# Physical Design Part

**Goal** → optimizing the placement of components like logic gates, transistors, Flipflops, memory elements etc in a large systems chip wafer so that power consumption is minimum, performance delay is reduced to minimum also.

→ a step in middle of ckt design in software simulation (say Quartus) and fabrication of the chip.

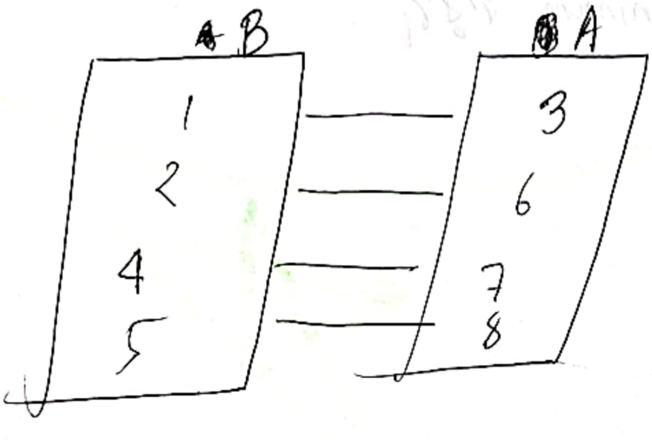
→ VLSI physical design flow → 6 টি steps

→ 1st step: Partitioning → Details will be discussed.

---

Block → Module বলে

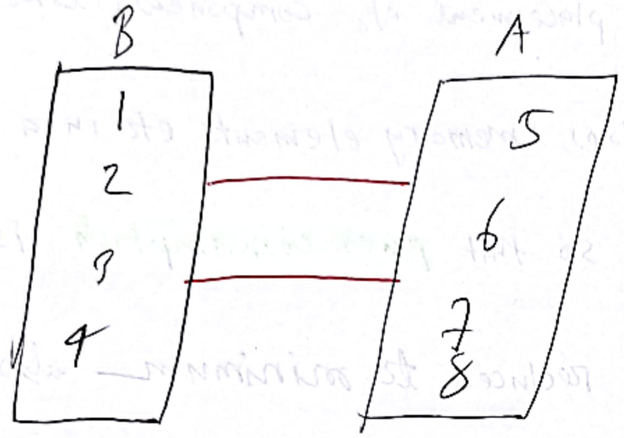
Gate → Node নাম দিয়ে থাকি।




→ $C_a$ ধরেছি:

4 টা external wires

$C_b$ অবস্থানে লাগে → only 2 wires

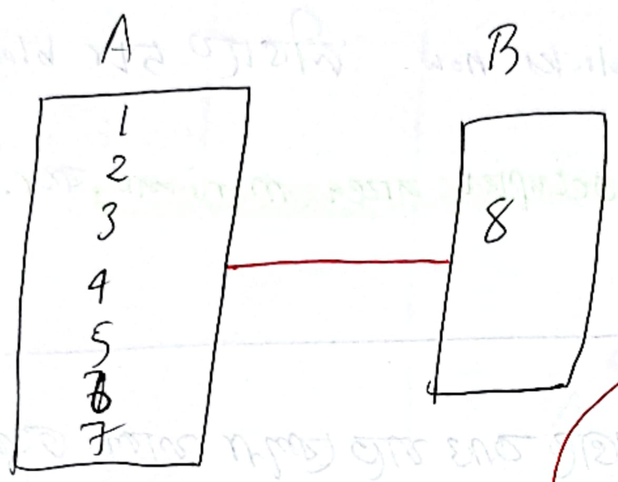



→ better due to lower delay (due to less dependency between B and A block).

→ Also, $C_a$ অবস্থানে → 4 wires → 4টা wire এর resistance এ $I^2R$ loss of power ও বেশি হবে।

Hence, goal → ২টা block এর connecting wire সংখ্যা minimum করা।

another way: → Not better design




try to keep no. of nodes/gates in both block equal/nearly equal for better design. otherwise, module size A and B becomes very imbalanced.

## Chip/floor Planning

let, I have 5 blocks now. এগুলোতে 5টা block বসাতে হবে (chip এর মধ্যে) ⇒ occupied area minimum করে সাজাতে।

---

Example prob → এমনভাবে সাজাও যাতে কোনো জায়গা আর blank না থাকে। → see slide-5

let, Block A has an area of — 4 unit

B " — 2 "

C — 3 "

---

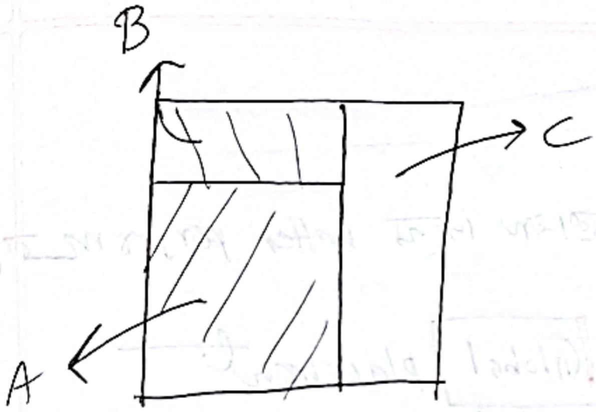
এমনভাবে width, height নিব যাতে পুরো area cover হয়।

A তে সম্ভব → 1 × 4, 4×2, 2×2 etc combo for area = 4 unit.

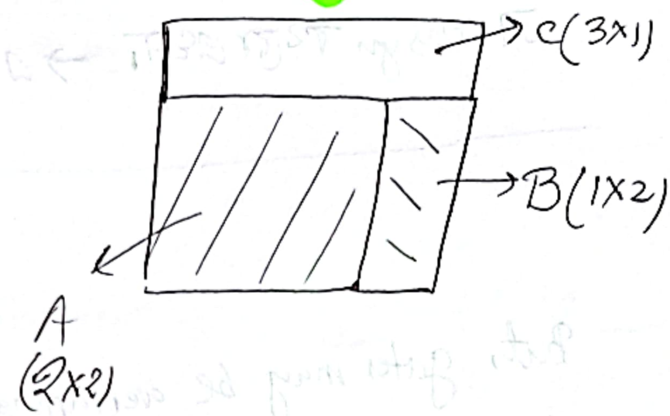
B " " → 1×2 or 2×1

C " " → 1×3 or 3×1 এমন।

See slide → A(2×2), B(2×1), C(1×3) নিয়ে optimum area পাইলাম।

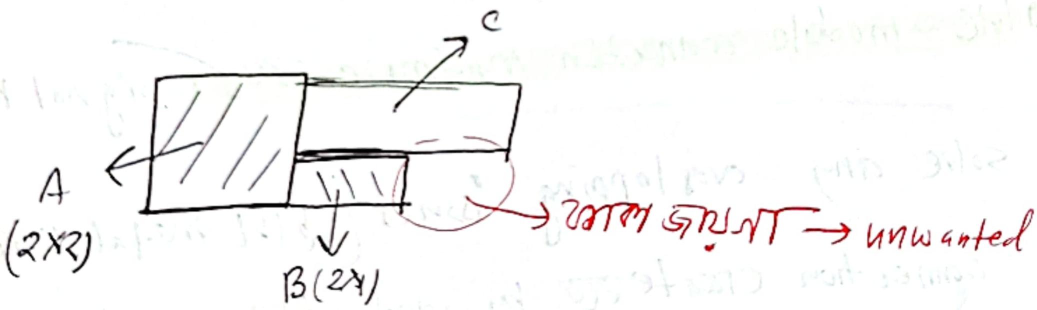



Alternative




এটি-ও সঠিক & correct.

if → A(2×2), B(2×1), C(3×1) এমনভাবে দিলে →




↳ not optimized design.

# Placement

now inside a module, তার gate কোথায় দিলে better perform করে তা design করতে হবে, → একে বলে Global placement

↓

software এটা করে দিবে।

But, gates may be overlapping with each other যা আবার solve করি in detailed placement. → যা real life এ দেয়া যাবে না।

---

# Signal Routing

→ module-module connection minimize করা = signal routing.

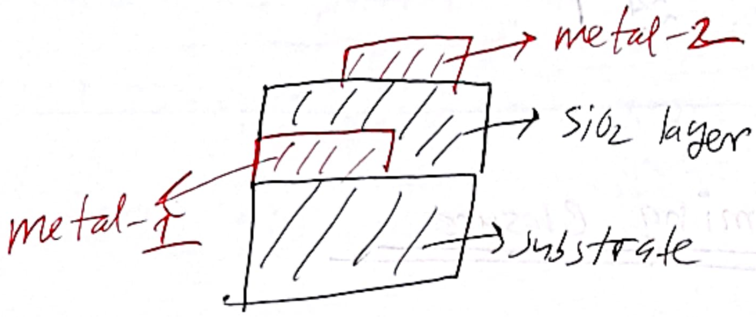
→ to solve any overlapping issues (2টি metal যখন overlap হলে connection create হয় by default) ⇒

either ঘুরিয়ে অন্যদিকে নিয়ে যাওয়া অথবা multiple metal layers use করা।

→ not good design

↓

good




no connection between 2 metal layers.

in slide-7, ~~[illegible]~~ এর horizontal connection done by metal-1
" vertical " " " metal-2

---

যদি Horizontal, vertical connection থাকে তাহলে metal-1, metal-2কে connect করা যাবে using via.

---

## CTS

farthest block গুলোতে clk যেতে long path hence delay বেশি হবে hence sequential logic ckt গুলো যখন clk edge পেতে কম বেশি time lag থাকে যার ফলে desired o/p পাওয়া না যায়। সমস্যা

solution: nearest logic block এ ঢুকার আগে extra delay introduce করা যাতে clk একই time এ nearest and farthest block এ received হয় ।

---

## Timing Closure

→ ensure that a signal is input to next block only after it is totally ready.

---

## Summary

You have designed a system with lots of gates and seq elements etc

→ partitioning into blocks or modules now

→ place these blocks in such manner that area is min —chip planning

→ Now, inside the block, placement করা যাতে delay কম হয়

→ signal routing

→ finally CTS to ensure even clk distribution

→ Timing Closure

# Algorithms for Partitioning

কোন গেইটকে block এ কাকে কাকে নিলে বা কী ভাবে কোন block নিলে minimum external connection হবে তা বুঝা হতেও তো পারে। Hence we develop an algorithm.

---

Initially একটা partitioning দেয়া থাকবে যা optimize করতে হবে।

See slide example → {3, 4, 5} একটা block বাকিগুলা অন্য block-এ।

for this initial pic → external connection আছে এখন = 3টা।
এটাকেই এখন minimize করা যাবে।

---

Graph representation → gate কে circle/node দিয়ে প্রকাশ করব।

Edge, E → কোনো node এ কতগুলো connection আছে তা।

in slide, $E(3) = 3$, $E(4) = 1$

দুই রকম হতে →
- Cut edge, $E_c$ → এক module হতে অন্য module এ যেই এমন connection (interblock)
- Non-cut Edge, $E_{nc}$ → এমন connection যা same block এই exist করে এমন (intra-block)

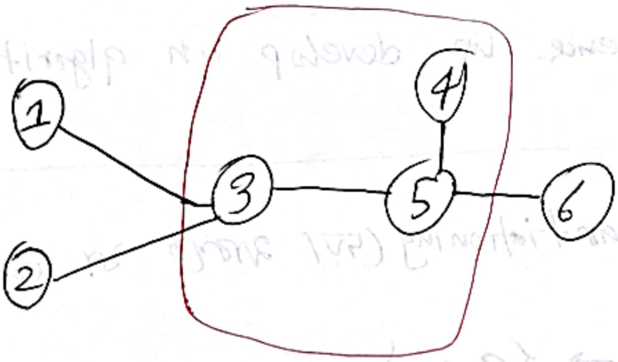
(3, 2), (3, 1) এ দুটো

(3, 5) এর মধ্যে

$E_c(3) = 2$, $E_{nc}(3) = \underline{1}$

(4, 5) এর মধ্যে

$E_c(4) = 0$, $E_{nc}(4) = \underline{1}$




---

Cost, D → যে কোনো node এর (no. of cut edge − no. of non-cut edge)

$D(3) = 2 - 1 = \underline{1}$

means, cost of moving a node.

$D(2) = 1 - 0 = \underline{1}$

$D(4) = 0 - 1 = -\underline{1}$

Cost > 0 হলে, node move করা উচিত

Cost < 0 হলে do not move the node out of its present module/block.

Gain $\Delta g$: এইবার ২টা node কে compare করবো → ২টা node যদি swap করি তাহলে তা better হয় কিনা তা measure করবো

→ এমন ২টা node এর gain calc. করবো যারা ২টা different block এ আছে। same block এর node একবারেও না।

---

$A \rightarrow \{3, 4, 5\}$

$B \rightarrow \{1, 2, 6\}$ in slide example.

A block এর যেকোনো node হতে পারে

its any node from block B

$$\Delta g(a, b) = D(a) + D(b) - 2\, c(a, b)$$

node a এর individual total cost হিসাব।

c(a,b) is the connection weight between a and b:
If an edge/connection exists between a and b, then c(a,b) = edge weight (here 1),

If 2 edge/connections exist between a and b, then c(a,b) =2

if no connection, then c(a,b) = 0.

---

let, node 3, 6 কে swap করতে চাই →

swap এর আগে, $D(3) = 2 - 1 = 1$, $D(6) = 1 - 0 = 1$

$c(3,6)$ = no direct connection exists = 0

hence, (3,6) swap করতে চাইলে, $\Delta g(3,6) = 1 + 1 - 2 \cdot 0$

$= 2$

$\Delta g(a,b)$ যতো বেশি হবে ততো beneficial for us.

---

cut-cost → একটা block এর সাথে অন্য block এর total connection. Here, cutcost = 3 in slide.

partition এ যে সব connection cut করে সেগুলো।

---

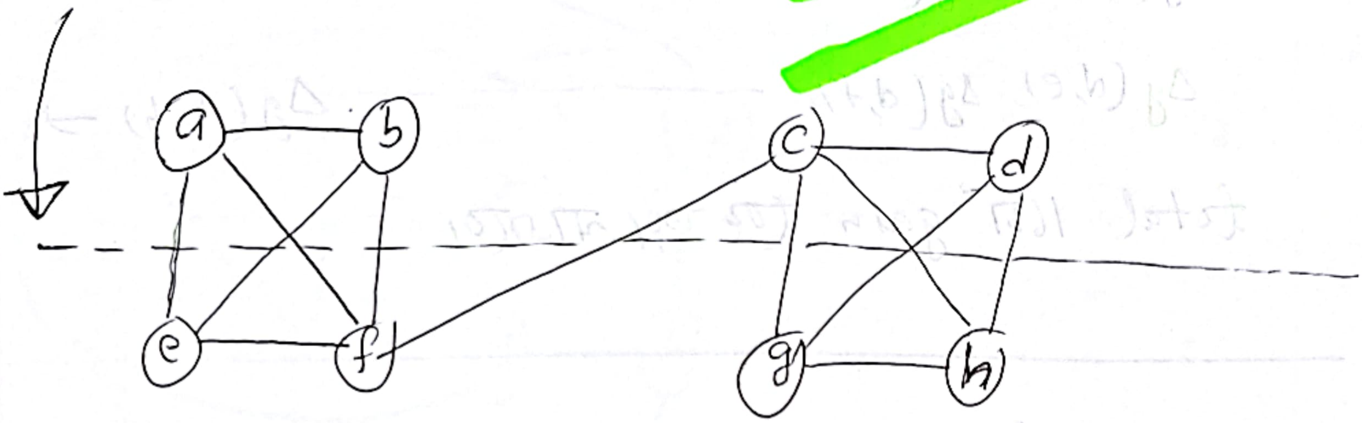
→ এটা কমালে আসলেই সব inter block connection নাই হবে। তাছাড়া যত বেশি a node এর cost কমাতে চাই অবশ্যই।

---

# Slide example - 2

# Imp

initial partition দেওয়া আছে in slide → cut cost = 9 now.




## First iteration:

1. সব block এর cost হিসাব:

A: $D(a) = 2 - 1 = 1$, $D(b) = 2 - 1 = 1$, $D(c) = 3 - 1 = 2$, $D(d) = 2 - 1 = 1$

B: $D(e) = 1$, $D(f) = 2$, $D(g) = 1$, $D(h) = 1$ same ভাবে

Then gain বের করে দেখবো কাকে কার সাথে swap করলে max profit হবে।

---

a এর সাথে b, c and d check করে লাভ নাই কারণ এরা same block এ আছে, অন্য block node এর সাথে gain check করবো আমরা।

$\rightarrow \Delta g(a,e) = D(a) + D(e) - 2c(a,e) = 1 + 1 - 2 \cdot 1 = 0 \rightarrow$ not profitable hence

$\Delta g(a,g) = 2$

$\Delta g(a,f) = 1 + 2 - 2 \cdot 1 = 1$

$\Delta g(a,h) = 2$

এরকমভাবে $\Delta g(b,e)$, $\Delta g(b,f)$, $\Delta g(b,g)$, $\Delta g(b,h)$,

$\Delta g(c,e)$, $\Delta g(c,f)$, ——————,

$\Delta g(d,e)$, $\Delta g(d,f)$, —————— $\Delta g(d,h) \rightarrow$

total 16টা gain বের করা লাগবে।

---

$\Delta g(c,e) = (2+1-0) = 3 \rightarrow$ max gain $\rightarrow$ $\Delta g_1$ নাম দিই, এটা

Hence, (c,e) কে swap করি। Always max gain বের করে হবে যতক্ষণ না swap করতে করতে max gain = 0 বা −1 এমন কিছু হবে।

---

⇒ 2nd figure এ swapped

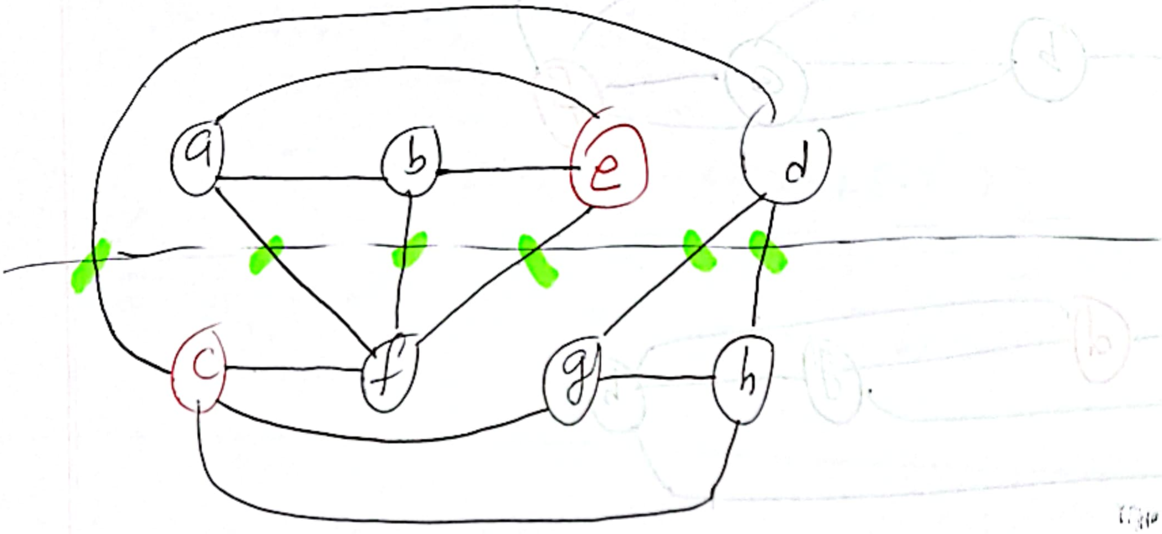
↓

Now, start 2nd iteration in same way now.

here, cutcost = 6টা। 9 হতে 6 টা হল মানে improved হচ্ছে।

যেই ২টা node already swap করে ফেলেছি তাদের আর cost calculation দরকার নাই। এখন বাকি Node গুলোর জন্য করা লাগবে।

মানে node c, e বাদ




A : $D(a) = 1-2 = -1$ , $D(b) = -1$ , $D(d) = 3$

B : $D(f) = 3-1 = 2$ , $D(g) = 1-2 = -1$ , $D(h) = -1$

$\Delta g(a,f) = -1+2-2\cdot 1 = -1$

$\Delta g(a,g) = -1-1-2\cdot 0 = -2$

$\Delta g(a,h) = -1-1-2\cdot 0 = -2$

$\Delta g(b,f) = 2+2-2\cdot 1 = 2$

$\Delta g(b,g) = -1-1-2\cdot 0 = -2$

$\Delta g(b,h) = -1-1-2\cdot 0 = -2$

---

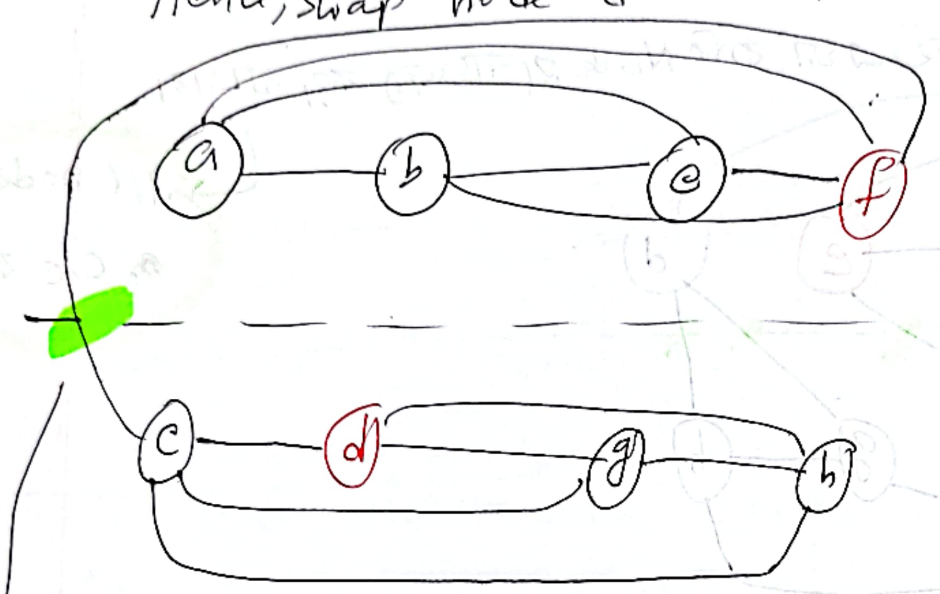
$\Delta g(d,f) = 3+2-2\cdot 0 = 5$ → এটা max

$\Delta g(d,g) = -1-1-2\cdot 1 = -4$

$\Delta g(d,h) = 3-1-2\cdot 1 = 0$

node e, node c এর হিসাব বাদ

Hence, swap node d and f. ($\Delta g_2 = \Delta g(d,f) = 5$)




→ now cut-cost = 1 only যা optimum. Still, we will have to continue iteration. 3rd iteration → এবার node e, c, f, d এরা সব বাদ। বাকিগুলোর জন্য gain calc. করবো।

$D(a) = -3,\quad D(b) = -3,\quad D(h) = -3$

$\Delta g(a,g) = -3-3-2.0 = -6$

$\Delta g(a,h) = -3-3-2.0 = -6$

$\Delta g(b,g) = -3-3-2.0 = -6$

$\Delta g(b,h) = -3-3-2.0 = -6$

$\therefore \Delta g_3 = (-6)$

Swap (a, g) now

[যেকোনো একটা নিলাম।]

see slide pic - after swapping (a, g) nodes →

এখানে only unswapped nodes → (b, h) এর gain calc.

and এদের swap করে এবার in 4th iteration.

→ $\Delta g_4$ আসে $= (-2)$

→ প্রত্যেক জোড়া node-ই swapping done.

---

## How to decide the optimum design solution?

cumulative max gain টা বের করে decide.

$G_1 = \Delta g_1 = 3$

$G_2 = \Delta g_1 + \Delta g_2 = 8 \longrightarrow$ max

$G_3 = \Delta g_1 + \Delta g_2 + \Delta g_3 = 3 + 5 - 6 = 2$

$G_4 = \Delta g_1 + \Delta g_2 + \Delta g_3 + \Delta g_4 = 3 + 5 - 6 - 2 = 0$

---

$G_2 \rightarrow$ max hence এর $G_m = 8$ when iteration 2nd step

এ জন্যে hence → $\boxed{\text{swap } (c, e) \text{ and } (f, d)}$

at $m^{th}$ iteration

if $G_m$ reaches zero or negative $\Rightarrow$ থেমে যাও আমরা,

---

One pass → ~~[illegible]~~ সব possible iteration শেষ করে $G_m$ বের করবো। if $G_m > 0$ → 2nd pass এ চলে যাবো otherwise → if $G_m \le 0$ → থেমে যাও in first pass.

---

~~pass [illegible] means iteration → last match এ যদি $G_m = G_2$ আর 2nd iteration এ আমার max gain $G_m$ পাইলাম hence 2nd pass পর্যন্ত করা লাগতো।~~

# How KL algorithm works

initial partition will be given

for ( pass = 1; $\cancel{G_m < 0}$ $G_m > 0$ ; pass = pass + 1) {

for (iter = 1; all swapping done $\neq$ ~~...~~ true ; iter = iter + 1) {

for each iteration, find max $\Delta g(a,b)$ and $G$ (cumulative gain) by calculating corresponding $\Delta g_{iter}$.

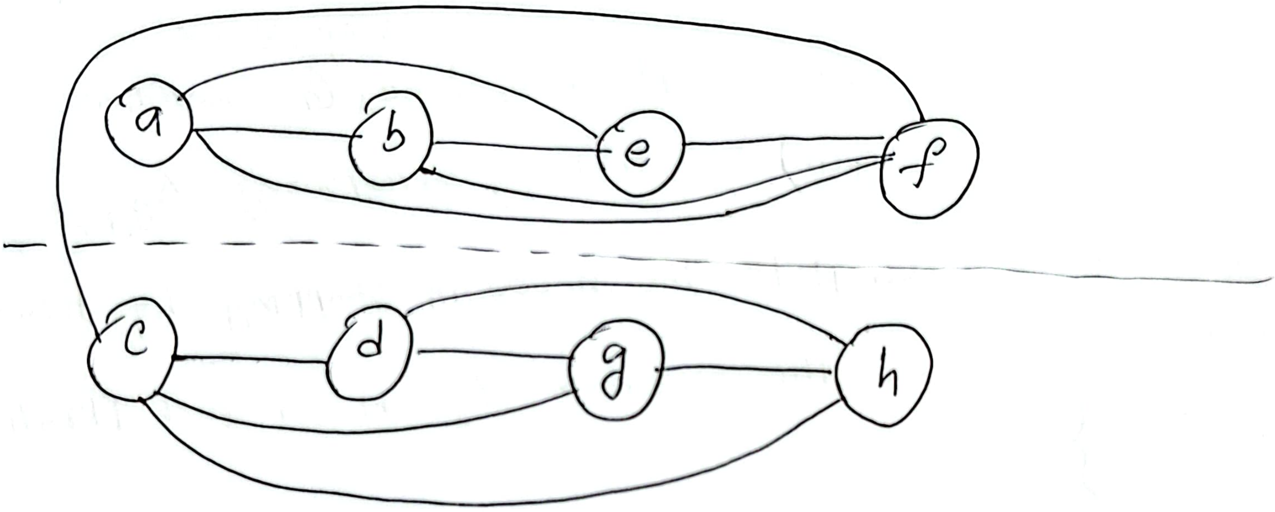
Complete the necessary swapping by focusing on the maximum $\Delta g(a,b)$ for each iteration.

}

For each pass → First find the iter value where you get max cumulative gain, $G_m$. if $G_m > 0$, the outer for loop will execute now. However, pass = 2 will start taking the structure/arrangement of the nodes where you got maximum cumulative gain, $G_m$. For example, if you get $G_m$ at iter = 2, pass = 2 will start after this iteration and total 2 swaps.

}

Hence, in our example-2 in the slide since $G_m = 8$ we need to go to 2nd pass now. We shall take the structure after 2nd iteration now. (means, after 2 swappings are done).




Now, run through all 4 iterations again and find the maximum cumulative gain $G_m$. if $G_m \leq 0$ now, you'll declare that the structure you took at the starting of the 2nd pass is the correct and optimized structure/arrangement for the nodes.