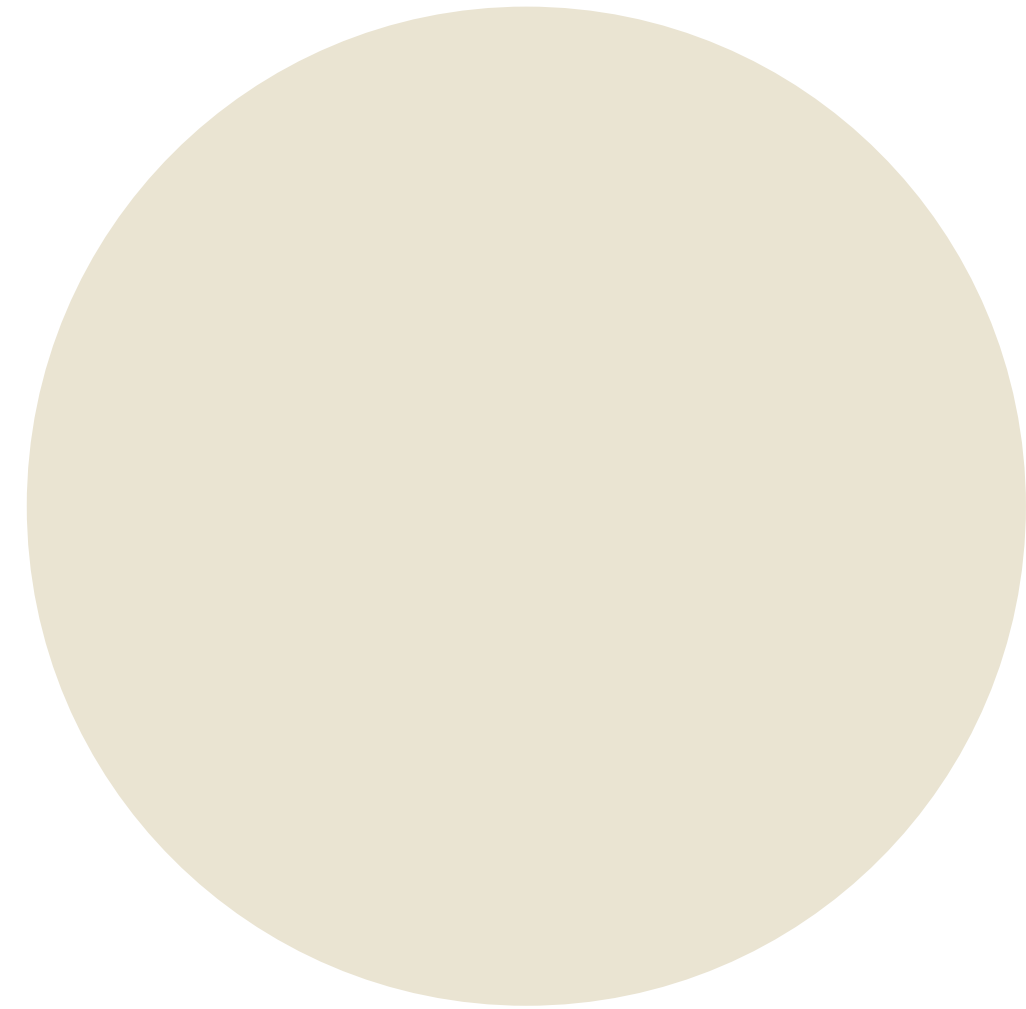




# ASA 9

● BRANCH AND BOUND



# PENGERTIAN

Branch and Bound (B&B) adalah salah satu metode algoritma untuk menyelesaikan masalah optimasi, baik minimasi maupun maksimasi, terutama dalam masalah pemrograman integer atau kombinatorial. Algoritma ini bekerja dengan membagi (branching) masalah menjadi sub-masalah yang lebih kecil dan membatasi (bounding) pencarian solusi berdasarkan estimasi nilai optimal.

Note: Branch and Bound mirip seperti BFS, tetapi queue yang digunakan adalah priority queue.

# KONSEP DASAR

## a. Branch (Pencabangan)

Membagi masalah besar menjadi sub-masalah yang lebih kecil, biasanya dalam bentuk pohon pencarian (search tree).

## b. Bound (Batasan)

Menentukan batas bawah atau atas dari solusi terbaik yang mungkin bisa dicapai oleh suatu cabang/sub-masalah.


## c. Pruning (Pemangkasan)

Menghapus atau mengabaikan cabang-cabang yang tidak dapat menghasilkan solusi lebih baik daripada solusi terbaik sementara (bound-nya tidak menjanjikan).

# STRATEGI

1. Mulai dari akar pohon yang mewakili masalah lengkap.
2. Cek bound solusi dari node saat ini.
3. Jika bound lebih buruk dari solusi terbaik sementara, pangkas node tersebut.
4. Jika tidak, cabang lebih lanjut dan ulangi proses.
5. Simpan solusi terbaik yang ditemukan sejauh ini.
6. Berakhir ketika seluruh node telah dieksplorasi atau dipangkas.

# PSEUDOCODE



```
def branch_and_bound(problem):  
    solusi_terbaik = None  
    antrian = [node_akar]  
  
    while antrian:  
        node = antrian.pop()  
  
        if solusi_dapat_ditingkatkan(node, solusi_terbaik):  
            if node_adalah_solusi(node):  
                solusi_terbaik = update_solusi_terbaik(node)  
            else:  
                for child in cabang(node):  
                    if bound(child) > nilai(solusi_terbaik):  
                        antrian.append(child)  
  
    return solusi_terbaik
```

# CONTOH: TSP

Masalah TSP adalah mencari rute terpendek yang melewati setiap kota sekali saja dan kembali ke kota awal.

Untuk kasus TSP, buat matriks jarak antar kota, lalu kurangi (reduce) matriks ini dengan cara:

- Kurangi setiap baris dengan nilai minimum di baris itu.
- Lalu kurangi setiap kolom dengan nilai minimum di kolom itu.
- Biaya pengurangan ini dijumlahkan → inilah bound awal dari node itu.

# CODE - TSP

[https://drive.google.com/file/d/1KSXDGonihrkRGqByZCDdYR4QQJOda6tA/view?usp=drive\\_link](https://drive.google.com/file/d/1KSXDGonihrkRGqByZCDdYR4QQJOda6tA/view?usp=drive_link)

# THANK YOU



● FOR YOUR NICE ATTENTION

---

link canva: [https://www.canva.com/design/DAGilKI7GW8/wXE9kZO88PUo-iWzQUyCAQ/edit?  
utm\\_content=DAGilKI7GW8&utm\\_campaign=designshare&utm\\_medium=link2&utm\\_source=sharebutton](https://www.canva.com/design/DAGilKI7GW8/wXE9kZO88PUo-iWzQUyCAQ/edit?utm_content=DAGilKI7GW8&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton)