

Struktur Data Pohon dan Graf

Informatika Kelas IX - Kurikulum Merdeka

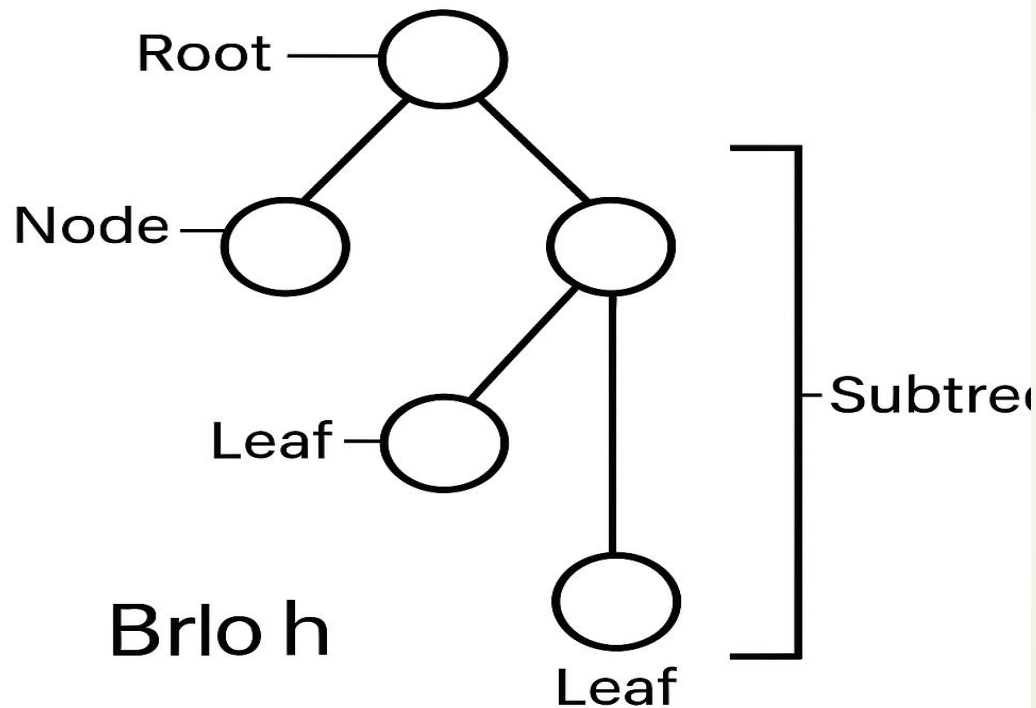
EKO WAHYUDI



Tujuan Pembelajaran

- Memahami konsep dasar struktur data pohon dan graf.
- Mengidentifikasi komponen pohon dan graf.
- Menjelaskan jenis dan traversal pohon.
- Menggunakan struktur data graf dalam kehidupan sehari-hari.
- Mewakili struktur pohon/graf dalam diagram sederhana.

Struktur Data Pohon





Struktur Data Pohon

Struktur hierarkis dengan node dan edge.

- Root : node paling atas.
- Node : elemen dalam pohon.
- Leaf : node tanpa anak.
- Subtree : bagian dari pohon yang juga merupakan pohon.



Jenis dan Traversal Pohon



Jenis Pohon:

- Binary Tree, BST, AVL Tree, Heap



Traversal:

- Preorder: Root \rightarrow Left \rightarrow Right
- Inorder: Left \rightarrow Root \rightarrow Right
- Postorder: Left \rightarrow Right \rightarrow Root

Penerapan Struktur Pohon

- Sistem File: folder dan file.
- Struktur Organisasi: CEO, manajer, karyawan.
- Ekspresi matematika: $(3+4)*5$ sebagai pohon.
- Kompilator: struktur sintaksis.

Konsep Pohon Silsilah Keluarga

Konsep Pohon Silsilah Keluarga

Dalam ilmu komputer, pohon silsilah keluarga bisa diwakili oleh struktur data pohon (tree). Setiap **simpul (node)** dalam pohon mewakili seseorang, dan **cabang (edge)** menunjukkan hubungan orang tua-anak. Biasanya, simpul orang tua berada di atas simpul anak-anaknya.

Berikut adalah karakteristik umum dari pohon silsilah keluarga sebagai struktur data:

- **Akar (Root):** Biasanya adalah individu tertua atau nenek moyang yang menjadi titik awal silsilah.
- **Simpul (Node):** Setiap simpul merepresentasikan individu dalam keluarga.
- **Orang Tua (Parent):** Simpul yang berada satu tingkat di atas simpul lainnya (misalnya, ayah atau ibu dari seseorang).
- **Anak (Child):** Simpul yang berada satu tingkat di bawah simpul lainnya (misalnya, anak dari seseorang).
- **Saudara (Sibling):** Simpul yang memiliki orang tua yang sama.
- **Daun (Leaf Node):** Simpul yang tidak memiliki anak, biasanya individu termuda dalam jalur silsilah tersebut.

Representasi dalam Pseudo-Code atau Struktur Data (Objek/Kelas):

```
class Individu:
    def __init__(self, nama, jenis_kelamin, tanggal_lahir=None):
        self.nama = nama
        self.jenis_kelamin = jenis_kelamin
        self.tanggal_lahir = tanggal_lahir
        self.pasangan = None
        self.anak_anak = [] # Daftar objek Individu

    def tambah_pasangan(self, pasangan):
        self.pasangan = pasangan
        pasangan.pasangan = self # Hubungan dua arah

    def tambah_anak(self, anak):
        self.anak_anak.append(anak)
```

```
# Membuat objek untuk setiap individu
adam = Individu("Adam", "Laki-laki")
hawa = Individu("Hawa", "Perempuan")
budi = Individu("Budi", "Laki-laki")
citra = Individu("Citra", "Perempuan")
dini = Individu("Dini", "Perempuan") # Asumsi Dini bukan keturunan langsung dari
galih = Individu("Galih", "Laki-laki") # Asumsi Galih bukan keturunan langsung dari
eko = Individu("Eko", "Laki-laki")
fani = Individu("Fani", "Perempuan")
hadi = Individu("Hadi", "Laki-laki")
indah = Individu("Indah", "Perempuan") # Asumsi Indah bukan keturunan langsung
joko = Individu("Joko", "Laki-laki")
kiki = Individu("Kiki", "Perempuan")
```


Representasi dalam Pseudo-Code atau Struktur Data (Objek/Kelas):

```
# Membangun hubungan dalam pohon
# Generasi 1 & 2
adam.tambah_pasangan(hawa)
adam.tambah_anak(budi)
adam.tambah_anak(citra)
hawa.tambah_anak(budi) # Hawa juga punya anak Budi dan Citra
hawa.tambah_anak(citra)

# Generasi 2 & 3
budi.tambah_pasangan(dini)
budi.tambah_anak(eko)
budi.tambah_anak(fani)
dini.tambah_anak(eko)
dini.tambah_anak(fani)

citra.tambah_pasangan(galih)
citra.tambah_anak(hadi)
galih.tambah_anak(hadi)
```

```
# Generasi 3 & 4
eko.tambah_pasangan(indah)
eko.tambah_anak(joko)
eko.tambah_anak(kiki)
indah.tambah_anak(joko)
indah.tambah_anak(kiki)

# Contoh penggunaan:
print(f"{adam.nama} dan {adam.pasangan.nama} memiliki anak:")
for anak in adam.anak_anak:
    print(f"- {anak.nama}")

print(f"\n{budi.nama} memiliki anak:")
for anak in budi.anak_anak:
    print(f"- {anak.nama}")

print(f"\n{eko.nama} memiliki anak:")
for anak in eko.anak_anak:
    print(f"- {anak.nama}")
```

Contoh : struktur silsilah keluarga

Nama-nama dalam Keluarga:

•Generasi 1 (Akar):

- Adam (Laki-laki)
- Hawa (Perempuan)

•Generasi 2 (Anak dari Adam & Hawa)

- Budi (Laki-laki)
- Citra (Perempuan)

•Generasi 3 (Anak dari Budi & Dini):

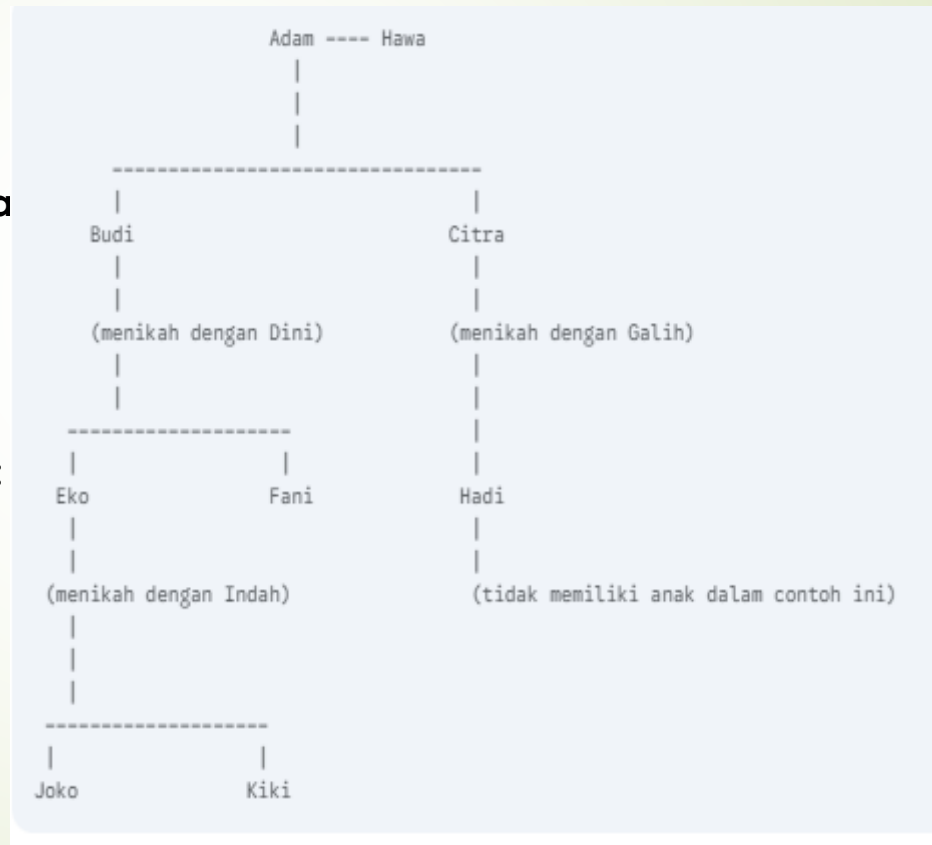
- Eko (Laki-laki)
- Fani (Perempuan)

•Generasi 3 (Anak dari Citra & Galih):

- Hadi (Laki-laki)

•Generasi 4 (Anak dari Eko & Indah):

- Joko (Laki-laki)
- Kiki (Perempuan)

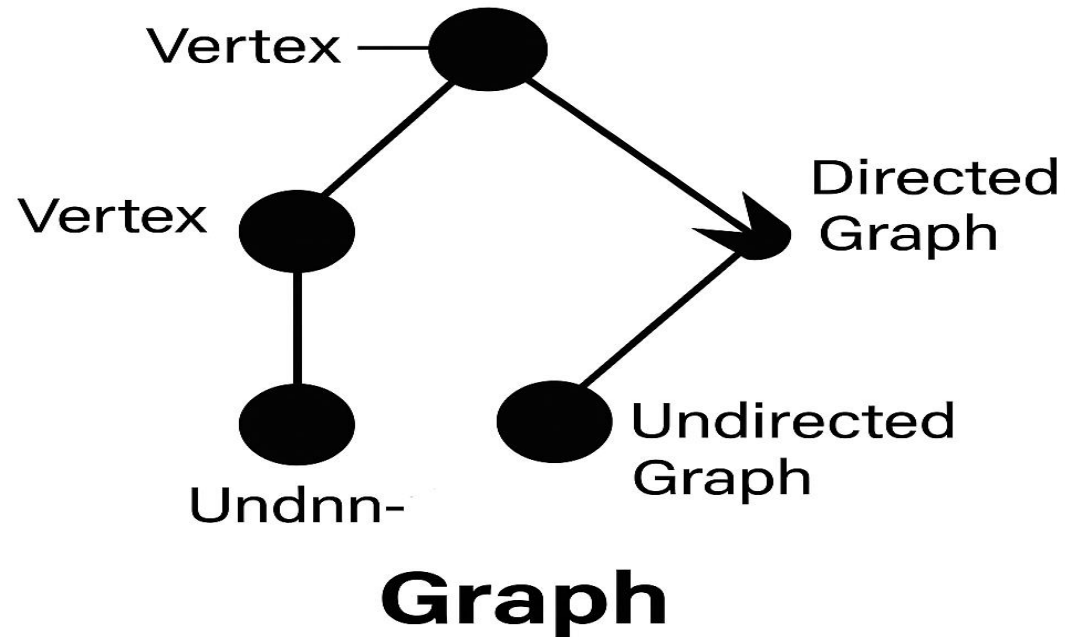




Kelebihan Representasi Pohon Silsilah

- **Visualisasi Jelas:** Memudahkan pemahaman hubungan antar individu.
- **Navigasi Efisien:** Mudah untuk melacak garis keturunan (dari orang tua ke anak) atau menemukan leluhur (dari anak ke orang tua, jika referensi dua arah disediakan).
- **Pencarian:** Memungkinkan pencarian individu, pasangan, atau anak-anak dengan relatif mudah.
- **Ekspansi Fleksibel:** Mudah untuk menambahkan anggota keluarga baru ke dalam silsilah.

Struktur Data graf





Struktur Data Graf

- Terdiri dari vertex (simpul) dan edge (tepi).
- Graf Berarah vs Tak Berarah.
- Graf Berbobot.
- Representasi: Adjacency Matrix & List.



Algoritma dalam Graf

- BFS: mengunjungi simpul setingkat terlebih dahulu.
- DFS: menjelajah sedalam mungkin dahulu.
- Dijkstra: pencarian rute terpendek.
- Deteksi siklus: mencari apakah ada loop dalam graf.



Penerapan Struktur Graf

- Jaringan Jalan: simpul = persimpangan, edge = jalan.
- Jaringan Sosial: simpul = individu, edge = hubungan.
- Jaringan Komputer: simpul = perangkat, edge = koneksi.



Aktivitas dan Penilaian



Aktivitas:

- Menggambar struktur folder dan organisasi.
- Simulasi traversal.



Penilaian:

- Kognitif: soal & LKPD.
- Psikomotorik: tugas visualisasi.
- Afektif: kerja kelompok.



Sumber Belajar dan Tugas



Sumber:

- Modul Kurikulum Merdeka.
- GeeksforGeeks, Shutterstock, YouTube.
- draw.io, Canva.



Tugas:

- Buat pohon keluarga.
- Visualisasi graf jaringan sosial sederhana.



Konsep Graf untuk Rute Jalan

- Dalam ilmu komputer, graf sangat cocok untuk memodelkan jaringan jalan. Setiap **simpul (node)** dalam graf akan merepresentasikan lokasi penting (misalnya, persimpangan jalan, gedung, atau titik minat), dan setiap **sisi (edge)** akan merepresentasikan jalan yang menghubungkan dua lokasi tersebut.



Berikut adalah karakteristik umum graf yang relevan untuk rute jalan:

- **Simpul (Node/Vertex):** Titik-titik lokasi. Dalam kasus ini, bisa berupa persimpangan jalan, gerbang sekolah, minimarket terdekat, atau halte bus.
- **Sisi (Edge):** Koneksi antar simpul. Ini adalah representasi dari jalan itu sendiri. Sisi bisa:
 - **Berbobot (Weighted):** Setiap sisi memiliki "bobot" atau nilai yang terkait dengannya, seperti jarak (dalam meter atau kilometer), waktu tempuh (dalam menit), atau biaya.
 - **Berarah (Directed):** Jika jalan adalah satu arah, maka sisi hanya bisa dilalui dari satu simpul ke simpul lainnya. Jika jalan dua arah, sisi bisa dianggap tidak berarah (undirected) atau direpresentasikan dengan dua sisi berarah yang berlawanan.

Contoh Graf Rute Jalan di Sekitar SMP Negeri 236 Jakarta

Titik-titik (Simpul/Node) Penting di Sekitar SMPN 236 Jakarta (Komplek PIK, Cakung):

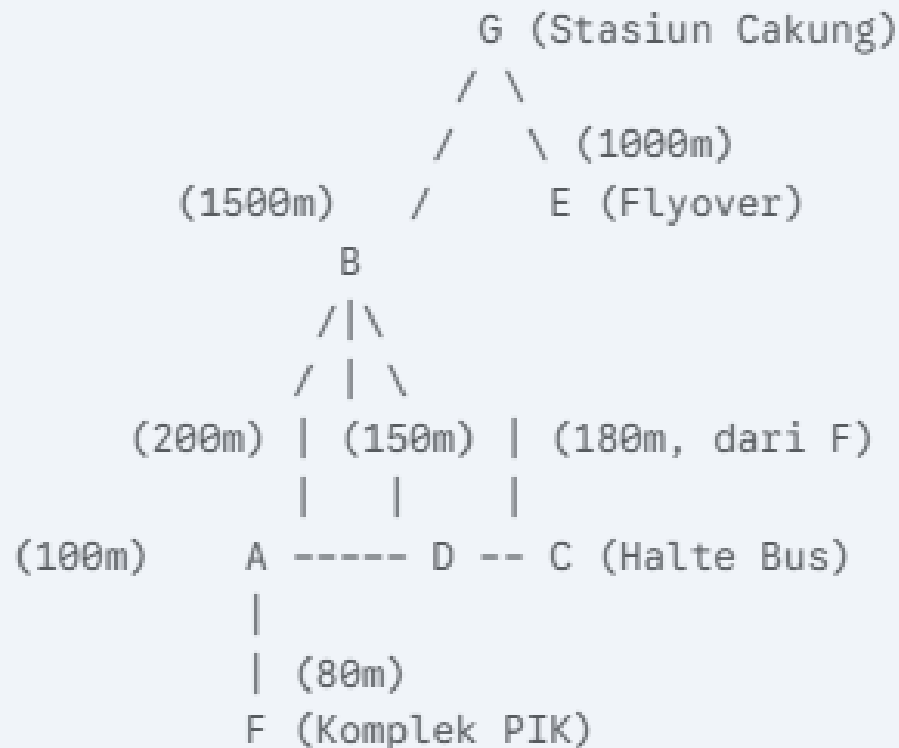
Kita akan menamai simpul-simpul ini dengan kode yang mudah diingat:

- **A:** Gerbang Utama SMPN 236 Jakarta
- **B:** Persimpangan Jl. Raya Penggilingan - Jl. Swadaya (arah KUA Cakung)
- **C:** Halte Bus TransJakarta Penggilingan (atau halte terdekat di Jl. Raya Penggilingan)
- **D:** Minimarket (Indomaret/Alfamart) di Jl. Raya Penggilingan
- **E:** Jembatan Layang (Flyover) Penggilingan/Persimpangan besar menuju Bintara/Walang
- **F:** Perumahan Komplek PIK (Jalan masuk ke dalam kompleks)
- **G:** Stasiun Cakung (titik akses transportasi KRL terdekat)

Jalan (Sisi/Edge) dan Bobot (Estimasi Jarak dalam Meter/Waktu dalam Menit):

- Kita akan asumsikan jalan dua arah untuk sebagian besar, kecuali disebutkan satu arah. Jarak dan waktu di sini adalah perkiraan.
- **A -- B:** Jl. Raya Penggilingan (depan sekolah menuju persimpangan Jl. Swadaya). **Jarak: 200m / Waktu: 3 menit.**
- **A -- D:** Jl. Raya Penggilingan (depan sekolah menuju minimarket). **Jarak: 100m / Waktu: 2 menit.**
- **A -- F:** Jalan masuk ke dalam Komplek PIK dari gerbang sekolah. Ini mungkin jalan internal kompleks. **Jarak: 80m / Waktu: 1 menit.**
- **B -- C:** Jl. Raya Penggilingan (dari persimpangan menuju halte bus). **Jarak: 150m / Waktu: 2 menit.**
- **B -- G:** Jl. Swadaya menuju Stasiun Cakung. Jalan ini cukup panjang dan ramai. **Jarak: 1500m / Waktu: 15-20 menit (jalan kaki), 5-7 menit (kendaraan).**
- **C -- E:** Jl. Raya Penggilingan (dari halte bus menuju Flyover Penggilingan). **Jarak: 300m / Waktu: 4 menit.**
- **D -- C:** Jl. Raya Penggilingan (dari minimarket menuju halte bus). **Jarak: 50m / Waktu: 1 menit.**
- **E -- G:** Dari Flyover Penggilingan menuju Stasiun Cakung (melalui jalan besar). Ini bisa jadi jalan satu arah tergantung jalur yang diambil di flyover. **Jarak: 1000m / Waktu: 10-12 menit.**
- **F -- B:** Jalan internal Komplek PIK yang bisa tembus ke Jl. Swadaya atau kembali ke Jl. Raya Penggilingan. Bisa jadi jalan satu arah dari F ke B. **Jarak: 180m / Waktu: 3 menit.**

Visualisasi Graf (Teks/Diagram Sederhana):



Catatan: Panah menunjukkan arah jika satu arah, garis tanpa panah berarti dua arah. Bobot bisa diubah antara jarak atau waktu sesuai kebutuhan.

Representasi dalam Pseudo-Code (Adjacency List):

```
# Setiap kunci adalah simpul (lokasi), dan nilai adalah daftar tuple (simpul_tujuan, jarak/waktu)

graf_rute_smp236_rev = {
    "A": [("B", "200m"), ("D", "100m"), ("F", "80m")],
    "B": [("A", "200m"), ("C", "150m"), ("G", "1500m")],
    "C": [("B", "150m"), ("D", "50m"), ("E", "300m")],
    "D": [("A", "100m"), ("C", "50m")],
    "E": [("C", "300m"), ("G", "1000m")],
    "F": [("A", "80m"), ("B", "180m)], # F ke B diasumsikan satu arah di diagram,
    "G": [("B", "1500m"), ("E", "1000m")]
}

# Contoh penggunaan:
print("Rute dari Gerbang Utama SMPN 236 (A):")
for tujuan, bobot in graf_rute_smp236_rev["A"]:
    print(f"- Menuju {tujuan} dengan jarak/waktu {bobot}")

print("\nRute dari Stasiun Cakung (G):")
for tujuan, bobot in graf_rute_smp236_rev["G"]:
    print(f"- Menuju {tujuan} dengan jarak/waktu {bobot}")
```