Nama : I Made Nobel Saputra

NIM : 2301020053

**6. a) Backward chaining untuk goal masalah\_isp pohon penelusuran lengkap + langkah detail**

Kita punya aturan relevan untuk **masalah\_isp**:

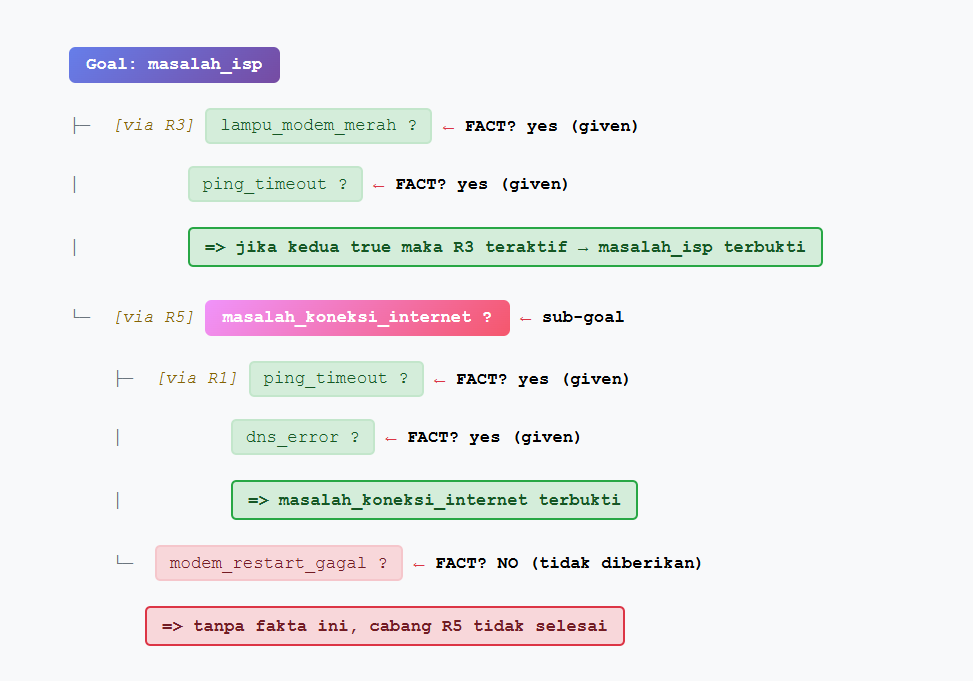
* **R3:** IF **lampu\_modem\_merah** AND **ping\_timeout** THEN **masalah\_isp**
* **R5:** IF **masalah\_koneksi\_internet** AND **modem\_restart\_gagal** THEN **masalah\_isp**

Facts awal:

* **ping\_timeout = True**
* **dns\_error = True**
* **lampu\_modem\_merah = True**

Strategi backward chaining: mulai dari goal **masalah\_isp**, cari aturan yang menghasilkan goal itu, lalu bukti pra-kondisi aturan itu (sub-goal) secara rekursif sampai mencapai fakta

Pohon penelusuran Gambar :



Goal awal: **masalah\_isp**

Cari aturan yang memberi **masalah\_isp** → ditemukan R3 dan R5.

Periksa R3:

* Butuh **lampu\_modem\_merah** dan **ping\_timeout**
* Cek fakta: **lampu\_modem\_merah = True** (diketahui)
* Cek fakta: **ping\_timeout = True** (diketahui)
* Karena kedua pra-kondisi terpenuhi, R3 membenarkan **masalah\_isp = True**
* **Selesai goal terbukti**

Periksa R5:

* Butuh **masalah\_koneksi\_internet** dan **modem\_restart\_gagal**
* **masalah\_koneksi\_internet** dapat dibuktikan oleh R1 karena **ping\_timeout** & **dns\_error** keduanya True → jadi **masalah\_koneksi\_internet = True**
* **modem\_restart\_gagal** **tidak ada** di facts awal → cabang R5 tidak bisa lengkap, kecuali fakta ini diberikan

Kesimpulan backward: **masalah\_isp** terbukti (paling cepat lewat R3)

# **b) Kedalaman maksimum penelusuran yang terjadi penjelasan**

Definisi praktis: hitung kedalaman sebagai jumlah **level sub-goal/aturan** dari root goal sampai mencapai fakta (berapa banyak langkah inferensi berurutan).

* Cabang R3: **masalah\_isp** → R3 → {**lampu modem merah, ping timeout**} (langsung fakta)  
   => kedalaman cabang ini = **2 level** (1 aturan, lalu fakta)
* Cabang R5 via R1: **masalah\_isp** → R5 → **masalah\_koneksi\_internet** → R1 → {**ping\_timeout,dns\_error**}  
   => kedalaman cabang ini = **3–4 level** tergantung cara penghitungan:
  + Jika mengukur jumlah aturan yang harus diterapkan berurutan: R1 lalu R5 = **2 aturan** (kedalaman 2 dalam hal rule-chaining)
  + Jika mengukur node-level (goal → aturan → sub-goal → aturan → fakta) maka hitungan mencapai **4 level**

Untuk jawaban singkat dan umum:  
 **Kedalaman maksimum penelusuran = 2 aturan berurutan** (yaitu R1 → R5) — atau jika dihitung level node sampai fakta, sampai **4 level**

Penjelasan: R3 adalah jalur pendek (1 aturan), R5 membutuhkan pembuktian **masalah\_koneksi\_internet** lewat R1 (jadi 2 aturan berantai)

# **c) Forward chaining dari facts yang sama kesimpulan yang didapat & perbandingan efisiensi**

Mulai dengan facts: **{ping\_timeout, dns\_error, lampu\_modem\_merah}**

Langkah forward chaining (aplikasi aturan yang pra-kondisinya terpenuhi):

1. **R1**: **ping\_timeout AND dns\_error** → kedua-duanya fakta → **tambahkan** **masalah\_koneksi\_internet**
2. **R3**: **lampu\_modem\_merah AND ping\_timeout** → kedua fakta ada → **tambahkan** **masalah\_isp**
3. **R2**: butuh **ping\_success** → tidak terpenuhi → tidak aktif
4. **R4**: butuh **ip\_conflict\_detected** → tidak ada → tidak aktif
5. **R5**: butuh **masalah\_koneksi\_internet** (sudah dihasilkan oleh R1) **dan** **modem\_restart\_gagal** (tidak ada) → tidak aktif
6. **R6**: butuh **slow\_response\_time** AND **ping\_success** → tidak terpenuhi.

Kesimpulan forward chaining dengan facts yang diberikan:

* **masalah\_koneksi\_internet = True** (dari R1)
* **masalah\_isp = True** (dari R3)

Perbandingan efisiensi untuk kasus ini:

* **Backward chaining** (goal-driven): memulai dari **masalah\_isp** dan langsung menemukan R3 yang membutuhkan fakta yang **sudah** ada → proses cepat, hanya memeriksa sedikit aturan. Jika sistem berhenti pada temuan pertama, sangat efisien.
* **Forward chaining** (data-driven): mengeksekusi semua aturan yang pra-kondisinya terpenuhi dan menghasilkan semua fakta turunan: juga menemukan **masalah\_isp** dan **masalah\_koneksi\_internet**. Untuk dataset kecil ini keduanya cepat, tapi forward chaining melakukan R1 juga walaupun R3 sudah cukup untuk membuktikan **masalah\_isp**. Jadi forward chaining mungkin melakukan inferensi ekstra yang tidak perlu jika tujuan hanya memverifikasi satu goal

**Kedalaman maksimum penelusuran = 2 aturan berurutan** (yaitu R1 → R5) — atau jika dihitung level node sampai fakta, sampai **4 level**.

Penjelasan: R3 adalah jalur pendek (1 aturan), R5 membutuhkan pembuktian **masalah\_koneksi\_internet** lewat R1 (jadi 2 aturan berantai)

# **d) 3 kelebihan dan 3 kekurangan backward chaining dibanding forward chaining dalam konteks troubleshooting ini**

Kelebihan backward chaining:

1. **Goal-directed / fokus** bekerja mundur dari masalah yang ingin dipecahkan (mis. **masalah\_isp**), sehingga biasanya menghindari inferensi yang tidak relevan
2. **Hemat sumber daya bila goal tunggal** hanya aturan yang relevan dengan goal yang dieksplorasi, cocok untuk troubleshooting yang bertanya "apa penyebab X?"
3. **Responsif untuk pertanyaan diagnostik interaktif** pengguna bisa meminta bukti untuk satu hipotesis, dan sistem langsung menelusuri bukti yang diperlukan

Kekurangan backward chaining:

1. **Tidak menemukan konsekuensi lain secara otomatis** jika ingin mengetahui semua masalah yang mungkin muncul dari facts, backward chaining harus dijalankan berulang untuk tiap goal
2. **Bergantung pada definisi goal** jika user tidak tahu goal yang tepat atau memberi goal yang keliru, backward chaining tidak membantu menemukan masalah tersembunyi.
3. **Rentan ke backtracking besar** jika banyak aturan alternatif dan banyak sub-goal yang belum diketahui, traversal bisa menjadi kompleks dan memerlukan banyak backtracking (terutama jika banyak cabang yang hampir cocok)

Perbandingan ringkas: backward bagus saat fokus diagnosa target, forward bagus saat ingin eksplorasi menyeluruh dari fakta ke banyak kemungkinan kesimpulan

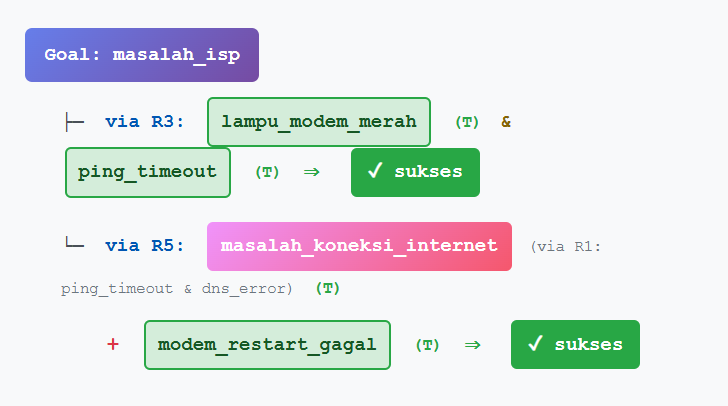
# **e) Kalau ditambahkan fact modem\_restart\_gagal = True, bagaimana jalur inferensi untuk goal masalah\_isp berubah?**

Facts jadi: **{ping\_timeout, dns\_error, lampu\_modem\_merah, modem\_restart\_gagal}**

Perubahan inferensi:

1. **Jalur via R3** tetap valid: **lampu\_modem\_merah + ping\_timeout** ⇒ **masalah\_isp**
2. **Jalur via R5** kini juga dapat lengkap:  
   * **masalah\_koneksi\_internet** diperoleh dari R1 (**ping\_timeout** & **dns\_error**) ⇒ **masalah\_koneksi\_internet = True**
   * **modem\_restart\_gagal = True** (baru ditambahkan) ⇒ kedua pra-kondisi R5 terpenuhi ⇒ R5 juga membenarkan **masalah\_isp**.
3. Hasil: **masalah\_isp** memiliki **dua bukti independen** (R3 dan R5). Sistem bisa menyimpulkan **masalah\_isp = True** dari kedua jalur, ini memberikan redundansi bukti (berguna untuk kepercayaan diagnosis

**Ilustrasi :**



7. **a) Derajat Keanggotaan**

**Kinerja = 65**

* **Buruk (0–50, trapesium):** nilai 65 sudah di luar batas, jadi **μ = 0**
* **Cukup (40–70, segitiga puncak 55):** berada di sisi kanan puncak, maka  
   μ = (70 – 65) / (70 – 55) = 5 / 15 = **0,33**
* **Baik (60–100, trapesium):** berada di sisi naik, maka  
   μ = (65 – 60) / (80 – 60) = 5 / 20 = **0,25**

**Absensi = 85%**

* **Rendah (0–70, trapesium):** di luar batas, jadi **μ = 0**
* **Sedang (60–90, segitiga puncak 75):** di sisi kanan puncak, maka  
   μ = (90 – 85) / (90 – 75) = 5 / 15 = **0,33**
* **Tinggi (80–100, trapesium):** di sisi naik, maka  
   μ = (85 – 80) / (90 – 80) = 5 / 10 = **0,5**

**Hasil akhir input:**

* Kinerja → Buruk = 0, Cukup = 0,33, Baik = 0,25
* Absensi → Rendah = 0, Sedang = 0,33, Tinggi = 0,5

### **b) Rule Aktif dan α-predikat**

Menggunakan operator **MIN (AND)**:

| **Rule** | **Aturan** | **α (MIN)** | **Output** |
| --- | --- | --- | --- |
| R1 | Baik ∧ Tinggi | min(0,25, 0,5) = **0,25** | Besar |
| R2 | Baik ∧ Sedang | min(0,25, 0,33) = **0,25** | Sedang |
| R3 | Cukup ∧ Tinggi | min(0,33, 0,5) = **0,33** | Sedang |
| R4 | Cukup ∧ Sedang | min(0,33, 0,33) = **0,33** | Kecil |
| R5 | Buruk | 0 | Kecil |

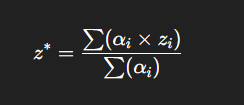
**Rule aktif:** R1, R2, R3, R4  
 **α aktif:** 0,25; 0,25; 0,33; 0,33

### **c) Defuzzifikasi (Metode Centroid)**

Gunakan nilai tengah:

* Kecil = 2,5 juta
* Sedang = 6 juta
* Besar = 11 juta

Rumus:



Perhitungan:  
 = (0,25×11 + 0,25×6 + 0,33×6 + 0,33×2,5) / (0,25 + 0,25 + 0,33 + 0,33)  
 = (2,75 + 1,5 + 2 + 0,83) / 1,16  
 = 7,08 / 1,16 = **6,07 juta rupiah**

**Hasil akhir (Centroid):** 6,07 juta rupiah

### **d) Mean of Maximum (MOM)**

Ambil α tertinggi dari masing-masing output:

* Kecil = 0,33
* Sedang = 0,33
* Besar = 0,25  
   Nilai maksimum α = 0,33 → dua kategori aktif (Kecil & Sedang)

Maka:  
 z = (2,5 + 6) / 2 = **4,25 juta rupiah**

**Hasil MOM:** 4,25 juta rupiah

**Perbandingan:**Metode **Centroid** lebih adil karena memperhitungkan semua rule dan menghasilkan nilai proporsional (kontinu)  
Sedangkan **MOM** cenderung kaku karena hanya mengambil puncak tertinggi, sehingga hasilnya lebih rendah dan kurang mencerminkan kontribusi keseluruhan kondisi

### **e) Cara Membuat Sistem Lebih Ketat**

1. **Naikkan batas fungsi “Baik”** misalnya dari 60–100 menjadi 70–100 agar hanya karyawan dengan kinerja sangat tinggi yang masuk kategori “Baik”
2. **Persempit area “Absensi Tinggi”** ubah dari 80–100 menjadi 90–100 supaya hanya karyawan dengan kehadiran hampir sempurna yang memicu bonus besar
3. (Opsional) Tambahkan rule baru seperti:  
    *IF Absensi < 80 THEN Bonus tidak lebih dari Sedang*

Dengan cara ini, sistem tidak mudah memberikan “Bonus Besar” kecuali performa dan kehadiran karyawan benar-benar tinggi

8. **a) Pengertian CF pakar dan CF user singkat & jelas**

* **CF pakar (CF(H))**: keyakinan (confidence) nilai yang diberikan oleh pakar pada hubungan antara *gejala (E)* dan *hipotesis/penyakit (H)* dalam basis pengetahuan. Nilainya berkisar −1 sampai +1 (di soal ini 0–1)
* **CF user (CF(E))**: tingkat keyakinan yang dilaporkan oleh pengguna/petani terhadap keberadaan gejala di lapangan. Juga berada pada skala 0–1

**Perbedaan utama:** CF pakar adalah tingkat kepercayaan relasional (seberapa kuat gejala mendukung hipotesis menurut pakar). CF user adalah seberapa yakin data/observasi itu benar pada kasus nyata. Dalam perhitungan CF(H,E) keduanya dikalikan untuk mendapatkan kontribusi nyata dari gejala itu terhadap hipotesis

## **b) Hitung CF(H,E) untuk setiap gejala (CF(H,E) = CF(H) × CF(E))**

Data dan perhitungan (digit-per-digit):

1. **daun\_menguning\_bertahap**CF(H) = 0.7 ; CF(user) = 0.8  
   CF(H,E) = 0.7 × 0.8 = 0.56
2. **pembuluh\_batang\_coklat**CF(H) = 0.9 ; CF(user) = 0.7  
   CF(H,E) = 0.9 × 0.7 = 0.63
3. **tanaman\_layu\_siang\_hari**CF(H) = 0.6 ; CF(user) = 0.9  
   CF(H,E) = 0.6 × 0.9 = 0.54
4. **pertumbuhan\_terhambat**CF(H) = 0.5 ; CF(user) = 0.6  
   CF(H,E) = 0.5 × 0.6 = 0.30

**Ringkasan kontribusi per-gejala:** 0.56, 0.63, 0.54, 0.30

## **c) Kombinasi CF gejala → CF final untuk “Layu Fusarium”**

Rumus kombinasi berurutan (dua nilai positif):



Kombinasikan sesuai urutan gejala yang diberikan:

1. Mulai dengan CF₁ = 0.56 (setelah gejala 1).  
    CF\_comb\_1 = 0.56
2. Gabungkan CF₂ = 0.63:



Hitung: 0.63 × 0.44 = 0.2772  
 → CF₁₂ = 0.56 + 0.2772 = **0.8372**

1. Gabungkan CF₃ = 0.54:



Hitung: 0.54 × 0.1628 = 0.087912  
 → CF₁₂₃ = 0.8372 + 0.087912 = **0.925112**

1. Gabungkan CF₄ = 0.30:



## **d) Keputusan berdasarkan threshold & rekomendasi tindakan**

Threshold yang diberikan:

* **CF ≥ 0.7** → diagnosis **diterima** (yakin)
* **CF ≥ 0.5** → diagnosis **mungkin** (perlu investigasi)

**Hasil:** CF\_final ≈ **0.9476** ≥ 0.7 → **Diagnosis: Layu Fusarium DITERIMA (tinggi keyakinan)**

**Rekomendasi tindakan (prioritas & terukur):**

1. **Konfirmasi cepat**: ambil sampel akar/akar/batang untuk pemeriksaan laboratorium (isolasi Fusarium) jika fasilitas tersedia untuk validasi ilmiah
2. **Segera sanitasi & isolasi**: singkirkan tanaman yang sudah parah dari lahan untuk mencegah penyebaran. Jangan biarkan sisa tanaman terkontaminasi di lapangan
3. **Kontrol kultur & bibit**: gunakan bibit bebas patogen; hentikan penanaman di area terkontaminasi sampai tindakan perbaikan dilakukan
4. **Manajemen tanah**: rotasi tanaman dengan tanaman non-inang, peningkatan drainase, dan pertimbangkan solarisasi tanah bila memungkinkan
5. **Higiene**: desinfeksi alat, sepatu, dan area kerja untuk mencegah penyebaran mekanis
6. **Resistensi & penggantian varietas**: pertimbangkan varietas toleran/resisten Fusarium pada tanam berikutnya
7. **Catat & monitor**: dokumentasikan insiden dan lakukan pengamatan berkala untuk melihat efektivitas tindakan

Catatan: Fusarium adalah patogen tanah (vascular wilt) fungisida daun jarang efektif; langkah saniter dan manajemen tanah lebih penting

## **e) Menggabungkan CF dari 2 pakar untuk gejala yang sama (Pakar 1: 0.9, Pakar 2: 0.7)**

Metode umum (kombinasi sekuensial untuk keyakinan searah, keduanya positif):



Hitung (urutkan dengan Pakar 1):



Jadi **CF gabungan dari dua pakar = 0.97**