Programowanie deklaratywne

Artur Michalski
Informatyka II rok

Plan wykładu

- Wprowadzenie do języka Prolog
- Budowa składniowa i *interpretacja programów prologowych*
- Listy, operatory i operacje arytmetyczne
- Sterowanie mechanizmem nawrotów
- Predefiniowane procedury prologowe
- Styl i technika programowania w Prologu

Budowa składniowa i znaczenie programów w Prologu

- Deklaratywna interpretacja programu prologowego
- Proceduralna interpretacja programu prologowego
- Porządek kauzul prologowych i celów
- Podsumowanie

Deklaratywna i proceduralne interpretacja programu prologowego

Rozważmy klauzulę `P` postaci: `P:-Q,R.`

- Interpretacja deklaratywna klauzuli `P`:
  P *jest prawdziwe wtedy, gdy* ***Q*** *i* ***R*** *są prawdziwe.*
  albo:
  Z ***Q*** *i* ***R*** *wynika* ***P****.*
- Interpretacja proceduralna klauzuli `P`:
  Żeby osiągnąć cel ***P****, najpierw musisz osiągnąć podcel* ***Q****, a potem podcel* ***R****.*
  albo:
  Spełnienie ***P*** *wymaga najpierw spełnienia* ***Q*** *a potem* ***R****.*

  Różnica: interpretacja proceduralna określa nie tylko logiczny związek między nagłówkiem reguły i jej ciałem, ale również *porządek* w jakim mają być osiągane podcele.

Deklaratywna interpretacja programu prologowego

Deklaratywna interpretacja programu prologowego określa czy dany cel jest spełniony, jeśli tak, to dla jakich wartości zmiennych.

Instancja klauzuli to taka klauzula, w której z każdą zmienną związano już jakiś term.

Przykład
Klauzula:

```
ma_dziecko(X) :- rodzic(X,_).
```

i jej przykładowe instancje:

```
ma_dziecko(piotr):- rodzic(piotr,Y1).
ma_dziecko(jan):- rodzic(jan,mala(ewa)).
```

Deklaratywna interpretacja programu prologowego

Formalna definicja interpretacji deklaratywnej

Dla danego programu prologowego i celu *G*:
Cel *G jest prawdziwy* (spełniony albo wynika logicznie z programu) wtedy i tylko wtedy, gdy:

- istnieje w programie klauzula C taka, że:
- istnieje instancja *J* klauzuli *C* taka, że:
  - zachodzi uzgodnienie nagłówka instancji *J* z *G*, oraz
  - wszystkie podcele w ciele instancji *J są prawdziwe* (spełnione) przy tym uzgodnieniu

Deklaratywna interpretacja programu prologowego

Interpretacja deklaratywna dla celu złożonego

W przypadku celów złożonych (ich koniunkcja), *lista celów jest spełniona*, gdy wszystkie jej cele są równocześnie spełnione dla tych samych podstawień (wyników uzgodnień) zmiennych.

Deklaratywna interpretacja programu prologowego

Rodzaje celów złożonych:

Koniunkcja celów - lista celów oddzielonych przecinkiem - wszystkie cele muszą być spełnione.

Dysjunkcja celów - lista celów oddzielonych średnikiem - wystarczy, że jeden z celów zostanie spełniony.

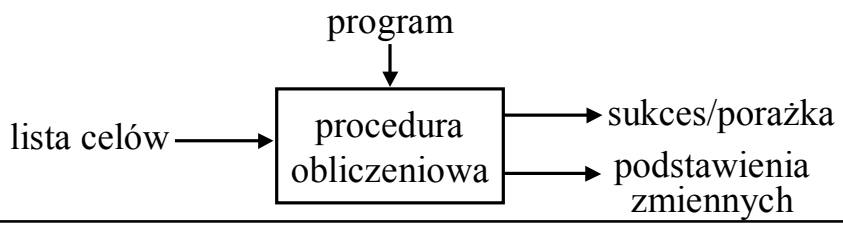
Klauzula postaci: `P:-Q;R.`
jest równoważna dwóm klauzulom: `P:-Q.` i `P:-R.`

Koniunkcja celów ma wyższy priorytet niż dysjunkcja celów.

Proceduralna interpretacja programu prologowego

Interpretacja proceduralna programu prologowego określa *jak* Prolog odpowiada na pytania (*w jaki sposób* spełnia cele).

Interpretacja proceduralna oznacza wykonanie procedury obliczeniowej, która doprowadzi do spełnienia listy celów z uwzględnieniem danego programu prologowego.



Proceduralna interpretacja programu prologowego

Przykład interpretacji proceduralnej

Zbiór klauzul:

```
big(bear).                  % klauzula 1
big(elephant).              % klauzula 2
small(cat).                 % klauzula 3
brown(bear).                % klauzula 4
black(cat).                 % klauzula 5
gray(elephant).             % klauzula 6
dark(X) :- black(X).        % klauzula 7
dark(X) :- brown(X).        % klauzula 8
```

Pytanie (cel): `dark(X),big(X).`

Proceduralna interpretacja programu prologowego

Przebieg procedury spełniania celu:

1) Początkowa lista celów: **`dark(X),big(X).`**
2) Analiza całego programu od początku do końca i poszukiwanie klauzuli, której nagłówek można uzgodnić z pierwszym celem: **`dark(X)`**. Klauzula 7:
   `dark(X) :- black(X).`
   Zamiana pierwszego celu na zainicjowaną podstawieniami zmiennych treść klauzuli 7:
   `black(X),big(X).`
3) Analiza programu w poszukiwaniu uzgodnień dla celu **`black(X)`**. Uzgodnienie klauzuli 5: **`black(cat)`**. Klauzula nie ma treści, więc lista celów po zastosowaniu podstawień zmiennych (**`X=cat`**) ma postać: **`big(cat).`**

Proceduralna interpretacja programu prologowego

Przebieg procedury spełniania celu (ciąg dalszy):

4) Analiza całego programu w poszukiwaniu klauzuli, której nagłówek można uzgodnić z: **`big(cat)`**. Brak takiej klauzuli. Nawrót do kroku 3) i anulowanie podstawienia: **`X=cat`**. Lista celów ma znów postać:
   `black(X),big(X).`
   Kontynuacja przeszukiwania programu poniżej klauzuli 5 zakończona porażką - brak innych sposobów spełnienia celu: **`black(X)`**. Nawrót do kroku 2) i przeszukiwanie poniżej klauzuli 7. Uzgodnienie dla klauzuli 8:
   `dark(X) :- brown(X).`
   Nowa lista celów: **`brown(X),big(X).`**

Proceduralna interpretacja programu prologowego

Przebieg procedury spełniania celu (ciąg dalszy):

5) Analiza całego programu w poszukiwaniu klauzuli, której nagłówek można uzgodnić z: **`brown(X)`**. Uzgodnienie dla **`brown(bear)`** i wiązanie zmiennej: **`X=bear`**. Klauzula pozbawiona treści, więc lista celów zmniejsza się do: **`big(bear)`**.
6) Poszukiwanie uzgodnień dla celu **`big(bear)`** zakończone sukcesem - mamy taki fakt! Klauzula nie ma treści, więc lista celów zmniejsza się do listy pustej. Cel główny spełniony dla podstawienia zmiennych: **`X=bear`**.
7) Cel główny spełniony - rozwiązanie: **`X=bear`**.

Proceduralna interpretacja programu prologowego

Formalna definicja interpretacji proceduralnej

Dla danego programu prologowego i celu *G1,G2,...,Gm*:

1) jeśli lista celów jest pusta, to koniec z *sukcesem*
2) jeśli lista celów nie jest pusta, to kontynuuj operację ANALIZA.
3) ANALIZA: Przeszukiwanie całego programu (od początku do końca) do pierwszej klauzuli *C*, której nagłówek można uzgodnić z pierwszym celem *G1*. Jeśli nie ma takiej klauzuli, to koniec z *porażką*. Jeśli taka klauzula jest i ma postać:
   H :- B1,B2, ..., Bn.
   to zmiana nazw zmiennych w *C* na unikalne zmienne (nowy wariant tej klauzuli: *C'*), różne od zmiennych zawartych w liście *G1,G2,...,Gm*:
   H' :- B1',B2',...,Bn'. wariant *C* oznaczony jako *C'*

Proceduralna interpretacja programu prologowego

Formalna definicja interpretacji proceduralnej

ciąg dalszy ANALIZY:
Uzgodnienie *G1* z *H'* i rezultat w postaci zbioru podstawień *S*.

W liście celu głównego *G1,G2,...,Gm* zamieniamy *G1* na listę *B1',B2',...,Bn'* i otrzymujemy nową listę celów:
B1',B2',...,Bn',G2,...,Gm
(jeśli *C* jest faktem wtedy *n=0* i nowa lista celów jest krótsza niż lista pierwotna; zmniejszanie się tej listy doprowadzi w końcu do listy pustej i tym samym osiągnięcia celu głównego).

Zamieniamy zmienne na nowej liście celów zgodnie z podstawieniami ze zbioru *S* i otrzymujemy ostateczną listę:
B1'',B2'',...,Bn'',G2',...,Gm'

Proceduralna interpretacja programu prologowego

Formalna definicja interpretacji proceduralnej

4) Wykonujemy rekurencyjnie powyższą procedurę ANALIZA dla nowej listy celów *B1'',B2'',...,Bn'',G2',...,Gm'* . Jeśli realizacja tego celu zakończy się sukcesem, to realizacja celu pierwotnego *G1,G2,...Gm* również kończy się sukcesem. Jeśli lista celów *B1'',B2'',...,Bn'',G2',...,Gm'* zakończy się porażką, to porzucamy dalsze przetwarzanie tej listy i powracamy do kontynuowania operacji ANALIZA dla pozostałej części programu prologowego, czyli przeglądamy program, poczynając od klauzuli następnej po klauzuli *C* (*C* jest klauzulą, której użyliśmy jako ostatniej) i próbujemy wykorzystać inną klauzulę, której nagłówek można uzgodnić z celem *G1*.

Proceduralna interpretacja programu prologowego

Formalna definicja interpretacji proceduralnej

Uwagi do definicji:
1. Procedura nie określa w jaki sposób otrzymywany jest ostateczny zbiór podstawień zmiennych *S*. Zbiór podstawień, który prowadzi do spełnienia pierwszego celu z listy podlega najczęściej dalszym uściśleniom w wyniku realizacji kolejnych celów.
2. Kiedy rekurencyjne wywołanie procedury prowadzi do porażki dokonujemy *nawrotu* do tego miejsca w programie, w którym wybrano złą klauzulę (klauzulę *C*), porzucając wszystkie rezultaty jej przetwarzania, włącznie z dokonanymi podstawieniami zmiennych. Taka realizacja celów gwarantuje systematyczną analizę wszystkich alternatywnych ścieżek przetwarzania, prowadzących do spełnienia celu lub kończy się stwierdzeniem (po sprawdzeniu ich wszystkich), że cel nie jest spełniony.

Porządek klauzul prologowych i celów

Porządek klauzul w programie prologowym ma kluczowe znaczenie dla efektywności i skuteczności programu.

Przykład

Klauzula poprawna deklaratywnie

```
p:- p.
```

Pytanie:

```
?- p.
```

Efekt: nieskończona pętla!!!

Program poprawny w sensie interpretacji deklaratywnej może być bezużyteczny ze względu na interpretację proceduralną.

Porządek klauzul prologowych i celów

Wpływ porządku klauzul na realizację celu programu.

Wnioski:

Program prologowy może nie znaleźć rozwiązania, nawet jeżeli rozwiązanie to istnieje (pętle nieskończone!).

Program prologowy może być poprawny w sensie deklaratywnym, ale błędny w sensie proceduralnym.

Istnieją ogólne metody eliminacji nieskończonych pętli (bezproduktywnych dróg poszukiwania rozwiązania), które można zastosować z programie prologowym.

Porządek klauzul prologowych i celów

Wpływ porządku klauzul na realizację celu

Przykład

```
przodek(P,D):-rodzic(P,D).                    %prb
przodek(P,D):-rodzic(P,R),przodek(R,D).       %prp
```

Możliwe modyfikacje porządku:

- zmiana kolejności klauzul *prb* i *prp* w programie
- zmiana kolejności podcelów w ciele klauzuli *prp*

Porządek klauzul prologowych i celów

Przykład c.d.

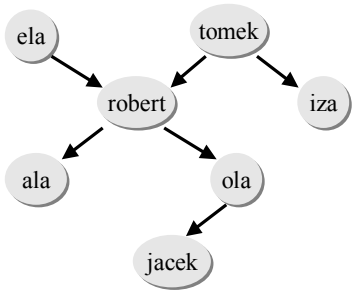
Efekt: cztery warianty tego samego programu - identyczne w sensie deklaratywnym, lecz różne w sensie proceduralnym.

```
przodek1(P,D):- rodzic(P,D).
przodek1(P,D):- rodzic(P,R),przodek1(R,D).

przodek2(P,D):- rodzic(P,R),przodek2(R,D).
przodek2(P,D):- rodzic(P,D).

przodek3(P,D):- rodzic(P,D).
przodek3(P,D):- przodek3(P,R),rodzic(R,D).

przodek4(P,D):- przodek4(P,R),rodzic(R,D).
przodek4(P,D):- rodzic(P,D).
```

Porządek klauzul prologowych i celów

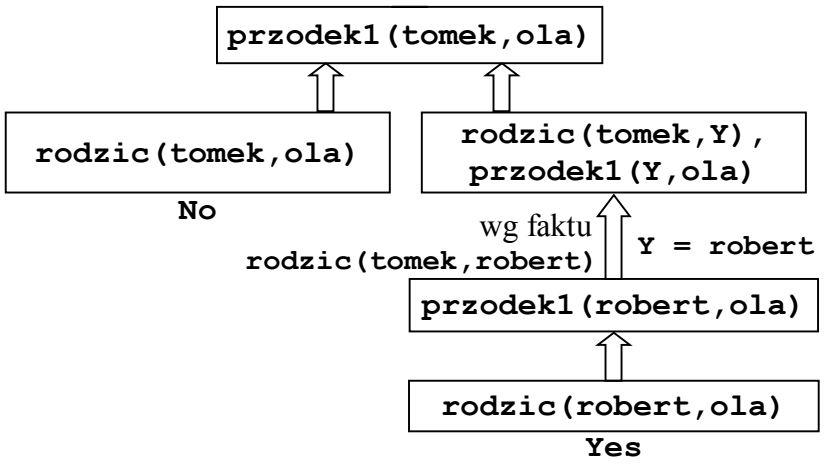
Przykład c.d.

```
?-przodek1(tomek,ola).
Yes

?-przodek2(tomek,ola).
Yes

?-przodek3(tomek,ola).
Yes

?-przodek4(tomek,ola).
ERROR: out of local stack
```



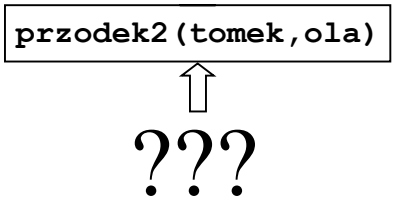
Porządek klauzul prologowych i celów

Przykład c.d.



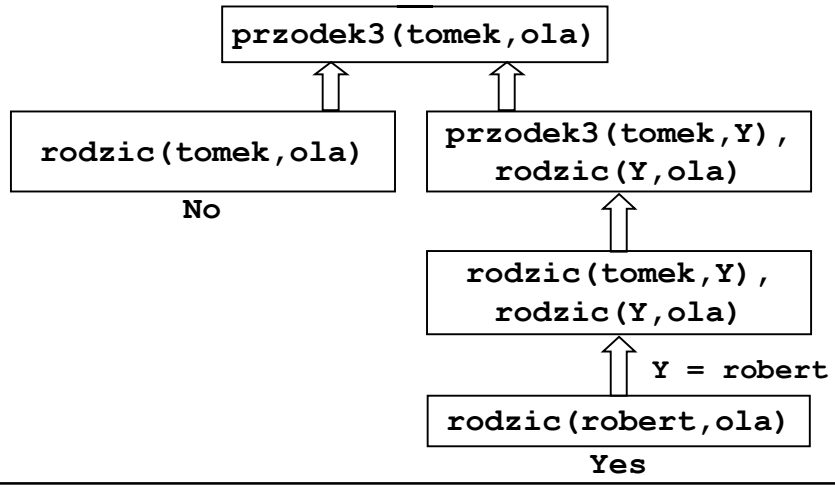
Porządek klauzul prologowych i celów

Przykład c.d.



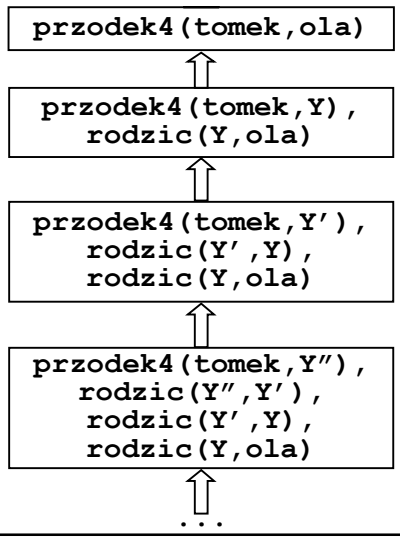
Porządek klauzul prologowych i celów

Przykład c.d.



Porządek klauzul prologowych i celów

Przykład c.d.



Porządek klauzul prologowych i celów

Przykład c.d.

Podsumowanie

przodek1 - najlepsza wersja

przodek2 - wersja o najgorszej efektywności, ale skuteczna

przodek3 - nie zawsze skuteczna, np. **?-przodek3(iza,jacek).**

przodek4 - wersja nieskuteczna (nieskończona pętla!)

W definicji klauzuli prologowej należy najpierw korzystać z klauzul, opisujących proste relacje między obiektami, zanim użyjemy relacji bardziej złożonych (rekurencyjnych).

Porządek klauzul prologowych i celów

Przykład c.d.

```
przodek1(P,D):- rodzic(P,D).
przodek1(P,D):- rodzic(P,R),przodek1(R,D).
```
rodzic → rodzic → przodek1

```
przodek2(P,D):- rodzic(P,R),przodek2(R,D).
przodek2(P,D):- rodzic(P,D).
```
rodzic → przodek2 → rodzic

```
przodek3(P,D):- rodzic(P,D).
przodek3(P,D):- przodek3(P,R),rodzic(R,D).
```
rodzic → przodek3 → rodzic

```
przodek4(P,D):- przodek4(P,R),rodzic(R,D).
przodek4(P,D):- rodzic(P,D).
```
przodek4 → rodzic → rodzic

Porządek klauzul prologowych i celów

Czy interpretacja deklaratywna programu jest potrzebna? Czy nie byłoby lepiej ograniczyć się tylko do interpretacji proceduralnej skoro poprawność programu w pierwszej nie jest jednoznaczna z poprawnością w drugiej?

Postęp w językach programowania wskazuje na konieczność porzucania języków proceduralnych na rzecz języków deklaratywnych, w których łatwiej i jaśniej można sformułować wiele zadań i w których ciężar przetwarzania w większym stopniu spoczywa na systemie programowania niż programiście.

Język programowania Prolog jest tylko pewnym krokiem uczynionym w kierunku „czystego" programowania deklaratywnego.

Podsumowanie

- Semantyka deklaratywna w Prologu określa czy dla danego programu cel jest spełniony, i jeśli tak, to dla jakich podstawień zmiennych
- Przecinek między klauzulami oznacza koniunkcję celów, a średnik - dysjunkcję celów
- Semantyka proceduralna w Prologu to procedura spełniania listy celów w kontekście danego programu. Procedura ta stwierdza fałszywość lub prawdziwość listy celów i podaje ewentualne podstawienia zmiennych. Procedura korzysta z mechanizmu nawrotów i analizuje alternatywne rozwiązania
- „Czysta" semantyka deklaratywna nie zależy od kolejności klauzul i kolejności celów w klauzulach

Podsumowanie

- Semantyka proceduralna jest ściśle określona przez kolejność klauzul i celów, która może mieć decydujący wpływ na efektywność i skuteczność programu
- Mając dany poprawny w sensie deklaratywnym program możemy zwiększyć jego efektywność poprzez zmianę kolejności klauzul i celów, nie naruszając jego poprawności deklaratywnej. Jest to dobra metoda wykrywania i eliminacji nieskończonych pętli w programie
- Istnieją inne ogólne techniki (np. z dziedziny Szt. Int.) eliminacji nieskończonych pętli