Pohon dari Sudut Pandang Teori Graf

Ari Wardana
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
email: if16065@students.if.itb.ac.id

Abstract *– Pohon didefinisikan sebagai suatu graf tak berarah terhubungkan (connected undirected graph) yang tidak mengandung rangkaian sederhana. Pohon adalah bentuk khusus dari suatu graf yang banyak diterapkan untuk berbagai keperluan. Misalnya struktur organisasi suatu perusahaan, silsilah suatu keluarga, skema sistem gugur suatu pertandingan, dan ikatan kimia suatu molekul adalah jenis graf yang tergolong sebagai pohon. Pada pohon, simpul-simpul yang berderajat satu dinamakan daun (leave), sedangkan simpul yang derajatnya lebih besar daripada satu dinamakan simpul cabang (branch node) atau simpul internal (internal node) dan kumpulan pohon-pohon yang terpisahkan satu sama lain disebut hutan (forest).*

Kata Kunci *: Pohon, Graf*

1. PENDAHULUAN

Di antara sekian banyak konsep dalam teori graf, konsep pohon(tree) mungkin merupakan konsep yang paling penting, khususnya bagi orang yang tertarik dengan penerapan graf. Beberapa terapan, baik dalam bidang ilmu komputer maupun di luar ilmu komputer, yang telah mengkaji pohon secara intensif sebagai objek matematika. Dalam kehidupan sehari-hari, orang telah lama menggunakan pohon unutk menggambarkan hirarkhi. Misalnya, pohon silsilah, keluarga, struktur organisasi pertandingan, dan lain-lain. Para ahli bahasa menggunakan pohon unukt menguraikan kalimat yang disebut pohon parsing(*parse tree*).

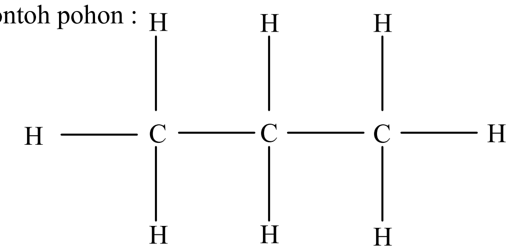
Pada makalah ini akan dibahas pohon dari sudut pandang teori graf.

2. DEFINISI POHON

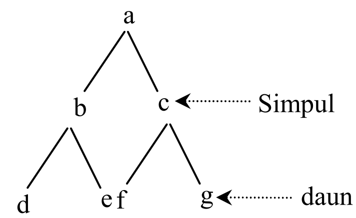
Pohon adalah graf ynag khusus. Definisi pohon adalh sebagai berikut:

> "**Pohon** adalah graf tak-berarah terhubung yang tidak mengandung sirkuit"[1]

Contoh pohon :



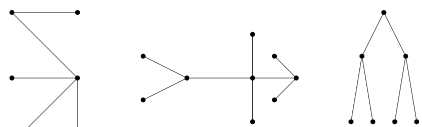
Gambar 1 : Ikatan Kimia



Gambar 2 : Diagram Pohon

Beberapa pohondapat dibuat menjadi hutan.

Hutan (*forest*) adalah :
- kumpulan pohon yang saling lepas, atau
- graf tidak terhubung yang tidak mengandung sirkuit. Setiap komponen di dalam graf terhubung tersebut adalah pohon.



Gambar 3 : Hutan yang terdiri dari tiga buah pohon

3. SIFAT-SIFAT POHON

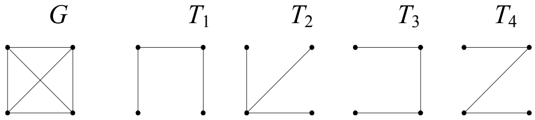
Teorema. Misalkan $G = (V, E)$ adalah graf tak-berarah sederhana dan jumlah simpulnya n. Maka, semua pernyataan di bawah ini adalah ekivalen:

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki $m = n - 1$ buah sisi.
4. G tidak mengandung sirkuit dan memiliki $m = n - 1$ buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan.

Teorema di atas dapat dikatakan sebagai definisi lain dari pohon.

4. POHON MERENTANG

Pohon merentang (*spanning tree*) dari graf terhubung adalah upagraf merentang yang berupa pohon.
Pohon merentang diperoleh dengan memutus sirkuit di dalam graf.

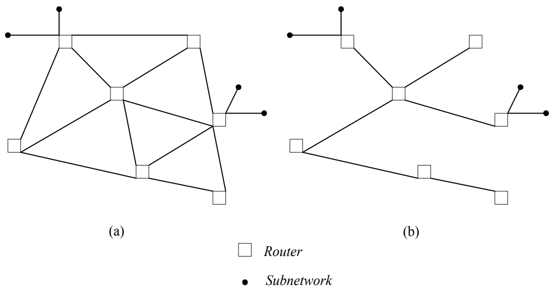


Gambar 4 : Pohon merentang T_1, T_2, T_3, dan T_4 yang dibentuk dari graf G

Setiap graf terhubung mempunyai paling sedikit satu buah pohon merentang. Graf tak-terhubung dengan k komponen mempunyai k buah hutan merentang yang disebut hutan merentang (*spanning forest*).

Aplikasi Pohon Merentang

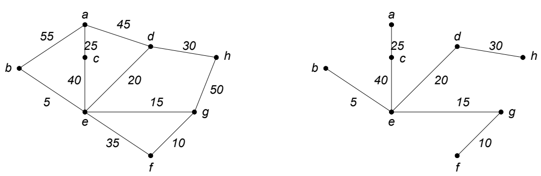
1. Jalan-jalan seminimum mungkin yang menghubungkan semua kota sehingga setiap kota tetap terhubung satu sama lain.
2. Perutean (*routing*) pesan pada jaringan komputer.



(a) Jaringan komputer, (b) Pohon merentang *multicast*
Gambar 5 : Perutean (routing)

Pohon Rentang Minimum

- Graf terhubung-berbobot mungkin mempunyai lebih dari 1 pohon merentang.
- Pohon rentang yang berbobot minimum – dinamakan **pohon merentang minimum** (*minimum spanning tree*).



Gambar 6 : Pohon merentang minimum dari sebuah graf

5. ALGORITMA PRIM

Langkah 1: ambil sisi dari graf G yang berbobot minimum, masukkan ke dalam T.

Langkah 2: pilih sisi (u, v) yang mempunyai bobot minimum dan bersisian dengan simpul di T, tetapi (u, v) tidak membentuk sirkuit di T. Masukkan (u, v) ke dalam T.

Langkah 3: ulangi langkah 2 sebanyak $n - 2$ kali.

- Jumlah langkah seluruhnya di dalam algoritma Prim adalah

$$1 + (n - 2) = n - 1 \qquad (1)$$

yaitu sebanyak jumlah sisi di dalam pohon rentang dengan n buah simpul.

```
procedure Prim(input G : graf, output T :
pohon)
{ Membentuk pohon merentang minimum T dari
graf terhubung-berbobot G.
Masukan: graf-berbobot terhubung G = (V, E),
dengan |V|= n
Keluaran: pohon rentang minimum T = (V, E')
}
Deklarasi
  i, p, q, u, v : integer

Algoritma
  Cari sisi (p,q) dari E yang berbobot
terkecil
  T ← {(p,q)}
  for i←1 to n-2 do
    Pilih sisi (u,v) dari E yang bobotnya
terkecil namun
    bersisian dengan simpul di T
    T ← T ∪ {(u,v)}
  endfor
```

6. ALGORITMA KRUSKAL

(Langkah 0: sisi-sisi dari graf sudah diurut menaik berdasarkan bobotnya – dari bobot kecil ke bobot besar)

Langkah 1: T masih kosong

Langkah 2: pilih sisi (u, v) dengan bobot minimum yang tidak membentuk sirkuit di T. Tambahkan (u, v) ke dalam T.

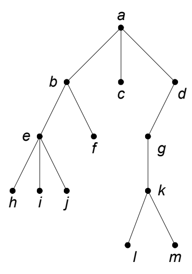
Langkah 3: ulangi langkah 2 sebanyak $n - 1$ kali.

```
procedure Kruskal(input G : graf, output T :
pohon)

{ Membentuk pohon merentang minimum T dari
graf terhubung –berbobot G.

Masukan: graf-berbobot terhubung G = (V, E),
dengan /V/= n

Keluaran: pohon rentang minimum T = (V, E')

}

Deklarasi

  i, p, q, u, v : integer


Algoritma

   ( Asumsi: sisi-sisi dari graf sudah diurut
menaik

     berdasarkan bobotnya – dari bobot kecil
ke bobot

     besar)

  T ← {}

  while jumlah sisi T < n-1 do

    Pilih sisi (u,v) dari E yang bobotnya
terkecil

    if (u,v) tidak membentuk siklus di T
then

       T ← T ∪ {(u,v)}

    endif

  endfor
```

7. POHON BERAKAR

Suatu graf dinamakan pohon berarah bila arah rusuknya diabaikan dan suatu pohon berarah dinamakan pohon berakar (rooted tree) bila ada tepat satu simpul yang berderajat masuk 0, dan semua simpul lain berderajat masuk 1. simpul berderajat masuk 0 dinamakan akar, simpul berderajat keluar 0 dinamakan daun, sedangkan simpul yang berderajat masuk 1 tetapi derajat keluarnya tidak 0 disebut simpul cabang.

Terminologi pada Pohon Berakar



Gambar 7 : Pohon Beakar

1. **Anak (*child* atau *children*) dan Orangtua (*parent*)**

   b, *c*, dan *d* adalah anak-anak simpul *a*,
   a adalah orangtua dari anak-anak itu

2. **Lintasan (*path*)**

   Lintasan dari *a* ke *j* adalah *a*, *b*, *e*, *j*.

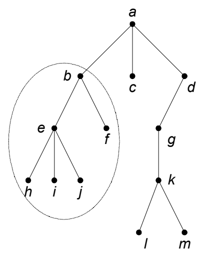
   Panjang lintasan dari *a* ke *j* adalah 3.

3. **Saudara kandung (*sibling*)**

   f adalah saudara kandung *e*,

   tetapi, *g* bukan saudara kandung *e*, karena orangtua mereka berbeda.

4. **Upapohon (*subtree*)**



Gambar 8: upa-pohon

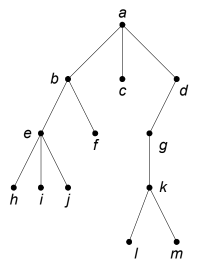
5. **Derajat (*degree*)**

   Derajat sebuah simpul adalah jumlah upapohon (atau jumlah anak) pada simpul tersebut.

   Derajat *a* adalah 3, derajat *b* adalah 2,
   Derajat *d* adalah satu dan derajat *c* adalah 0.

   Jadi, derajat yang dimaksudkan di sini adalah derajat-keluar.

   Derajat maksimum dari semua simpul merupakan derajat pohon itu sendiri. Pohon di atas berderajat 3



Gambar 9 : Pohon berderajat 3

6. **Daun (*leaf*)**

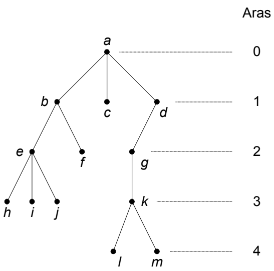
   Simpul yang berderajat nol (atau tidak mempunyai anak) disebut **daun**. Simpul *h*, *i*, *j*, *f*, *c*, *l*, dan *m* adalah daun.

   Simpul Dalam (*internal nodes*)

   Simpul yang mempunyai anak disebut **simpul**

dalam. Simpul *b*, *d*, *e*, *g*, dan *k* adalah simpul dalam.

7. **Aras (*level*) atau Tingkat**



Gambar 10 : Pohon beraras

8. **Tinggi (*height*) atau Kedalaman (*depth*)**

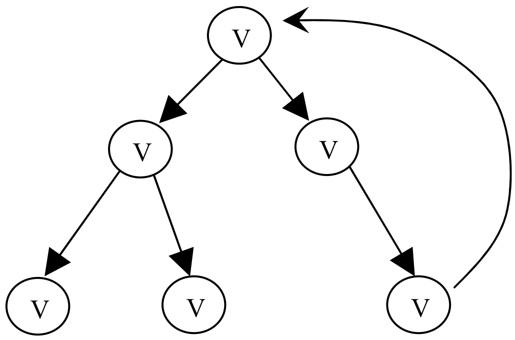
Aras maksimum dari suatu pohon disebut **tinggi** atau **kedalaman** pohon tersebut. Pohon di atas mempunyai tinggi 4.

Teorema Pohon Berakar

Teorema 1 (Teorema geometrik pohon)

Bila (T,V_0) adalah pohon berakar (T adalah relasi dan V_0 adalah akar) maka:

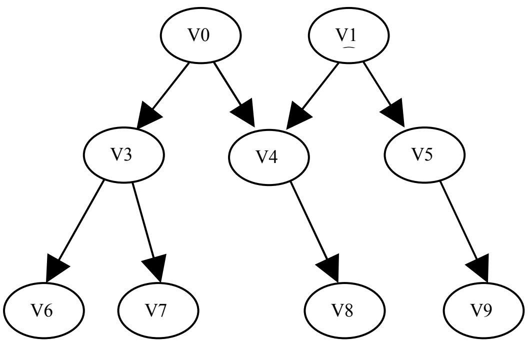
- Tidak ada siklus dalam T



Gambar 11 : Bukan Pohon Berakar

Gambar di atas bukanlah suatu pohon berakar karena ada suatu siklus dari V_0 - V_2 - V_3 kembali ke V_0.

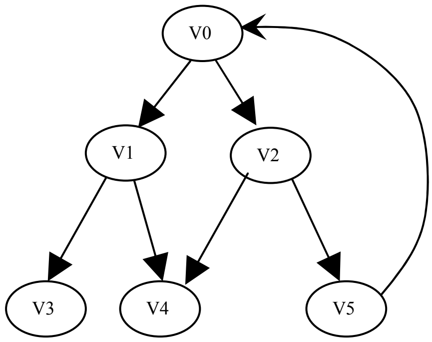
- V_0 merupakan satu-satunya akar dari T
  Tidak ada akar selain V_0 pada suatu pohon.



Gambar 12 : Bukan Pohon Berakar

Gambar di atas bukanlah pohon berakar karena mempunyai 2 akar pohon yaitu V_0 dan V_1.

- Tiap simpul di T kecuali V_0 memiliki derajat masuk satu sedangkan V0 berderajat masuk 0.

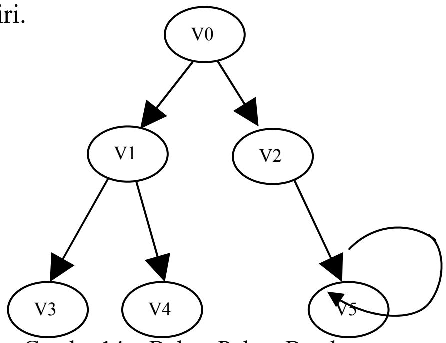


Gambar 13 : Bukan Pohon Berakar

Gambar di atas bukanlah suatu pohon berakar karena akarnya (V_0) berderajat masuk 1 dan ada simpul lain yang berderajat masuk 2 yaitu V_4.

Teorema 2

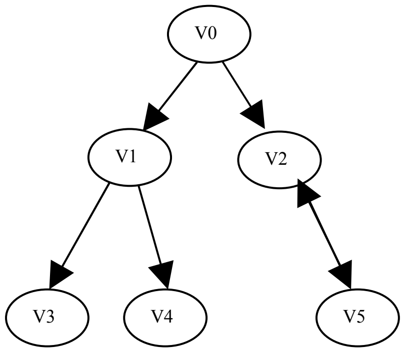
- Irreflexive
  Setiap simpul tidak berelasi dengan simpul itu sendiri.



Gambar14 : Bukan Pohon Berakar

Gambar di atas bukanlah pohon berakar karena simpul V5. Berelasi dengan dirinya sendiri.

- Asymmetric
  Relasi yang terjadi antar simpul bukanlah merupakan relasi bolak-balik (relasi satu arah).



Gambar 15 : Bukan Pohon Berakar

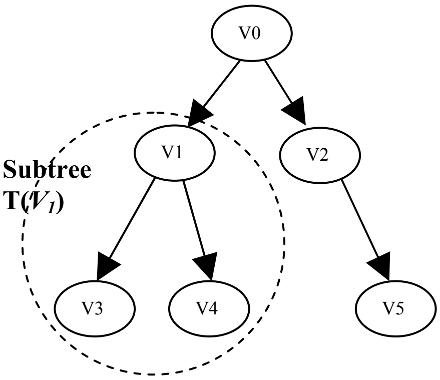
Gambar di atas bukan pohon berakar karena V_2 dan V_5 berelasi bolak – balik.

- Jika (a, b) $\in$ T dan (b, c) $\in$T, maka (a, c) $\notin$T
  Bila b berelasi dengan a dan bila c berelasi dengan b, maka c tidak memiliki relasi dengan a.

Teorema 3

Bila (T, v_o) adalah pohon berakar dan $v \in$ T maka :

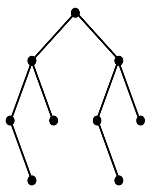
- T(v) juga pohon berakar dengan akar v . T(v) juga subtree dari T dengan awal v.



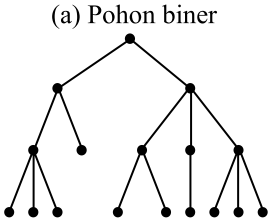
Gambar 16 : Pohon dengan SubPohon-nya

Sebuah pohon berakar yang simpul cabangnya memiliki paling banyak n anak (maksimal), disebut dengan pohon n-er *(n-ary tree).*Dan sebuah pohon n-er dikatakan teratur bila setiap simpul cabangnya tepat memiliki n anak.

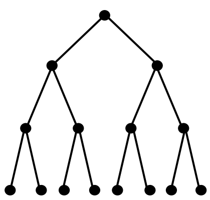
Contoh:



(a) Pohon biner



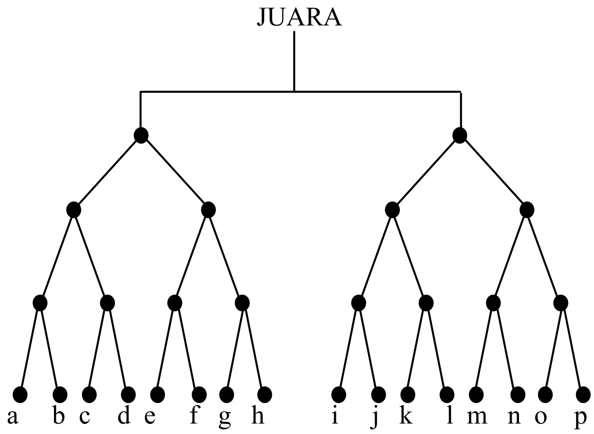
(b) Pohon terner



(c) Pohon biner teratur

Gambar 17 : Pohon n-er

Hubungan antara banyakya simpul cabang dengan banyaknya daun pada suatu pohon m-er teratur bisa kita lihat pada contoh berikut. Misalkan ada sebuah turnamen, pada setiap pertandingan menggunakan sistem gugur. Ada 16 klub peserta turnamen, sehingga pada akhir turnamen hanya tersisa satu tim yang menjadi juara. Bila kita tuangkan jadwal pertandingannya dalam bentuk grafik, ini merupakan contoh sebuah pohon biner teratur dimana setiap simpul cabang tepat memiliki 2 anak. Maka kita dapat menemukan bahwa jumlah pertandingan yang dilangsungkan adalah 15 pertandingan (satu lebih sedikit daripada jumlah klub peserta).



Gambar 18 : Diagram Kompetisi

Bila i menyatakan banyak simpul cabang, dan t menyatakan banyaknya daun, maka diperoleh hubungan :

$$i = t - 1 \qquad (2)$$

hasil ini dapat diperluas untuk pohon m-er teratur lainnya menjadi :

$$(m-1)\, i = t - 1 \qquad (3)$$

Contoh :

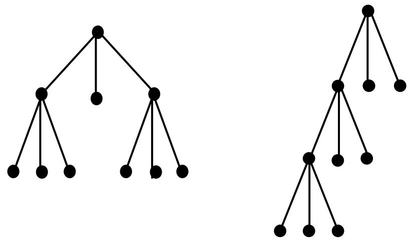
Bila sebuah komputer dapat menghitung 3 buah bilangan sekaligus dengan sebuah instruksi, berapa instruksikah yang dibutuhkan untuk menjumlahkan 7 buah bilangan ?

Jawab :

dengan menggunakan rumus yang sudah kita definisikan diatas, maka:

$$(m-1)\, i = t - 1 \text{ ; } m=3 \text{ dan } t=7 \qquad (4)$$
$$(3-1)\, i = 7 - 1 \qquad (5)$$
$$i = 3 \qquad (6)$$

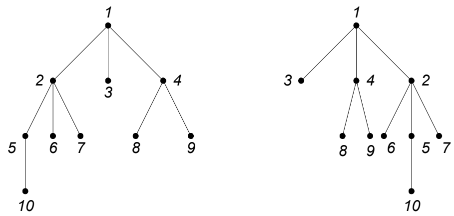
Ini berarti komputer harus melakukan 3 kali instruksi untuk menjumlahkan ketujuh bilangan tersebut dan dapat digambarkan sebagai berikut (dalam dua cara)



Gambar 19 : Cara perhitungan komputer

9. POHON TERURUT

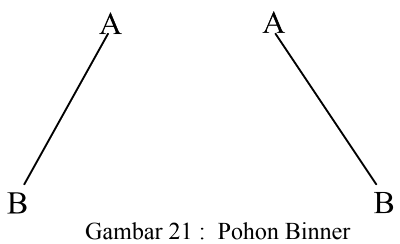
Pohon berakar yang urutan anak-anaknya penting disebut **pohon terurut** (*ordered tree*).



Gambar 20 : Pohon Terurut

10. POHON BINNER

Pohon biner merupakan jenis pohon m-er yang simpul cabangnya memiliki maksimal dua anak. Karena anak dari suatu cabang maksimal hanya dua, maka anak cangan ini dinamakan anak cabang kiri atau anak cabang kanan. Dalam pohon biner, cabang kiri dan kanan ini dibedakan (untuk pohon secara umum tidak). Bukan hny cabangnya saja, bahkan urutan cabang kiri atau vabang kanan pun dibedakan. Perhatikan gambar dibawah ini, kedua contoh ini merupakan pohon biner yang berbeda.

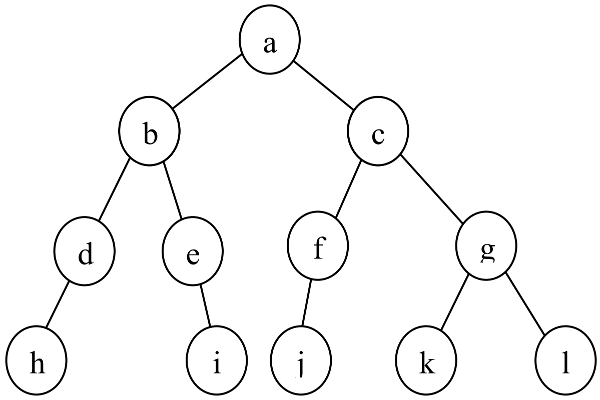


Gambar 21 : Pohon Binner

Penelusuran pohon biner

Penelusuran pohon berakar ada 3 macam :

1. Preorder
   - Kunjungi akar
   - Telusuri cabang kiri
   - Telusuri cabang kanan
2. Inorder
   - Telusuri cabang kiri
   - Kunjungi akar
   - Telusuri cabang kanan
3. Postorder
   - Telusuri cabang kiri
   - Telusuri cabang kanan
   - Kunjungi akar

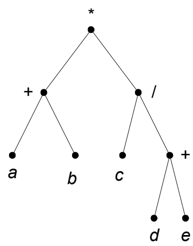


Gambar 22 : Pohon Binner

Preorder : a-b-d-h-e-i-c-f-i-g-k-l
Inorder : h-d-b-e-i-a-j-f-c-k-g-l
Postorder : h-d-i-e-b-j-f-k-l-g-c-a

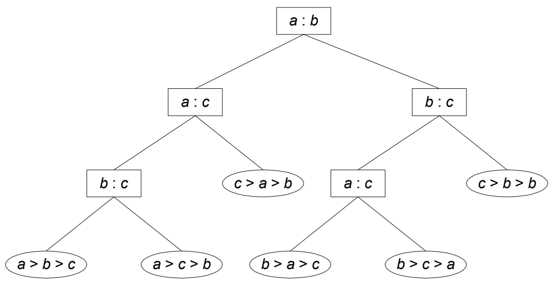
Terapan Pohon Biner

1. Pohon Ekspresi



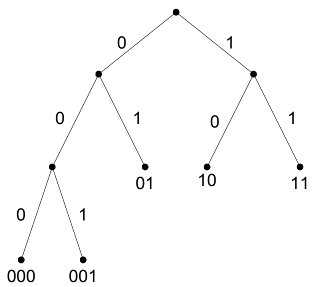
Gambar 23 : Pohon ekspresi dari $(a + b)*(c/(d + e))$

2. Pohon Keputusan



Gambar 24 : Pohon keputusan untuk mengurutkan 3 buah elemen

3. Kode Awalan



Gambar 25 : Pohon biner dari kode prefiks { 000, 001, 01, 10, 11}

4.

5. Kode Huffman

Tabel Kode ASCII

Simbol	Kode ASCII
A	01000001
B	01000010
C	01000011
D	01000100

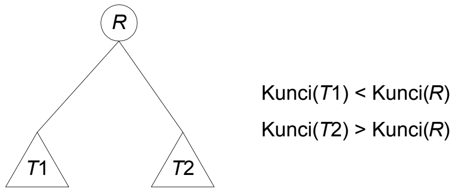
rangkaian bit untuk string ‘*ABACCDA’*:
01000001010000010010000010100000110100000
110100010001000001
atau 7 × 8 = 56 bit (7 *byte*).

Tabel 1. Kekerapan dan kode Huffman untuk *string ’ABACCDA’*

Simbol	Kekerapan	Peluang	Kode Huffman
A	3	3/7	0
B	1	1/7	110
C	2	2/7	10
D	1	1/7	111

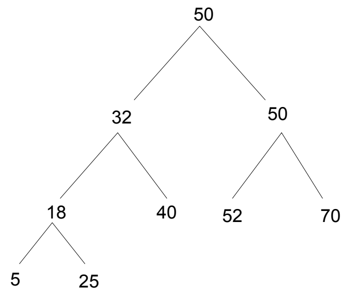
rangkaian bit untuk *’ABACCDA’*:
0110010101110
hanya 13 bit!

6. Pohon Pencarian Binner



Gambar 26 : pohon pencarian binner

Data : 50, 32, 18, 40, 60, 52, 5, 25, 70



Gambar 27 : Pohon pencarian Binner untuk data di atas

11. POHON BERLABEL

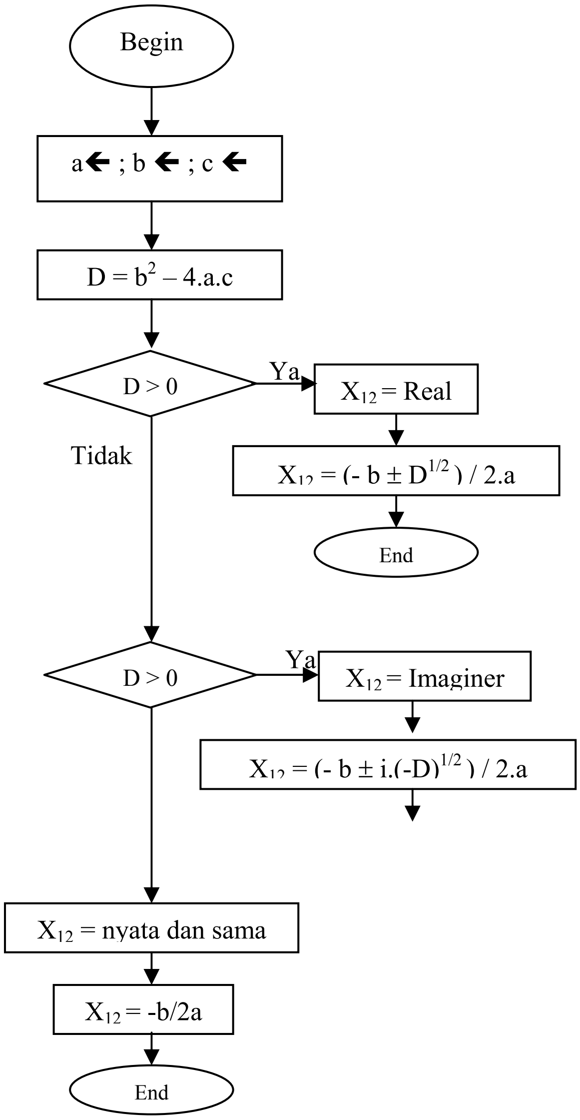
Pohon berlabel digunakan untuk mengindikasikan bahwa diagram tersebut digunakan untuk tujuan tertentu. Misalnya digunakan untuk merepresentasikan flowchart pada program, digunakan pada penggunaan senarai berantai pada C++, dan sebagainya. Label digunakan untuk memberi keterangan pada simpul, simpul tersebut mempunyai fungsi tertentu.

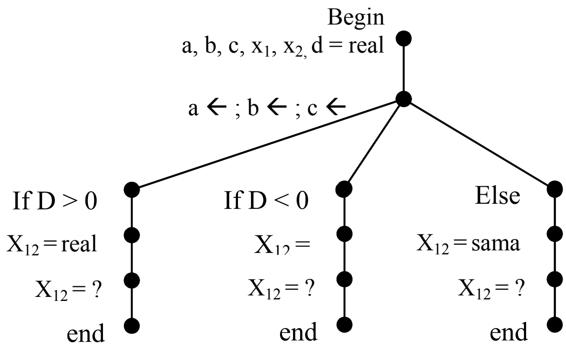
Aplikasi pada flowchart dari program pencarian dari akar – akar persamaan kuadrat, yang mendapat input a, b, c dimana ketiga variabel tersebut digunakan pada persamaan $ax^2 + bx + c = 0$. Pertama dilakukan pencarian parameter D untuk menentukan apakah hasilnya berupa bilangan real, imaginer atau real dan sama nilainya. Ketiga pilihan tersebut digunakan sebagai simpul.

Programnya :

- Mulai
- Pendeklarasian variabel
- a$\leftarrow$; b $\leftarrow$; c $\leftarrow$ (in
- put dari user)
- $D = b^2 – 4.a.c$
- if $D > 0$ then
  - akar akarnya nyata
  - $X_{12} = (- b \pm D^{1/2}) / 2.a$
  - selesai
- if $D < 0$ then
  - akar akarnya imaginer
  - $X_{12} = (- b \pm i.(-D)^{1/2}) / 2.a$
  - selesai
- else
  - akarnya sama
  - $X_{12} = - b / 2a$

selesai





Gambar 28 : Diagram Program

12. KESIMPULAN

1. **Pohon** adalah graf tak-berarah terhubung yang tidak mengandung sirkuit
2. **Pohon** memiliki sifat-sifat tertentu
3. Pohon dapat trebentuk dari n-er simpul
4. Pohon biner merupakan jenis pohon m-er yang simpul cabangnya memiliki maksimal dua anak.

DAFTAR REFERENSI

[1] Munir, Rinaldi. (2006). Diktat Kuliah IF2153 Matematika Diskrit. Program Studi Teknik Informatika, Institut Teknologi Bandung.

[2] http://www.informatika.org/~rinaldi/Matdis/2006-2007/Makalah/
Diakses tanggal 29 Desember 2007, pukul 14.05 WIB

[3] http://www.mti.ugm.ac.id/~adji/courses/resources/lectures/DiscMath/
Diakses tanggal 29 Desember 2007, pukul 14.30 WIB