

Tugas Besar

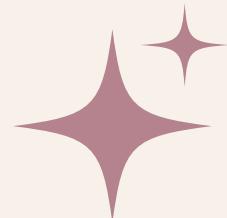
Matematika Diskrit FOL

Bentrok Penjadwalan

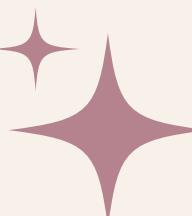
Ujian ETS

Annisa Reida Kamilaini - 241524032

Shofiana Winocita - 241524061



Analisis Isu



Problem Statement

Dari hasil analisis Fishbone, dapat dirumuskan sebuah pertanyaan spesifik untuk diselesaikan dengan FOL, yaitu:

“Bagaimana menyusun penjadwalan waktu ETS agar semua mata kuliah di setiap program studi dan tingkat dapat terujikan, tanpa ada terjadinya bentrok antara ruangan yang digunakan, kelas yang menjalani ujian, dan pengawas yang ditugaskan?”



Pemodelan Logika Orde Pertama (FOL)

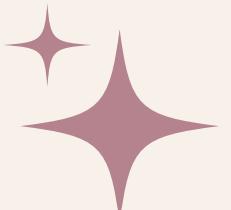


Struktur KB

1

Predikat

1. **jumlah_ruang(Gedung, Jumlah)**
2. **ketersediaan_ruang(Ruang, Status)**
3. **jadwal_penggunaan(Ruang, Waktu, Status)**
4. **mata_kuliah(MK, Tingkat, Tipe)**
5. **jumlah_kelas(Prodi, Tingkat, Jumlah)**
6. **kelas_dijadwalkan(Kelas, MK, Waktu)**
7. **kelas_di_ruang(Kelas, Ruang, Waktu)**
8. **pengawas(Pengawas, Status)**
9. **ditugaskan(Pengawas, Kelas, Waktu)**
10. **kebutuhan_ruang(TipeMK, JenisRuang)**



★ Struktur KB

2 Fakta

- 1) jumlah_ruang(gedung_jtk, 6).
- 2) ketersediaan_ruang(ruang_201, terpakai).
- 3) ketersediaan_ruang(ruang_301, tersedia).
- 4) ketersediaan_ruang(ruang_401, tersedia).
- 5) jadwal_penggunaan(ruang_201, jam_9, terpakai).
- 6) jadwal_penggunaan(ruang_301, jam_9, tersedia).
- 7) jadwal_penggunaan(ruang_401, jam_14, tersedia).
- 8) mata_kuliah(mk_algoritma, tingkat_1, teori).
- 9) mata_kuliah(mk_jarkom, tingkat_2, teori).
- 10) mata_kuliah(mk_basisdata, tingkat_1, teori).
- 11) jumlah_kelas(tif, tingkat_1, 7).
- 12) jumlah_kelas(tko, tingkat_1, 2).
- 13) pengawas(pengawas_andi, hadir).
- 14) pengawas(pengawas_budi, tidak_hadir).
- 15) pengawas(pengawas_citra, hadir).

- 16) kelas_dijadwalkan(kelasA, mk_algoritma, jam_9).
- 17) kelas_dijadwalkan(kelasB, mk_jarkom, jam_9).
- 18) kelas_dijadwalkan(kelasC, mk_basisdata, jam_14).
- 19) butuh_ruang(kelasA).
- 20) butuh_ruang(kelasB).
- 21) butuh_ruang(kelasC).
- 22) kelas_di_ruang(kelasA, ruang_201, jam_9).
- 23) kelas_di_ruang(kelasB, ruang_201, jam_9).
- 24) kelas_di_ruang(kelasC, ruang_401, jam_14).
- 25) ditugaskan(pengawas_andi, kelasA, jam_9).
- 26) ditugaskan(pengawas_budi, kelasB, jam_9).
- 27) ditugaskan(pengawas_citra, kelasC, jam_14).
- 28) kebutuhan_ruang(teori, ruang_kelas).
- 29) kebutuhan_ruang(praktek, lab).

★ Struktur KB

3 Rules

Rule 1 : Bentrok Ruangan

```
bentrok_ruangan(K1, K2) :-  
    kelas_di_ruang(K1, R, W),  
    kelas_di_ruang(K2, R, W),  
    K1 @< K2.
```

Rule 2 : Ruangan Tidak Dapat Digunakan

```
ruang_tidak_dapat_dipakai(R, W) :-  
    jadwal_penggunaan(R, W, terpakai),  
    ketersediaan_ruang(R, terpakai).
```

Rule 3 : Kekurangan Ruang

```
kekurangan_ruang(Prodi, Tingkat) :-  
    jumlah_kelas(Prodi, Tingkat, JK),  
    jumlah_ruang(gedung_jtk, JR),  
    JK > JR.
```

Rule 4 : Potensi Bentrok Ruang

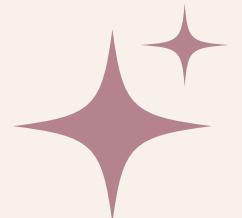
```
potensi_bentrok(K, R, W) :-  
    butuh_ruang(K),  
    kelas_di_ruang(K, R, W),  
    ruang_tidak_dapat_dipakai(R, W).
```

Rule 5 : Pengawas Tidak Bisa Mengawasi

```
pengawas_tidak_bisa(P, K) :-  
    ditugaskan(P, K, _W),  
    pengawas(P, tidak_hadir).
```

Rule 6 : Kelas Bermasalah

```
kelas_bermasalah(K) :-  
    bentrok_ruangan(K, _).  
kelas_bermasalah(K) :-  
    bentrok_ruangan(_, K).  
kelas_bermasalah(K) :-  
    potensi_bentrok(K, _, _).
```



Struktur KB

3

Rules

Rule 7 : Jadwal Bermasalah

`jadwal_bermasalah(K) :-`

`kelas_bermasalah(K),`

`kelas_dijadwalkan(K, _, _).`

Rule 8 : Perlu Penjadwalan Ulang

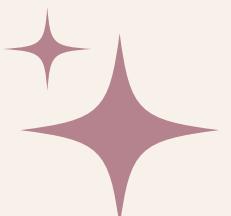
`perlu_penjadwalan_ulang(K) :-`

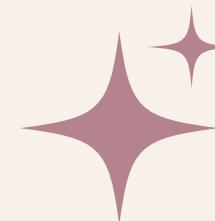
`jadwal_bermasalah(K).`

Rule 9 : Butuh Pengawas Pengganti

`butuh_pengawas_pengganti(K) :-`

`pengawas_tidak_bisa(_, K).`





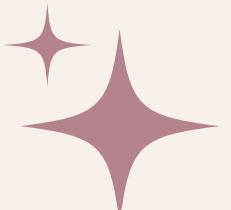
Rantai 3 Langkah

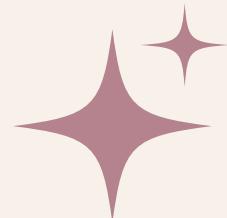
bentrok_ruangan(K, X) → kelas_bermasalah(K) → jadwal_bermasalah(K)
→ perlu_penjadwalan_ulang(K)

Dalam Notasi FOL formal:

$\forall K ((bentrok_ruangan(K, X) \rightarrow kelas_bermasalah(K)) \wedge (kelas_bermasalah(K) \rightarrow jadwal_bermasalah(K)) \wedge (jadwal_bermasalah(K) \rightarrow perlu_penjadwalan_ulang(K)))$

Rantai ini menunjukkan bahwa jika suatu kelas mengalami bentrok ruangan, maka kelas tersebut bermasalah, menyebabkan jadwal ujian bermasalah, dan akhirnya memerlukan penjadwalan ulang.





Unifikasi (Mekanisme)

Rule kompleks yang dianalisis:

Rule 4 : Potensi Bentrok Ruang

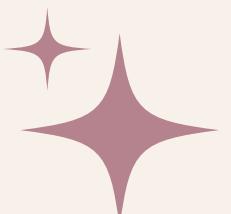
potensi_bentrok(K, R, W) :-

butuh_ruang(K),

kelas_di_ruang(K, R, W),

ruang_tidak_dapat_dipakai(R, W).

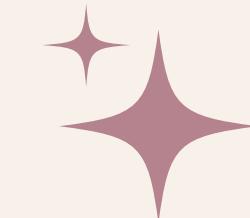
Elemen	Predikat pada Rule	Fakta di KB	Substitusi
1	butuh_ruang(K)	butuh_ruang(kelasA)	{ K / kelasA }
2	kelas_di_ruang(kelasA, R, W)	kelas_di_ruang(kelasA, ruang_201, jam_9)	{ R / ruang_201, W / jam_9 }
3	ruang_tidak_dapat_dipakai(R,W)	ruang_tidak_dapat_dipakai(ruang201, jam_9)	{ R / ruang_201, W / jam_9 }





Resolusi (Teori)

Rule yang dianalisis: Rule 8 : Perlu Penjadwalan Ulang
perlu_penjadwalan_ulang(K) :-
jadwal_bermasalah(K).



Bentuk FOL:

$$\forall K \ jadwal_bermasalah(K) \rightarrow perlu_penjadwalan_ulang(K)$$

Eliminasi Implikasi:

$$\forall K (\neg jadwal_bermasalah(K) \vee perlu_penjadwalan_ulang(K))$$

Clausal Form:

$$\{\neg jadwal_bermasalah(K) \vee perlu_penjadwalan_ulang(K)\}$$

Negasi Tujuan:

$$\neg perlu_penjadwalan_ulang(kelasA)$$

Himpunan Klausu:

Klausu hasil CNF Rule 8: $\neg jadwal_bermasalah(K) \vee perlu_penjadwalan_ulang(K)$

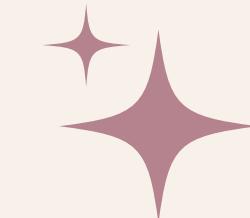
Fakta hasil inferensi sebelumnya: $jadwal_bermasalah(kelasA)$

Negasi tujuan: $\neg perlu_penjadwalan_ulang(kelasA)$



Resolusi (Teori)

Rule yang dianalisis: Rule 8 : Perlu Penjadwalan Ulang
perlu_penjadwalan_ulang(K) :-
jadwal_bermasalah(K).



Bentuk FOL:

$$\forall K \ jadwal_bermasalah(K) \rightarrow perlu_penjadwalan_ulang(K)$$

Eliminasi Implikasi:

$$\forall K (\neg jadwal_bermasalah(K) \vee perlu_penjadwalan_ulang(K))$$

Clausal Form:

$$\{\neg jadwal_bermasalah(K) \vee perlu_penjadwalan_ulang(K)\}$$

Negasi Tujuan:

$$\neg perlu_penjadwalan_ulang(kelasA)$$

Himpunan Klausu:

Klausu hasil CNF Rule 8: $\neg jadwal_bermasalah(K) \vee perlu_penjadwalan_ulang(K)$

Fakta hasil inferensi sebelumnya: $jadwal_bermasalah(kelasA)$

Negasi tujuan: $\neg perlu_penjadwalan_ulang(kelasA)$

★ Resolusi (Teori)

Proses Resolusi

Resolusi antara: $\neg jadwal_bermasalah(kelasA) \vee perlu_penjadwalan_ulang(kelasA)$

dan $\neg perlu_penjadwalan_ulang(kelasA)$

menghasilkan: $\neg jadwal_bermasalah(kelasA)$

Resolusi lanjutan dengan klausa: $jadwal_bermasalah(kelasA)$

Menghasilkan klausa kosong.

Dapat disimpulkan secara formal bahwa $perlu_penjadwalan_ulang(kelasA)$ terbukti benar secara logis melalui metode resolusi.

Hasil Inferensi dan Solusi

★ Hasil Uji Wajib

1. Query: Apakah ruangan bentrok? (ruang & waktu sama)

```
▼ Apakah ruangan bentrok? (ruang & waktu sama)

bentrok_ruangan(X, Y)

Uji: Apakah ruangan bentrok? (ruang & waktu sama)

 VALID
Hasil binding variabel:
1. X = kelasA, Y = kelasB
```

2. Query: Apakah ruangan tidak bisa dipakai?

```
▼ Apakah ruangan tidak bisa dipakai?

ruang_tidak_dapat_dipakai(R, W)

Uji: Apakah ruangan tidak bisa dipakai?

 VALID
Hasil binding variabel:
1. R = ruang_201, W = jam_9
```

★ Hasil Uji Wajib

3. Query: Apakah kekurangan ruangan(per prodi-tingkat)?

▼ Apakah kekurangan ruangan (per prodi-tingkat)?

`kekurangan_ruang(P, T)`

Uji: Apakah kekurangan ruangan (per prodi-tingkat)?

VALID

Hasil binding variabel:

1. P = `tif`, T = `tingkat_1`

4. Query: Adakah potensi bentrok?

▼ Adakah potensi bentrok (kelas ditempatkan di ruang terpakai)?

`potensi_bentrok(K, R, W)`

Uji: Adakah potensi bentrok (kelas ditempatkan di ruang terpakai)?

VALID

Hasil binding variabel:

1. K = `kelasA`, R = `ruang_201`, W = `jam_9`

2. K = `kelasB`, R = `ruang_201`, W = `jam_9`

★ Hasil Uji Wajib

5. Query: Apakah pengawas tidak bisa hadir untuk kelas yang ditugaskan?

▼ Apakah pengawas tidak bisa hadir untuk kelas yang ditugaskan?

`pengawas_tidak_bisa(P, K)`

Uji: Apakah pengawas tidak bisa hadir untuk kelas yang ditugaskan?

VALID

Hasil binding variabel:

1. P = pengawas_budi, K = kelasB

6. Query: Apakah kelas bermasalah? (bentrok atau ruang invalid)

▼ Apakah kelas bermasalah? (bentrok atau ruang invalid)

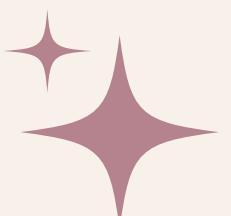
`kelas_bermasalah(K)`

Uji: Apakah kelas bermasalah? (bentrok atau ruang invalid)

VALID

Hasil binding variabel:

1. K = kelasA
2. K = kelasB
3. K = kelasA
4. K = kelasB



★ Hasil Uji Wajib

7. Query: Apakah jadwal bermasalah karena bentrok?

```
▼ Apakah jadwal bermasalah?  
  
jadwal_bermasalah(K)  
  
Uji: Apakah jadwal bermasalah?  
  
✓ VALID  
Hasil binding variabel:  
1. K = kelasA  
2. K = kelasB  
3. K = kelasA  
4. K = kelasB
```

8. Query: Perlukah penjadwalan ulang? (Rantai 3 Langkah)

```
▼ Perlukah penjadwalan ulang? (Rantai 3 Langkah)  
  
perlu_penjadwalan_ulang(K)  
  
Uji: Perlukah penjadwalan ulang? (Rantai 3 Langkah)  
  
✓ VALID  
Hasil binding variabel:  
1. K = kelasA  
2. K = kelasB  
3. K = kelasA  
4. K = kelasB
```

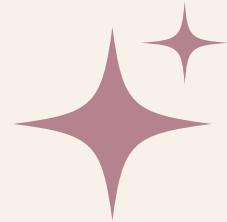
★ Hasil Uji Wajib

9. Query: Apakah butuh pengawas pengganti?

The screenshot shows a Prolog query interface. At the top, there is a dropdown menu with the option "Apakah butuh pengawas pengganti?". Below it, a button labeled "butuh_pengawas_pengganti (K)". A modal window titled "Uji: Apakah butuh pengawas pengganti?" is open. Inside the modal, there is a green checkmark icon followed by the word "VALID". Below this, the text "Hasil binding variabel:" is displayed, followed by "1. K = kelasB".

10. Query Kustom

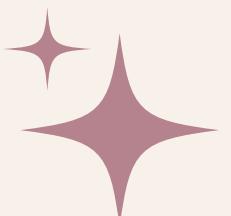
The screenshot shows a Prolog query interface titled "Query Kustom". It has a text input field asking "Masukkan query Prolog (contoh: kelas_di_ruang(K,R,W).)" with the example query "kelas_di_ruang(K,R,W)" already entered. Below the input field is a button labeled "Jalankan Query Kustom". A modal window is open, showing a green checkmark icon followed by the word "VALID". The text "Hasil binding variabel:" is displayed, followed by three results: "1. K = kelasA, R = ruang_201, W = jam_9", "2. K = kelasB, R = ruang_201, W = jam_9", and "3. K = kelasC, R = ruang_401, W = jam_14".



Analisis Validitas

Fallacy yang berhasil disanggah oleh Knowledge Base (KB) ini adalah hasty generalization (generalisasi keliru), yaitu anggapan bahwa semua kelas yang berlangsung pada waktu yang sama pasti bermasalah dan harus dijadwalkan ulang.

Hasil inferensi menunjukkan bahwa meskipun terdapat bentrok ruangan antara kelasA dan kelasB pada jam_9, tidak semua kelas otomatis dikategorikan bermasalah. KB hanya menetapkan suatu kelas bermasalah jika memenuhi kondisi logis tertentu melalui rantai inferensi, seperti adanya bentrok nyata atau penggunaan ruangan yang tidak valid serta kebutuhan ruang kelas tersebut. Dengan demikian, KB menolak kesimpulan instan berbasis satu indikator dan memastikan keputusan penjadwalan ulang diambil secara logis, selektif, dan terstruktur.



Kesimpulan dan Rekomendasi

- 1 kelas yang terdeteksi mengalami bentrok ruangan dan berujung pada status perlu_penjadwalan_ulang sebaiknya langsung diproses oleh sistem penjadwalan ulang otomatis
- 2 setiap penjadwalan ujian perlu melewati tahap validasi ruang untuk memastikan ruang yang dipilih benar-benar tersedia
- 3 setiap penjadwalan ujian perlu melewati tahap validasi ruang untuk memastikan ruang yang dipilih benar-benar tersedia



Terima Kasih

