4. Strategi Pencarian Tanpa Informasi (Blind Search)

shinta Oktaviana R Skom Mkom
shinta.oktaviana@tik.pnj.ac.id

STRATEGI PENCARIAN UNINFORMED

- Sebuah strategi pencarian didefinisikan dengan memilih urutan ekspansi node.
- Strategi dievaluasi sepanjang dimensi berikut:
  - kelengkapan: apakah selalu mencari solusi jika ada?
    completeness
  - kompleksitas waktu: jumlah node yang dihasilkan
    time complexity
  - kompleksitas ruang: jumlah maksimum node dalam memori
    - *space complexity*
  - optimalitas: apa selalu menemukan solusi yang paling murah?
    optimality

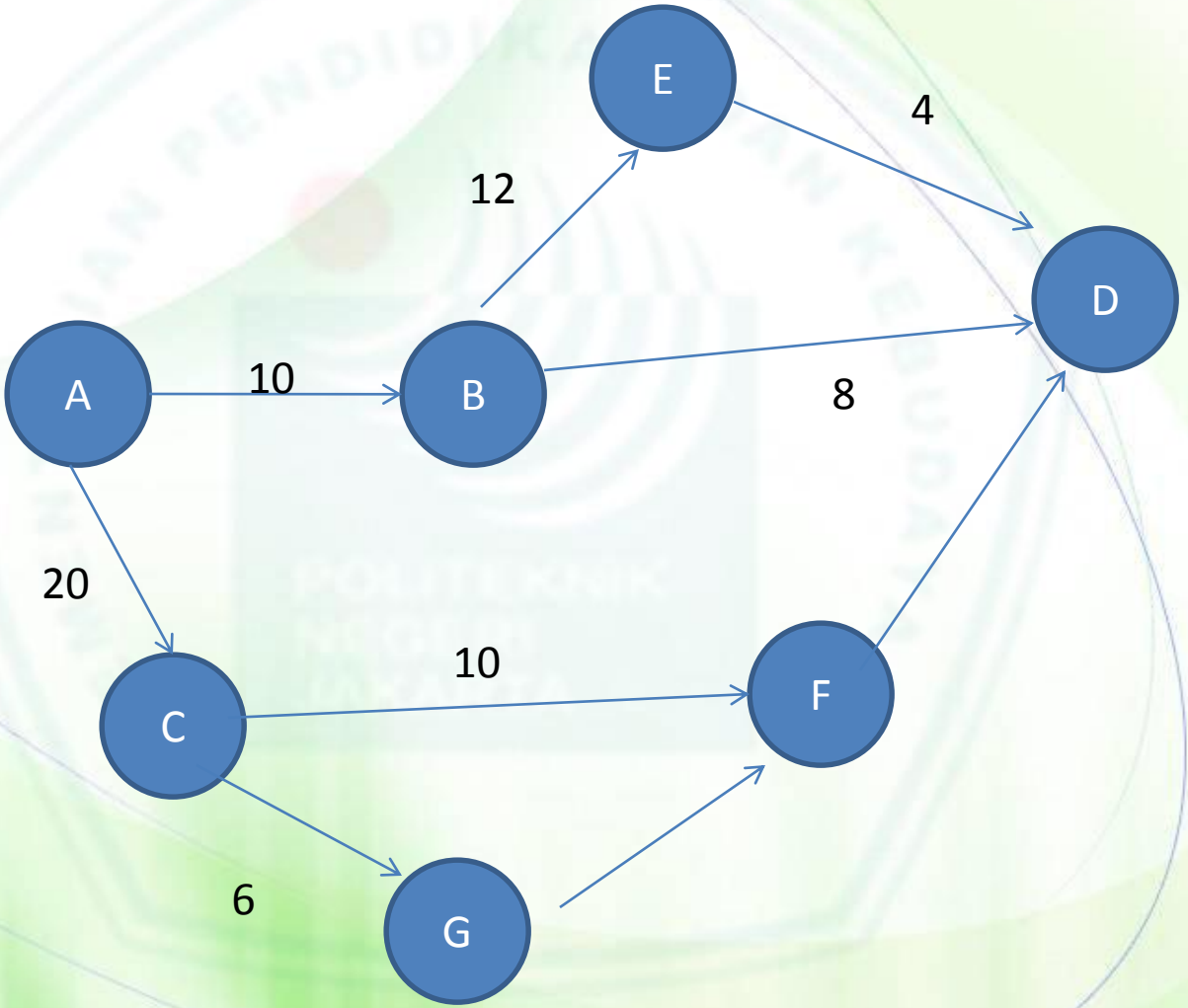
kompleksitas waktu dan ruang diukur dalam hal:

- *b*: faktor percabangan maksimum search tree
- *d*: kedalaman solusi yang paling murah
- *m*: panjang maksimum setiap path (mungkin ∞)

Strategi pencarian uninformed

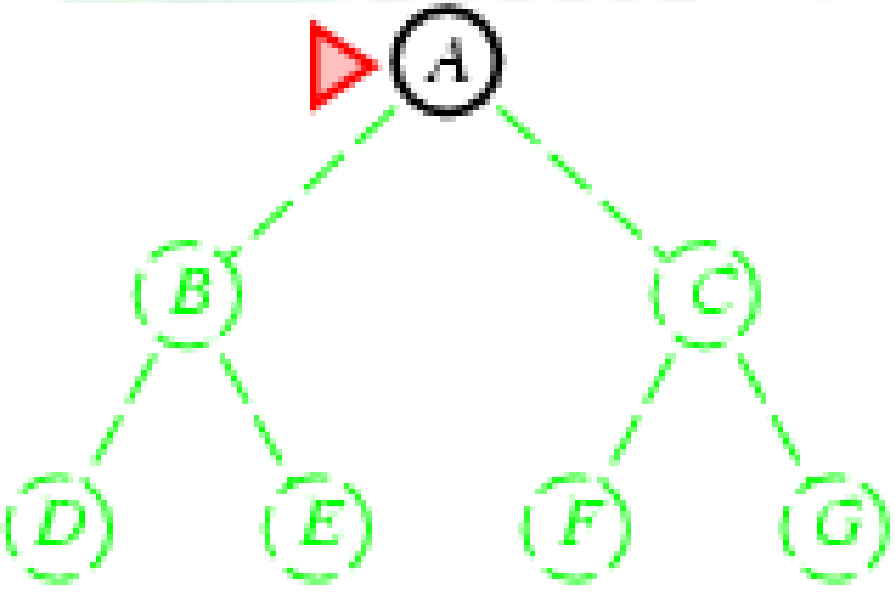
1. Breadth-first search
2. Uniform-cost search
3. Depth-first search
4. Depth-limited search
5. Iterative Deepening search
6. Bidirectional search

Contoh problem dengan representasi graph

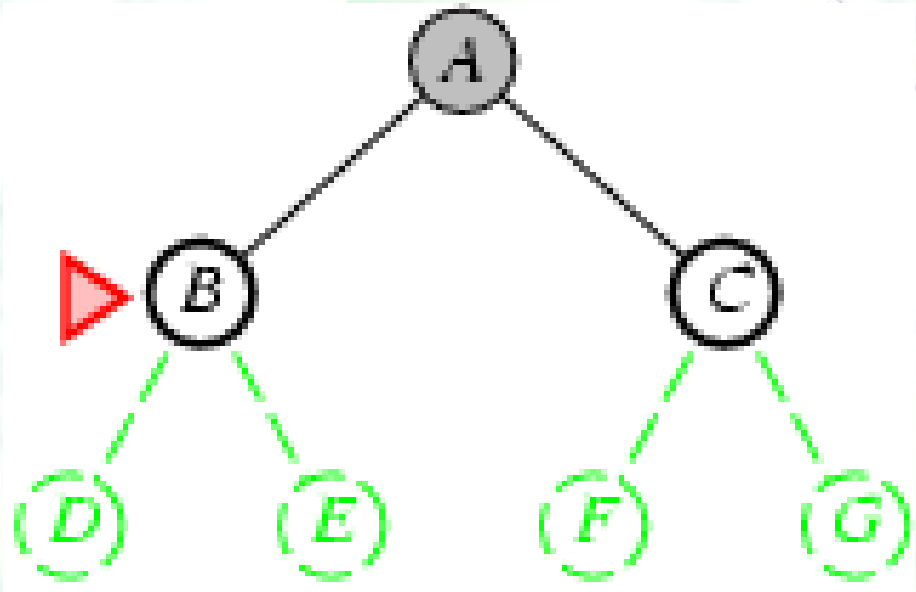


Breadth-first search

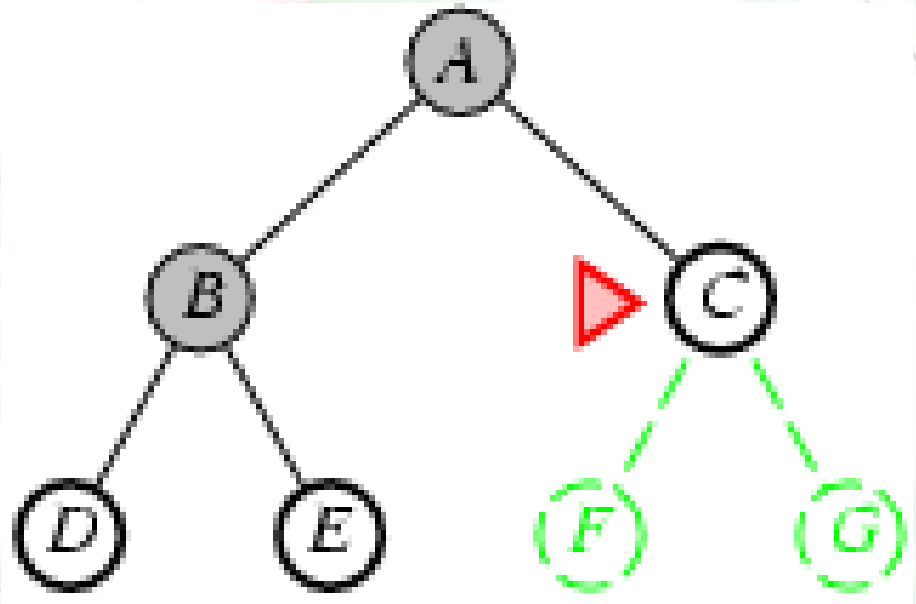
- Memperluas node terdangkal yang belum diekspansi
- start: A goal: D



Breadth-first search

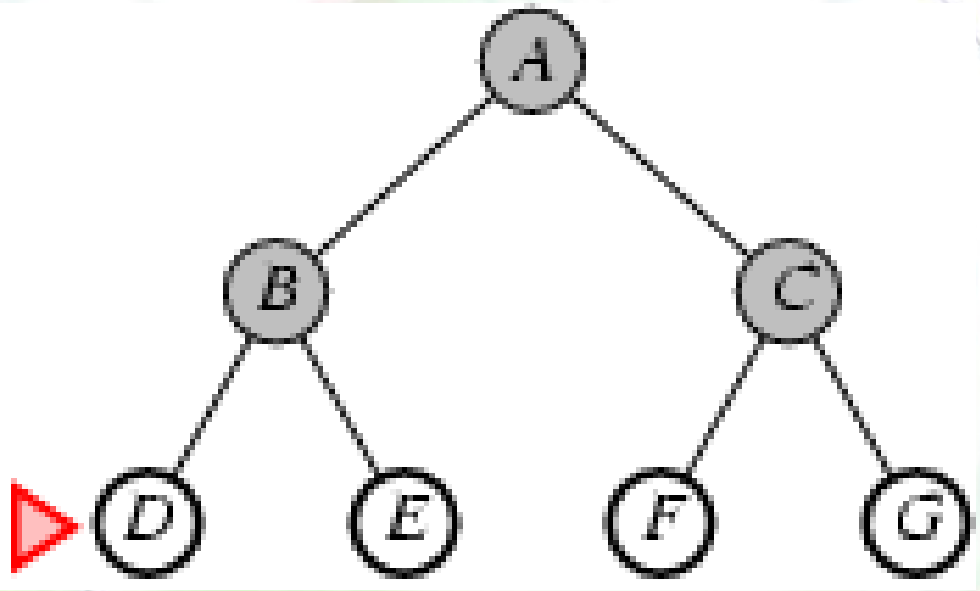


Breadth-first search



Breadth-first search

d = 2, m = 4



Kompleksitas BFS

Depth	Nodes	Time	Memory
0	1	1 millisecond	100 bytes
2	111	.1 seconds	11 kilobytes
4	11,111	11 seconds	1 megabyte
6	10^6	18 minutes	111 megabytes
8	10^8	31 hours	11 gigabytes
10	10^{10}	128 days	1 terabyte
12	10^{12}	35 years	111 terabytes
14	10^{14}	3500 years	11,111 terabytes

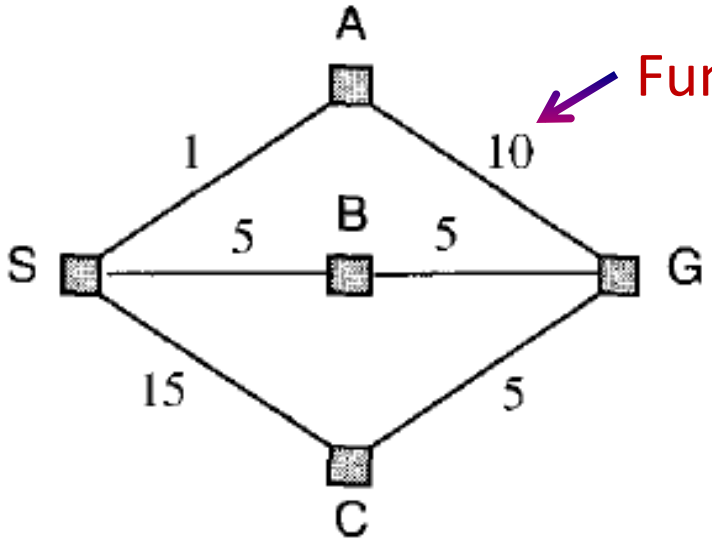
Figure 3.12 Time and memory requirements for breadth-first search. The figures shown assume branching factor $b = 10$; 1000 nodes/second; 100 bytes/node.

Uniform Cost Search (UCS)

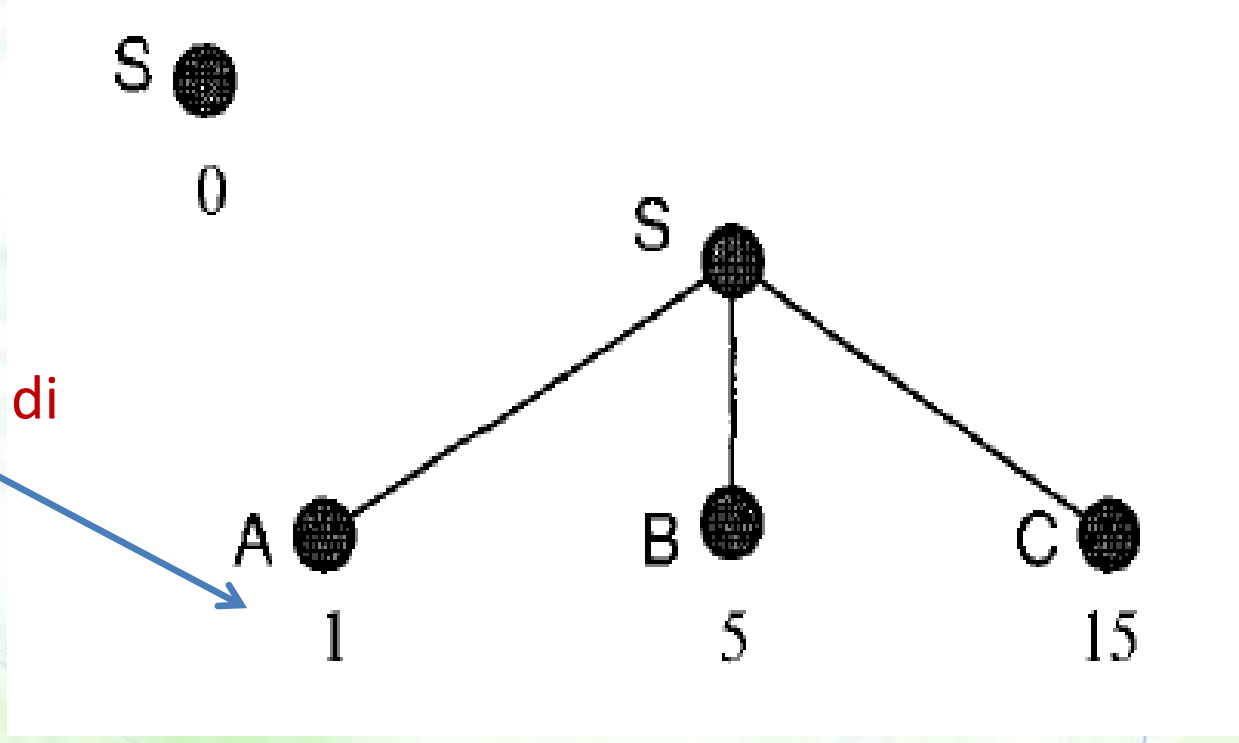
- Sama seperti BFS dengan tambahan pembentukan tree diurutkan berdasarkan cost yang paling murah/least-cost
- Urutan ekspansi seperti BFS
- Implementasi: tree/queue diurutkan berdasarkan least-cost
- Menggunakan fungsi g(n) sebagai fungsi yang menentukan cost terendah dari solusi yang dihasilkan.

Contoh UCS

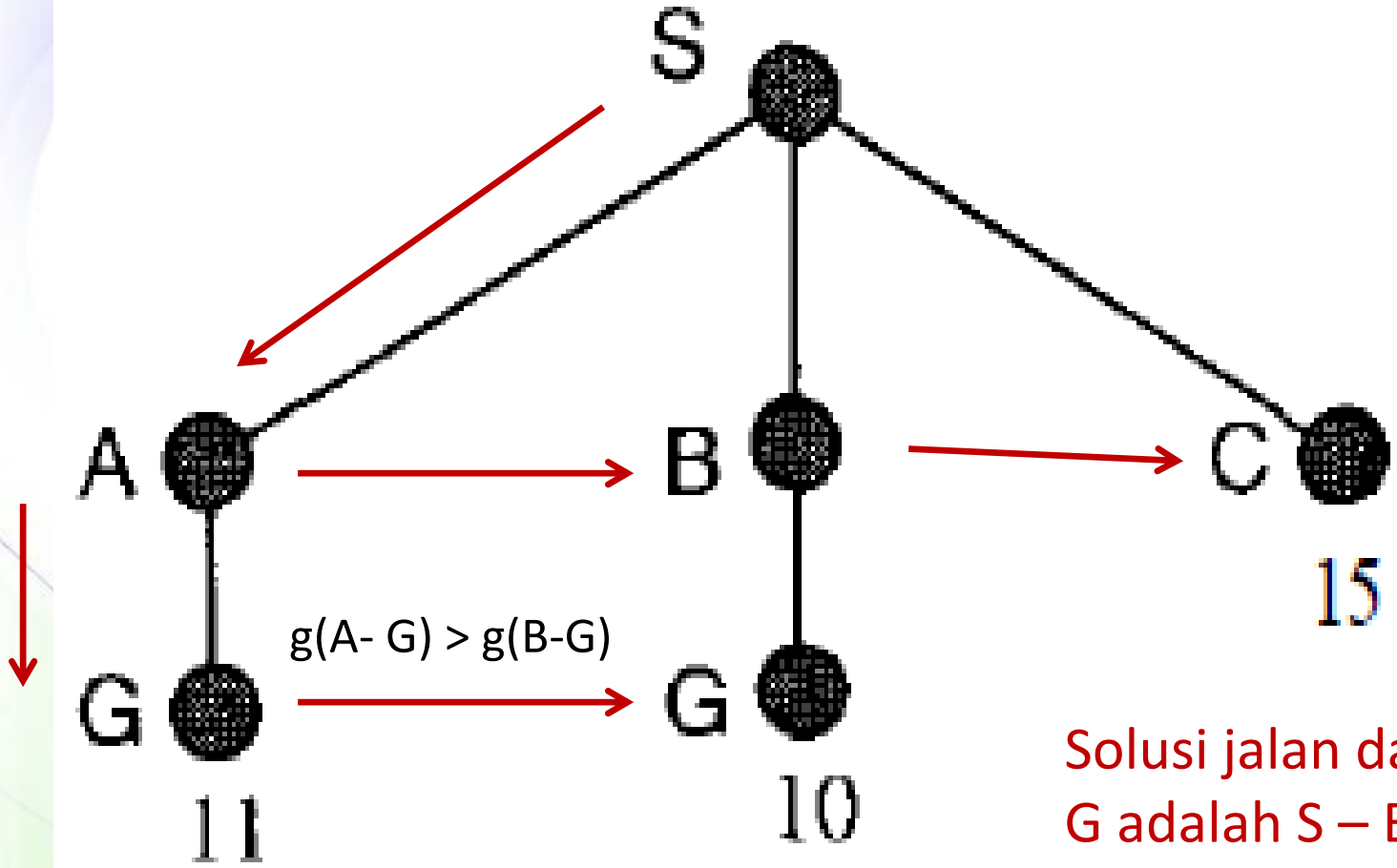
Mencari rute dari S ke G



- g(a) < g(b) < g(c)



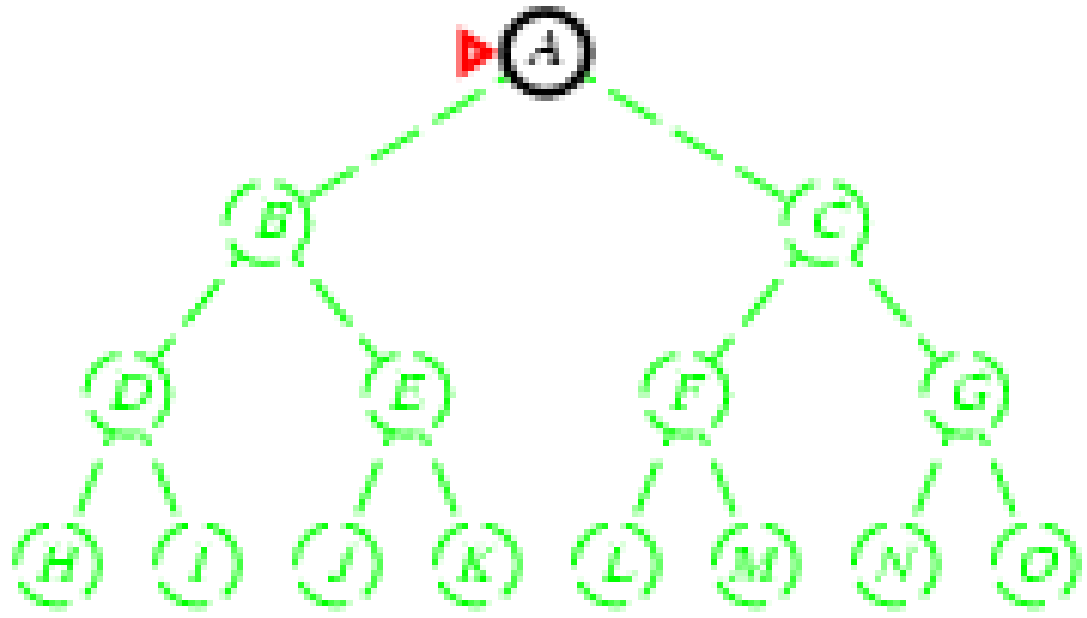
Nilai node terkecil berada di sebelah kiri



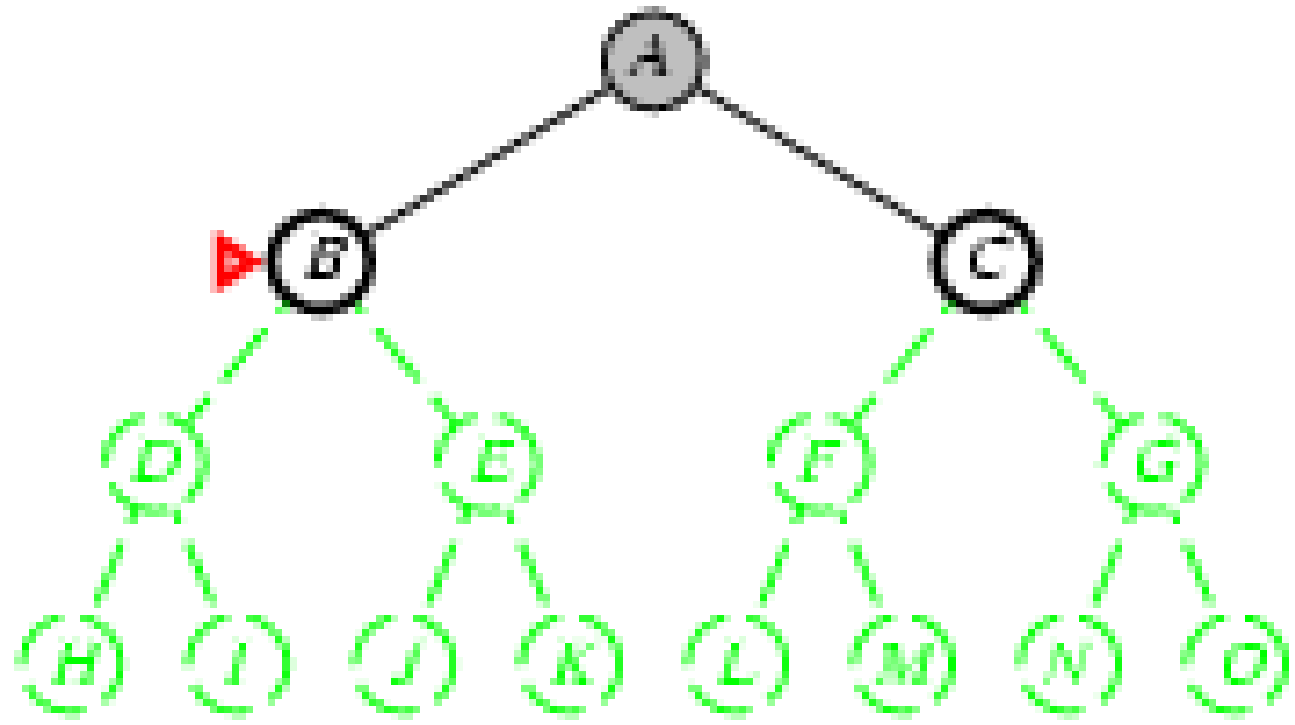
Solusi jalan dari S ke G adalah S – B - G

Depth-first search

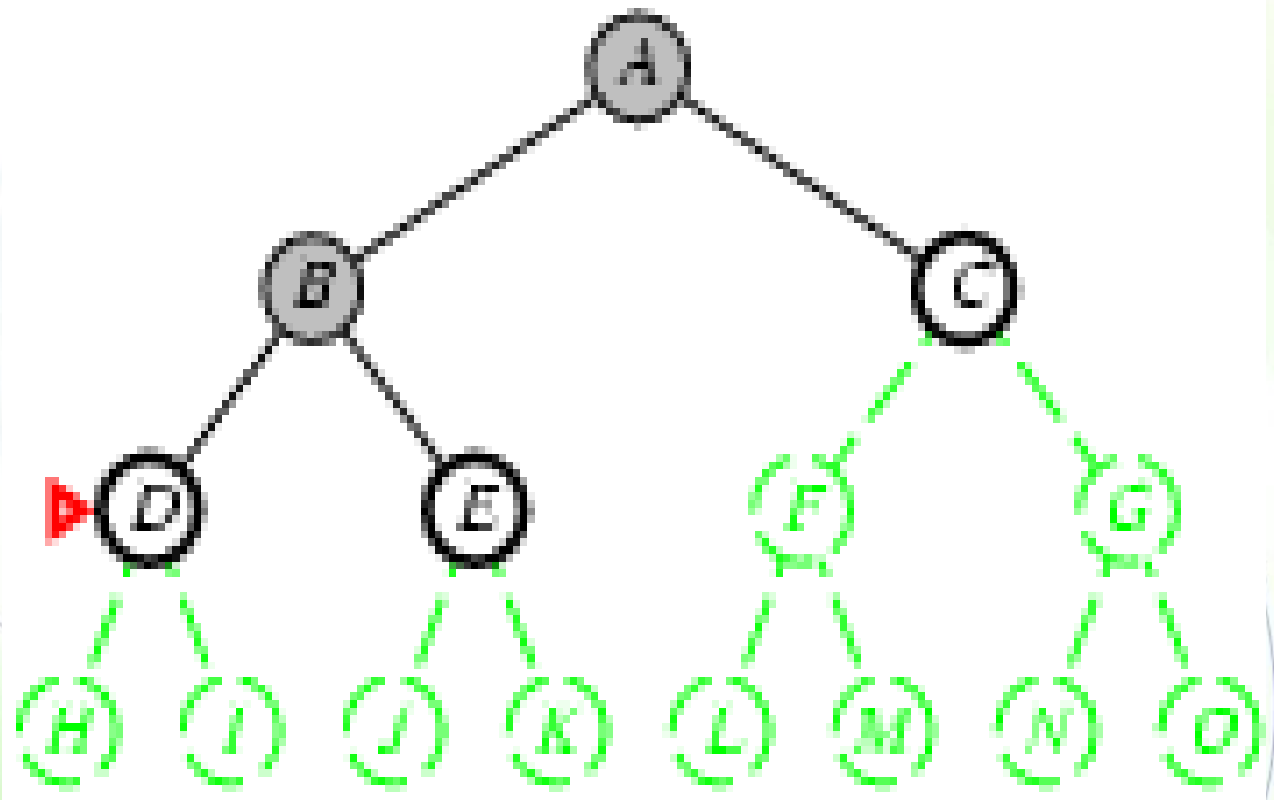
- Pencarian unexpanded node terdalam
- Start: A; goal: M
- Kompleksitas memory lebih baik dari pada BFS karena tidak semua node disimpan di memory.



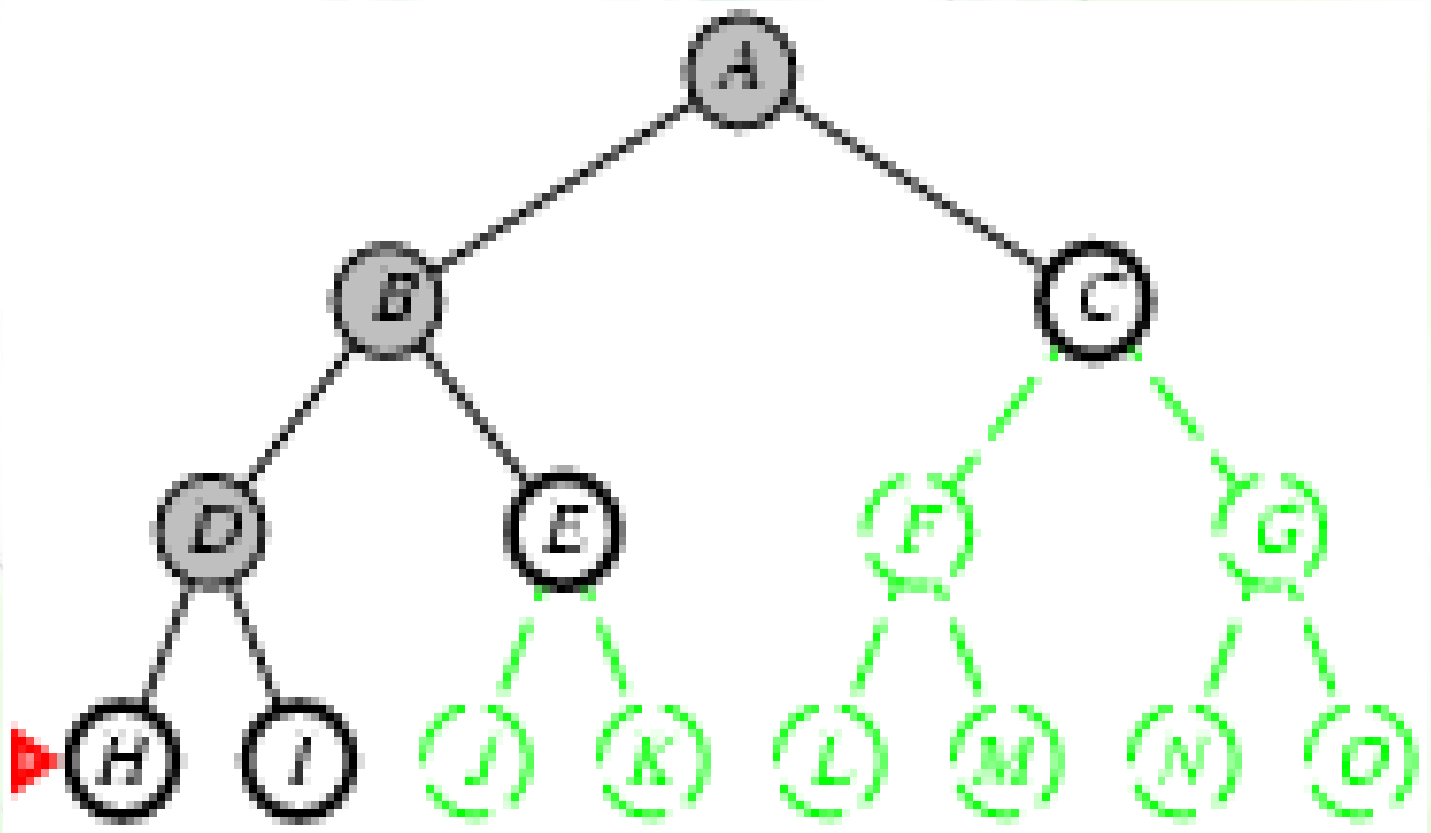
Depth-first search



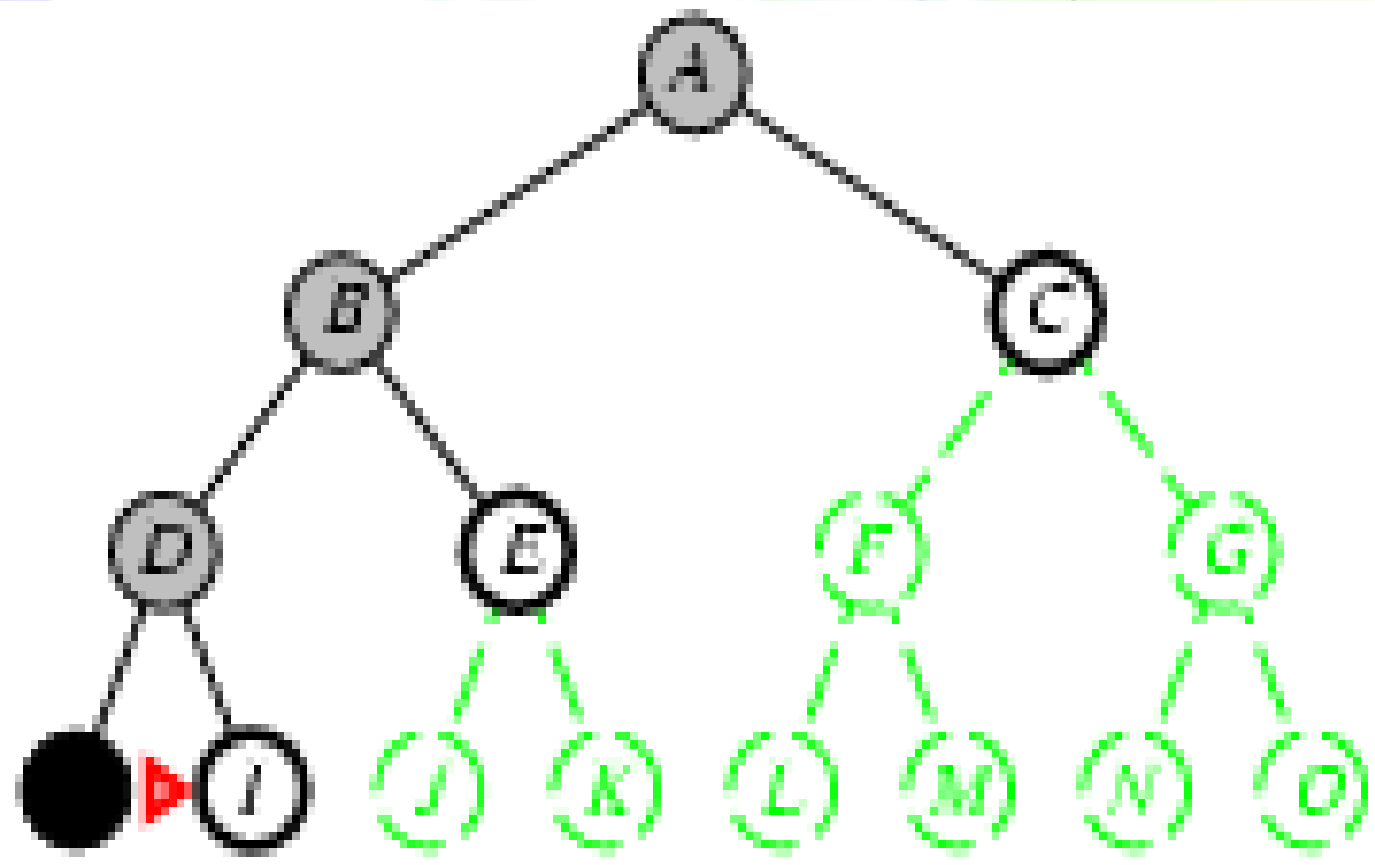
Depth-first search



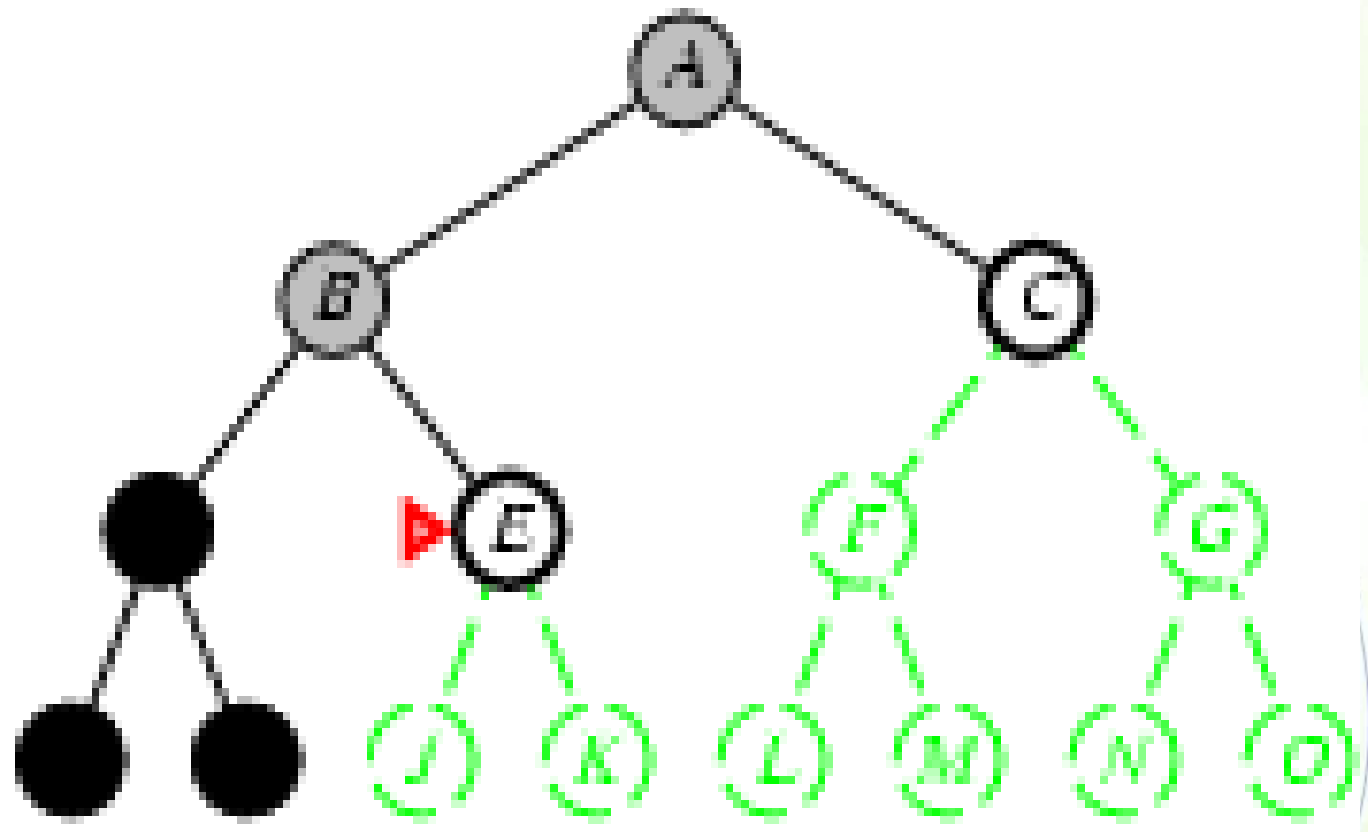
Depth-first search



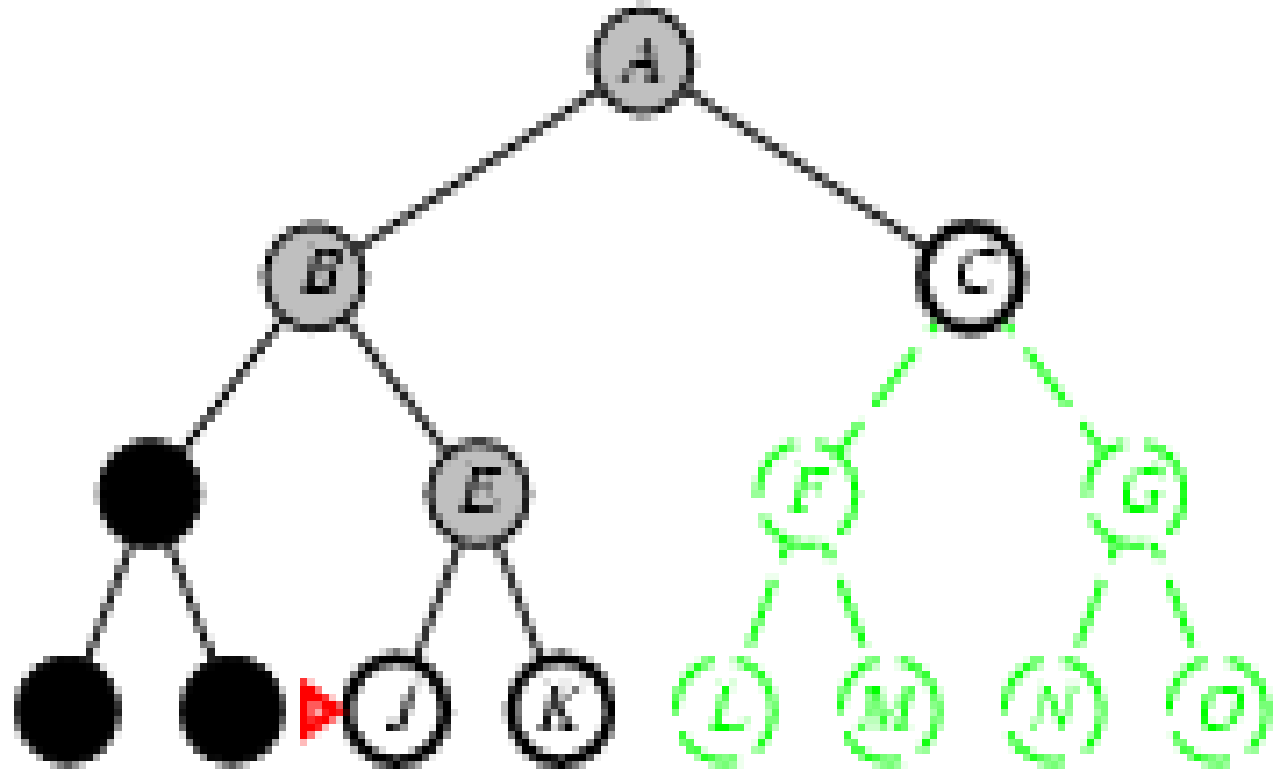
Depth-first search



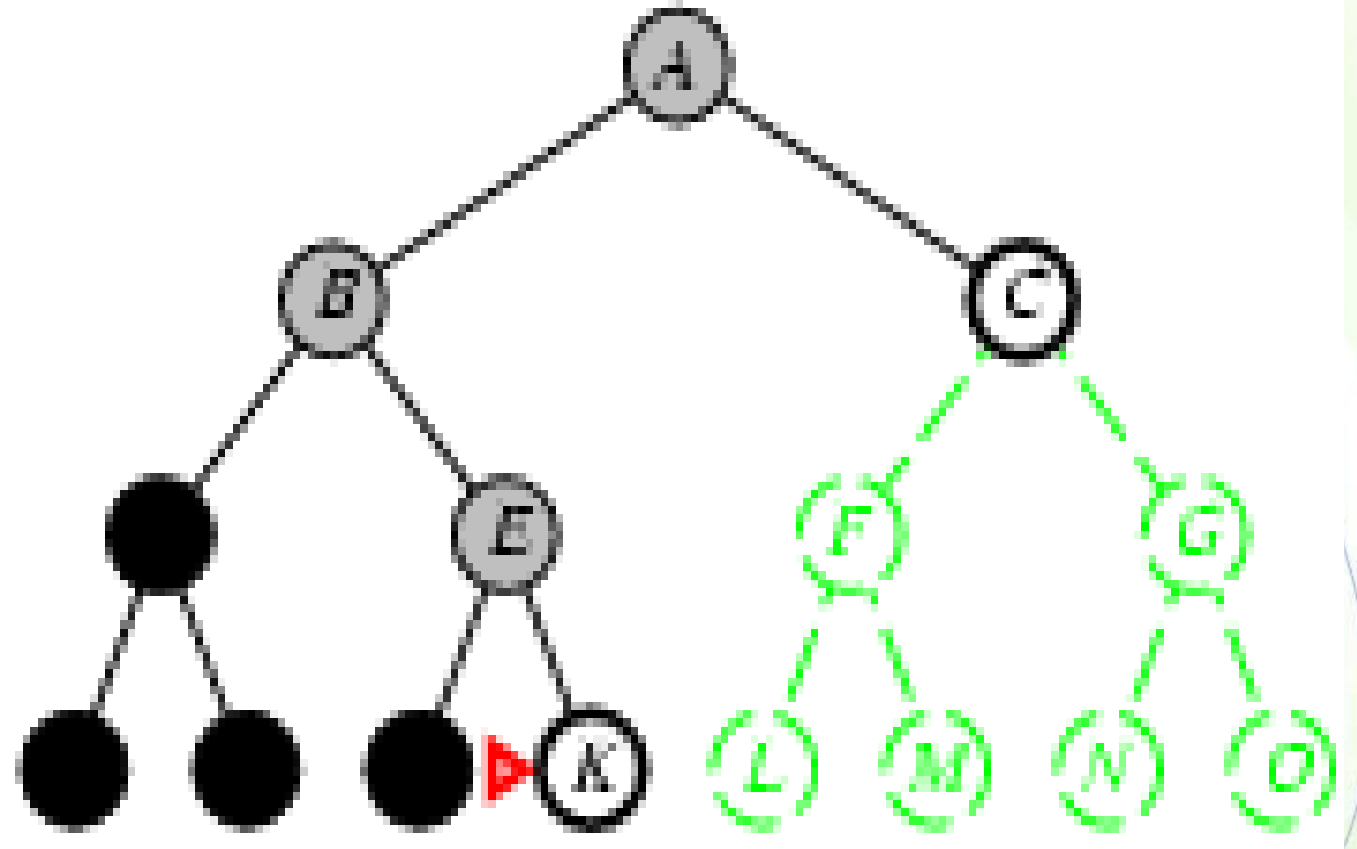
Depth-first search



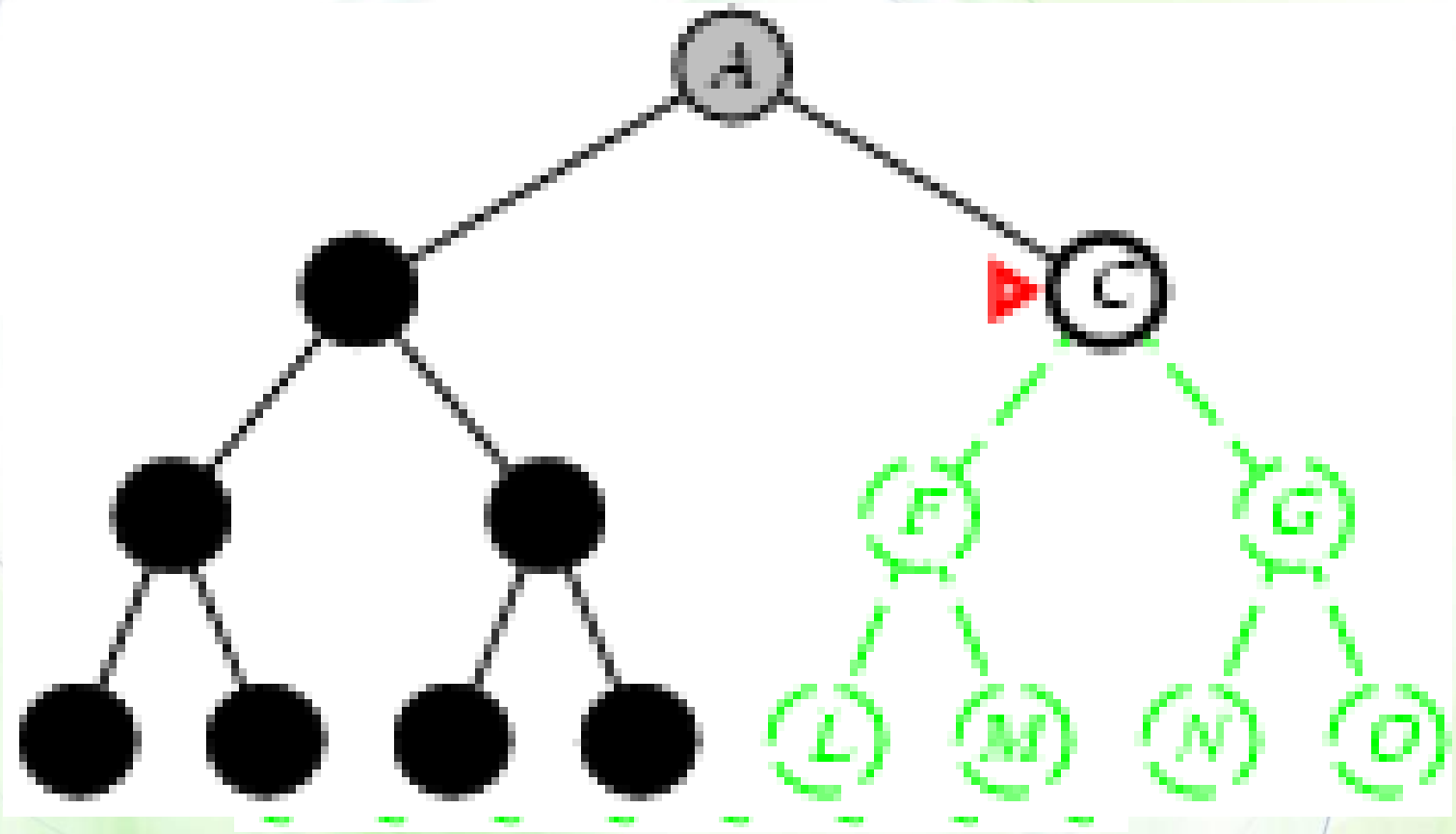
Depth-first search



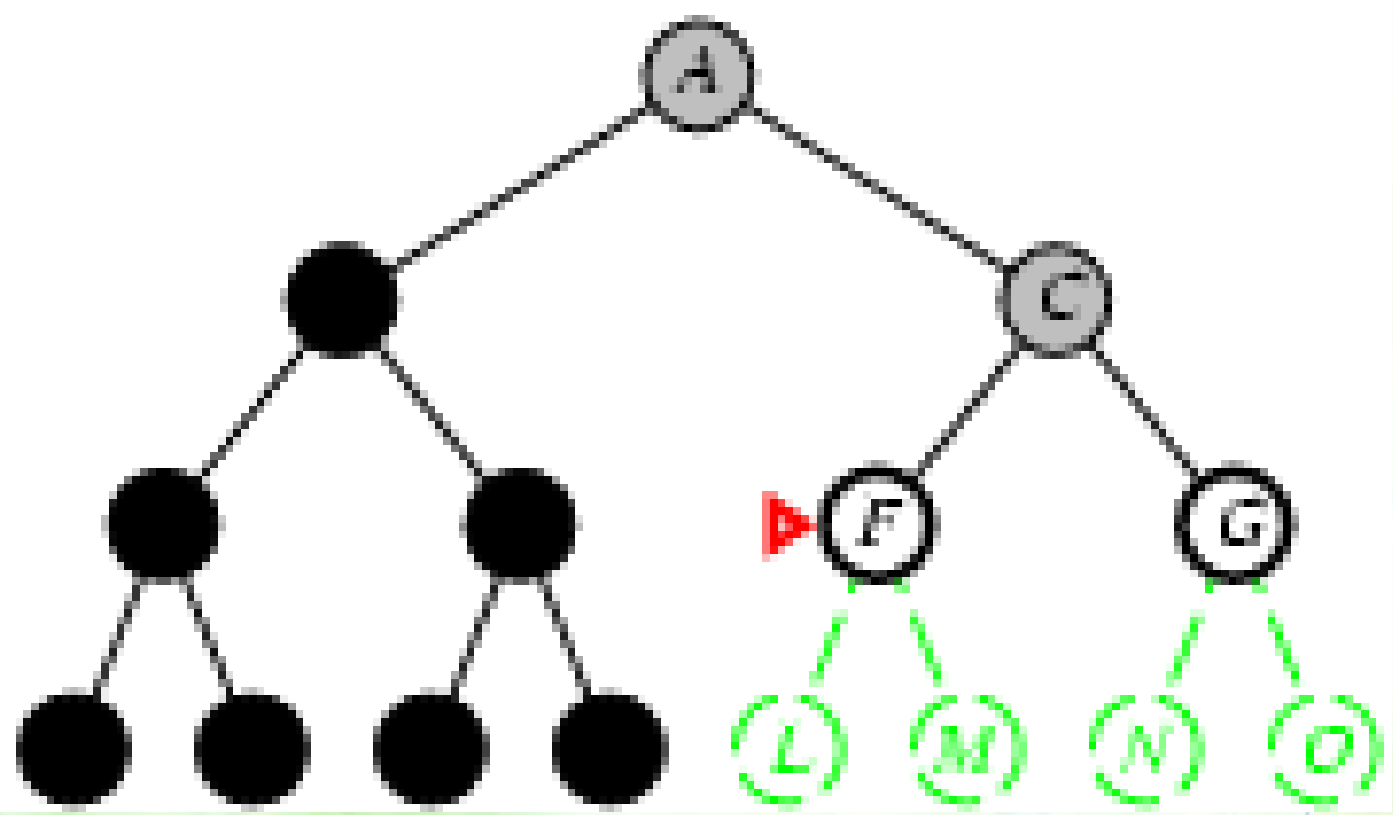
Depth-first search



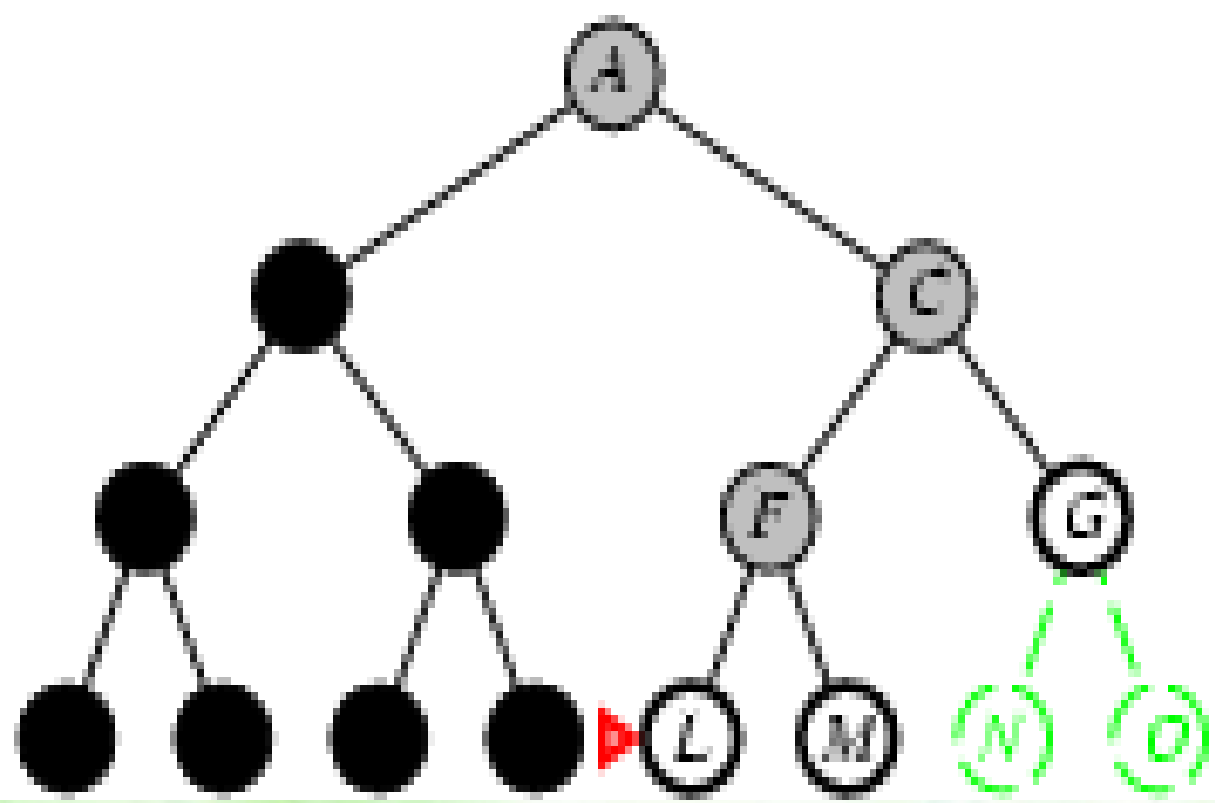
Depth-first search



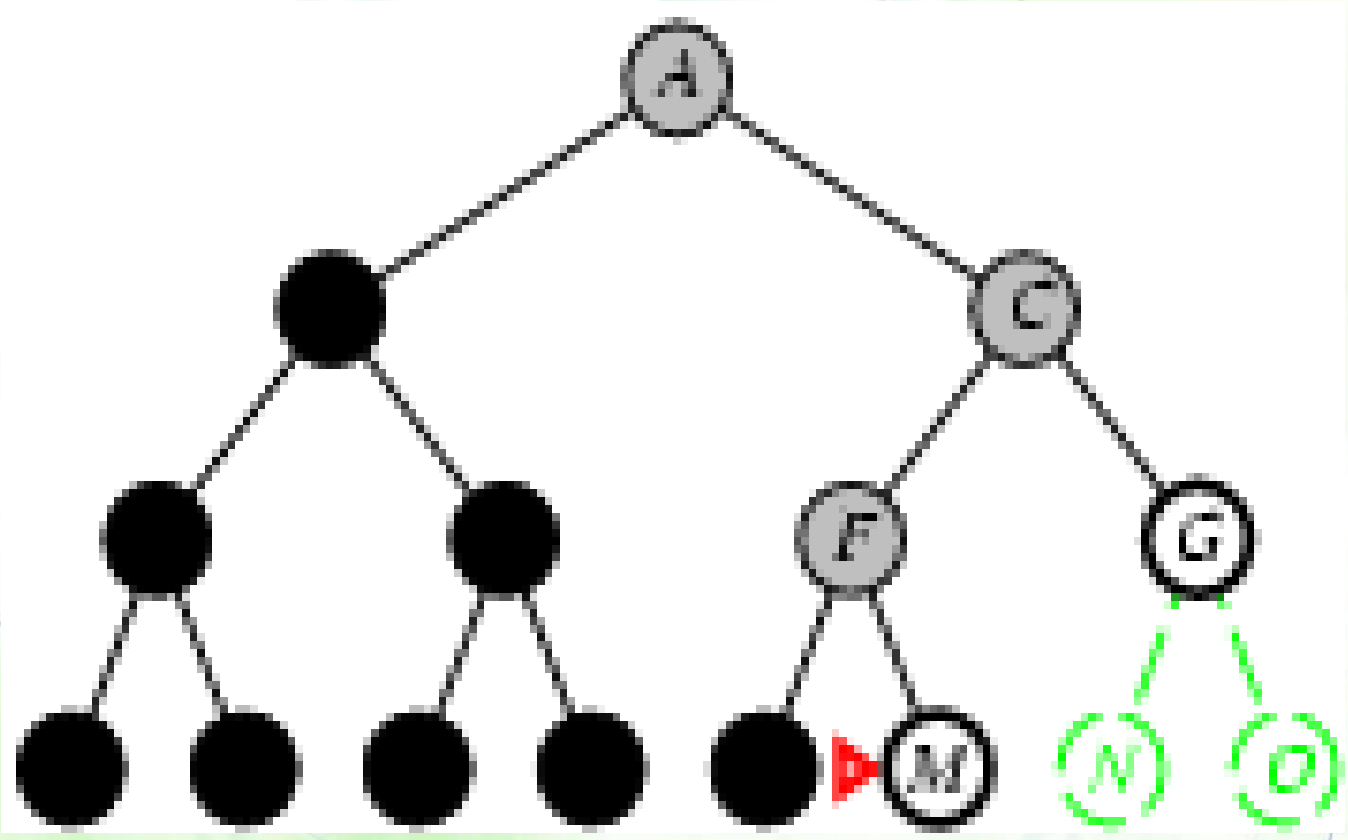
Depth-first search



Depth-first search



Depth-first search



Depth Limited Search (DLS)

- Merupakan strategi DFS dengan batas kedalaman tree *l yang didefinisikan sebelumnya.*
- Bertujuan untuk meminimalisir worse case yang terjadi pada DFS.
- Completeness tergantung pada batas kedalaman solusi yang ditentukan. Semakin kecil nilai yang ditentukan, maka kemungkinan pencarian solusi tidak komplit.

Iterative Deepening Search (IDS)

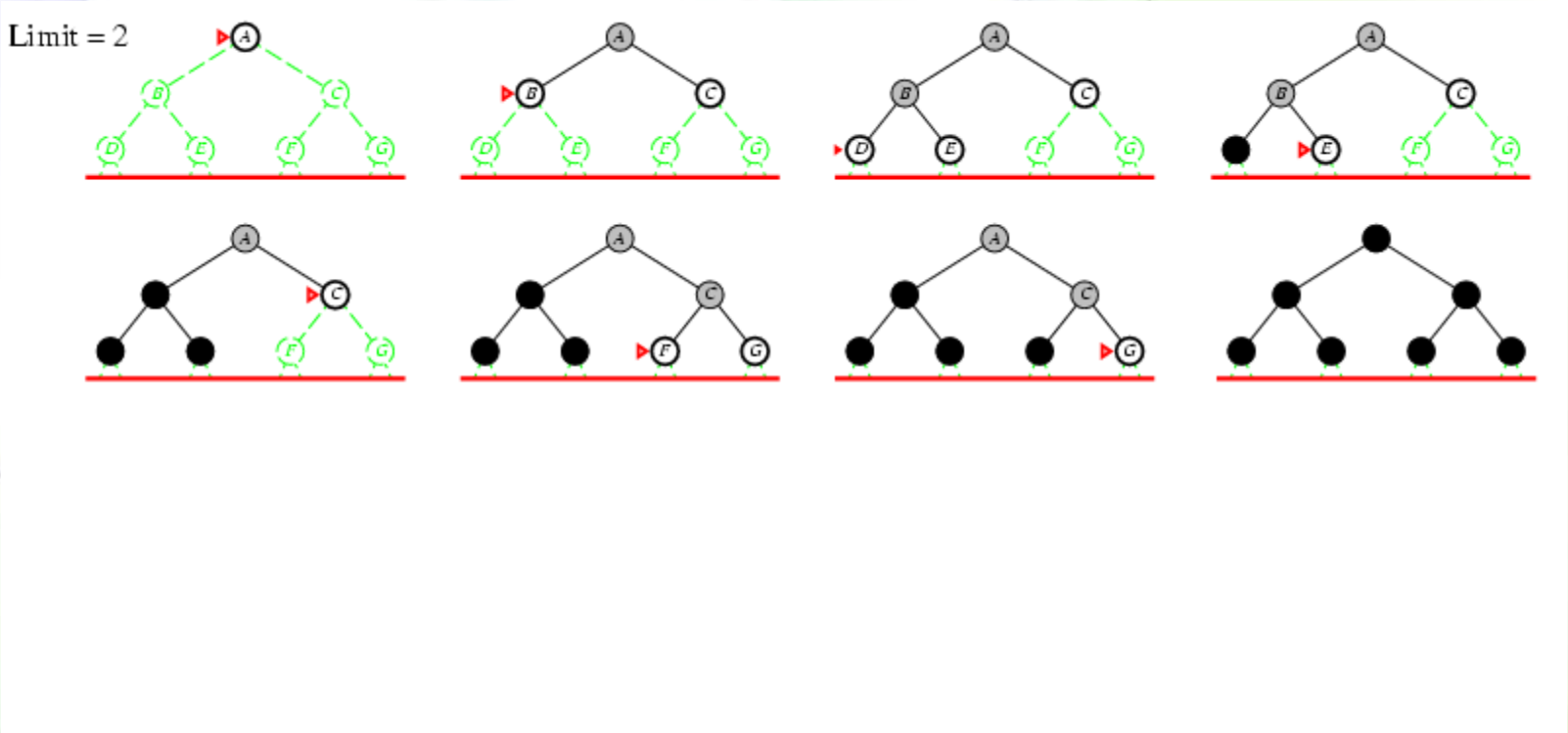
- Kerena nilai fungsi limit terbaik pada DLS baru diketahui setelah kita menemukan solusi terbaik.
- Prinsip dari strategi ini adalah melakukan pencarian DLS secara bertahap dengan nilai l yang ditambahkan pada setiap iterasinya.
- Strategi ini mengkombinasikan keuntungan BFS dan DFS (kelengkapan dan kompleksitas ruang linear dijamin). Lakukan pencarian DLS dengan l = 0,1,2, ... sampai tidak cutoff

Iterative deepening search / =0

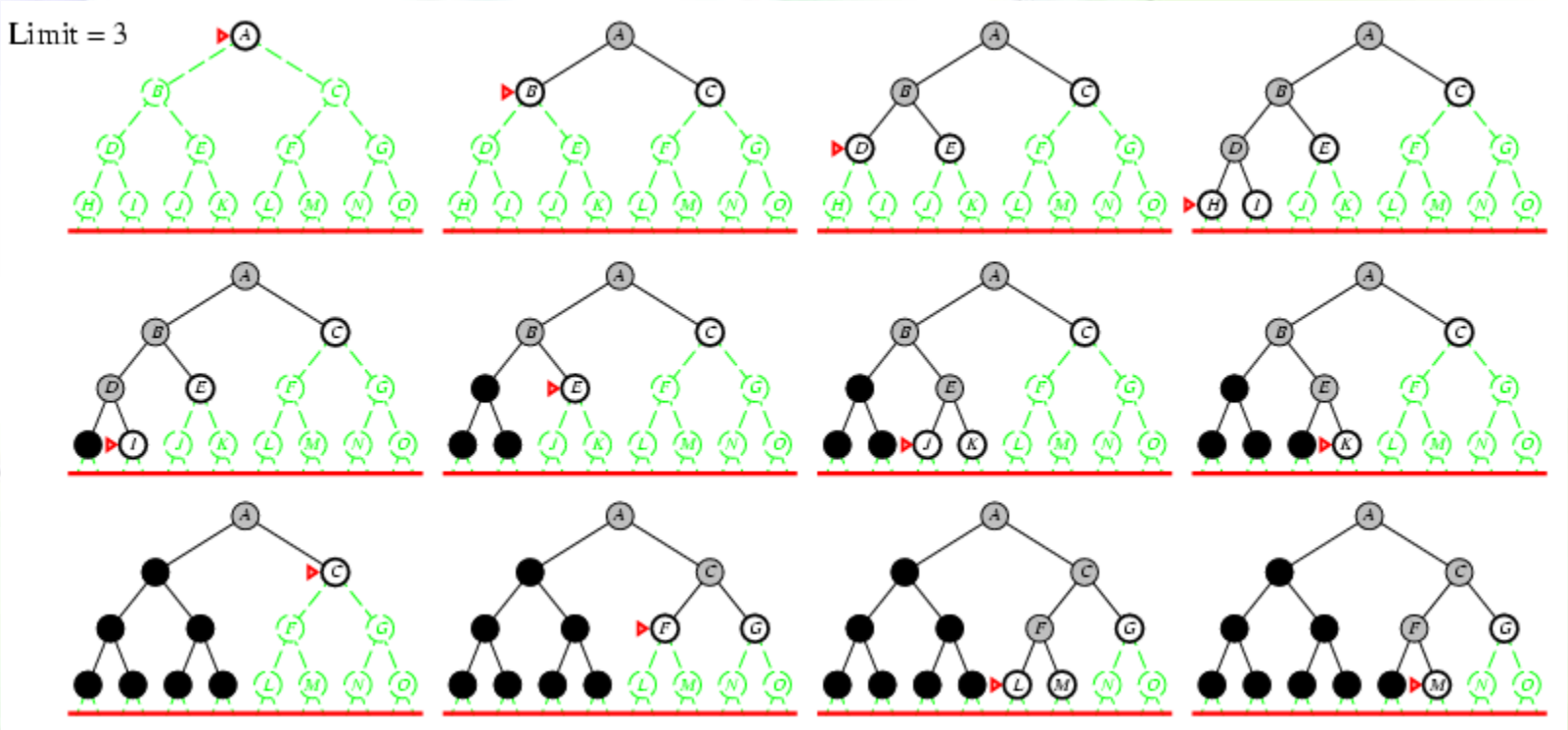
Limit = 0

Iterative deepening search / =1

Iterative deepening search / =2



Iterative deepening search / =3



Bidirectional Search

- Pencarian dilakukan dari arah node asal dan node tujuan
- Kompleksitas ruang dan waktu akan lebih kecil
- Kelemahannya adalah mencari nilai fungsi dari node pendehulu.
- Sulit diimplementasikan jika memiliki lebih dari 1 node solusi

Ringkasan Uninformed Search

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Time	b^d	b^d	b^m	b^l	b^d	$b^{d/2}$
Space	b^d	b^d	bm	bl	bd	$b^{d/2}$
Optimal?	Yes	Yes	No	No	Yes	Yes
Complete?	Yes	Yes	No	Yes, if $l > d$	Yes	Yes

Figure 3.18 Evaluation of search strategies. b is the branching factor; d is the depth of solution; m is the maximum depth of the search tree; l is the depth limit.

Rangkuman

- Perumusan masalah biasanya membutuhkan abstrak rincian dunia nyata untuk menentukan ruang state yang bisa dieksplorasi.
- Agen penyelesaian problem dapat memecahkan masalah dan mencapai tujuan melalui strategi pohon pencarian (searching tree).
- Berbagai strategi pencarian uninformed: BFS, UCS, DFS, DLS, IDS
- IDS hanya menggunakan ruang linear dan tidak banyak waktu lebih dari algoritma uninformed lainnya.

Tugas Kelompok

Bandingkan 2 buah algoritma dalam mencari solusi pada problem missionaries and canibal

Hint : e-book hal 88 no 3.4