Metoda Searching

Blind Search

Dosen

RAHMIATI

Metode Searching

Pencarian : suatu proses mencari solusi dari sebuah masalah melalui sekumpulan kemungkinan ruang keadaan (state space)

Blind Searching / Uninformed Search

- Pencarian buta (tidak ada informasi awal yang dipakai dalam proses pencarian)

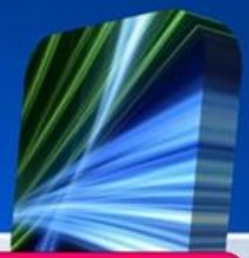
Heuristic Searching / Informed Search

- Pencarian terbimbing (Ada informasi awal yang dipakai dalam proses pencarian)

Mengapa menggunakan pencarian buta?

1. Ada kasus dimana tidak ada informasi awal yang dapat digunakan
2. Jawaban yang kita cari hanya bisa diketahui pada saat kita menemukannya

Blind Searching (Pencarian Buta) / Uninformed Search



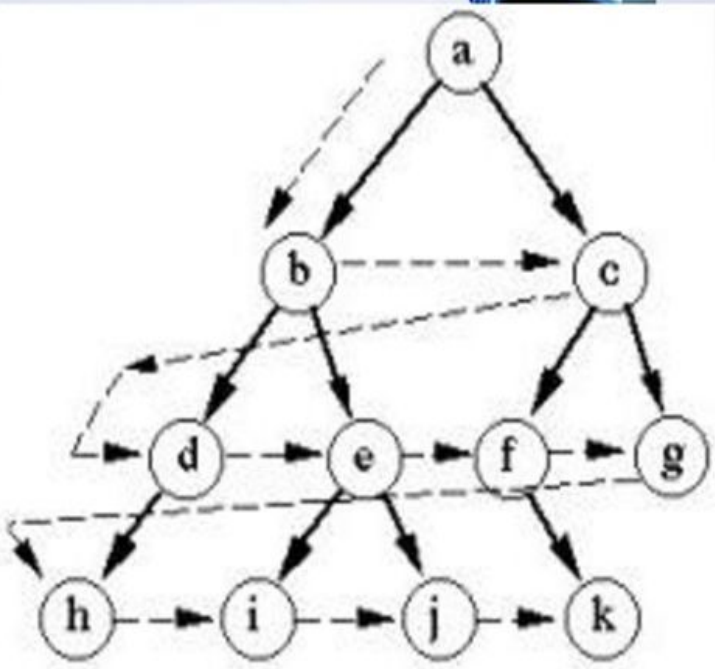
- **Breadth-First Search (BSF)**
- **Depth-First Search (DFS)**
- Uniform Cost Search (UCS)
- Depth Limited Search (DLS)
- Iterative Deeping Depth First Search (IDDFS)
- Bidirectional Search (BS)

Breadth-First Search (BFS)

- Dimulai dari Level 0 yaitu **simpul akar** A (root node A) menuju ke **Tujuan (Goal)** adalah simpul K

-Menggunakan prinsip **First In First Out (FIFO)**

-Menggunakan Prinsip **Queue / Antrian**



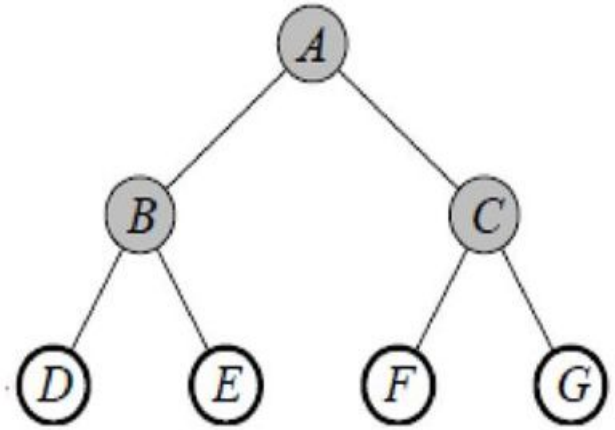
Breadth-first search

Breadth-First Search (BFS)

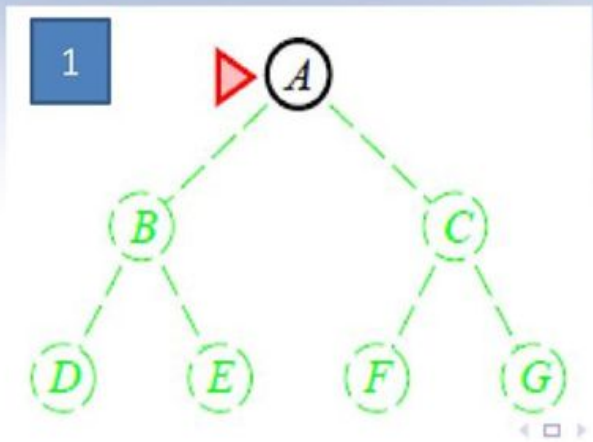
- Pencarian dilakukan pada semua simpul dalam setiap level secara berurutan dari kiri ke kanan
- Jika pada suatu level belum ditemukan solusi (tujuan yang dicari) maka pencarian dilanjutkan pada level berikutnya (n+1)
- Contoh

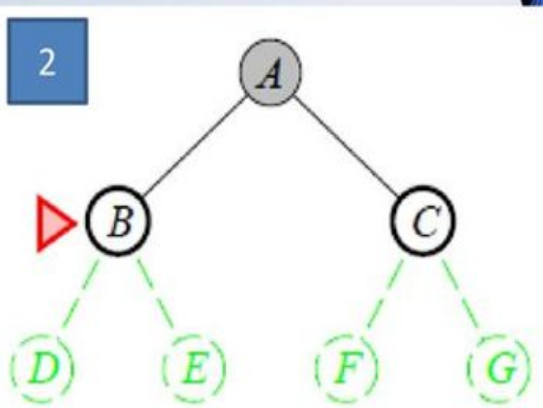
  Root : A

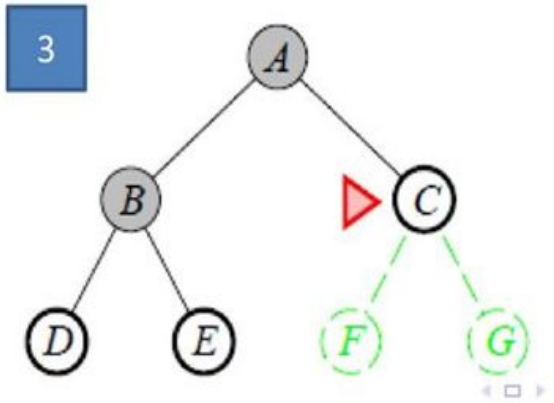
  Goal : D

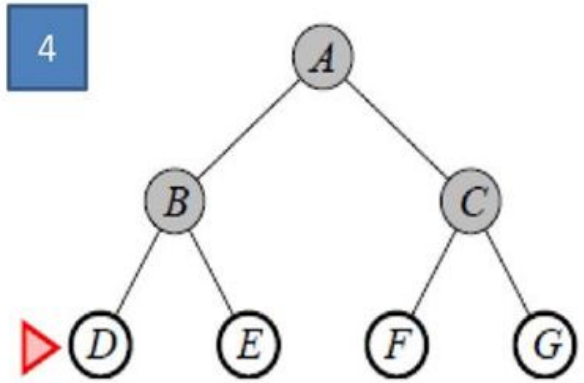


Tahap Breadth-First Search (BFS)







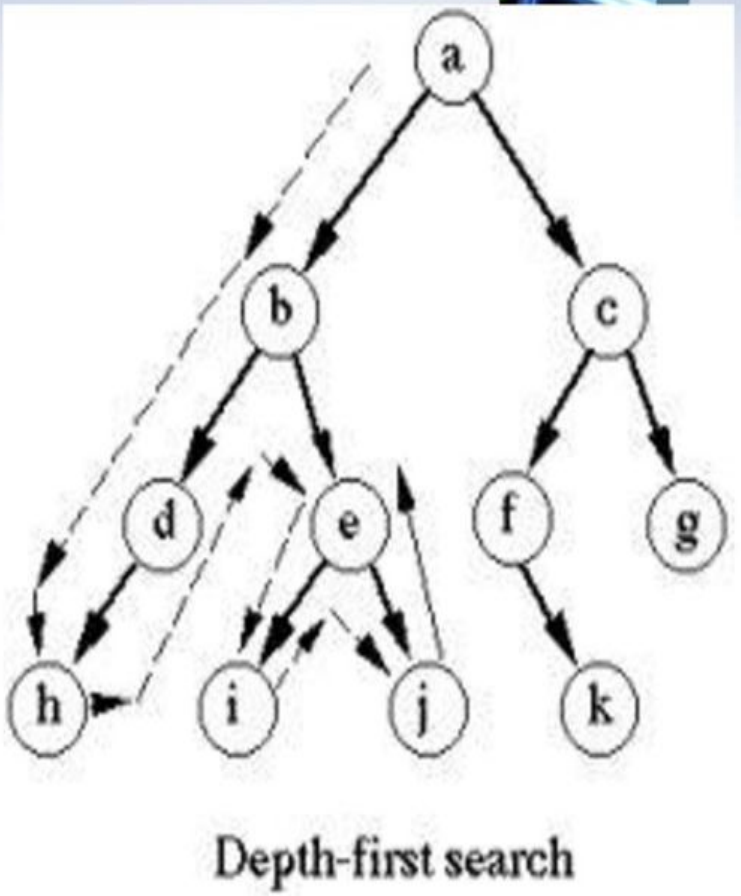


Depth-First Search (DFS)

-Dimulai dari Level 0 yaitu **simpul akar** A (root node A) dan **Tujuan (Goal)** adalah simpul J

-Menggunakan prinsip **Last In First Out (LIFO)**

-Menggunakan Prinsip **Stack / Tumpukan**



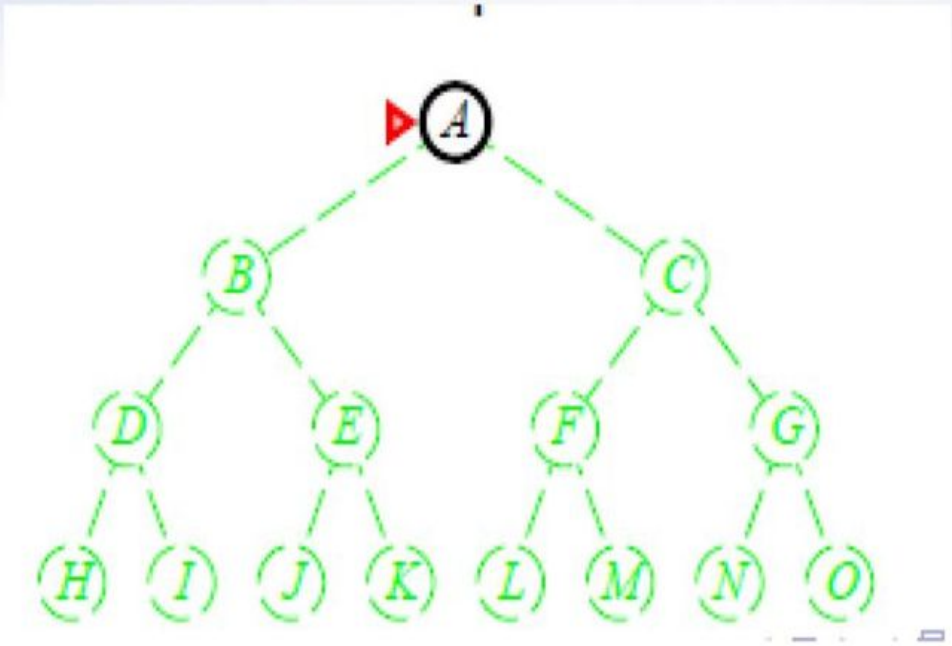
Depth-first search

Depth-First Search (DFS)

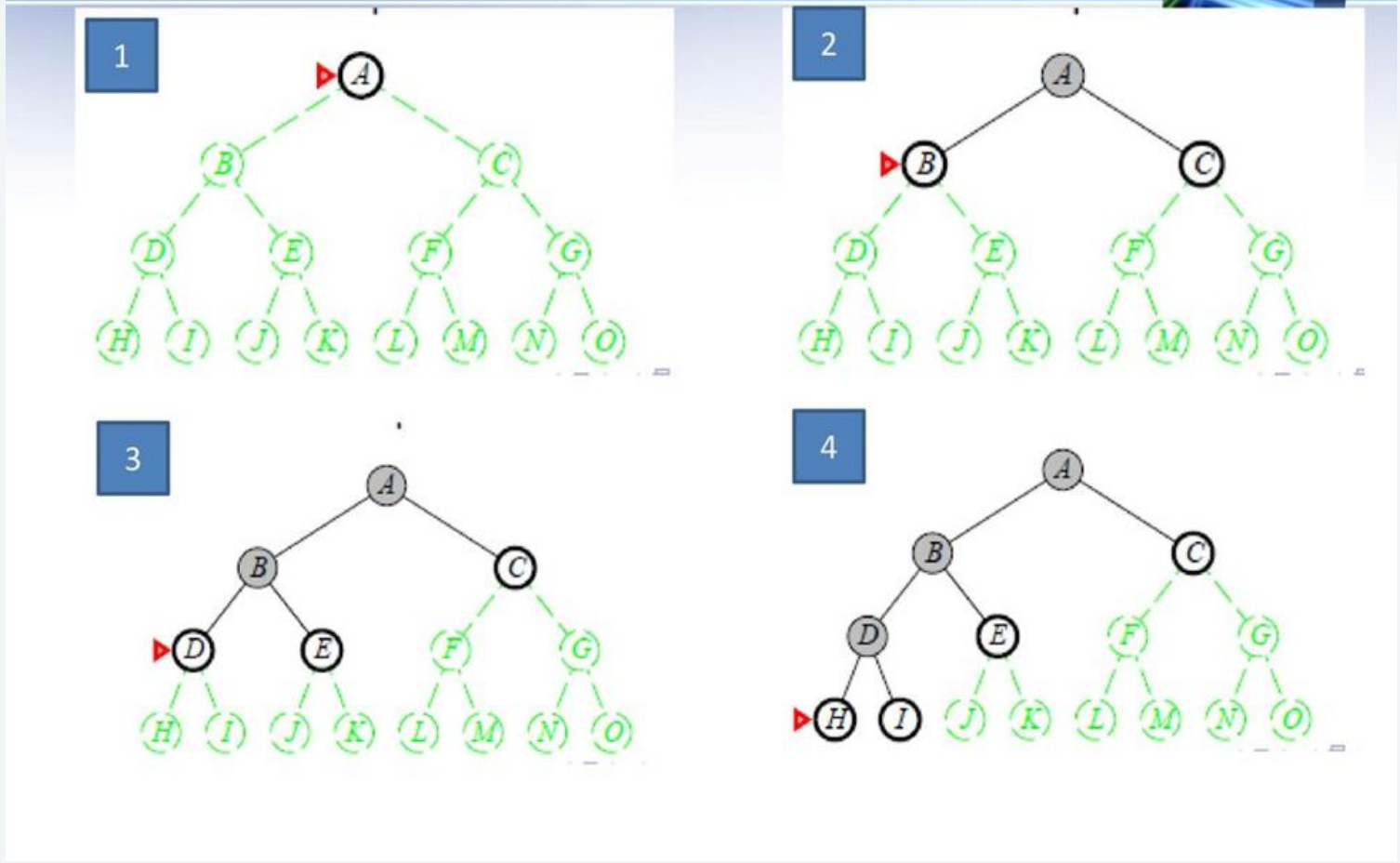
- Pencarian dilakukan pada suatu simpul dalam **setiap level dari yang paling kiri.**
- Jika pada **level yang terdalam**, solusi **belum ditemukan**, maka pencarian **dilanjutkan pada simpul sebelah kanan** dan simpul yang kiri dapat dihapus dari memori.
- Jika pada level yang **paling dalam tidak ditemukan solusi**, maka pencarian **dilanjutkan pada level sebelumnya**.

Depth-First Search (DFS)

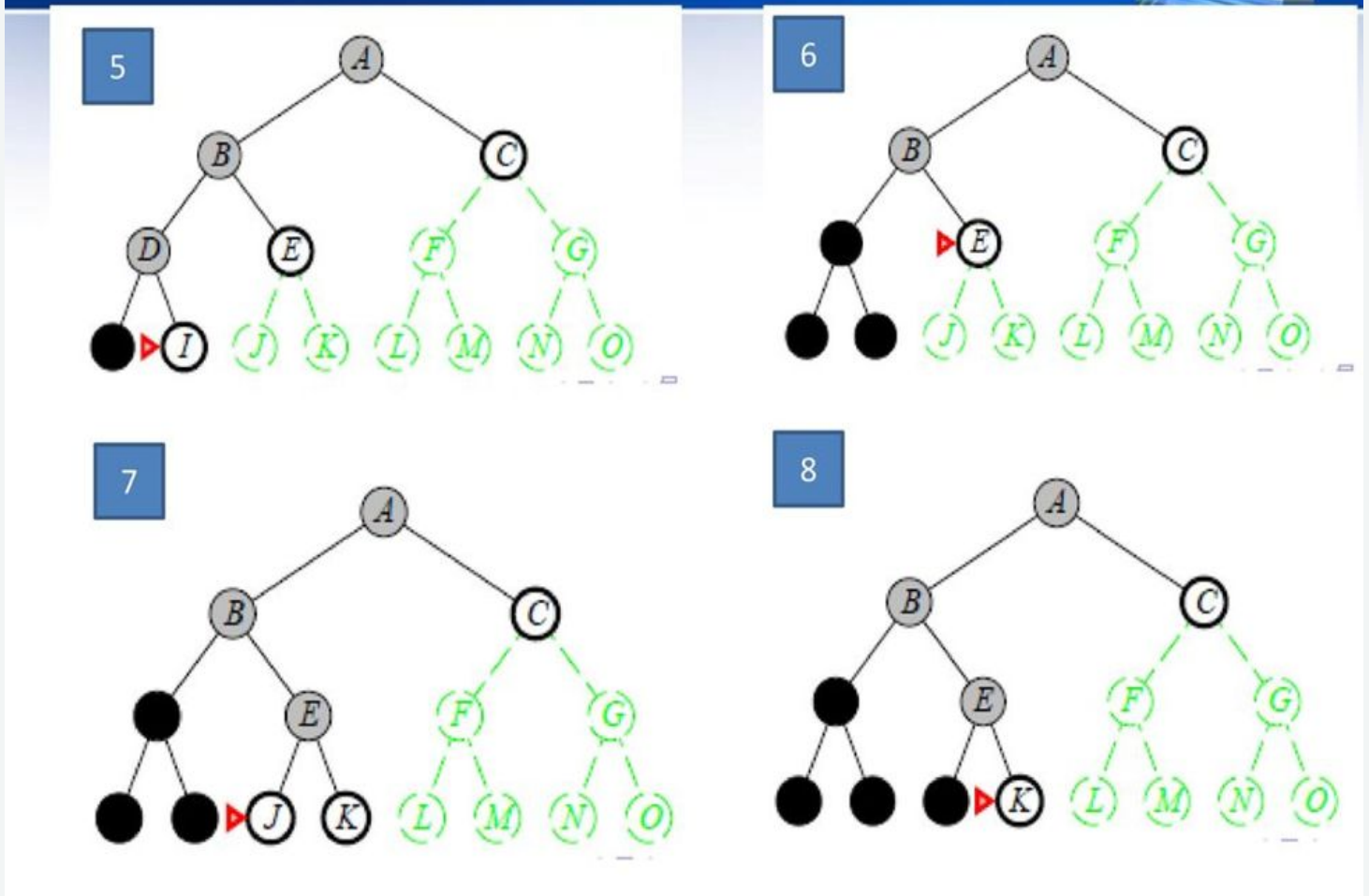
- Contoh :
- Root : A
- Goal : M



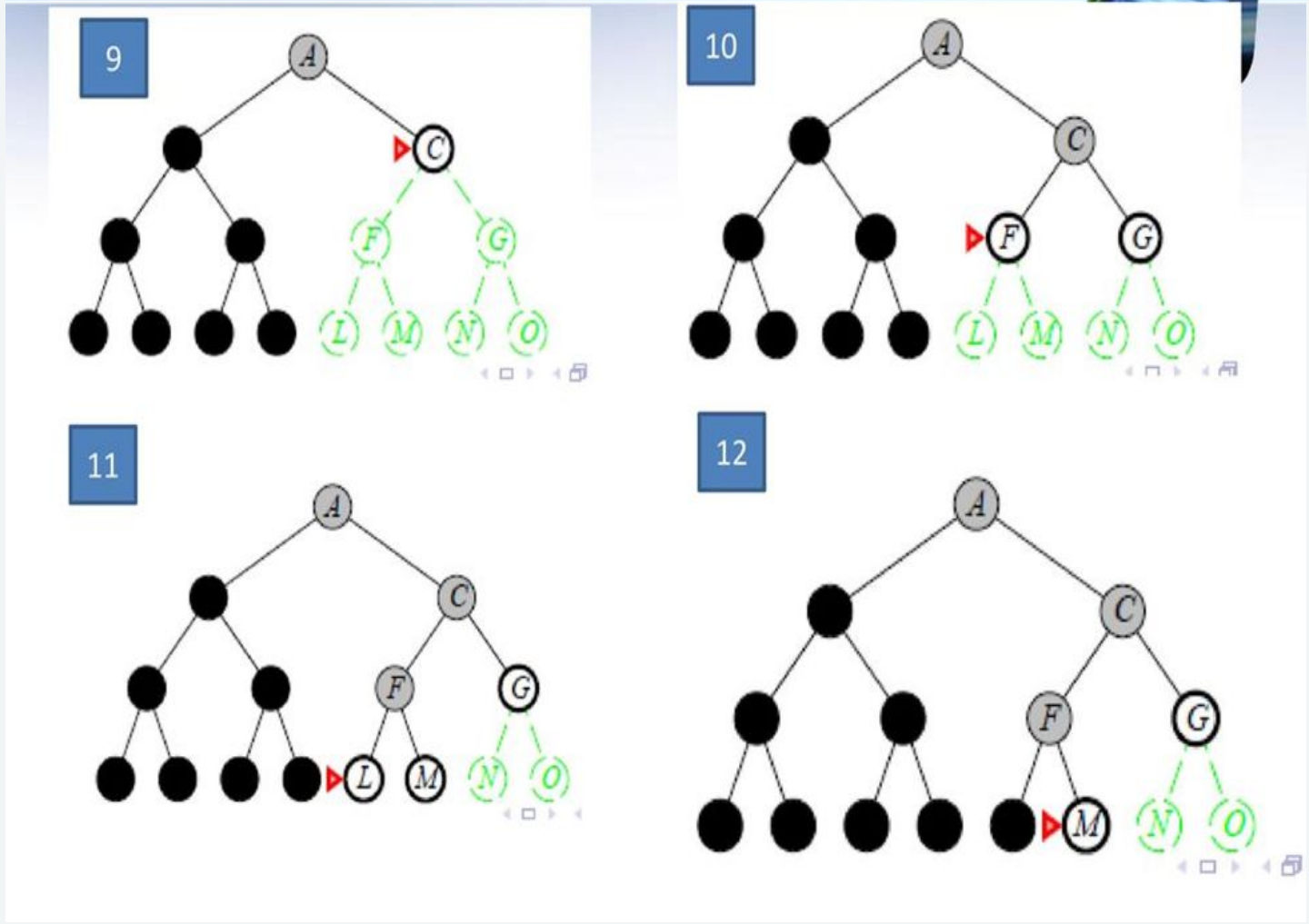
Tahap Depth-First Search (DFS)



Tahap Depth-First Search (DFS)



Tahap Depth-First Search (DFS)



Uniform Cost Search (UCS)

- Konsepnya hampir sama dengan BFS, bedanya adalah bahwa BFS menggunakan urutan level yang paling rendah sampai yang paling tinggi, sedangkan UCS menggunakan urutan biaya dari yang paling kecil sampai yang terbesar.
- UCS berusaha menemukan solusi dengan total biaya terendah yang dihitung berdasarkan biaya dari simpul asal menuju ke simpul tujuan.

Depth Limited Search (DLS)

- Metode ini berusaha mengatasi kelemahan DFS (tidak *complete* karena ketika proses pencarian menemui *infinite state space*) dengan membatasi kelemahan maksimum dari suatu jalur solusi yaitu dengan batas *depth* pada level tertentu semenjak awal pencarian.
- Tetapi, sebelum menggunakan DLS, kita harus tahu berapa level maksimum dari suatu solusi

Iterative Deeping Depth First Search (IDDFS)

- IDS merupakan metode yang menggabungkan kelebihan BFS (Complete dan Optimal) dengan kelebihan DFS (space complexity rendah atau membutuhkan sedikit memori)
- Tetapi konsekuensinya adalah time *complexity*nya menjadi tinggi untuk menemukan berapa *depth limit* terbaik untuk sampai pada *goal* yang dicari. Langkahnya menambah limit secara bertahap, mulai dari 0,1,2, dan seterusnya.

Bidirectional Search (BS)

- Pencarian dilakukan dari dua arah : pencarian maju (dari start ke goal) dan pencarian mundur (dari goal ke start).
- Ketika dua arah pencarian telah sampai pada simpul yang sama, maka solusi telah ditemukan, yaitu dengan cara menggabungkan kedua jalur yang bertemu.

PERFORMA METODE PENCARIAN

Completeness

- Apakah metode tersebut menjamin adanya solusi jika solusinya ada ?

Time Complexity

- Berapa lama waktu yang diberikan untuk menemukan solusi tersebut ?

Space Complexity

- Berapa banyak memori yang dibutuhkan untuk menemukan solusi tersebut?

Optimally

- Apakah metode tersebut menjamin menemukan solusi terbaik jika terdapat beberapa solusi yang berbeda ?

Perbandingan Algoritma Uninformed Search

Kriteria	BFS	DFS	UCS	DLS	IDDFS	BS
Waktu	b^d	b^m	b^d	b^l	b^d	b^(d/2)
Tempat	b^d	b*m	b^d	b*l	b*d	b^(d/2)
Optimal ?	Ya	Tidak	Ya	Ya	Ya	Ya
kelengkapan	ya	Tidak	ya	Ya (jika l ≥d)	Ya	Ya

Keterangan :
b : jumlah maksimal cabang tree
d : kedalaman pada least-cost solution
m : kedalaman maksimum pada state-space (bisa bernialai infinity)
l : nilai cut off pada kedalaman tree

SEKIAN DAN TERIMA KASIH