Szkoła Gospodarstwa Wiejskiego
Wydział Zastosowań Informatyki i Matematyki

PARADYGMATY PROGRAMOWANIA

Jednostka 9 – Programowanie logiczne: reguły, wnioskowanie i Prolog (cz. 2)

1. Cel i zakres zajęć

Celem zajęć jest:

- pogłębienie znajomości **paradygmatu logicznego**,
- poznanie mechanizmów **wnioskowania logicznego** i **rekurencji w regułach**,
- zrozumienie różnicy między faktami, regułami i zapytaniami,
- nauczenie się budowania **bazy wiedzy** i zadawania złożonych pytań w Prologu,
- zapoznanie z podstawami mechanizmu **backtrackingu** i unifikacji.

Zakres tematyczny:

- język Prolog: fakty, reguły, zapytania,
- unifikacja, dopasowywanie wzorców,
- rekurencyjne reguły i relacje,
- backtracking – mechanizm wyszukiwania rozwiązań,
- wnioskowanie logiczne w praktyce.

---

2. Wprowadzenie teoretyczne

2.1 Programowanie logiczne

W programowaniu logicznym program składa się z:

- **faktów** — opisują relacje, które są zawsze prawdziwe,
- **reguł** — wyrażają zależności logiczne pomiędzy faktami,
- **zapytania** — służą do wnioskowania nowych informacji.

Silnik logiczny **nie wykonuje instrukcji**, tylko **próbuje udowodnić zapytania** na podstawie faktów i reguł.

2.2 Fakty i reguły – przypomnienie

```
rodzic(jan, anna).
rodzic(anna, piotr).

przodek(X,Y) :- rodzic(X,Y).
przodek(X,Y) :- rodzic(X,Z), przodek(Z,Y).
```

Zapytanie:

```
?- przodek(jan, piotr).
true.
```

2.3 Unifikacja

Unifikacja to proces dopasowywania zmiennych i wartości w celu sprawdzenia zgodności wzorców.
Przykład:

```
?- rodzic(X, anna).
X = jan.
```

Silnik Prologa automatycznie szuka przypisań zmiennych, które sprawiają, że reguła i zapytanie są zgodne.

2.4 Backtracking

Prolog korzysta z mechanizmu **backtrackingu**, czyli systematycznego przeszukiwania wszystkich możliwych dopasowań.

Przykład:

```
kolor(czerwony).
kolor(zielony).
kolor(niebieski).

lubie(czerwony).
lubie(niebieski).

?- kolor(X), lubie(X).
X = czerwony ;
X = niebieski ;
false.
```

Prolog znajduje pierwsze dopasowanie, a następnie cofa się („backtrackuje”), aby znaleźć kolejne.

2.5 Rekurencja w regułach

Prolog obsługuje **rekurencję logiczną** — reguła może odwoływać się do samej siebie, co pozwala definiować relacje transitive (np. przodkowie, ścieżki w grafach).

---

3. Tutorial 1 – Rodzina i przodkowie (Prolog)

3.1 Fakty

```
rodzic(jan, anna).
rodzic(anna, piotr).
rodzic(jan, krzysztof).
rodzic(krzysztof, marek).
```

3.2 Reguły

```
dziecko(X,Y) :- rodzic(Y,X).

przodek(X,Y) :- rodzic(X,Y).
przodek(X,Y) :- rodzic(X,Z), przodek(Z,Y).
```

3.3 Zapytania

```
?- przodek(jan, marek).
true.

?- dziecko(anna, X).
X = jan.
```

---

4. Tutorial 2 – Operacje na zbiorach i strukturach

4.1 Listy w Prologu

```
suma([], 0).
suma([H|T], S) :- suma(T, S1), S is S1 + H.

?- suma([1,2,3,4], S).
S = 10.
```

Tu wykorzystujemy:

- **rekurencję** do przetwarzania list,
- **operator [H|T]** do rozbijania listy na głowę i ogon.

4.2 Definiowanie własnych relacji

Przykład relacji „należy do listy” (`member/2`):

```
nalezy(X, [X|_]).
nalezy(X, [_|T]) :- nalezy(X, T).
?- nalezy(3, [1,2,3,4]).
true.
```

```
?- nalezy(5, [1,2,3,4]).
false.
```

---

5. Zadania do samodzielnego wykonania

Zadanie 1 (obowiązkowe)

Zdefiniuj fakty i reguły opisujące **drzewo genealogiczne** kilku osób.
Napisz reguły:

- `dziecko/2`,
- `rodzenstwo/2`,
- `dziadek/2`,
- `przodek/2` (rekurencyjnie).
  Przetestuj je na różnych zapytaniach.

Zadanie 2 (obowiązkowe)

Zaimplementuj relację `nalezy/2` i `dlugosc/2` dla list w Prologu, używając rekurencji.

Zadanie 3 (dodatkowe)

Zaimplementuj prostą relację `sciezka/3`, która dla grafu zapisanego jako fakty `krawedz(A,B)` odpowiada, czy istnieje ścieżka z A do B. Użyj rekurencji.

---

6. Pytania kontrolne

1. Czym różnią się fakty od reguł?
2. Na czym polega unifikacja w Prologu?
3. Co robi mechanizm backtrackingu?
4. Jak działa rekurencja logiczna?
5. Czym różni się sposób rozwiązywania problemów w Prologu od podejścia imperatywnego?

---

7. Checklisty

Programowanie logiczne

- Fakty i reguły zdefiniowane poprawnie.
- Zapytania działają zgodnie z oczekiwaniami.
- Użyto rekurencji w co najmniej jednej regule.
- Wyniki zapytań pokrywają się z przewidywaniami logicznymi.
- Nie używano sztucznych „pętli” — jedynie relacje i dopasowania.

---

8. Wzór sprawozdania

Dodatkowo:

- Diagram drzewa genealogicznego lub grafu.
- Fakty, reguły, przykładowe zapytania i wyniki.
- Komentarz: jak backtracking wpływał na sposób uzyskiwania wyników.

Zakończenie

Po wykonaniu ćwiczenia student powinien:

- rozumieć, jak działa programowanie logiczne i wnioskowanie,
- znać mechanizmy unifikacji i backtrackingu,
- potrafić definiować relacje rekurencyjne,
- umieć formułować fakty i reguły w Prologu i zadawać zapytania,
- porównywać logiczne podejście z imperatywnym, funkcyjnym i obiektowym.