Breadth/Depth First Search (BFS/DFS) (Bagian 1)

Bahan Kuliah IF2211 Strategi Algoritmik

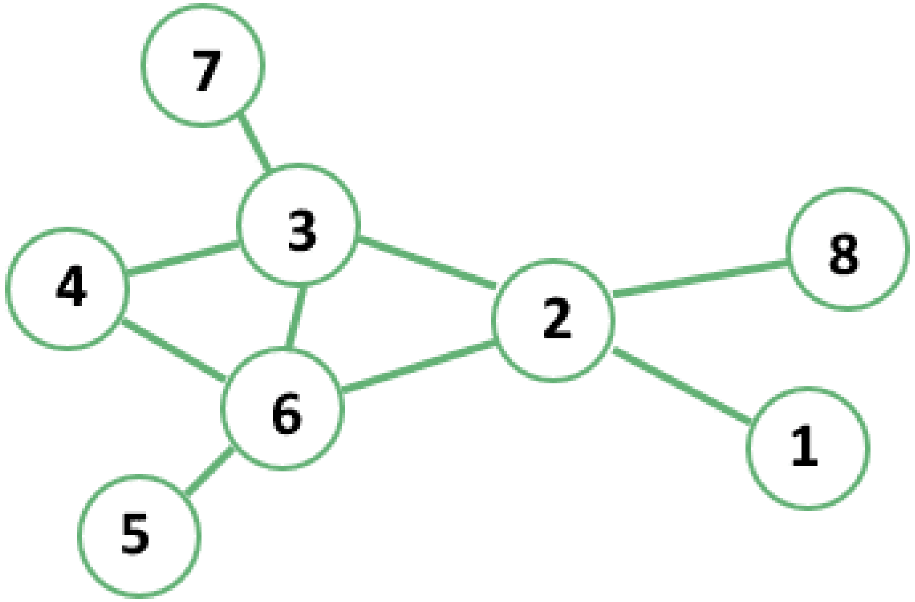
Oleh: Rinaldi Munir & Nur Ulfa Maulidevi

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika ITB
2021

Traversal Graf

- Algoritma traversal graf: mengunjungi simpul-simpul di dalam graf dengan cara yang sistematik
  - Pencarian melebar (*breadth first search*/BFS)
  - Pencarian mendalam (*depth first search*/DFS)

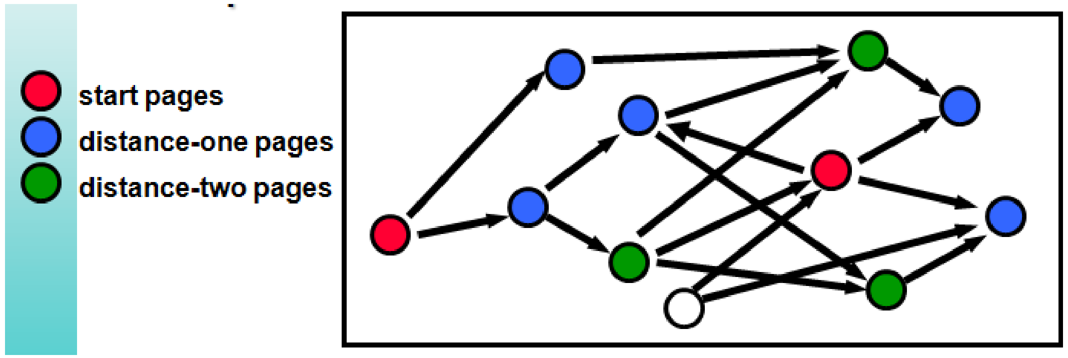
- Asumsi: graf terhubung



- Graf merupakan representasi persoalan

- Traversal graf artinya melakukan pencarian solusi persoalan yang direpresentrasikan dengan graf

social graph



Web page network

http://www.oreilly.de/catalog/9780596518172/toc.html

Algoritma Pencarian Solusi Berbasis Graf

- **Tanpa informasi (*uninformed/blind search*)**
  - Tidak ada informasi tambahan yang disediakan
  - Contoh: **DFS, BFS**, *Depth Limited Search*, *Iterative Deepening Search*, *Uniform Cost Search*

- **Dengan informasi (*informed Search*)**
  - Pencarian berbasis heuristik
  - Mengetahui *non-goal state* yang "lebih menjanjikan" daripada yang lain
  - Contoh: *Best First Search*, A*

Representasi Graf dalam Proses Pencarian

Dalam proses pencarian solusi, terdapat dua pendekatan:

1. Graf statis: graf yang sudah terbentuk sebelum proses pencarian dilakukan

   - graf direpresentasikan sebagai struktur data

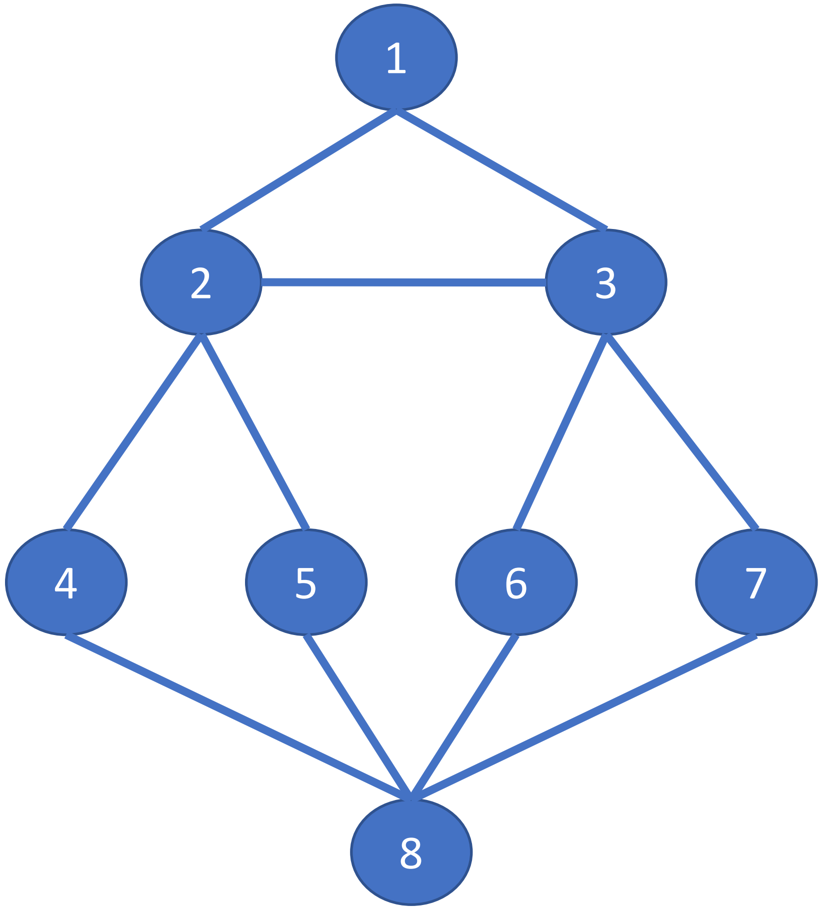
2. Graf dinamis: graf yang terbentuk saat proses pencarian dilakukan

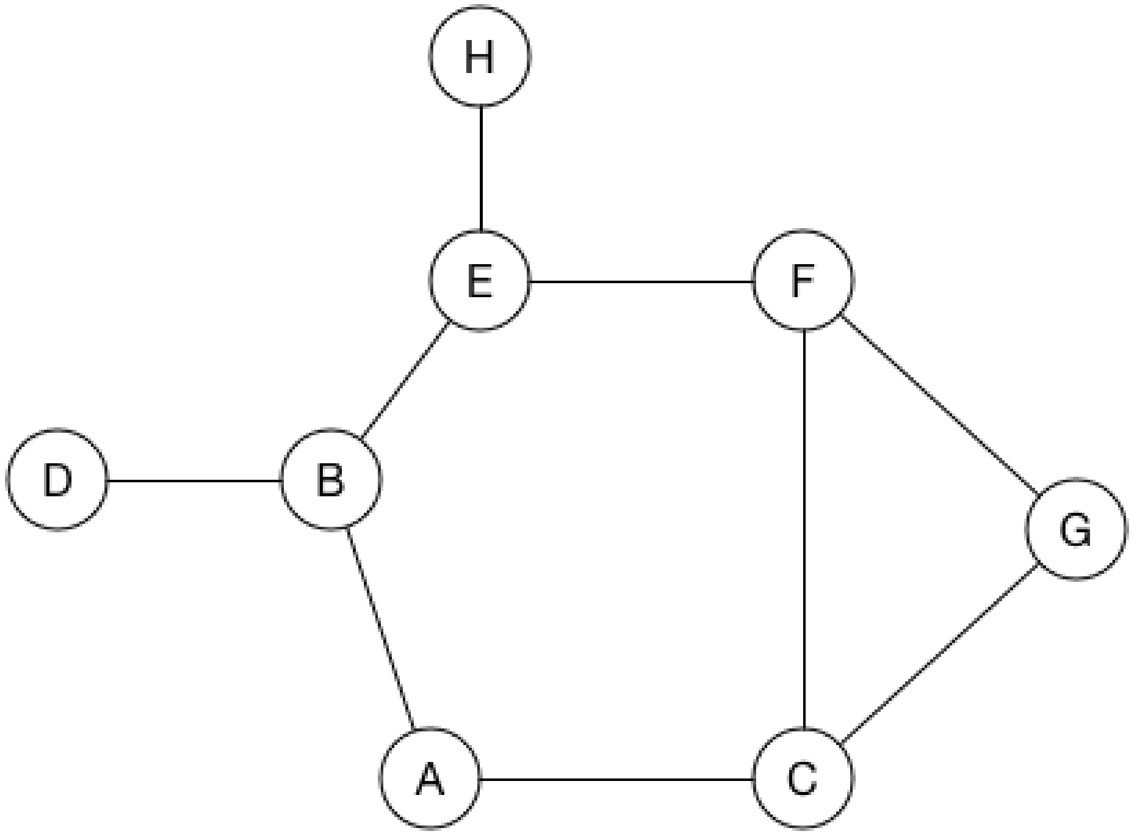
   - graf tidak tersedia sebelum pencarian, graf dibangun selama pencarian solusi

Graf Statis

Pencarian Melebar (BFS)

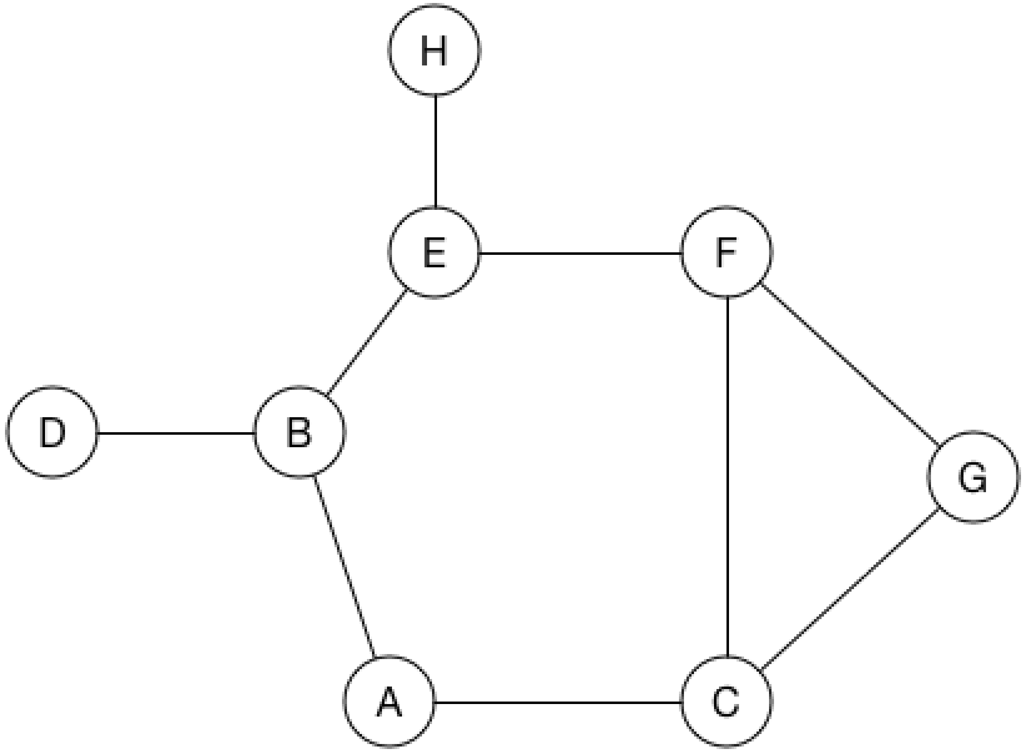
- Traversal dimulai dari simpul *v*.
- Algoritma:
  1. Kunjungi simpul *v*
  2. Kunjungi semua simpul yang bertetangga dengan simpul *v* terlebih dahulu.
  3. Kunjungi simpul yang belum dikunjungi dan bertetangga dengan simpul-simpul yang tadi dikunjungi, demikian seterusnya.

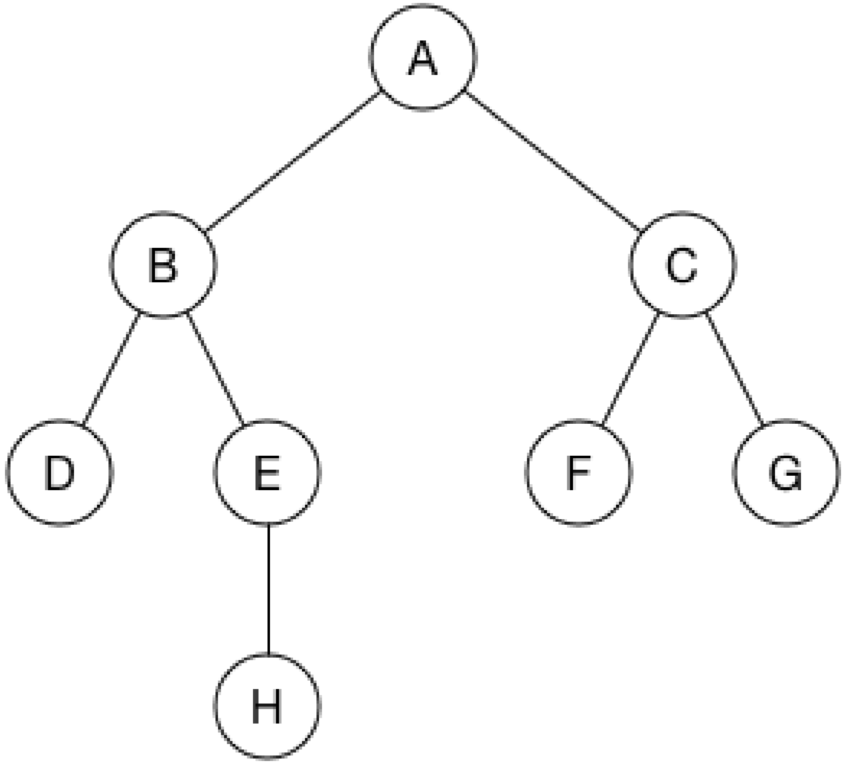




Urutan simpul-simpul yang dikunjungi secara BFS dari A → A, B, C, D, E, F, G, H

- Traversal secara BFS pada graf tersebut dapat digambarkans ebagai pohon BFS:

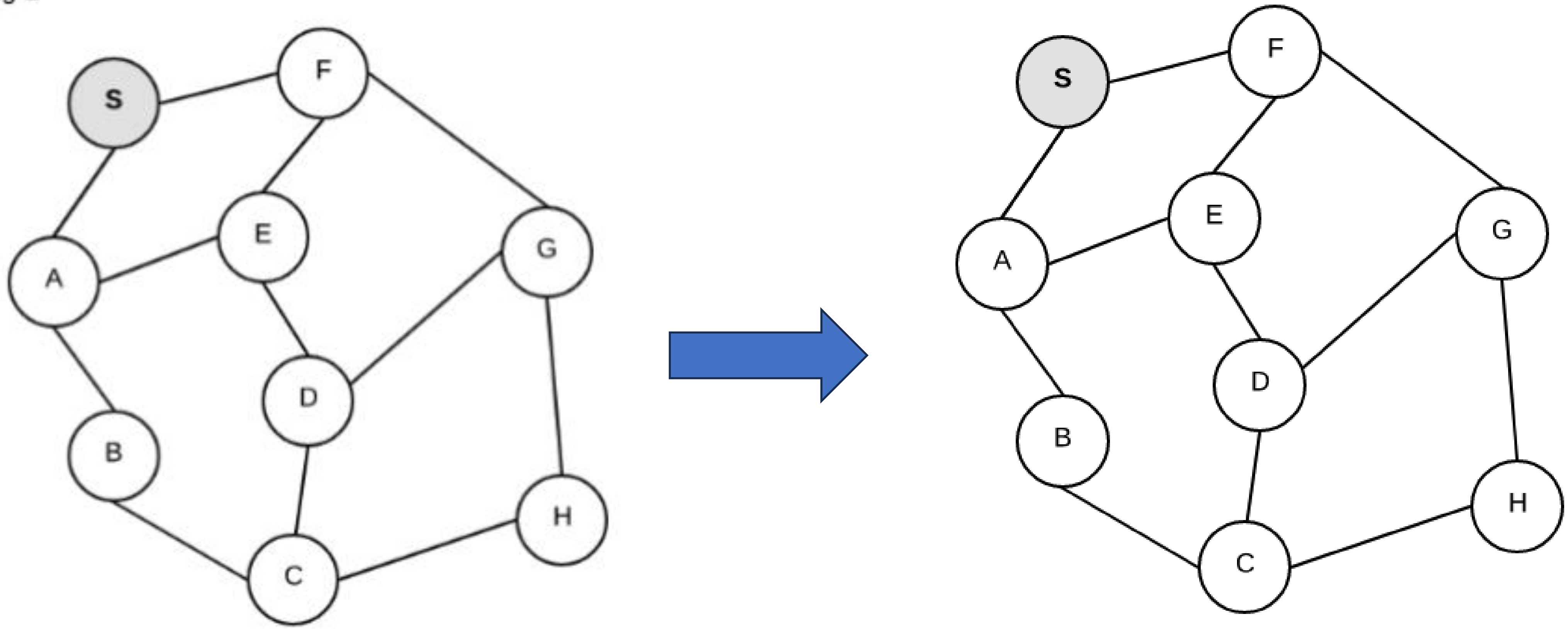




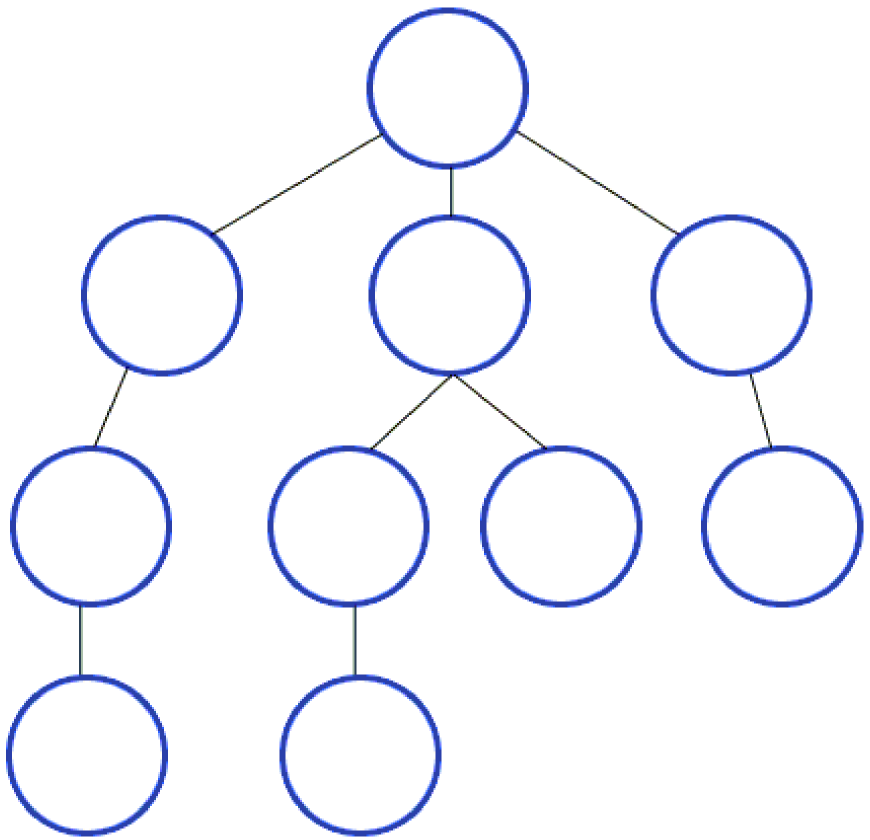
Urutan simpul-simpul yang dikunjungi secara BFS dari A → A, B, C, D, E, F, G, H

Fig 2

Fig 2



Urutan simpul-simpul yang dikunjungi secara BFS dari S → S, A, F, B, E, G, C, D, H



BFS: Struktur Data

1. Matriks ketetanggaan $A = [a_{ij}]$ yang berukuran nxn,
   a_{ij}= 1, jika simpul i dan simpul j bertetangga,
   a_{ij}= 0, jika simpul i dan simpul j tidak bertetangga.

2. Antrian q untuk menyimpan simpul yang telah dikunjungi.

3. Tabel Boolean, diberi nama "dikunjungi"

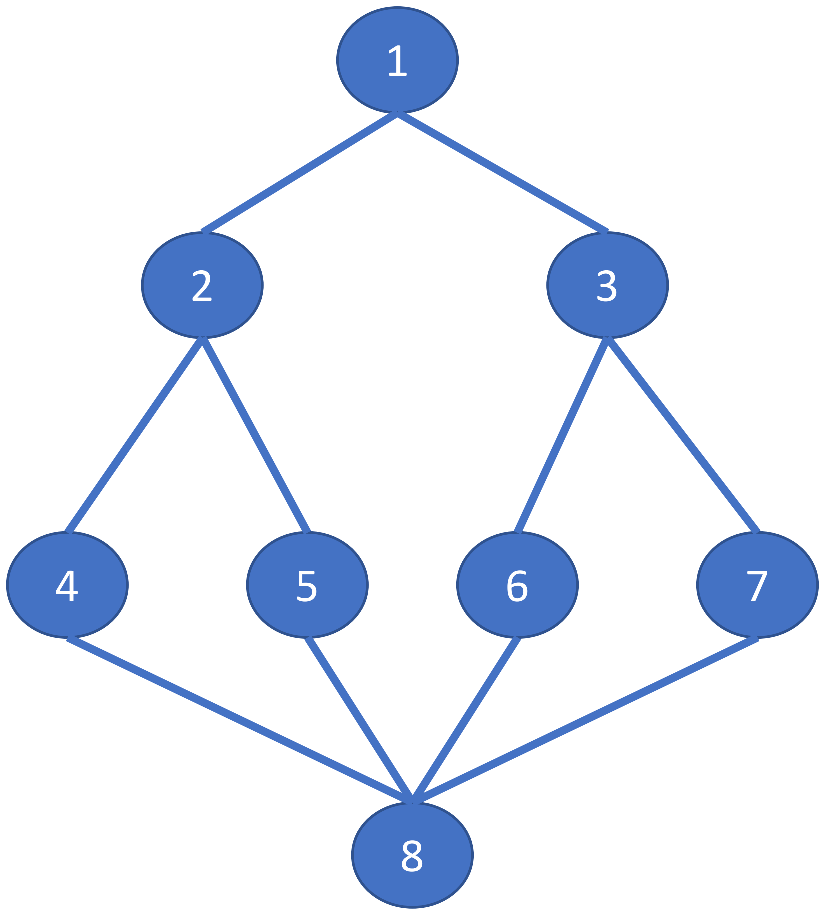
   `dikunjungi : array[l..n] of boolean`

   dikunjungi[i] = *true* jika simpul i sudah dikunjungi

   dikunjungi[i] = *false* jika simpul i belum dikunjungi

```
procedure BFS(input v:integer)
{ Traversal graf dengan algoritma pencarian BFS.

  Masukan: v adalah simpul awal kunjungan
  Keluaran: semua simpul yang dikunjungi dicetak ke layar
}
Deklarasi
   w : integer
   q : antrian;

   procedure BuatAntrian(input/output q : antrian)
   { membuat antrian kosong, kepala(q) diisi 0 }

   procedure MasukAntrian(input/output q:antrian, input v:integer)
   { memasukkan v ke dalam antrian q pada posisi belakang }

   procedure HapusAntrian(input/output q:antrian,output v:integer)
   { menghapus v dari kepala antrian q }

   function AntrianKosong(input q:antrian) → boolean
   { true jika antrian q kosong, false jika sebaliknya }

Algoritma:
   BuatAntrian(q)          { buat antrian kosong }

   write(v)                { cetak simpul awal yang dikunjungi }
   dikunjungi[v]←true      { simpul v telah dikunjungi, tandai dengan
                             true}
   MasukAntrian(q,v)      { masukkan simpul awal kunjungan ke dalam
                            antrian}

  { kunjungi semua simpul graf selama antrian belum kosong }
   while not AntrianKosong(q) do
       HapusAntrian(q,v)   { simpul v telah dikunjungi, hapus dari
                             antrian }
       for tiap simpul w yang bertetangga dengan simpul v do
             if not dikunjungi[w] then
                write(w)     {cetak simpul yang dikunjungi}
                MasukAntrian(q,w)
                dikunjungi[w]←true
             endif
      endfor
   endwhile
  { AntrianKosong(q) }
```

BFS: Ilustrasi



Iterasi	V	Q	dikunjungi							
			1	2	3	4	5	6	7	8
Inisialisasi	1	{1}	T	F	F	F	F	F	F	F
Iterasi 1	1	{2,3}	T	T	T	F	F	F	F	F
Iterasi 2	2	{3,4,5}	T	T	T	T	T	F	F	F
Iterasi 3	3	{4,5,6,7}	T	T	T	T	T	T	T	F
Iterasi 4	4	{5,6,7,8}	T	T	T	T	T	T	T	T
Iterasi 5	5	{6,7,8}	T	T	T	T	T	T	T	T
Iterasi 6	6	{7,8}	T	T	T	T	T	T	T	T
Iterasi 7	7	{8}	T	T	T	T	T	T	T	T
Iterasi 8	8	{}	T	T	T	T	T	T	T	T

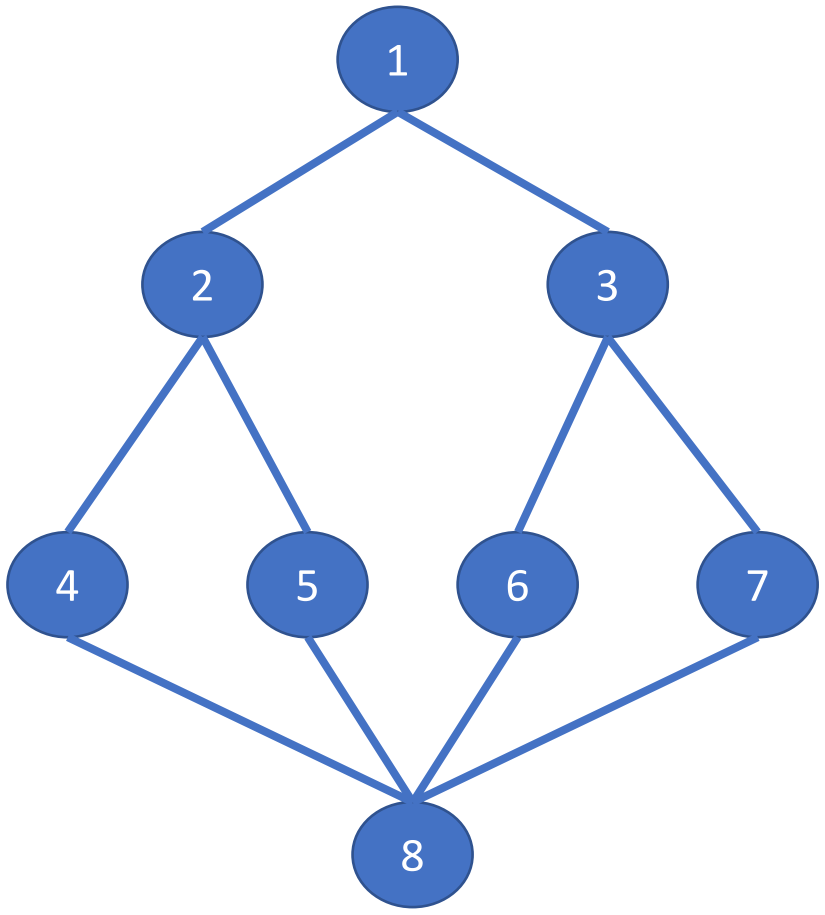
Urutan simpul2 yang dikunjungi: 1, 2, 3, 4, 5, 6, 7, 8

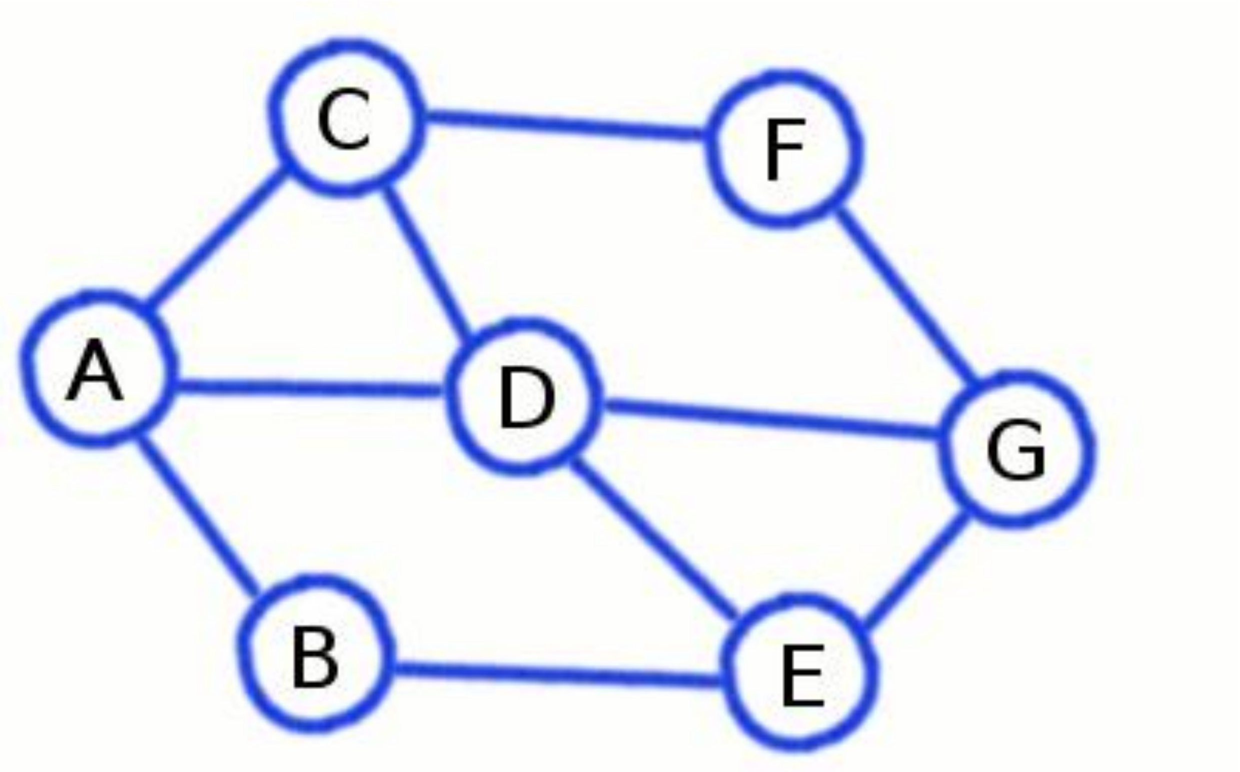
Pencarian Mendalam (DFS)

- Traversal dimulai dari simpul *v*.

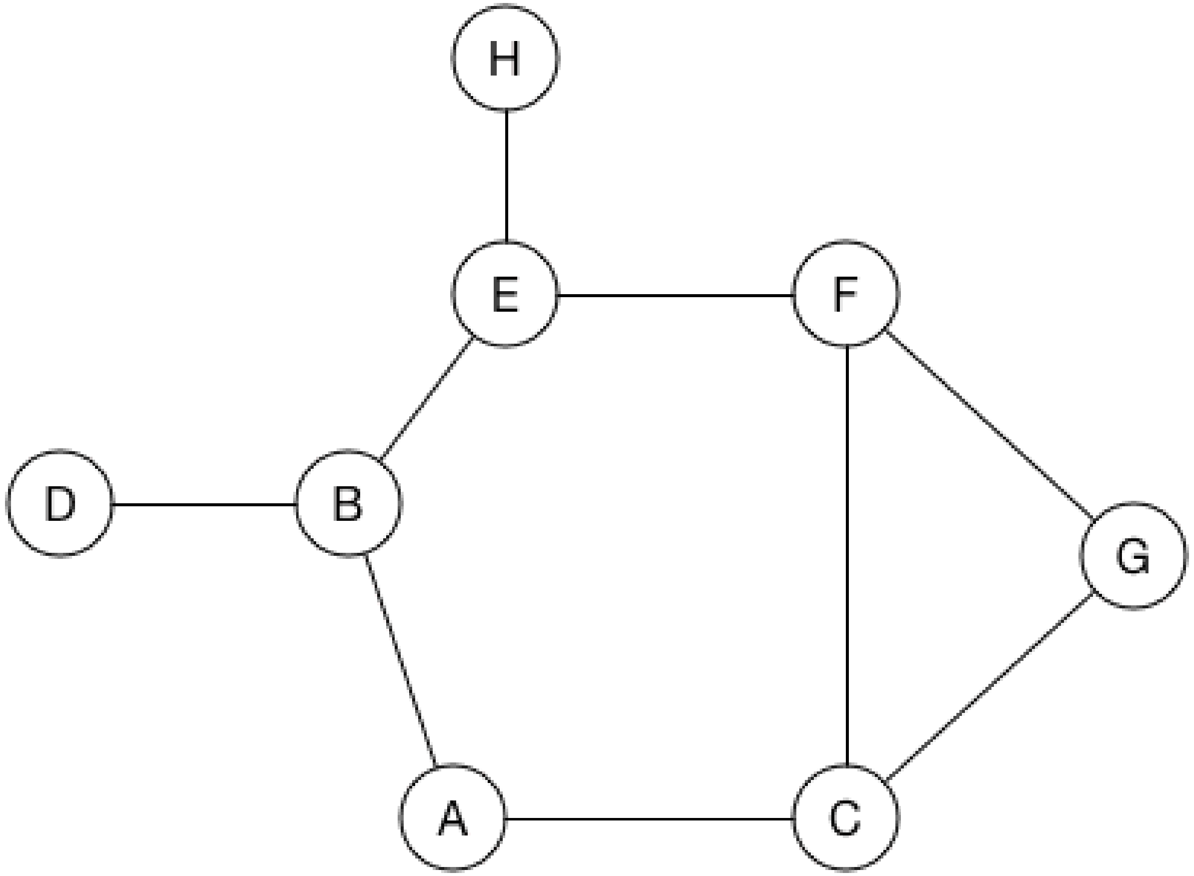
- Algoritma:

1. Kunjungi simpul *v*
2. Kunjungi simpul w yang bertetangga dengan simpul *v*.
3. Ulangi DFS mulai dari simpul *w*.
4. Ketika mencapai simpul *u* sedemikian sehingga semua simpul yang bertetangga dengannya telah dikunjungi, pencarian dirunut-balik (*backtrack*) ke simpul terakhir yang dikunjungi sebelumnya dan mempunyai simpul w yang belum dikunjungi.
5. Pencarian berakhir bila tidak ada lagi simpul yang belum dikunjungi yang dapat dicapai dari simpul yang telah dikunjungi.





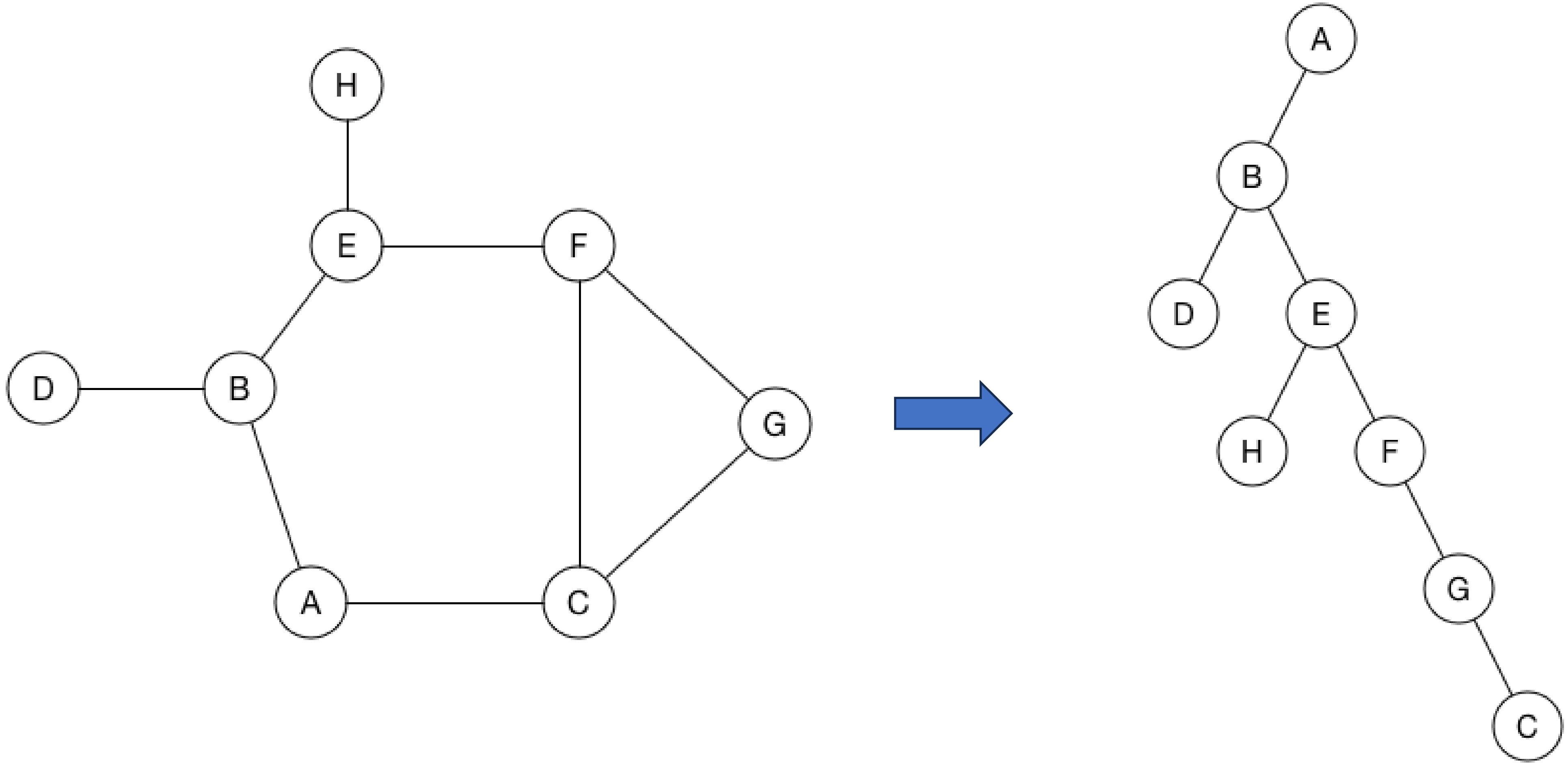
Urutan simpul-simpul yang dikunjungi secara DFS dari A → A, B, E, D, C, F, G



Urutan simpul-simpul yang dikunjungi secara DFS dari A → A, B, D, E, H, F, G, C

(atau: A, B, D, E, F, C, G, H   jika sesuai urutan abjad lebih dahulu)

- Traversal secara DFS pada graf tersebut dapat digambarkan sebagai pohon DFS:

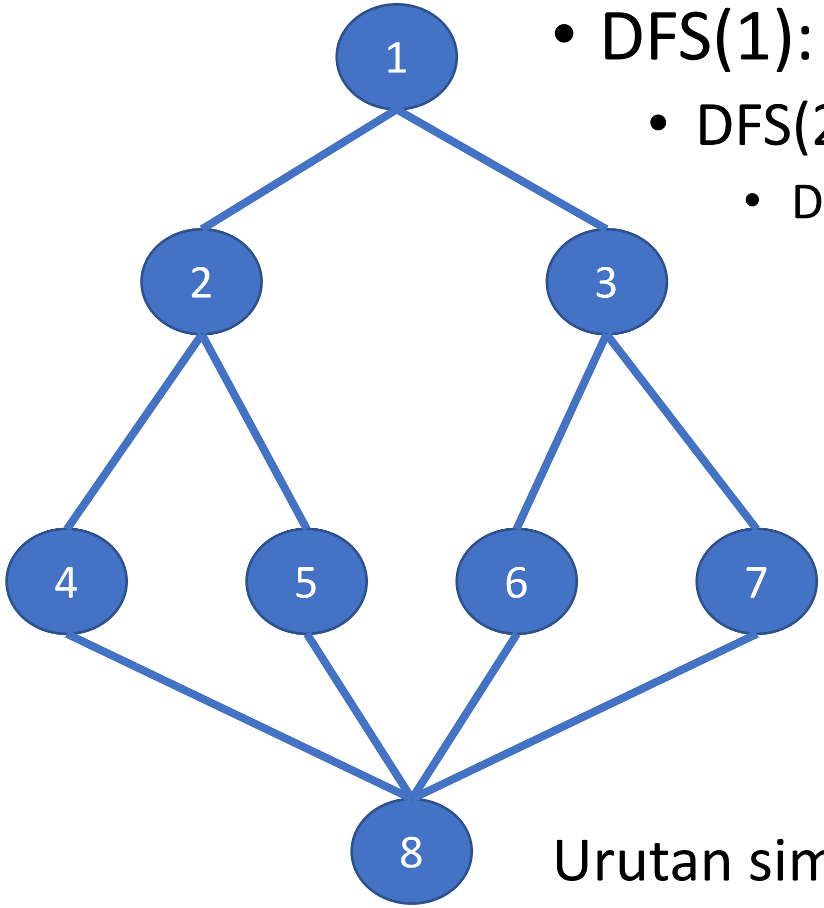


Urutan simpul-simpul yang dikunjungi secara DFS dari A → A, B, D, E, H, F, G, C

DFS

```
procedure DFS(input v:integer)
{Mengunjungi seluruh simpul graf dengan algoritma pencarian DFS

Masukan: v adalah simpul awal kunjungan
Keluaran: semua simpulyang dikunjungi ditulis ke layar
}
Deklarasi
   w : integer

Algoritma:
   write(v)
   dikunjungi[v]←true
   for w←1 to n do
      if A[v,w]=1 then {simpul v dan simpul w bertetangga }
          if not dikunjungi[w] then
             DFS(w)
          endif
      endif
   endfor
```

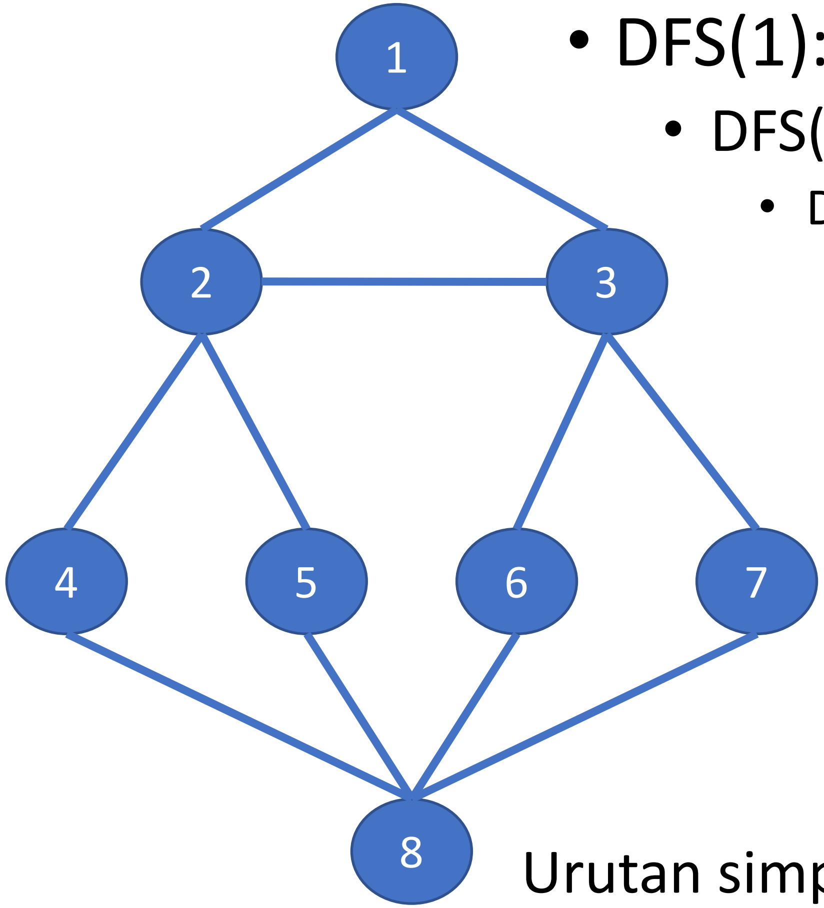
DFS: Ilustrasi 1



- DFS(1): v=1; dikunjungi[1]=true; DFS(2)
  - DFS(2): v=2; dikunjungi[2]=true; DFS(4)
    - DFS(4): v=4; dikunjungi[4]=true; DFS(8)
      - DFS(8): v=8; dikunjungi[8]=true; DFS(5)
        - DFS(5): v=5; dikunjungi[5]=true
        - DFS(6): v=6; dikunjungi[6]=true; DFS(3)
          - DFS(3): v=3; dikunjungi[3]=true; DFS(7)
            - DFS(7): v=7; dikunjungi[7]=true

Urutan simpul2 yang dikunjungi: 1, 2, 4, 8, 5, 6, 3, 7

DFS: Ilustrasi 2

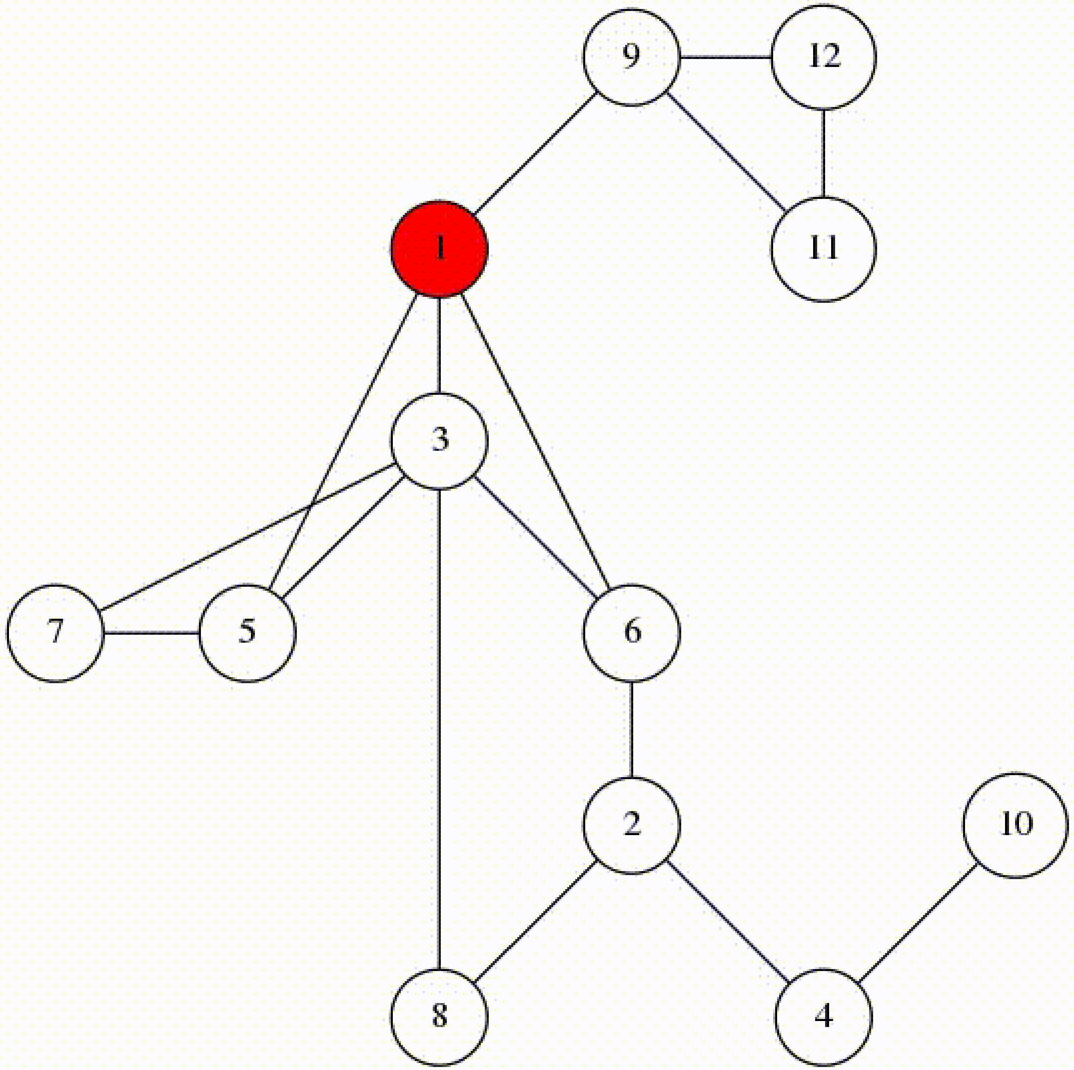


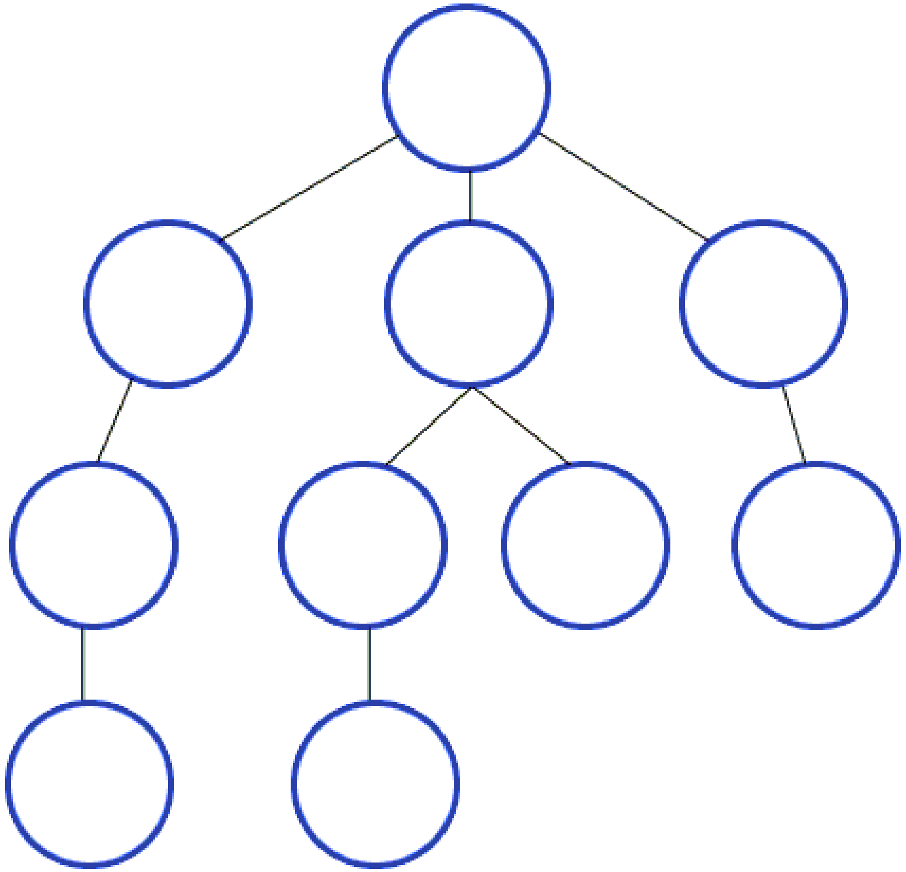
- DFS(1): v=1; dikunjungi[1]=true; DFS(2)
  - DFS(2): v=2; dikunjungi[2]=true; DFS(3)
    - DFS(3): v=3; dikunjungi[3]=true; DFS(6)
      - DFS(6): v=6; dikunjungi[6]=true; DFS(8)
        - DFS(8): v=8; dikunjungi[8]=true; DFS(4)
          - DFS(4): v=4; dikunjungi[4]=true;

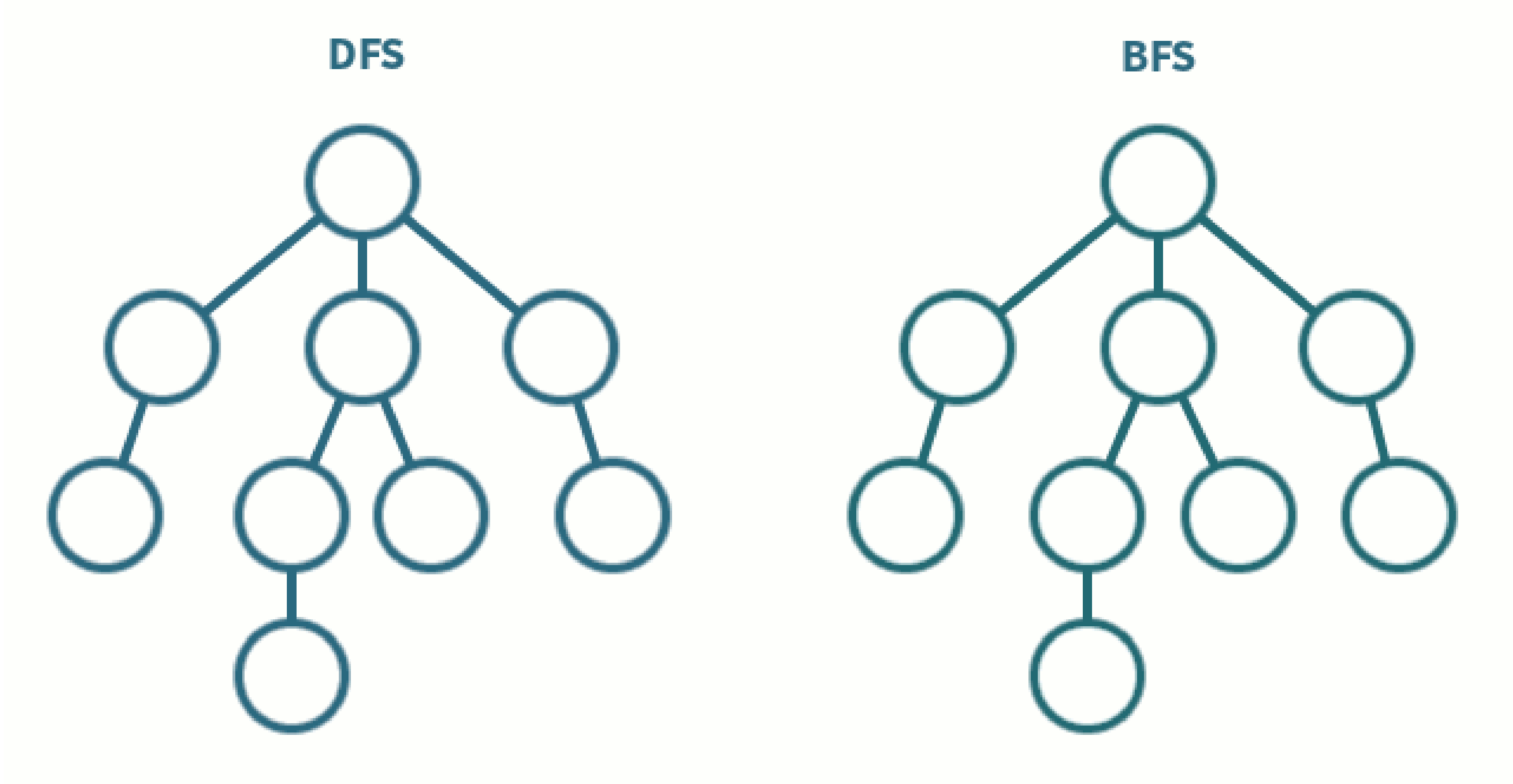
          DFS(8): DFS(5)
          - DFS(5): v=5; dikunjungi[5]=true

          DFS(8): DFS(7)
          - DFS(7): v=7; dikunjungi[7]=true

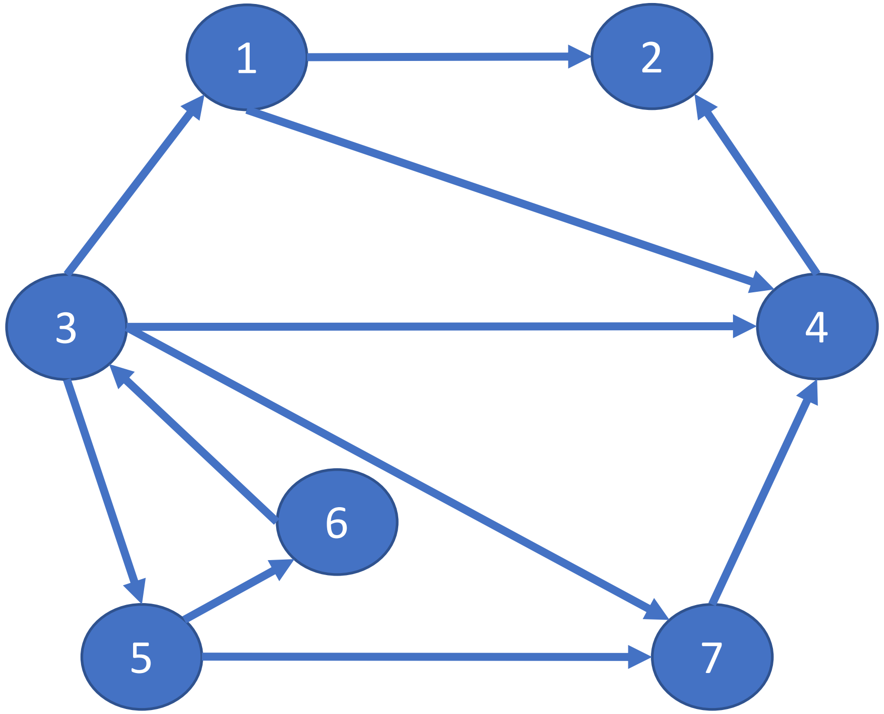
Urutan simpul2 yang dikunjungi: 1, 2, 3, 6, 8, 4, 5, 7


9
12
1
11
3
7
5
6
2
10
8
4




DFS
BFS

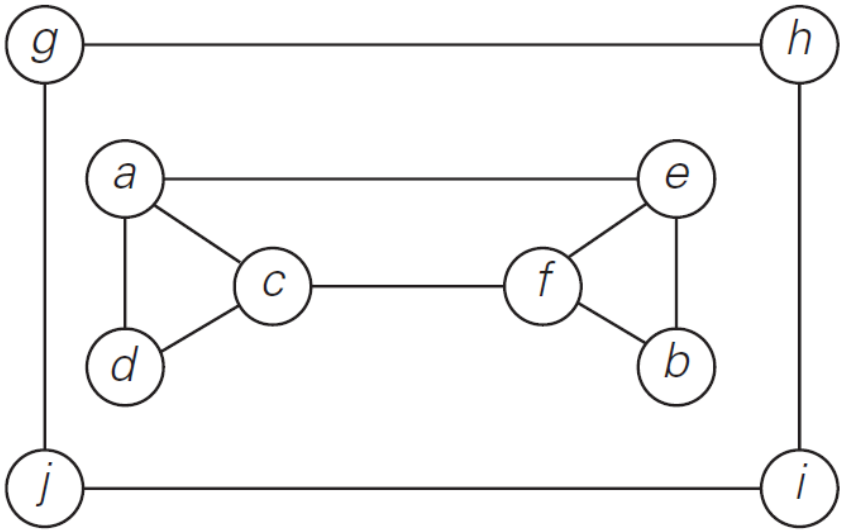
Contoh (hal 113)



- Khusus untuk graf berarah, beberapa simpul mungkin tidak dapat dicapai dari simpul awal. Coba dengan simpul yang belum dikunjungi sebagai simpul awal. (hal 113)

- DFS (1): 1-2-4-3-5-6-7
- BFS (1): 1-2-4-3-5-7-6

Contoh Lain



- Bagaimana penelusuran graf dengan BFS?

- Bagaimana Penelusuran graf dengan DFS

Penerapan BFS dan DFS: Citation Map

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

ELSEVIER

A novel robust scaling image watermarking scheme based on Gaussian Mixture Model

Maryam Amirmazlaghani [a,*], Mansoor Rezghi [b], Hamidreza Amindavar [c]

[a] Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran
[b] Department of Computer Science, Tarbiat Modares University, Tehran, Iran
[c] Department of Electrical Engineering, Amirkabir University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Available online 24 October 2014

Keywords:
Gaussian Mixture Model (GMM)
Statistical modeling
Maximum Likelihood detector
Wavelet transform
L-curve method

ABSTRACT

In this paper, we propose a novel scaling watermarking scheme in which the watermark is embedded in the low-frequency wavelet coefficients to achieve improved robustness. We demonstrate that these coefficients have significantly non-Gaussian statistics that are efficiently described by Gaussian Mixture Model (GMM). By modeling the coefficients using the GMM, we calculate the distribution of watermarked noisy coefficients analytically and we design a Maximum Likelihood (ML) watermark detector using channel side information. Also, we extend the proposed watermarking scheme to a blind version. Consequently, since the efficiency of the proposed method is dependent on the good selection of the scaling factor, we propose L-curve method to find the tradeoff between the imperceptibility and robustness of the watermarked data. Experimental results demonstrate the high efficiency of the proposed scheme and the performance improvement in utilizing the new strategy in comparison with the some recently proposed techniques.

1. Introduction

Nowadays, we encounter easy distribution and sharing of digital media due to easy access to the Internet. However, it has made the protection and authentication of multimedia contents and copyright to be of a great concern. Digital watermarking which embeds hidden secondary data into digital multimedia products, has been applied as a technology for postdistribution protection of digital media. Imperceptibility and robustness are two main requirements of watermarking schemes and usually there is a trade off between them. Watermarks have two categories of roles: In the first category, the main goal is to determine whether a specific watermark is present or not in the received media content (integrity verification) (Cheng & Huang, 2003; Merhav & Sabbag, 2008). In the second category, the embedded watermark is considered as a hidden unknown message which should be decoded

watermark retrieval (Mahbubur Rahman, Omair Ahmad, & Swamy, 2009). There are several methods of watermark embedding, such as through quantization (Chen & Wornell, 2001; Okman & Akar, 2007), additive (Mahbubur Rahman et al., 2009; Mairgiotis, Galatsanos, & Yang, 2008), and multiplicative (Barni, Bartolini, Rosa, & Piva, 2001; Cheng & Huang, 2003; Cox, Kilian, Leighton, & Shammoon, 1997; Ng & Garg, 2005). In multiplicative watermarks, the power of the watermark is proportional to the corresponding image feature samples. So, multiplicative watermarks are image content dependent and they are more robust than additive watermarking methods. Another embedding approach is based on scaling. In the scaling based watermarking, the watermark data is embedded into the cover media by slightly scaling the cover (Akhaee et al., 2009).

The watermark is often embedded in a transformed domain. The transforms usually employed for digital watermarking are

References

Akhaee, M. A., Sahraeian, S. M. E., & Marvasti, F. (2010). Contourlet-based image watermarking using optimum detector in a noisy environment. *IEEE Transactions on Image Processing, 19*, 967–980.

Akhaee, M. A., Sahraeian, S. M. E., Sankur, B., & Marvasti, F. (2009). Robust scaling-based image watermarking using maximum-likelihood decoder with optimum strength factor. *IEEE Transactions on Multimedia, 11*, 822–833.

Allili, M. S. (2012). Wavelet modeling using finite mixtures of generalized Gaussian distributions: Application to texture discrimination and retrieval. *IEEE Transactions on Image Processing, 21*, 1452–1464.

Amirmazlaghani, M., Amindavar, H., & Moghaddamjoo, A. R. (2009). Speckle suppression in SAR images using the 2-D GARCH model. *IEEE Transactions on Image Processing, 18*, 250–259.

Barni, M., Bartolini, F., Rosa, A. D., & Piva, A. (2001). A new decoder for the optimum recovery of nonadditive watermarks. *IEEE Transactions on Image Processing, 10*, 755–766.

Barni, M., Bartolini, F., Rosa, A. D., & Piva, A. (2003). Optimum decoding and detection of multiplicative watermarks. *IEEE Transactions on Signal Processing, 51*, 1118–1123.

Cheng, Q., & Huang, T. S. (2003). Robust optimum detection of transform domain multiplicative watermarks. *IEEE Transactions on Signal Processing, 51*, 906–924.

Chen, B., & Wornell, G. W. (2001). Quantization index modulation: A class of provably good methods for digital watermarking and information embedding. *IEEE Transactions on Information Theory, 47*, 1423–1443.

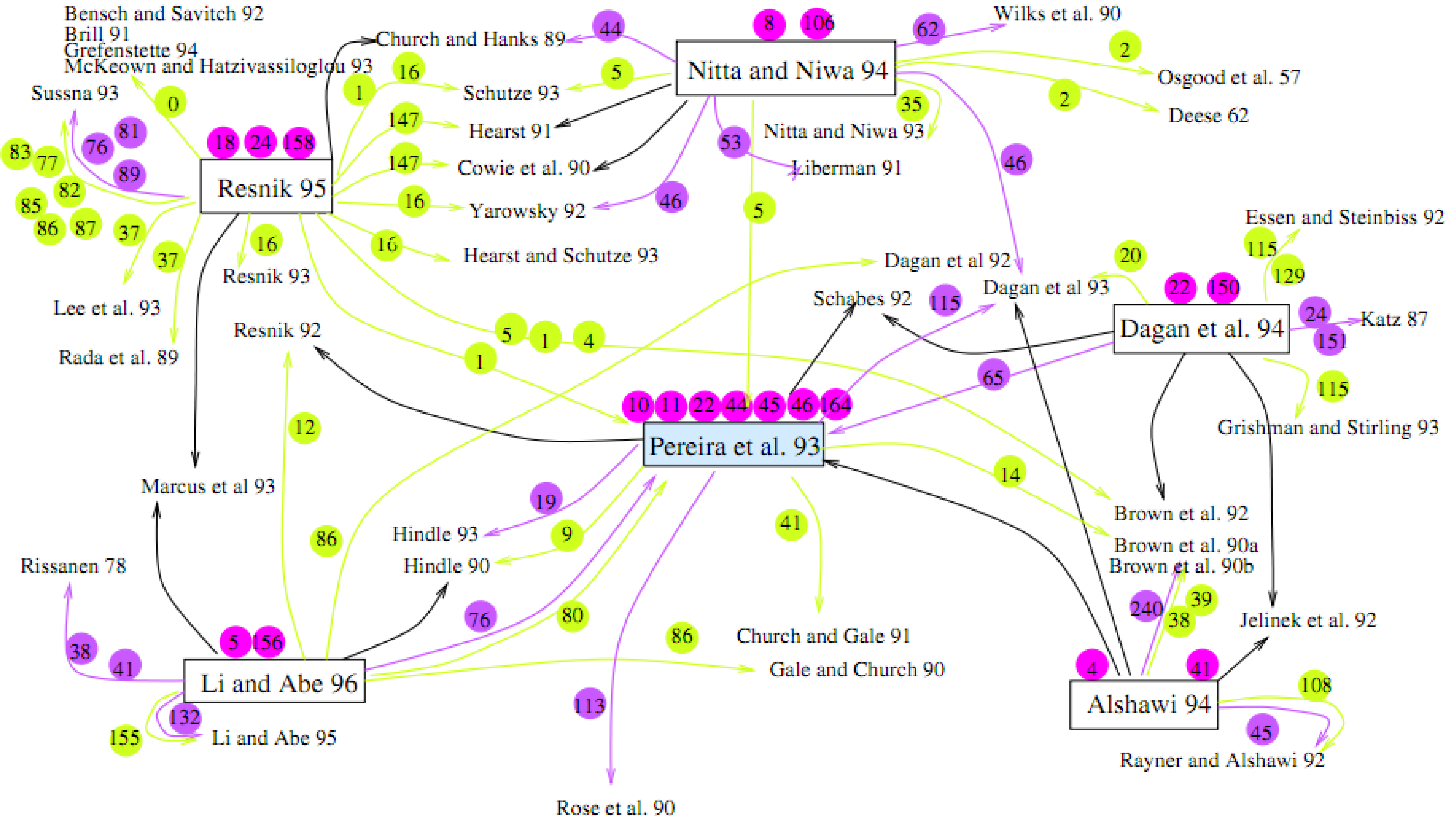
Cox, I., Kilian, J., Leighton, T., & Shammoon, T. (1997). Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing, 6*, 1673–1687.

Doncel, V. R., Nikolaidis, N., & Pitas, I. (2007). An optimal detector structure for the Fourier descriptors domain watermarking of 2D vector graphics. *IEEE Transactions on Visualization and Computer Graphics, 13*, 851–863.

Donoho, D. L. (1994). Ideal spatial adoption by wavelet shrinkage. *Biometrika, 81*, 425–455.

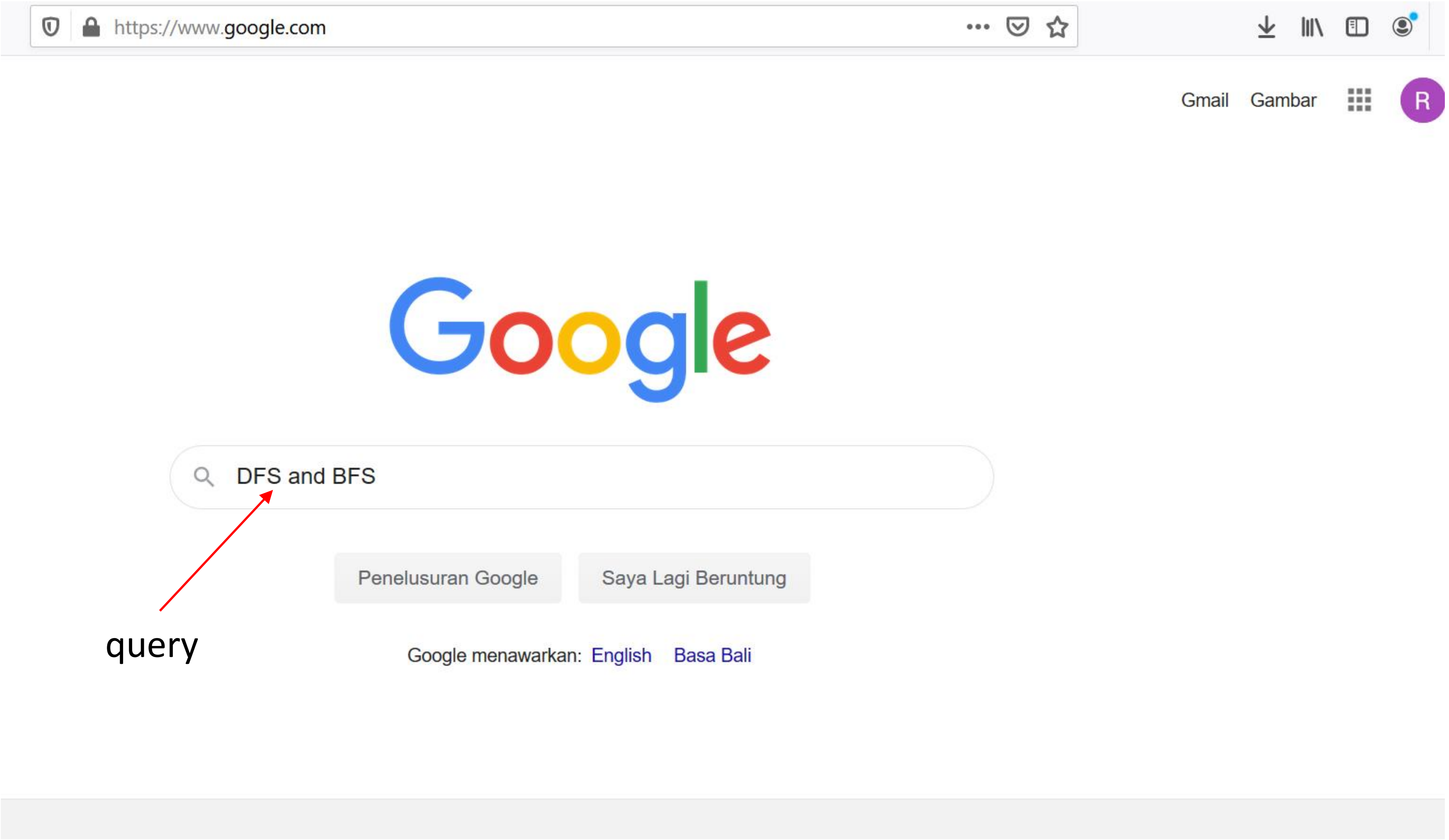
Gembicki, F., & Haimes, Y. (1975). Approach to performance and sensitivity multiobjective optimization: The goal attainment method. *IEEE Transactions on Automatic Control, AC-20*, 769–771.

Citation Map:



Sumber: Teufel (1999), Argumentative Zoning

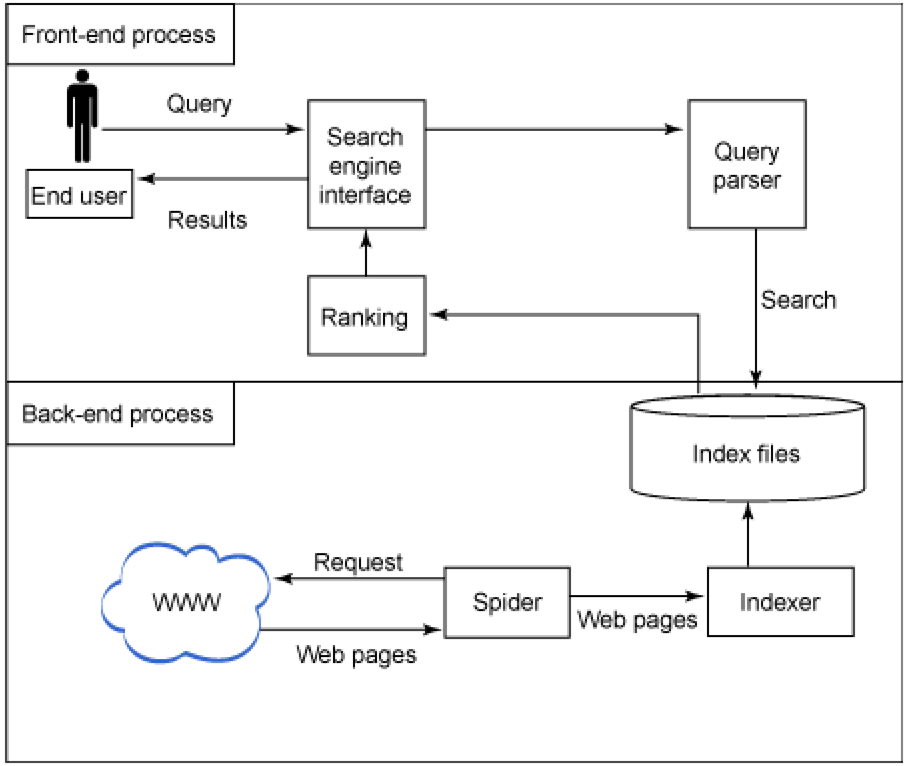
Penerapan BFS dan DFS: *Web Spider*



query

Penerapan BFS dan DFS: *Web Spider*

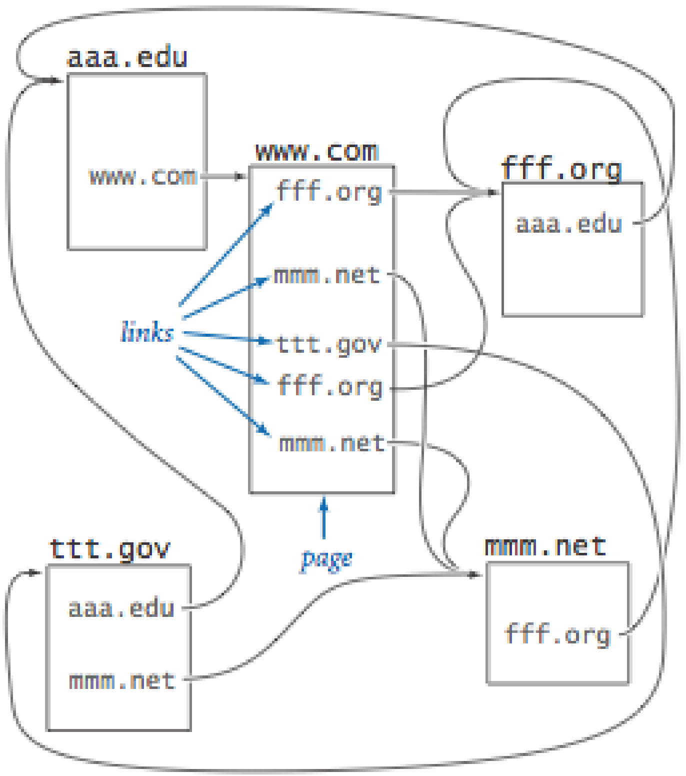
Arsitektur umum mesin pencari



http://www.ibm.com/developerworks/web/library/wa-lucene2/

- Secara periodik, *web spider* menjejalahi internet untuk mengunjungi halaman-halaman web

Web Spider: Penjelajahan Web

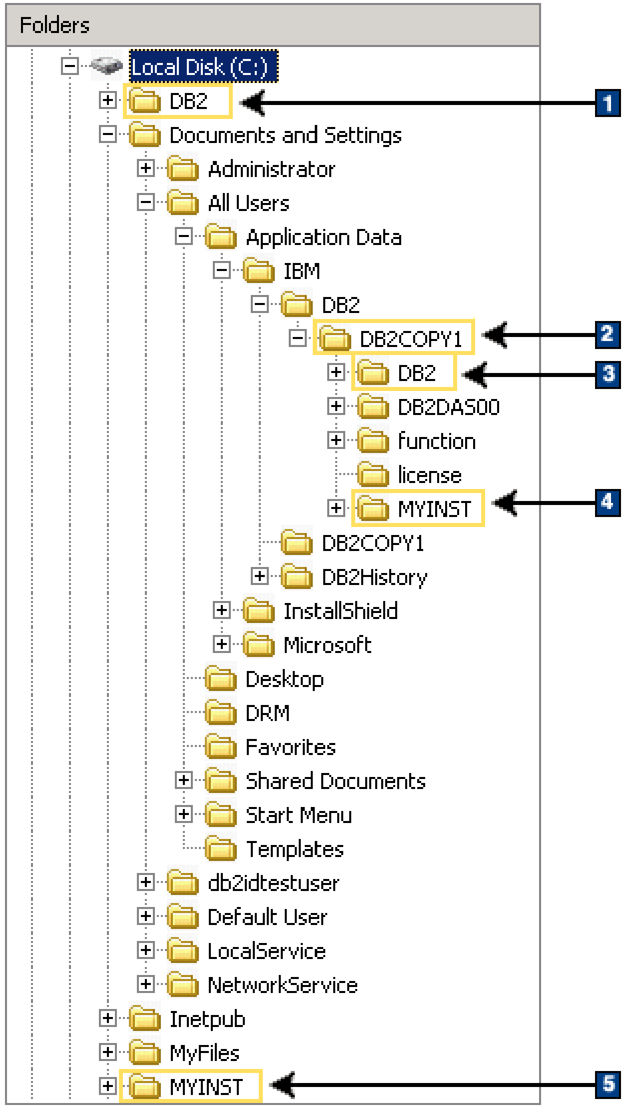


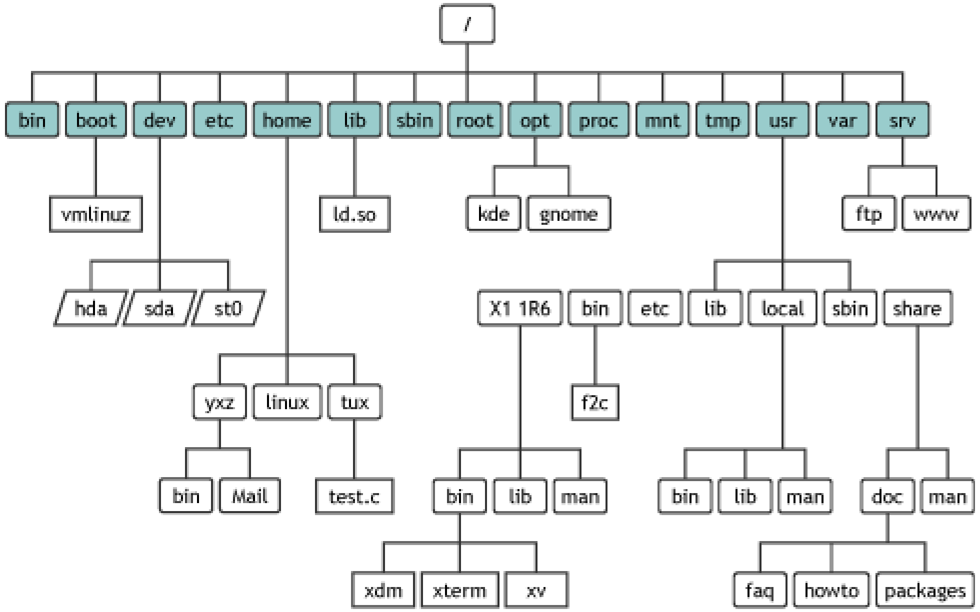
Pages and links

http://introcs.cs.princeton.edu/java/16pagerank/

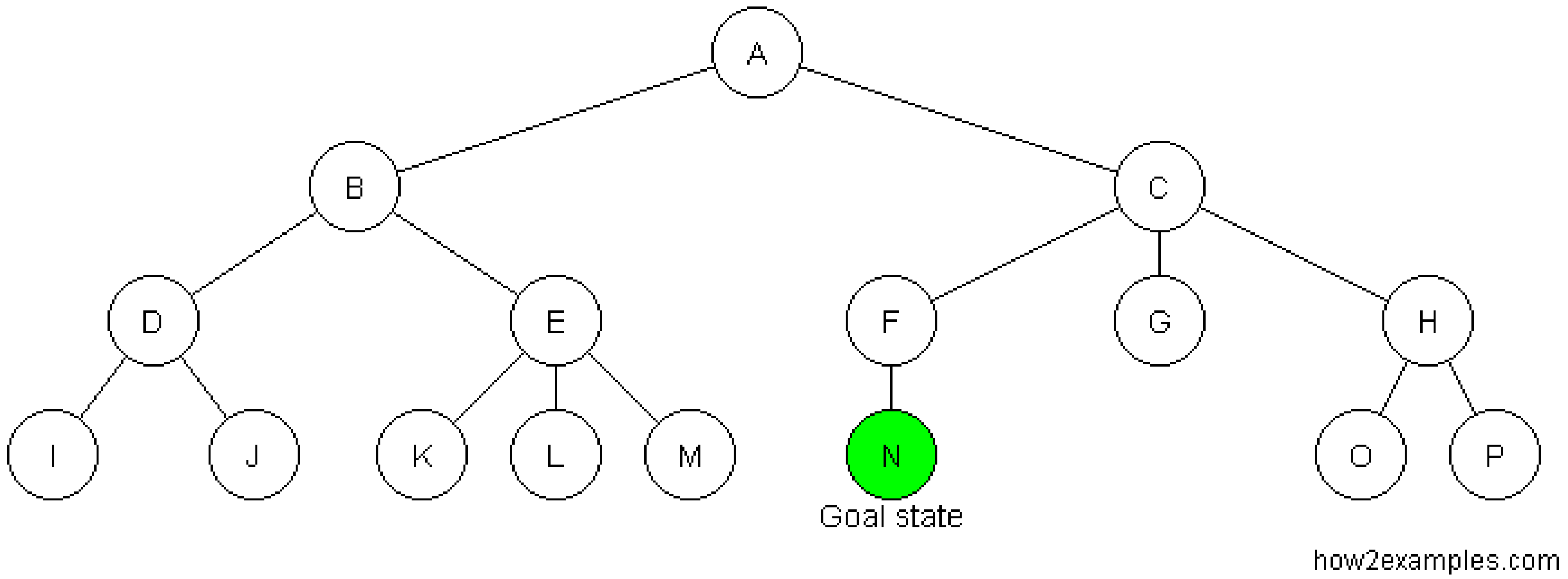
- Halaman web dimodelkan sebagai graf berarah
  - Simpul menyatakan halaman web (web page)
  - Sisi menyatakan link ke halaman web
- Bagaimana teknik menjelajahi web? Secara DFS atau BFS
- Dimulai dari web page awal, lalu setiap link ditelusuri secara DFS sampai setiap web page tidak mengandung link.

DFS dan BFS untuk penelusuran direktori (folder)

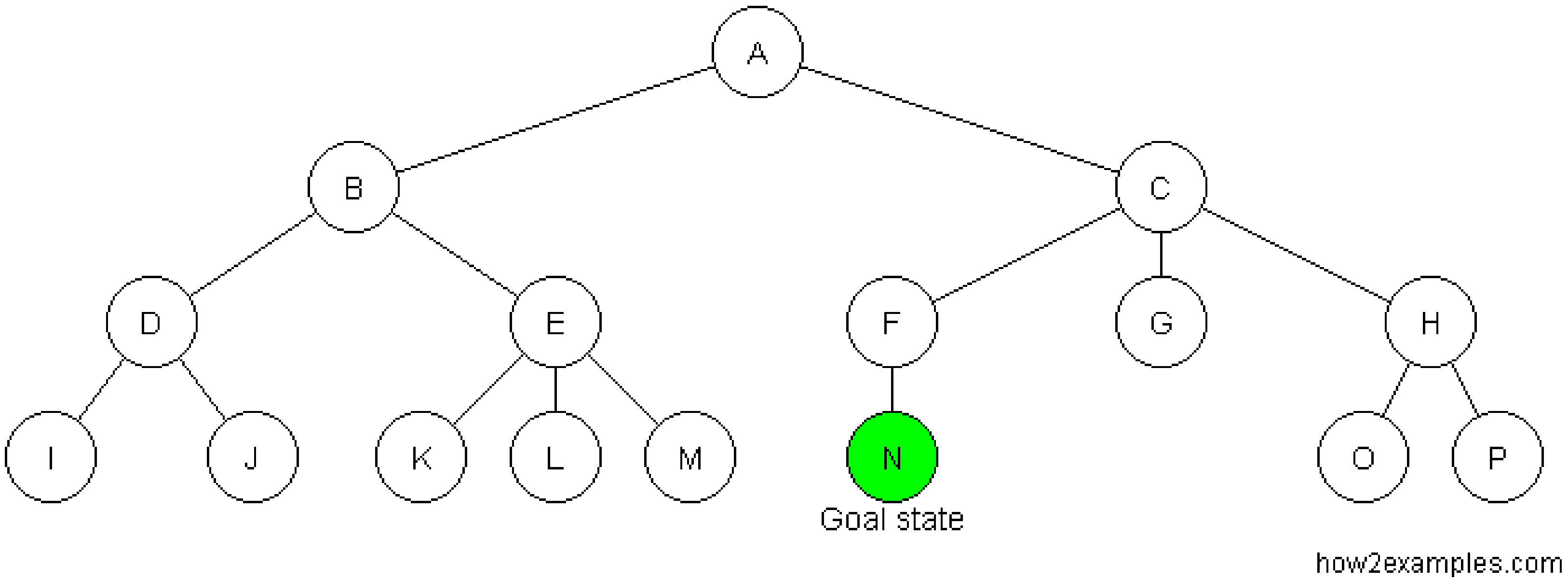




Pencarian dokumen di dalam direktori (folder) secara BFS



Pencarian dokumen di dalam direktori (folder) secara DFS



BERSAMBUNG