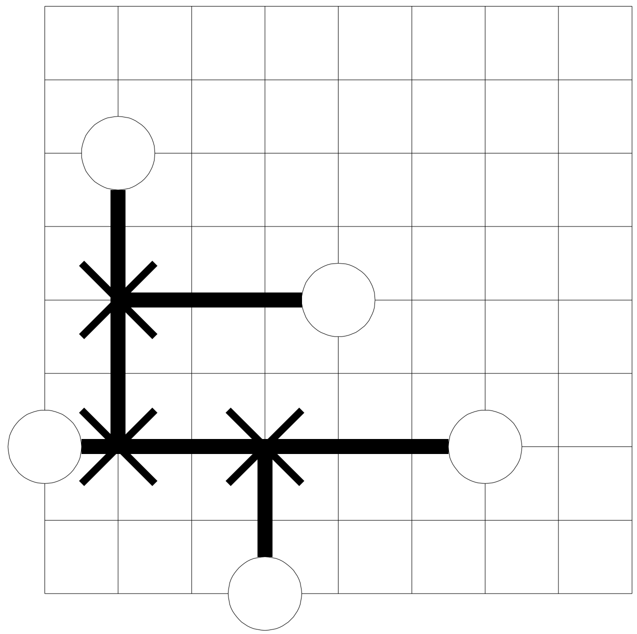


L_R=19

- **Sonderfall von MRST**
  - **In P via Prim's Algorithmus (im Buch 3.4.4)**
    - **Vollständiger planarer Graph**

Längenmetriken 3

- Rechtw. Steiner-minimaler Baum (RSMT)



$L_S=15$

$L_R/L_S = 1{,}26$

- RSMT-Berechnung ist NP-vollständig
  - Annäherung durch MSRT: max. 1.5x so lang
  - Bessere Näherungen existieren

Längenmetriken 4

- **Quadratischer Euklidischer Abstand**
  - **Arbeitet auf Zellen, nicht auf Netzen**
    - **Für Clique-Modell geeignet**

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\gamma_{ij}\left[(x_i-x_j)^2+(y_i-y_j)^2\right]$$

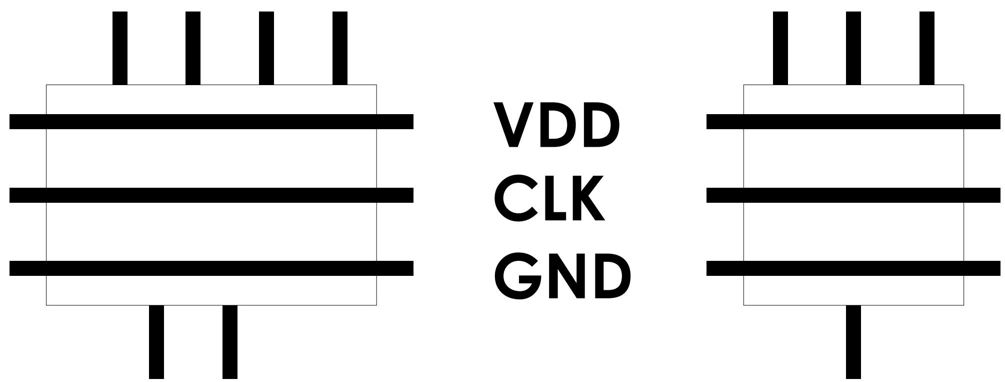
- **γij**
  - **=0 wenn (v_i, v_j) $\notin E$**
  - **=|(v_i,v_j)|: Gewichtet nach Anzahl Kanten**
  - **< |(v_i,v_j)|: nicht nur Einzelleitungen**

Platzierungsprobleme

- Standardzellen
  - Semi-Custom
- Building Block
  - Teilweise Full-Custom möglich
- MPGA/FPGA
  - Auf vorgegebene Strukturen

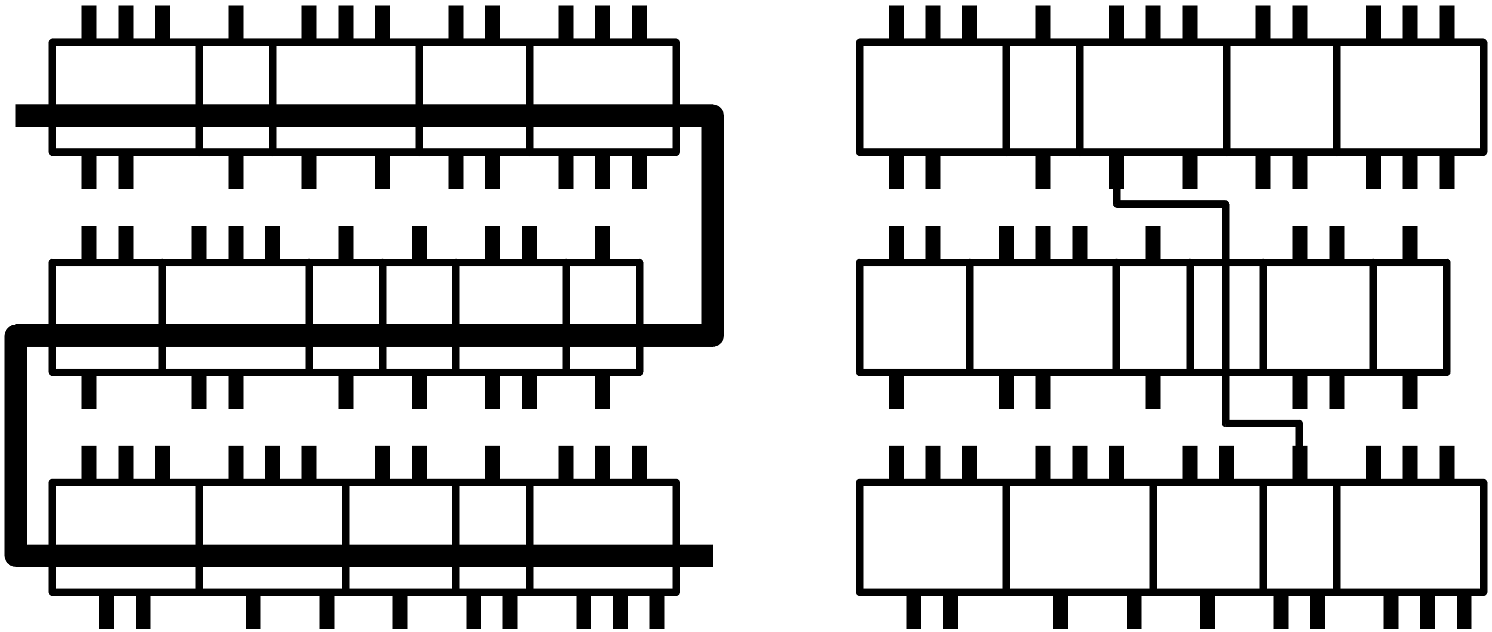
Standardzellen 1

- Standardzellen (Semi-Custom)
  - Kleinere Schaltungen (Gatter) aus Bibliothek
  - Festes Layout
    - Grösse
    - Terminal-Anordnung
  - Anreihbar in Zeilen
    - Logistische Signale



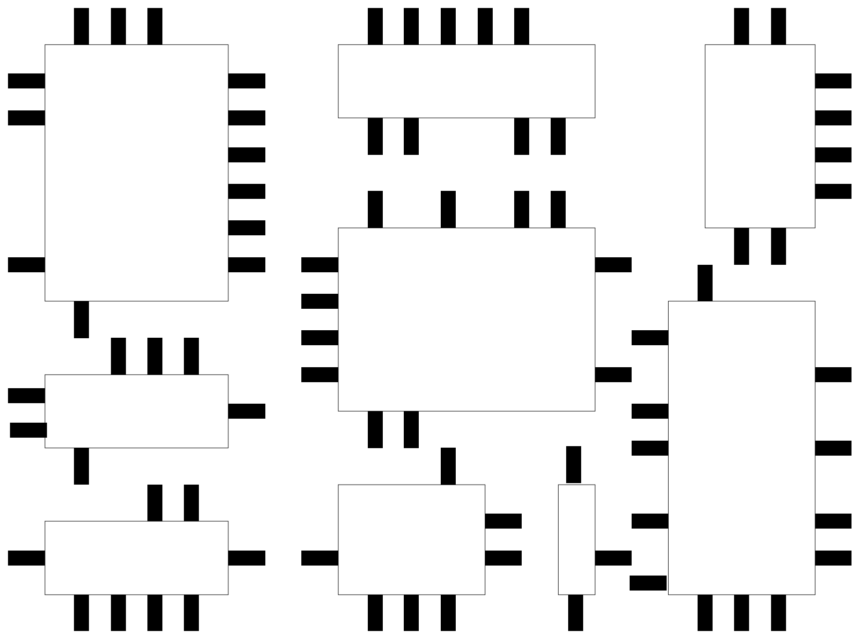
Standardzellen 2

- Zeilenweise Anordnung
- Verdrahtung zwischen Zeilen
- Ausnahmen
  - Angrenzende Verbindungen (abutment)
  - Durchleitungen (feedthroughs)



Building Blocks 1

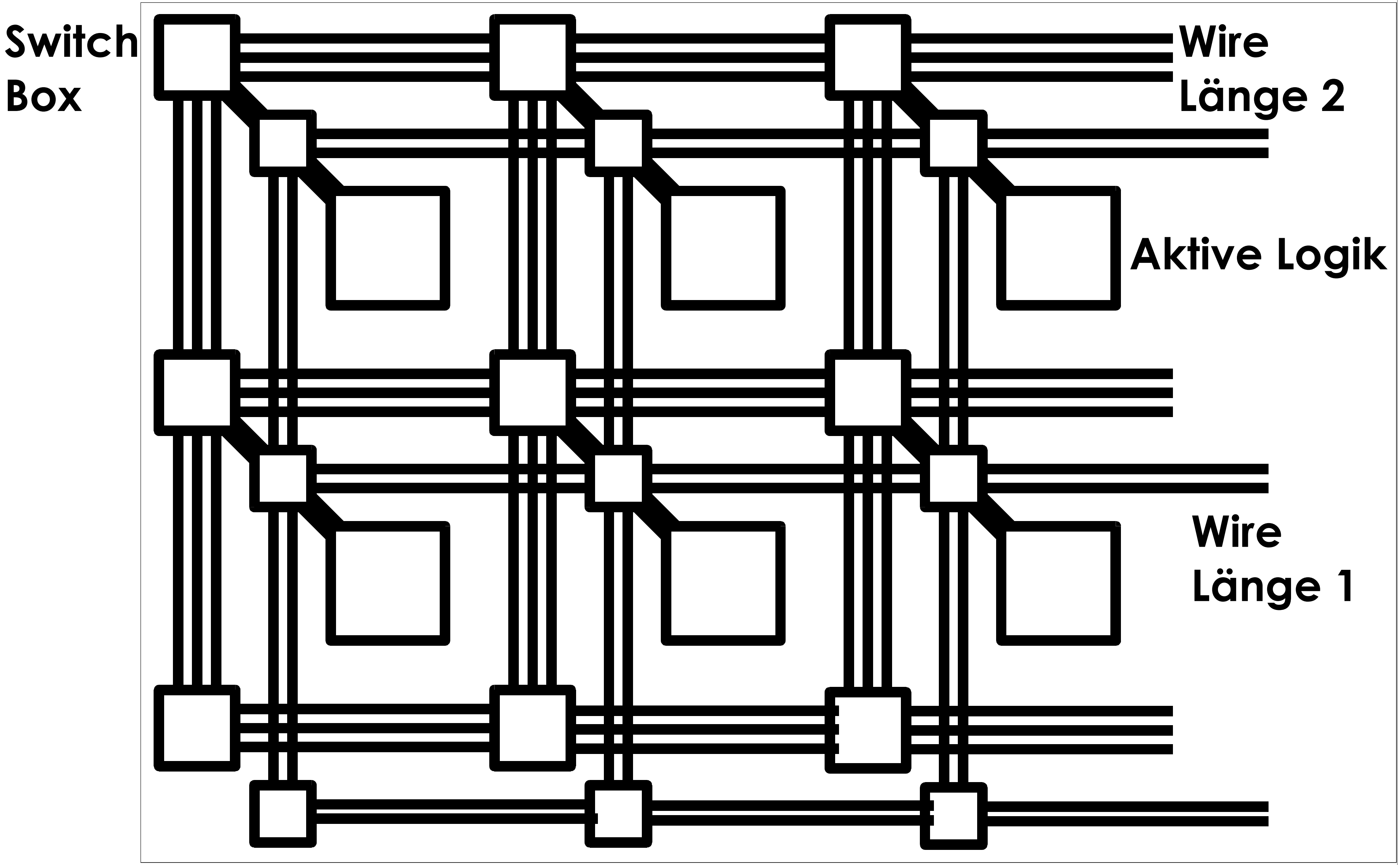
- Mehr Flexibilität
  - Kann auch Full-Custom Teile enthalten
  - Automatisch generierte Blöcke (z.B. RAM)
- Verdrahtungskanäle an allen Seiten



MPGA/FPGA 1

- Mask-Programmable Gate Array
  - Modebezeichnung: Structured ASIC
- Field-Programmable Gate Array
- Feste Anordnung von
  - Logik
  - Verdrahtung
- Anpassung auf Anwendung
  - MPGA: Beim Hersteller (Metalllagen)
  - FPGA: Beim Anwender (Programmierung)

MPGA/FPGA 2



Längenmaße und Platzierung

MPGA/FPGA 3

- Sehr ähnlich zu UPP
- Aber: Segmentierte Verbindungen
  - Mehrere Verdrahtungslängen
- Verzögerung abhängig von
  - Anzahl durchlaufener Switch Boxes
  - Last (Fan-Out)
- Feste Verdrahtungskapazität
- ➢ Nicht jede Platzierung verdrahtbar
- Verdrahtbarkeit in Kostenfunktion

Platzierungsverfahren 1

- Konstruktiv
  - Zellkoordinaten sind nach einmaligem Plazierungsschritt fest
- Iterativ
  - Zellkoordinaten können beliebig oft geändert werden
- Kombination
  - Konstruktive Startlösung
  - Dann iterative Verbesserung

Mögliche Optimierungsziele 1

- Minimale Verdrahtungsfläche
- Minimale Verdrahtungslänge
- Schnellste Schaltung
  - Timing-driven
- Anzahl von Leitungen durch Schnittlinie
- Verdrahtbare Schaltung
- Geringes Übersprechen
  - Zwischen Leitungen

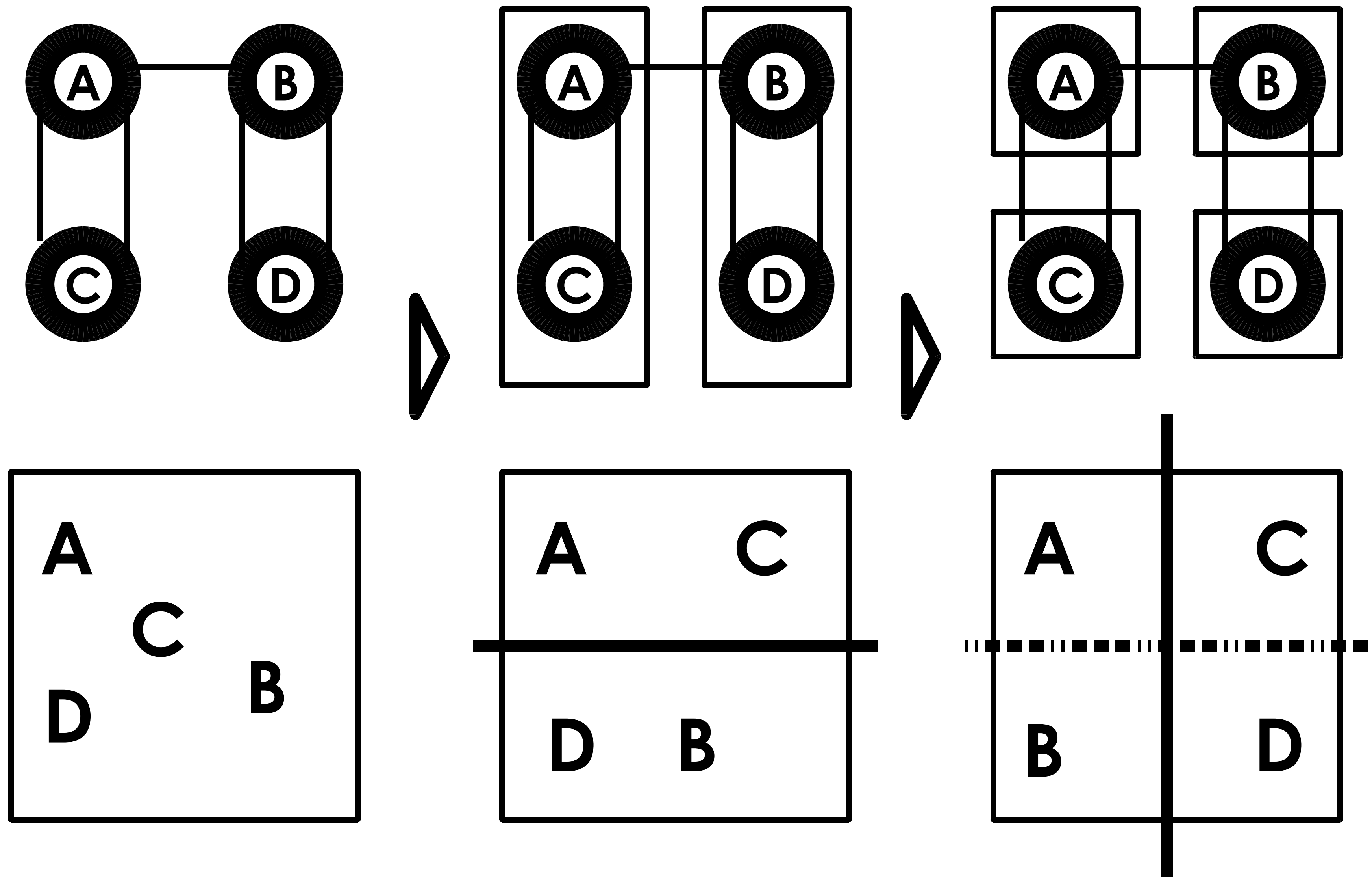
Konstruktive Platzierung 1

- Viele Methoden
- Top-Down Verfahren
  - Starten mit kompletter Schaltung
  - Aufteilen in immer kleinere Probleme
  - Beispiel: Min-Cut
- Bottom-Up Verfahren
  - Beginnen mit einzelnen Zellen
  - Zusammenfügen von Teillösungen
  - Beispiel: Clustering

Min-Cut Platzierung 1

- Idee
  - Teile Schaltung in zwei Hälften auf
  - Minimiere die Anzahl der Netze dazwischen
    - MinCut: Minimiere Gewicht *durchschnittener* Netze
  - Teile auch Layoutfläche nach jedem Schnitt
  - Ordne Schaltungshälften Layouthälften zu
    - Horizontal und Vertikal, i.d.R. abwechselnd
  - Wiederhole bis Abbruch
    - z.B. Nur noch eine Zelle in Partition

Min-Cut Plazierung 2



Min-Cut Plazierung 3

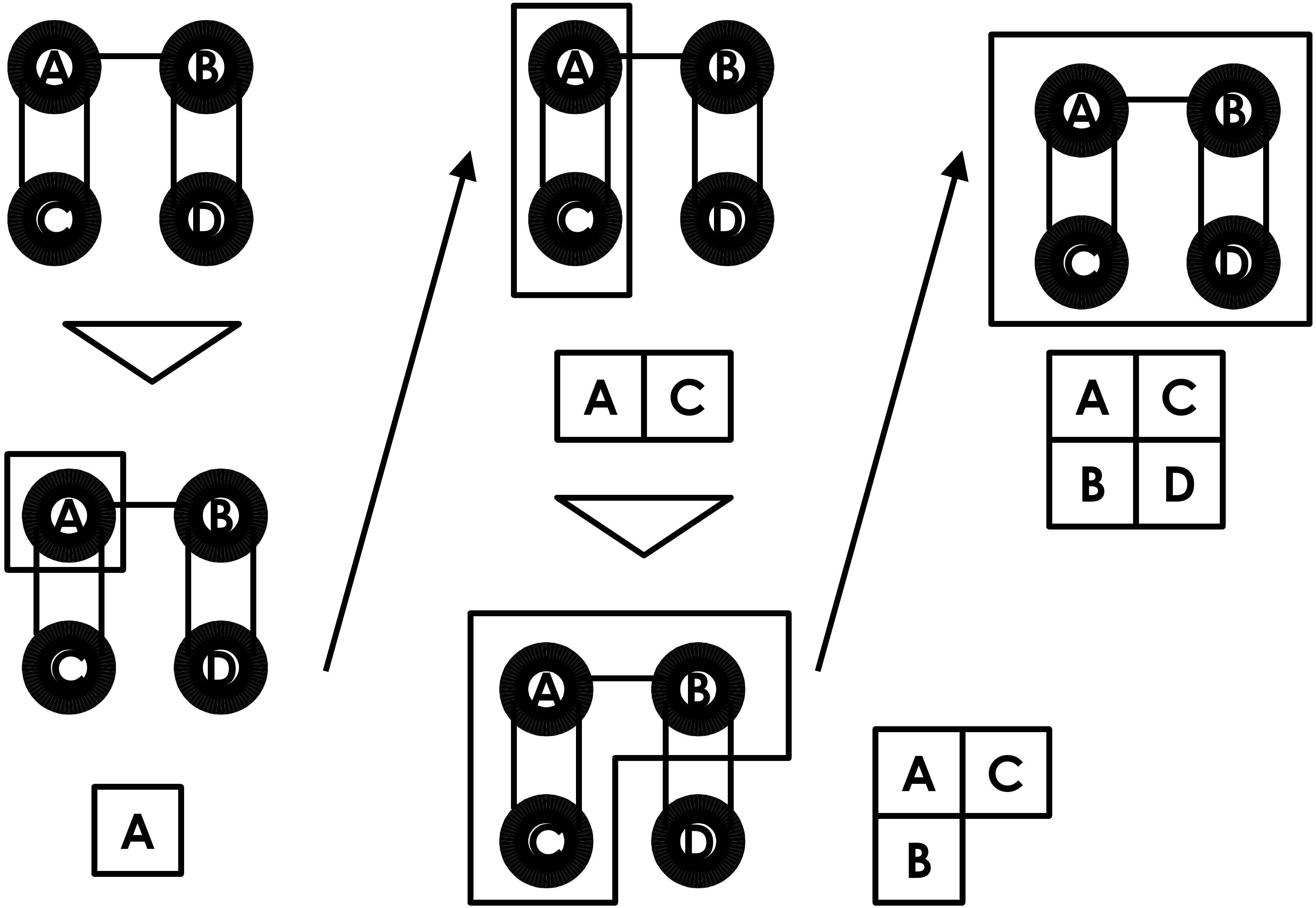
- Aufteilen des Graphen
  - Standardalgorithmen
- Zuweisung der Partitionen an Layout
  - Einschließlich Richtung der Aufteilung
  - Verschiedene Heuristiken
  - Beispiele:
    - Berücksichtige bereits zugewiesene Partitionen
    - Berücksichtige Chip-I/O-Pads

Platzierung mit Clustering 1

- Beginne mit einer Startzelle als Cluster
- Finde angeschlossene Zelle(n)
- Ordne Zelle(n) „nahe“ um Cluster an
- Füge neue Zellen dem Cluster hinzu

- Entscheidungen:
  - Welche Zellen(n) hinzufügen?
  - Auf welche Art nahegelegen anordnen?

Platzierung mit Clustering 2



Iterative Verbesserung 1

- „Kleine“ Veränderung bestehender Lösung
  - Ändere die Position von Zelle(n)
  - Falls besseres Ergebnis: Immer übernehmen
  - Schlechter: Unter Umständen übernehmen

  ➔ Abhängig von Suchverfahren!

Iterative Verbesserung 2

```
iterative_improvement () {
  s := initial_configuration();
  c := s.cost();
  while (!stop()) {
        s' := s.perturb();
        c':= s'.cost();
        if ( c.accept( c' ) )
            s := s';
  }
}
```

- **initial_configuration**
- **cost**
- **stop**
  - **z.B. #Iterationen**
  - **komplexer möglich**
- **accept**
  - **Nachbarsuche**
  - **Simulated Annealing**
  - **Tabu-Suche**

Iterative Verbesserung 3

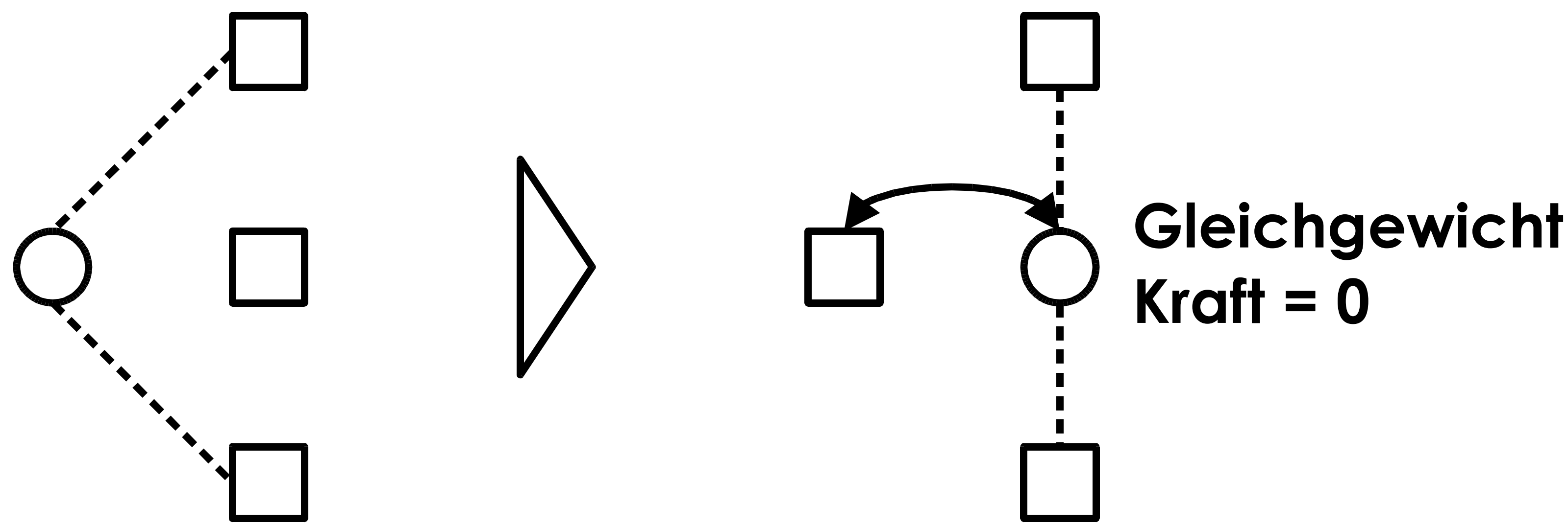
- perturb
  - Bei UPP: einfach, z.B. Positionstausch
    - Bei Standardzellen oder Building Block:
      - Unterschiedliche Zellgrößen, Überlappung möglich

Iterative Verbesserung 4

- Vorgehensweisen
  - Überlappung erlaubt, aber höhere Kosten
    - Bereinige beste gefundene Lösung am Ende
    - Möglicherweise drastische Verschlechterung
  - Beseitige Überlappung direkt nach jedem Zug
    - Bei BB sehr aufwendig, bei SC machbar
    - Aber so genauere Kostenberechnung möglich
  - Erzeuge nur überlappungsfreie Lösungen
    - Züge unter Umständen sehr viel aufwendiger

Iterative Verbesserung 5

- Alternativen zu zufäligem Zellaustausch
  - Kräfte-gesteuerte Auswahl des Partners
  - Bestimme Idealposition der Zelle
    - Reduziere durch Netze ausgeübte Anziehungskraft
  - Tausche dann mit Zelle auf Idealposition



Iterative Verbesserung 6

- **Berechnung des Schwerpunktes**
  - **Verwendet Cliquen-Modell $G(V, E)$**
  - **γ_{ij}: Gewicht von $(i,j) \in E$, $\gamma_{ij} = 0$ falls $(i, j) \notin E$**
  - **Bestimme Schwerpunkt (x_i^g, y_i^g) der Zelle *i***

$$x_i^g = \frac{\sum_j \gamma_{ij} x_j}{\sum_j \gamma_{ij}} \quad \text{Gewichteter Durchschnitt} \quad y_i^g = \frac{\sum_j \gamma_{ij} y_j}{\sum_j \gamma_{ij}}$$

  - **Bewege Zelle *i* dorthin**
  - **Was tun, wenn dort schon andere Zelle liegt?**
    - **Bewege andere Zelle auf ihren Schwerpunkt**
    - **Erzeugt Folge von Zügen, ggf. Tabu-Mechanismus**

Partitionierung 1

- Aufteilen eines Graphen
- Hier motiviert durch Plazierung
  - Min-Cut
- Andere Anwendungen
  - Aufteilen einer Schaltung auf mehrere Chips
  - Verkleinern der Problemgröße
    - Vorbearbeitung vor anderem Algorithmus
- Viele Verfahren
  - Beispiel: Kernighan-Lin

Kernighan-Lin Partitionierung 1

- **Problem**
  - **Gewichteter, ungerichteter Graph $G(V,E)$**
  - **$|V| = 2n$**
  - **γ_{ab}: Gewicht von $(a,b) \in E$, $\gamma_{ab}=0$ bei $(a,b) \notin E$**
  - **Finde Mengen A und B mit**
    - **$A \cup B = V$, $A \cap B = \emptyset$, $|A| = |B| = n$**
  - **Minimiere**

$$\sum_{(a,b) \in A \times B} \gamma_{ab}$$

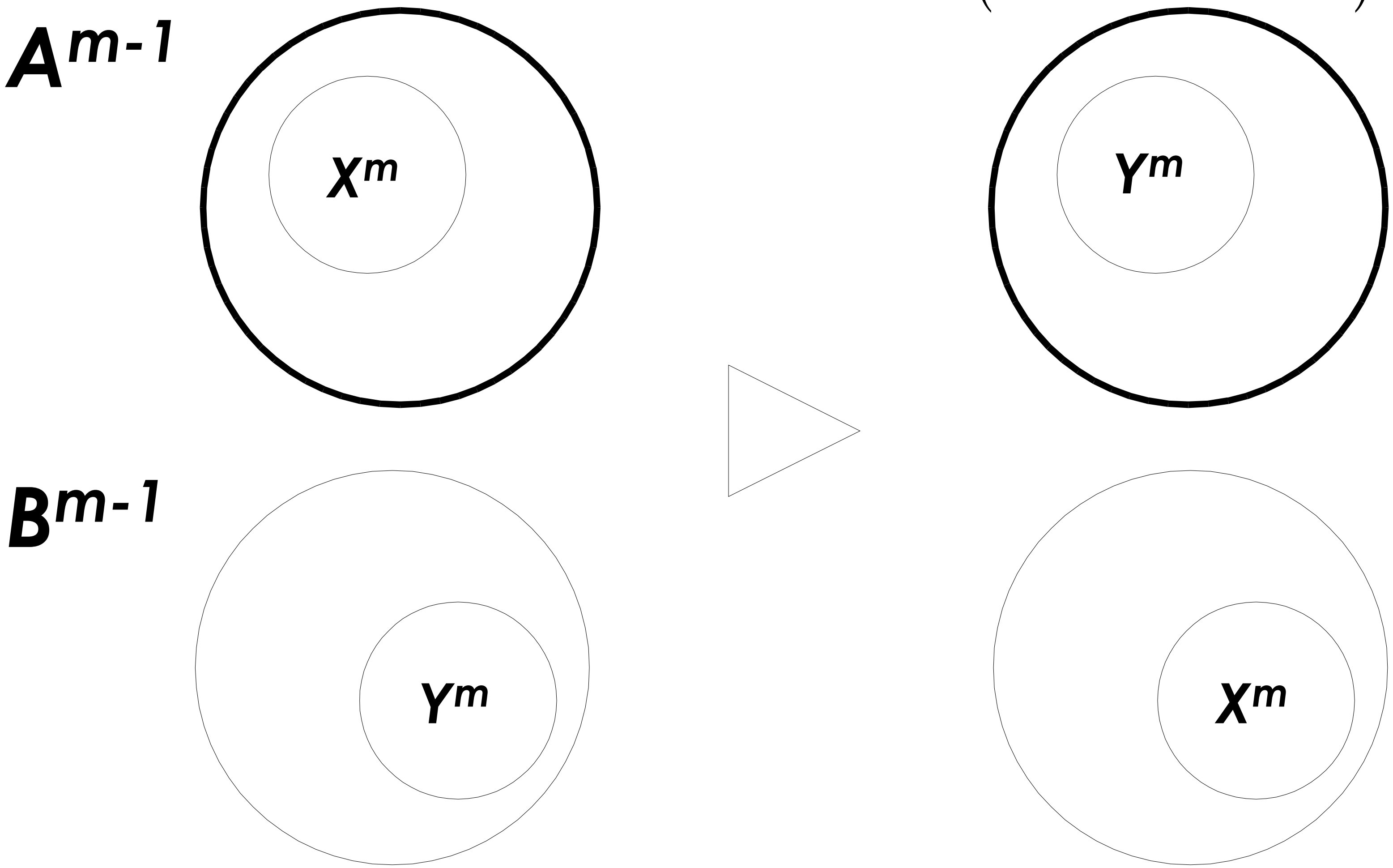
- **Arbeitet auf Cliquen-Modell**

Kernighan-Lin Partitionierung 2

- Partitionierungsproblem ist NP-vollständig
- KL ist eine Heuristik
  - Im praktischen Einsatz bewährt
- Vorgehensweise
  - Anfangslösung bestehend aus A^0 und B^0
    - I.d.R. nicht optimal
  - Isoliere Untermengen von A^{m-1} und B^{m-1}
  - Tausche diese aus um A^m und B^m zu bestimmen
  - Wiederhole, solange Verbesserung erreichbar

Kernighan-Lin Partitionierung 3

$$A^m = (A^{m-1} \setminus X^m) \cup Y^m$$



$$B^m = (B^{m-1} \setminus Y^m) \cup X^m$$

Kernighan-Lin Partitionierung 4

- Optimum immer in einem Schritt erzielbar
  - Bei geeignetem X^m und Y^m
- Problem: Wie X^m und Y^m bestimmen?
  - Schwer zu finden

➔ Suche Lösung in mehreren Schritten
  - Wiederhole, bis keine Verbesserung mehr
- Anzahl Schritte unabhängig von n
  - In der Praxis <= 4.

Kernighan-Lin Partitionierung 5

- Konstruktion von X^m und Y^m
- Externe Kosten

$$E_a = \sum_{y \in B^{m-1}} \gamma_{ay} \quad , a \in A^{m-1}$$

- Interne Kosten

$$I_a = \sum_{x \in A^{m-1}} \gamma_{ax} \quad , a \in A^{m-1}$$

- Analog für B

Kernighan-Lin Partitionierung 6

- $D_a = E_a - I_a$ für $a \in A^{m-1}$ (desirability)
  - >0: Knoten sollte nach *B* getauscht werden
  - <0: Knoten sollte in *A* bleiben
- Verbesserung Δ der Schnittkosten
  - Bei Austausch von $a \in A^{m-1}$ und $b \in B^{m-1}$

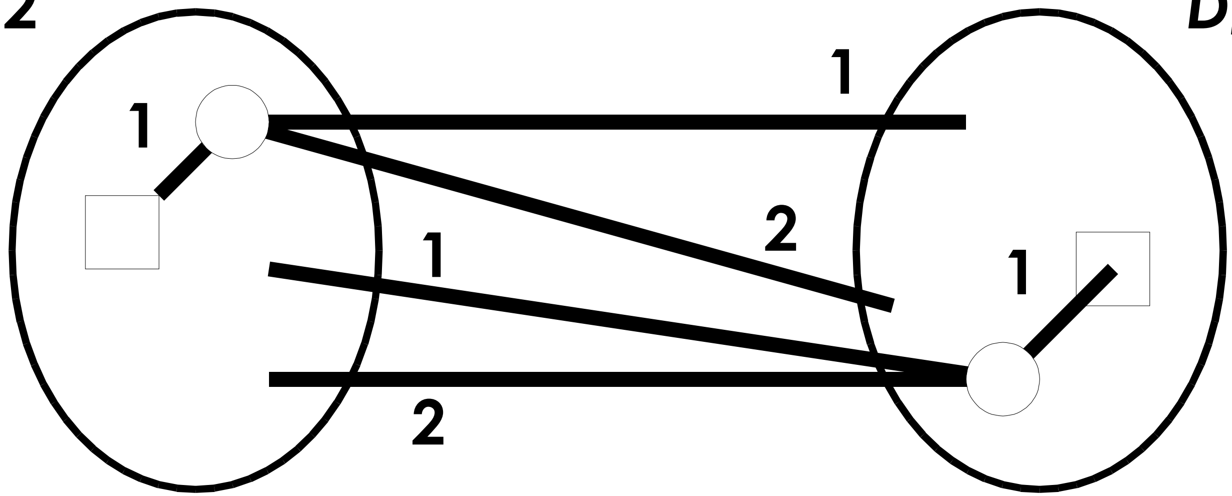
$$\Delta = D_a + D_b - 2\gamma_{ab}$$

  - Δ kann negativ sein!

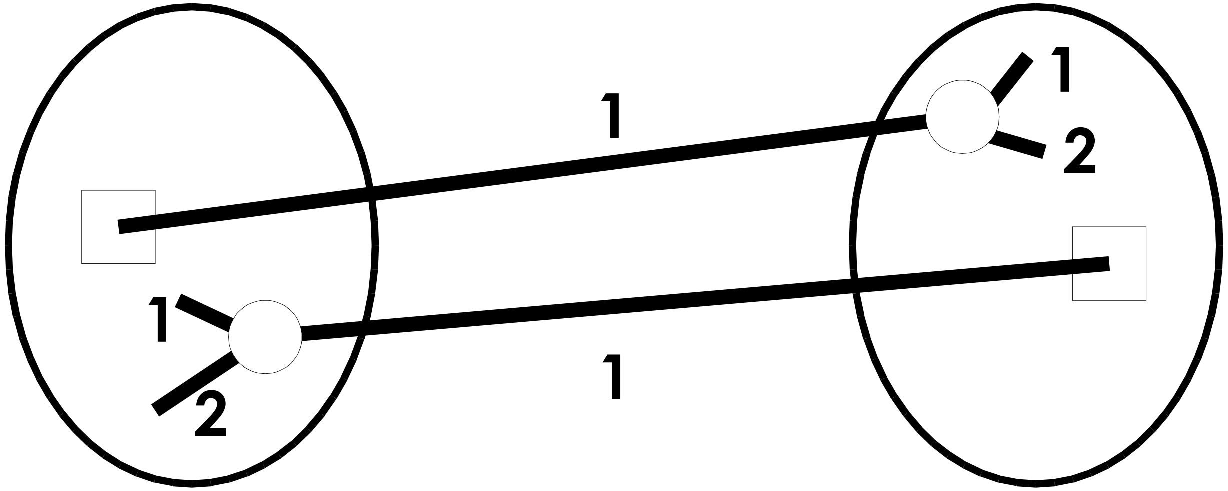
Kernighan-Lin Partitionierung 7

$D_a=3-1=2$

$D_b=3-1=2$

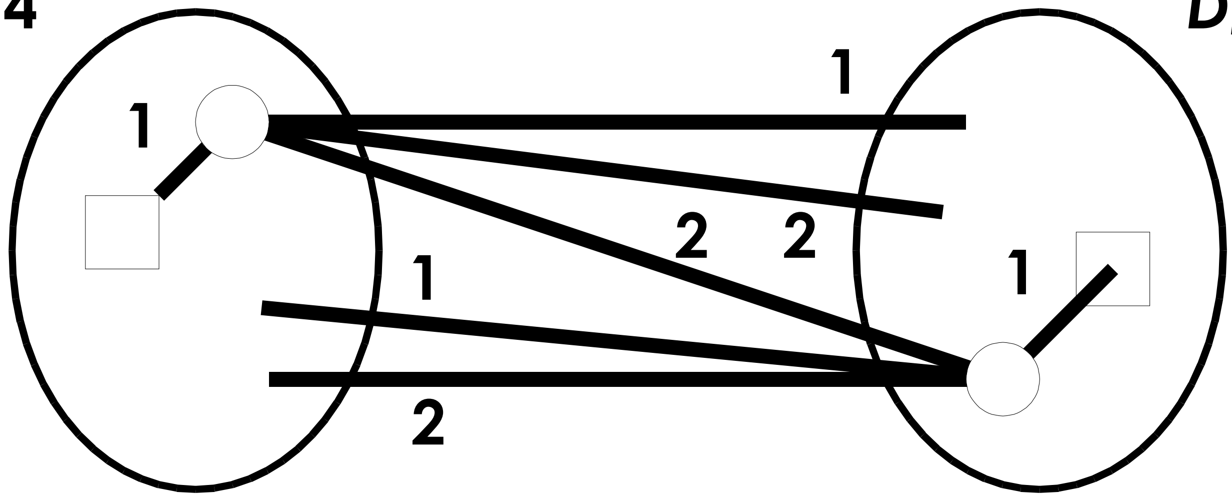


$$\Delta = D_a + D_b - 2\,\gamma_{ab} = 2 + 2 - 0 = 4$$

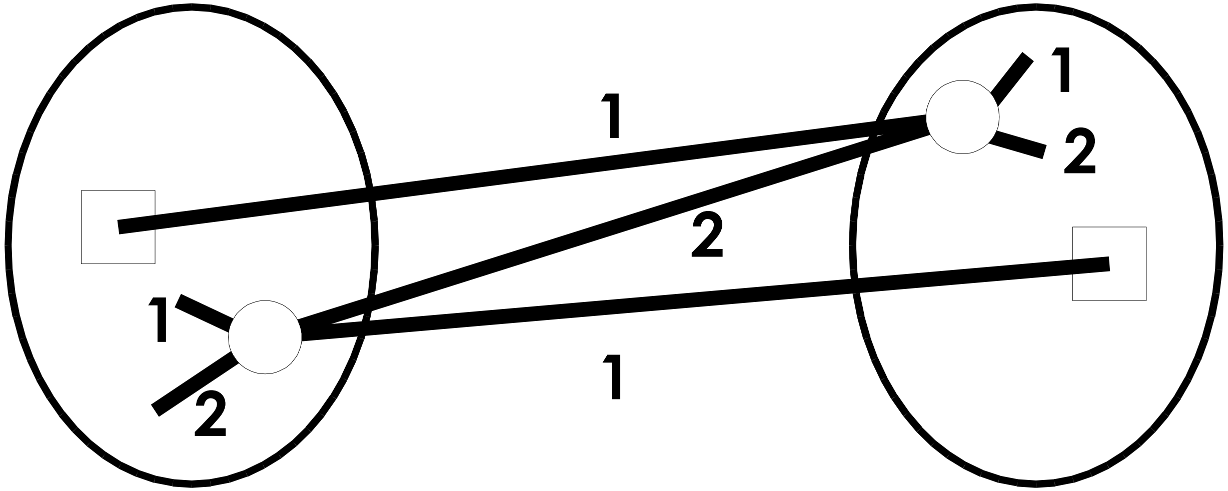


Kernighan-Lin Partitionierung 8

$D_a = 5 - 1 = 4$ $D_b = 5 - 1 = 4$



$$\Delta = D_a + D_b - 2\,\gamma_{ab} = 4 + 4 - 2 \cdot 2 = 4$$



Kernighan-Lin Partitionierung 9

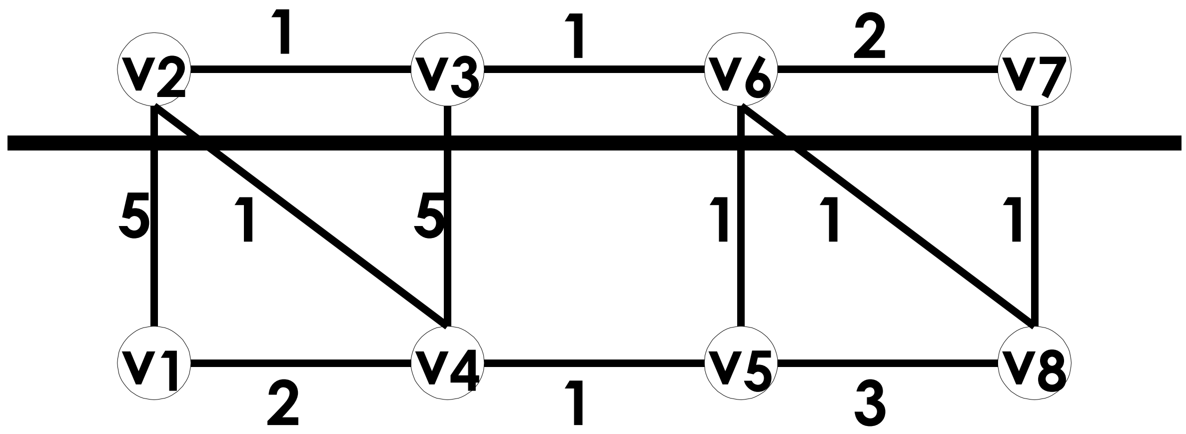
$$D_x = E_x - I_x$$
$$= E_x^{old} + \gamma_{ax} - \gamma_{bx} - (I_x^{old} - \gamma_{ax} + \gamma_{bx})$$
$$= D_x^{old} + 2\gamma_{ax} - 2\gamma_{bx}$$

initialize(A^0, B^0);
m := 1;
do {
  foreach $a \in A^{m-1}$
    "berechne D_a";
  foreach $b \in B^{m-1}$
    "berechne D_b"
  for (i:=1; i <= n; ++i) {
    "finde freie $a_i \in A^{m-1}$, $b_i \in B^{m-1}$ mit
      $\Delta_i := D_{a_i} + D_{b_i} - 2\,\gamma_{a_i b_i}$ maximal"
    "sperre a_i und b_i"
    foreach "freies" $x \in A^{m-1}$
      $D_x := D_x + 2\,\gamma_{x a_i} - 2\,\gamma_{x b_i}$;
    foreach "freies" $y \in B^{m-1}$
      $D_y := D_y - 2\,\gamma_{y a_i} + 2\,\gamma_{y b_i}$;
  }
  "finde ein k mit $\sum_{i=1}^{k} \Delta_i$ ist max."
  $G := \sum_{i=1}^{k} \Delta_i$
  if (G > 0) {
    $X^m := \{a_1, \ldots a_k\}$;
    $Y^m := \{b_1, \ldots, b_k\}$;
    $A^m := (A^{m-1} \setminus X^m) \cup Y^m$;
    $B^m := (B^{m-1} \setminus Y^m) \cup X^m$;
    "entsperre alle Knoten in A^m and B^m"
    m := m + 1;
  }
} **while** (G > 0);

Kernighan-Lin Partitionierung 10

- Δ_i kann negativ werden
- $\sum \Delta_i$ kann zeitweise auch negativ sein
  - Dicht verbundene Teilmengen
    - Keine Verbesserung bei Austausch von Einzelknoten
    - Erst bei Austauch der gesamten Teilmenge

Kernighan-Lin Partitionierung 11

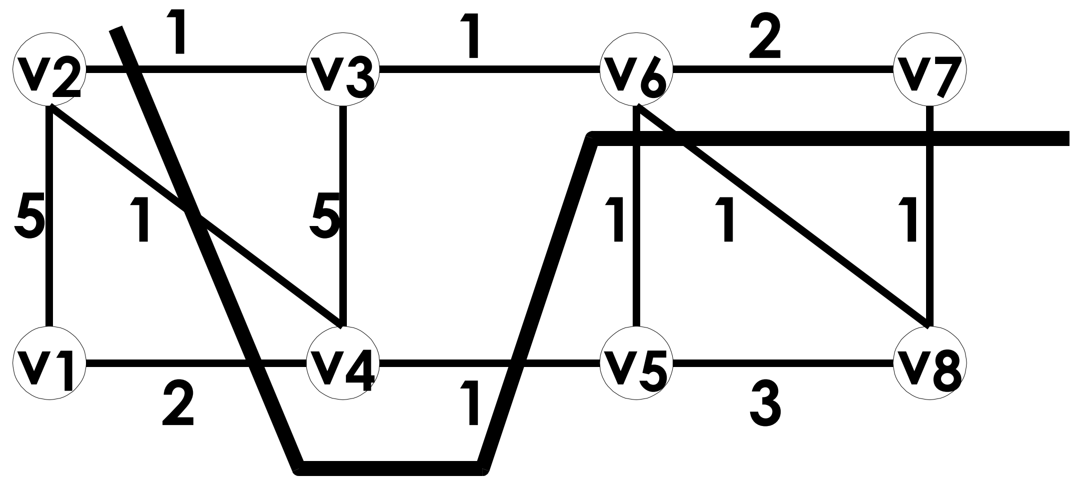


i	A^0: v2	v3	v6	v7	B^0: v1	v4	v5	v8	Δ_i
1	$\boxed{5}$	3	-1	-1	3	$\boxed{3}$	-3	-1	6
2		-5	-1	$\boxed{-1}$	-3		$\boxed{-1}$	-1	-2
3		-5	$\boxed{1}$		-3			$\boxed{3}$	2
4		$\boxed{-3}$			$\boxed{-3}$				-6

$\sum \Delta_i$

6 4 6 0

Kernighan-Lin Partitionierung 12



i	A^1 v3	v4	v6	v7	B^1 v1	v2	v5	v8	Δ_i
1	-5	$\boxed{-1}$	-1	-1	-3	-3	-1	$\boxed{-1}$	-2
2	$\boxed{5}$		-3	-3	-7	-5	$\boxed{3}$		8
3			$\boxed{-3}$	-3	$\boxed{-7}$	-7			-10
4				$\boxed{1}$		$\boxed{3}$			4

$\sum \Delta_i$

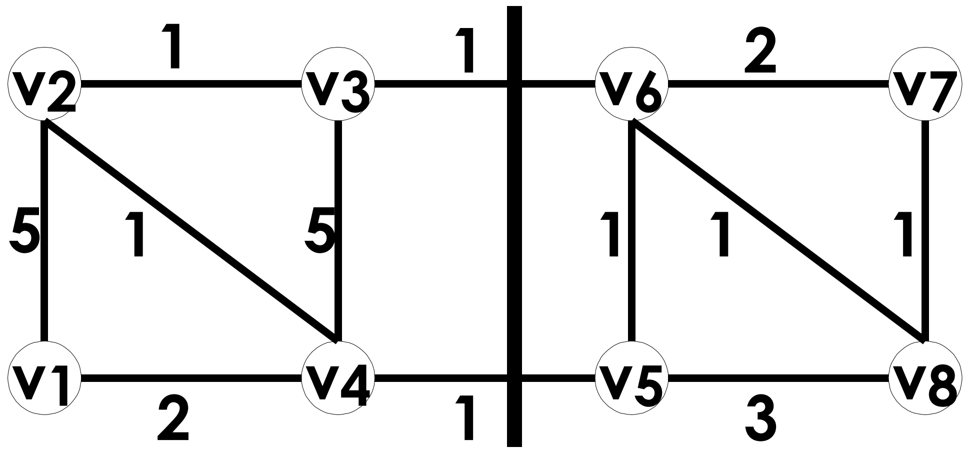
-2

6

-4

0

Kernighan-Lin Partitionierung 13



- Danach keine Verbesserung mehr in G
- Innere Schleife: *n* Iterationen
  - Finden des Paares mit bestem Δ: $O(n^2)$
  - Nach Δ sortiert: $O(n \log n)$

→ $O(n^3)$ oder $O(n^2 \log n)$

Kernighan-Lin Partitionierung 14

- KL: Lokale Suche mit variabler Nachbarschaft
- Schnellere Verfahren
  - Fiduccia-Mattheyses (FM)
    - Wesentlich schneller: O(n)
    - Aber schlechtere Qualität der Lösungen
  - QuickCut (QC): avg. O($|E| \log n$)
    - Gleiche Qualität wie KL
- Diverse Alternativen
  - Spectral Partitioning, Multi-Level-FM, ...

Weiteres Vorgehen

- Diesen Freitag keine Veranstaltung!

- Montag
  - Für 4SWS'ler: Abgabe 1. Aufgabe, 23:59 MET
    - Ausnahmsweise noch *ohne* Gruppennummer
    - Bitte in der Mail auch Klarnamen angeben
      - Nicht nur E-Mail-Adressen!
  - Keine Vorbereitung der Vorlesung
    - Vorstellung eines realen Platzierungswerkzeugs

Zusammenfassung

- Übung Timing-Analyse
- Längenmaße für Netze
- Platzierungsverfahren
  - Zieltechnologien
  - Konstruktiv
  - Iterativ
- Partitionierung: Min-Cut
  - Anwendung für Platzierung
  - Kernighan-Lin