Laporan Praktikum Kecerdasan Buatan

Semester Ganjil 2024/2025

Jurusan Teknik Elektro
Program Studi S1 Teknik Elektro
Fakultas Teknik
Universitas Tidar

Praktikum ke- : 6
Judul Praktikum : *Forward Chaining* dan *Backward Chaining*

Nama : Jerico Christianto
NIM : 2220501082

“Laporan praktikum ini saya kerjakan dan selesaikan dengan sebaik-baiknya tanpa melakukan tindak kecurangan. Apabila di kemudian hari ditemukan adanya kecurangan pada laporan ini, maka saya bersedia menerima konsekuensinya.”

Tertanda,

Jerico Christianto

2220501082

A. Tujuan Praktikum

1. Memahami konsep dasar *Forward Chaining* dan *Backward Chaining* dalam kecerdasan buatan
2. Menerapkan algoritma tentang *Forward Chaining* dan *Backward Chaining* dalam kecerdasan buatan

B. Langkah Praktikum

1. Terdapat rule seperti berikut :

Rule 1: IF Y & D == True
THEN Z == True
Rule 2: IF X, B, E == True
THEN Y == True
Rule 3: IF A == True
THEN X == True
Rule 4 IF C == True
THEN L == True
Rule 5 IF L & M == True
THEN N == True

2. Terdapat fakta-fakta seperti berikut :

A = True
B = True
C = True
D = True
E = True
X = False
L = False
Y = False
Z= False
M = False
N = False

3. Menuliskan kode program seperti berikut

```
[1]: fakta = {
         'A' : True,
         'B' : True,
         'C' : True,
         'D' : True,
         'E' : True,
         'X' : False,
         'L' : False,
         'Y' : False,
         'Z' : False,
         'M' : False,
         'N' : False,
     }
```

```
rule = {
    'R1' : False,
    'R2' : False,
    'R3' : False,
    'R4' : False,
    'R5' : False,
}

rules_fired = []
i=0
is_change = True
while is_change == True:
    is_change = False
    i+=1
    print('iterasi ke ',i)
    #Rule 1
    if rule['R1'] is False:
        if fakta['Y'] is True and fakta['D'] is True:
            fakta['Z'] = True
            is_change = True
            rules_fired.append('R1')
            rule ['R1'] = True
        print('Rule ke-1:',rule['R1'])

    #Rule 2
    if rule['R2'] is False:
        if fakta['X'] is True and fakta['B'] is True and fakta['E'] is True:
            fakta['Y'] = True
            is_change = True
            rules_fired.append('R2')
            rule ['R2'] = True
        print('Rule ke-2:',rule['R2'])

    #Rule 3
    if rule['R3'] is False:
        if fakta['A'] is True:
            fakta['X'] = True
            is_change = True
            rules_fired.append('R3')
            rule ['R3'] = True
        print('Rule ke-3:',rule['R3'])

    #Rule 4
    if rule['R4'] is False:
        if fakta['C'] is True:
            fakta['L'] = True
            is_change = True
            rules_fired.append('R4')
            rule ['R4'] = True
        print('Rule ke-4:',rule['R4'])

    #Rule 5
    if rule['R5'] is False:
        if fakta['L'] is True and fakta['M'] is True:
            fakta['N'] = True
            is_change = True
            rules_fired.append('R5')
            rule ['R5'] = True
print(rules_fired)
print("Kesimpulan Z: ", fakta['Z'])
```

C. Hasil Praktikum

1. Menambahkan array fakta dan rule

```
[1]: fakta = {
        'A' : True,
        'B' : True,
        'C' : True,
        'D' : True,
        'E' : True,
        'X' : False,
        'L' : False,
        'Y' : False,
        'Z' : False,
        'M' : False,
        'N' : False,
    }

    rule = {
        'R1' : False,
        'R2' : False,
        'R3' : False,
        'R4' : False,
        'R5' : False,
    }
```

Kode program tersebut merupakan kode untuk menambahkan fakta yang diketahui dan menambahkan array untuk rule.

2. Mendeklarasikan array

```
rules_fired = []
i=0
is_change = True
while is_change == True:
    is_change = False
    i+=1
    print('iterasi ke ',i)
```

Kode ini digunakan untuk membuat daftar array bernama rules_fired yang akan mencatat semua aturan yang sudah terbukti benar berdasarkan informasi yang kita punya. Angka i berfungsi sebagai penanda untuk menghitung berapa kali proses pencarian aturan benar dilakukan. is_change adalah tanda untuk menunjukkan apakah ada aturan baru yang terbukti benar atau tidak. Selama ada aturan baru yang terbukti benar, proses pencarian akan terus berulang.

3. Menambahkan rule

```
    #Rule 1
    if rule['R1'] is False:
        if fakta['Y'] is True and fakta['D'] is True:
            fakta['Z'] = True
            is_change = True
            rules_fired.append('R1')
            rule ['R1'] = True
        print('Rule ke-1:',rule['R1'])
```

```
    #Rule 2
    if rule['R2'] is False:
        if fakta['X'] is True and fakta['B'] is True and fakta['E'] is True:
            fakta['Y'] = True
            is_change = True
            rules_fired.append('R2')
            rule ['R2'] = True
        print('Rule ke-2:',rule['R2'])

    #Rule 3
    if rule['R3'] is False:
        if fakta['A'] is True:
            fakta['X'] = True
            is_change = True
            rules_fired.append('R3')
            rule ['R3'] = True
        print('Rule ke-3:',rule['R3'])

    #Rule 4
    if rule['R4'] is False:
        if fakta['C'] is True:
            fakta['L'] = True
            is_change = True
            rules_fired.append('R4')
            rule ['R4'] = True
        print('Rule ke-4:',rule['R4'])

    #Rule 5
    if rule['R5'] is False:
        if fakta['L'] is True and fakta['M'] is True:
            fakta['N'] = True
            is_change = True
            rules_fired.append('R5')
            rule ['R5'] = True
        print('Rule ke-5:',rule['R5'])
```

Kode ini digunakan untuk memasukkan aturan baru ke dalam sistem yang akan digunakan untuk menarik kesimpulan.

4. Menampilkan semua rules dan kesimpulan

```
print(rules_fired)
print("Kesimpulan Z: ", fakta['Z'])
```

Program ini akan menunjukkan semua aturan yang digunakan dan kesimpulan akhir yang didapat setelah kita bertanya tentang fakta Z.

5. Output yang dihasilkan yaitu seperti berikut :

```
iterasi ke  1
Rule ke-1: False
Rule ke-2: False
Rule ke-3: True
Rule ke-4: True
Rule ke-5: False
iterasi ke  2
Rule ke-1: False
Rule ke-2: True
Rule ke-5: False
iterasi ke  3
Rule ke-1: True
Rule ke-5: False
iterasi ke  4
Rule ke-5: False
['R3', 'R4', 'R2', 'R1']
Kesimpulan Z:  True
```

Program ini akan menampilkan jumlah iterasi yang diperlukan dalam proses inferensi untuk mendapatkan kesimpulan dari fakta Z, serta hasil akhir dari inferensi tersebut.

D. Kendala Praktikum

Tidak terdapat kendala pada praktikum kali ini. Kode program dapat dijalankan dengan lancar dan hasilnya sesuai.

E. Studi Kasus

Membuat kode program untuk menyelesaikan contoh kasus seperti di bawah dengan menggunakan metode forward chaining :

```
Rule 1: IF Menyusui = tidak AND
           Kaki > 4 AND
           Warna bulu = polos
        THEN Binatang = belalang

Rule 2: IF Menyusui = ya AND
           Kaki = 4 AND
           Berdarah = panas
      THEN Kelas = mamalia

Rule 3: IF Makanan = binatang
         THEN kategori = carnivora

Rule 4: IF Makanan = tumbuhan
         THEN Kategori = herbivora

Rule 5: IF Kelas = mamalia AND
           Warna bulu = berbintik AND
           Tinggi > 80 centimeter
      THEN Binatang = Harimau
Rule 6: IF Kelas = mamalia AND
           Warna bulu = polos AND
           Tinggi > 80 centimeter
      THEN Binatang = Kuda

Rule 7: IF Kelas = mamalia AND
           Kategori = herbivora AND
           Warna bulu = bergaris AND
           Tinggi > 80 centimeter
      THEN  Binatang = Zebra
Rule 8: IF Kelas = mamalia AND
           Kategori = carnivora AND
           Warna bulu = polos OR berbintik OR bergaris AND
           Tinggi < 80 centimeter
      THEN  Binatang = Kucing

Rule 9: IF Kaki = 2 AND
           Berdarah = panas AND
           Warna bulu = berbintik OR
           Menyusui = tidak
      THEN Binatang = elang

Rule 10: IF Berdarah = dingin AND
            Tinggi < 80 centimeter
       THEN Binatang = salmon
```

Kode Program :

```
[11]: fakta = {
    'Menyusui': False,
    'Kaki': 2,
    'Berdarah': 'panas',
    'Warna_bulu': 'berbintik',
    'Tinggi': 30,
    'Makanan': 'binatang',
    'Kelas': None,
    'Kategori': None,
    'Binatang': None
}

rules = {
    'R1': lambda: not fakta['Menyusui'] and fakta['Kaki'] > 4 and fakta['Warna_bulu'] == 'polos',  # Belalang
    'R2': lambda: fakta['Menyusui'] and fakta['Kaki'] == 4 and fakta['Berdarah'] == 'panas',      # Mamalia
    'R3': lambda: fakta['Makanan'] == 'binatang',                                                 # Carnivora
    'R4': lambda: fakta['Makanan'] == 'tumbuhan',                                                # Herbivora
    'R5': lambda: fakta['Kelas'] == 'mamalia' and fakta['Warna_bulu'] == 'berbintik' and fakta['Tinggi'] > 80,  # Harimau
    'R6': lambda: fakta['Kelas'] == 'mamalia' and fakta['Warna_bulu'] == 'polos' and fakta['Tinggi'] > 80,      # Kuda
    'R7': lambda: fakta['Kelas'] == 'mamalia' and fakta['Kategori'] == 'herbivora' and fakta['Warna_bulu'] == 'bergaris' and fakta['Tinggi'] > 80, # Zebr
    'R8': lambda: fakta['Kelas'] == 'mamalia' and fakta['Kategori'] == 'carnivora' and (fakta['Warna_bulu'] in ['polos', 'berbintik', 'bergaris'])
          and fakta['Tinggi'] < 80,  # Kucing
    'R9': lambda: fakta['Kaki'] == 2 and fakta['Berdarah'] == 'panas' and (fakta['Warna_bulu'] == 'berbintik' or not fakta['Menyusui']),  # Elang
    'R10': lambda: fakta['Berdarah'] == 'dingin' and fakta['Tinggi'] < 80  # Salmon
}

# Fungsi untuk menerapkan forward chaining
def forward_chain():
    is_change = True
    rules_fired = []

    while is_change:
        is_change = False

        # Rule 1: Belalang
        if fakta['Binatang'] is None and rules['R1']():
            fakta['Binatang'] = 'belalang'
            is_change = True
            rules_fired.append('R1')

        # Rule 2: Mamalia
        if fakta['Kelas'] is None and rules['R2']():
            fakta['Kelas'] = 'mamalia'
            is_change = True
            rules_fired.append('R2')

        # Rule 3: Carnivora
        if fakta['Kategori'] is None and rules['R3']():
            fakta['Kategori'] = 'carnivora'
            is_change = True
            rules_fired.append('R3')

        # Rule 5: Harimau
        if fakta['Binatang'] is None and rules['R5']():
            fakta['Binatang'] = 'Harimau'
            is_change = True
            rules_fired.append('R5')

        # Rule 6: Kuda
        if fakta['Binatang'] is None and rules['R6']():
            fakta['Binatang'] = 'Kuda'
            is_change = True
            rules_fired.append('R6')

        # Rule 7: Zebra
        if fakta['Binatang'] is None and rules['R7']():
            fakta['Binatang'] = 'Zebra'
            is_change = True
            rules_fired.append('R7')

        # Rule 8: Kucing
        if fakta['Binatang'] is None and rules['R8']():
            fakta['Binatang'] = 'Kucing'
            is_change = True
            rules_fired.append('R8')

        # Rule 9: Elang
        if fakta['Binatang'] is None and rules['R9']():
            fakta['Binatang'] = 'elang'
            is_change = True
            rules_fired.append('R9')

        # Rule 10: Salmon
        if fakta['Binatang'] is None and rules['R10']():
            fakta['Binatang'] = 'salmon'
            is_change = True
            rules_fired.append('R10')

    print("Aturan yang diterapkan:", rules_fired)
    print("Fakta yang diketahui:", fakta)

# Jalankan forward chaining
forward_chain()
```

Program diatas akan mencoba mencari kesimpulan dari beberapa fakta yang telah ditentukan yaitu tidak menyusui, berkaki 2, berdarah panas, warna bulu berbintik, tinggi 30 cm, dan memakan binatang.

Hasil :

```
Aturan yang diterapkan: ['R3', 'R9']
Fakta yang diketahui: {'Menyusui': False, 'Kaki': 2, 'Berdarah': 'panas', 'Warna_bulu': 'berbintik', 'Tinggi': 30, 'Makanan': 'binatang', 'Kelas': None,
'Kategori': 'carnivora', 'Binatang': 'elang'}
```

Program berhasil menarik kesimpulan makhluk tersebut sebagai seekor elang berdasarkan kumpulan fakta yang diberikan dan aturan-aturan yang telah didefinisikan sebelumnya.

Kesimpulan :

Dari studi kasus tersebut didapatkan bahwa *forward chaining* akan menarik kesimpulan yang dimulai dari fakta-fakta yang diketahui dahulu dan secara bertahap menerapkan aturan-aturan untuk menghasilkan kesimpulan baru. Proses ini berlanjut hingga tidak ada lagi aturan yang dapat diterapkan atau hingga tujuan yang diinginkan tercapai.