1. Wprowadzenie do środowiska SWI Prolog

Obecnie mamy dostępnych kilka implementacji języka Prolog, jedną z nich jest środowisko Jana Wielemaker'a: SWI–Prolog, http://www.swi-prolog.org.
Na zajęciach będziemy korzystać z tego środowiska. Praca w nim jest nieco podobna do pracy w powłoce unixowej. Program składa się z dwóch części: bazy wiedzy i programu. Baza wiedzy zapisana jest w pliku: **baza_wiedzy.pl,** program natomiast to odwołanie do bazy wiedzy, które wpisujemy w linii komend środowiska SWI-Prolog.

Budowa programów

Elementy składniowe programu:

- stałe: stałe znakowe a także liczby (atomy),
- niewiadome/szukane (odpowiadają tzw. zmiennym logicznym, które nie mają zbyt wiele wspólnego ze zmiennymi w klasycznych językach programowania),
- termy: symbole funkcyjne wraz z argumentami, obiekty strukturalnie złożone.

Program składa się z:

- szeregu klauzul (ang. clause), wyróżniamy:
  - fakty (klauzule proste, formuły atomowe/atomiczne),
  - reguły (klauzule złożone),
- celu (ang. goal).

2. Fakty w języku Prolog

Fakty, to prawdziwe stwierdzenia o obiektach i powiązaniach między nimi:
symbol_predyktu(obiekt$_1$,obiekt$_2$,…,obiekt$_n$).

- lubi(ola, kino).
- mężczyzna(tomek).
- rodzic(zofia, marcin).

Podane fakty odczytujemy następująco:

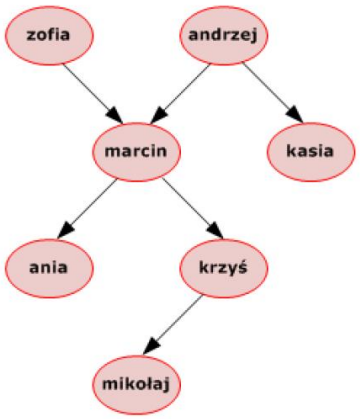
- Ola lubi kino.
- Tomek to mężczyzna.
- Zofia jest rodzicem Marcina.

Należy pamiętać, że:

- nazwy relacji, jak i argumentów piszemy z małej litery.

- każdy fakt zakończony jest kropką.
- argumenty są oddzielone przecinkami. Ich ilość i kolejność występowania jest określona ściśle poprzez rodzaj związku ich wiążącego.

Przykład 1.
Schemat przykładowej relacji w rodzinie



Fakt, że Zofia jest rodzicem Marcina, w Prologu zapisujemy, jako:
rodzic(zofia, marcin).
Słowo **rodzic** to nazwa relacji, a argumentami są **zofia** i **marcin.**
Utwórzmy, zatem naszą pierwszą bazę wiedzy. (Uruchom środowisko SWI Prolog i utwórz nowy plik o nazwie rodzina.pl). Całe drzewo rodzinne zapiszemy zatem w postaci:
rodzic(zofia, marcin).
rodzic(andrzej, marcin).
rodzic(andrzej, kasia).
rodzic(marcin, ania).
rodzic(marcin, krzyś).
rodzic(krzyś, mikołaj).

Powyższa baza wiedzy zawiera sześć **klauzul,** każda z nich opisuje jeden fakt relacji bycia rodzicem.
Występują dwa typy klauzul:

- Klauzule złożone z pojedynczej struktury to stwierdzenia, które przyjmujemy jako fakt tak jak w przykładzie np. rodzic(zofia,marcin);
- Klauzule, które wyrażają definicję nowej relacji, nie przez wyliczenie powiązanych tą relacją obiektów, ale przez określenie wiążące ją z innymi relacjami.

Po utworzeniu powyższej bazy wiedzy możemy w środowisku SWI Prolog zadać pytania na temat relacji rodzinnych np.: Czy Marcin jest rodzicem Krzysia?, Czy Andrzej jest rodzicem Mikołaja? Itp. Pierwsze pytanie możemy zadać wpisując do kompilatora:
? – rodzic(marcin, krzyś).
Oczywiście Prolog znajdując taki fakt w bazie wiedzy odpowie:
true
Drugie pytanie zadamy analogicznie:
? – rodzic(andrzej, mikołaj).
Na to pytanie uzyskamy oczywiście odpowiedź:
false
ponieważ w bazie wiedzy nie mamy zapisanego faktu dotyczącego tej postaci. Analogicznie uzyskamy odpowiedź false zadając pytanie:
? – rodzic(andrzej, maciek).
Ponieważ imię maciek nie występuje w ogóle w naszym programie. Oprócz najprostszych pytań dotyczących prawdziwości podanych stwierdzeń możemy zadać bardziej ogólne pytania. Na przykład:
? – rodzic(X, marcin).
Pytanie to ma postać: Kto jest rodzicem marcina? W tym przypadku Prolog nie udzieli odpowiedzi true lub false, tylko poda wszystkie obiekty powiązane z obiektem marcin relacją rodzic. Uzyskana odpowiedź to:
X= andrzej

Możemy również zadać pytanie przeciwne: Jak nazywają się dzieci marcina. W języku Prolog będzie ono brzmiało:

? – rodzic(marcin, X).

Analizując przedstawiony na początku graf relacji rodzinnych widzimy, że na to pytanie jest więcej niż jedna odpowiedź. Prolog najpierw znajdzie rozwiązanie:

X=ania

Dlaczego Prolog nie wypisuje od razu wszystkich możliwych rozwiązań? Jest to związane z metodą poszukiwania rozwiązania. Jeśli chcemy uzyskać wszystkie rozwiązania, musimy to uzmysłowić naszemu programowi. W Swi–Prolog poszukiwanie pozostałych rozwiązań następuje po naciśnięciu symbolu **„;”**. Po znalezieniu każdego rozwiązania musimy powtórzyć tę operację.

X=krzyś

Więcej rozwiązań już nie ma. Możemy również zadawać pytania bardzo ogólnej natury: Kto jest czyim rodzicem?

? – rodzic(X,Y).

Prolog znajdzie wszystkie pary rodzic dziecko po kolei.

X=zofia

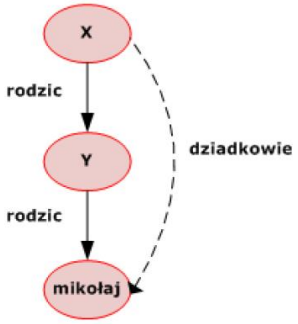
Y=marcin;

X=andrzej

Y=marcin;

X=andrzej

Y=kasia;



Możemy zatrzymać wypisywanie rozwiązań poprzez wpisanie **„.”** zamiast średnika.

Możemy spróbować zadać bardziej skomplikowane pytania np.: Kto jest dziadkiem mikołaja? Ponieważ w bazie wiedzy nie ma faktów opisujących relację bycia dziadkiem, a oczywistym jest możliwość odpowiedzi na to pytanie na podstawie faktów zawartych w naszej bazie wiedzy, musimy skonstruować zapytanie złożone:

a) Kto jest rodzicem mikołaja? Przypuśćmy, ze jest to osoba Y
b) Kto jest rodzicem Y? Przypuśćmy, ze jest to osoba X (poszukiwani dziadkowie)

Zadając pytanie złożone z kilku części, oddzielamy je przecinkiem (odpowiednik spójnika i). Inaczej możemy odczytać pytanie następująco: Znajdź X i Y spełniające warunki:

? – rodzic(Y, mikołaj), rodzic(X,Y).

Uzyskana odpowiedź to:

X=marcin

Y=krzyś

Kolejność zadawanych pytań nie ma znaczenia. Spróbujmy teraz dowiedzieć się jak nazywają się wnuki andrzeja.

? – rodzic(andrzej, X), rodzic(X, Y).

Odpowiedź to:

X=marcin

Y=ania;

X=marcin

Y=krzyś

Inne jeszcze pytanie, które możemy zadać to czy ania i krzyś mają takich samych rodziców. To pytanie również możemy zadać dwuetapowo:

a) Kto jest rodzicem ani? Nazwijmy go X
b) Czy ta sama osoba X jest rodzicem Krzysia?

To pytanie w Prologu przyjmie postać:

? – rodzic(X, ania), rodzic(X, krzyś).

Odpowiedź to:
X= Marcin

3. Reguły w języku Prolog

Reguła – warunkowe stwierdzenie o istnieniu zależności między obiektami:

nazwa_powiaz(obiekt, obiekt, …) if
nazwa_powiaz(obiekt, obiekt, …) and
nazwa_powiaz(obiekt, obiekt, …)
…. and
nazwa_powiaz(obiekt, obiekt, …).

```
lubi(ewa, X) if
        mezczyzna(X) and
        przystojny(X) and
        czyta(X, dostojewski).
```

Słowa kluczowe **„if”** oraz **„and”** mogą zostać zastąpione symbolami: „ **:-**” i „ **,** ”.

Rozszerzmy nasz przykładowy program o informację dotyczącą płci osób, które występują w programie. Możemy to zrobić na kilka sposobów na przykład dodając proste fakty do naszej bazy wiedzy.

```
kobieta(zofia).
kobieta(kasia).
kobieta(ania).
mężczyzna(andrzej).
mężczyzna(marcin).
mężczyzna(krzyś).
mężczyzna(mikołaj).
```

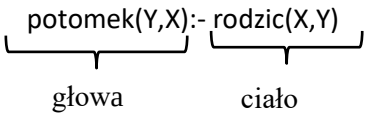
Relacje wprowadzone powyżej to mężczyzna i kobieta. Są to relacje **unarne,** czyli **jednoargumentowe.** Relacja rodzic to relacja binarna (dwuargumentowa), której argumentami jest para obiektów. Relacja jednoargumentowa może służyć do wyrażenia prostych własności rozważanych obiektów. Możemy oczywiście informację o płci osób wyrazić w inny sposób np. wprowadzając relację binarną płeć:

```
płeć(zofia, kobieta).
płeć(krzyś, mężczyzna).
płeć(mikołaj, mężczyzna).
```

Wybór sposobu opisu relacji zależy od twórcy programu. Należy jednak dość dokładnie zastanowić się nad tym, do czego program ma służyć i jakie będą konsekwencje takiej czy innej reprezentacji wiedzy. Dodajmy jeszcze do programu relację potomstwo, jako odwrócenie relacji rodzic. W najprostszy sposób możemy to zrobić poprzez wymienienie każdej pary potomek i rodzic. Powinniśmy jednak unikać dodawania do bazy wiedzy relacji, które można stworzyć bazując na wcześniej zdefiniowanych faktach. Możemy opisać relację potomek następującym logicznym stwierdzeniem:

Dla każdego X i Y,
Y jest potomkiem X jeśli
X jest rodzicem Y.

Powyższe stwierdzenie w sposób sformalizowany możemy zapisać jako tzw. regułę:



Fakt jest to coś co jest bez względu na zachodzące warunki prawdziwe. Natomiast reguła może przyjąć zarówno wartość prawda jak i fałsz w zależności od argumentów. Reguła składa się z dwóch części:

- Prawej strony reguły – części warunkowej,
- Części wynikowej reguły – lewej strony.

Lewa strona reguły jest inaczej nazywana głową, natomiast prawa ciałem.
Sprawdźmy teraz działanie stworzonej przez nas reguły.

? – potomek(kasia, andrzej).

Ponieważ w bazie wiedzy nie ma faktu opisującego relację potomek, do sprawdzenia prawdziwości podanego celu program musi wykorzystać stworzoną regułę. Zmienne X i Y zostaną ukonkretnione:

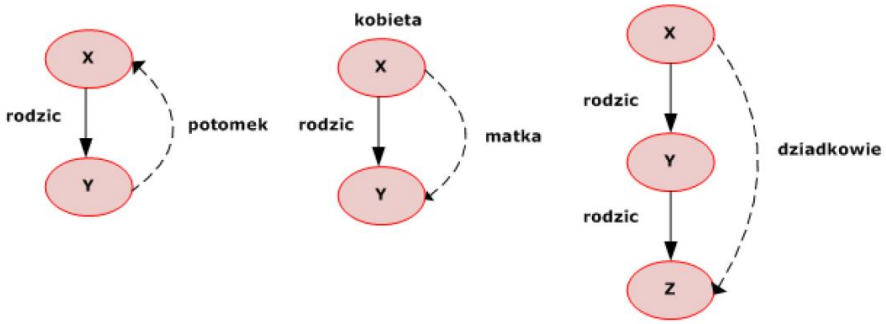
X = andrzej i Y = katarzyna

Po ukonkretnieniu otrzymujemy szczególny przypadek naszej ogólnej reguły.

potomek(katarzyna, andrzej):-rodzic(andrzej, katarzyna).

Teraz Prolog sprawdza, czy część warunkowa reguły jest prawdziwa, czyli czy zachodzi fakt rodzic(andrzej, katarzyna). Nowy cel jest banalny do wykazania, ponieważ jest zapisany jako fakt w programie. To oznacza, że część wynikowa reguły jest również prawdziwa i Prolog odpowie na nasze pytanie true.

Na poniższych grafach przedstawione zostały różne rodzaje relacji:



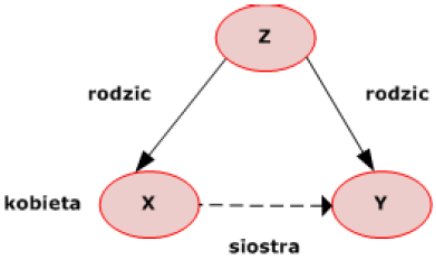
Relację matka można oprzeć na następującym logicznym stwierdzeniu:
Dla każdych X i Y,
X jest matką Y jeśli
X jest rodzicem Y i
X jest kobietą.
Które zapisane w Prologu ma postać poniższej reguły:

matka(X,Y):-rodzic(X,Y), kobieta(X).

Relację dziadkowie można oprzeć na stwierdzeniu:
Dla każdych X i Z,
X jest dziadkiem Z jeśli
X jest rodzicem Y i
Y jest rodzicem Z.
W Prologu powyższe stwierdzenie przybiera postać:

dziadkowie(X, Z):- rodzic(X, Y), rodzic(Y, Z).

Do naszej bazy wiedzy możemy dodać jeszcze relację bycia siostrą.
Dla każdych X i Y,
X jest siostrą Y jeśli
X i Y mają takich samych rodziców i
X jest kobietą.



Zapis sformalizowany:

siostra(X,Y):-
rodzic(Z,X),
rodzic(Z,Y),
kobieta(X).

Sprawdźmy na przykładzie działanie reguły siostra:

? – siostra(ania,krzyś).

Uzyskujemy na to pytanie odpowiedź zgodną z naszymi przewidywaniami, czyli true. Wydawałoby się, że reguła, którą stworzyliśmy działa poprawnie. Jest jednak pewna mała subtelność. Spróbujmy wypisać wszystkie pary rodzeństwa, z których jedno jest kobietą.

?- siostra(X,Y).

Uzyskujemy zaskakującą 1 – wszą odpowiedź: kasia jest siostrą samej siebie.

X = kasia,
Y = kasia;
X = ania,
Y = krzyś;

Dlaczego tak się stało? Konstruując regułę nie wspomnieliśmy, że X i Y to różne osoby. Możemy poprawić tę regułę korzystając z operatora „różne”.

X \= Y.

Zadania do samodzielnego rozwiązania

Zadanie 1.

1. Pobierz plik źródłowy do zależności rodzinnych ⇒ Rodzinka.pl
2. Narysuj graf zależności pomiędzy osobami.
3. Skompiluj źródło (['Rodzinka.pro'].).
    a. Dodaj do źródła następujące reguły.
    b. Potomek child, to relacja odwrotna do rodzica parent.
    c. Matka mother, to rodzic parent płci żeńskiej female.
    d. Dziadek grandparent, to rodzic parent czyjegoś rodzica parent.
    e. Siostra sister, to ktoś kto ma tego samego rodzica parent i jest kobietą female.
    f. Przodek predecessor - poprzez rekurencyjne sprawdzanie poprzednich pokoleń rodziców parent.

Zadanie 2.

Stwórz bazę wiedzy zawierającą kilka osób oraz ich ulubione sposoby spędzania wolnego czasu. Rozszerz stworzoną bazę wiedzy o informacje o wieku i utwórz regułę porównującą wiek.

Zadanie 3.

Stwórz bazę faktów o swojej rodzinie w postaci: osoba(imie, imie_ojca, imie_matki, plec, wiek). Następnie zdefiniuj reguły, które pozwolą sprawdzić czy dla danych dwóch osób zachodzi pokrewieństwo (np.: rodzeństwo, bracia, siostry, brat, siostra).