TERMINOLOGIJA PROGRAMIRANJA V PROLOGU
TERMINOLOGY OF PROLOG PROGRAMMING

INFORMATICA 2/88

Igor Kononenko
Fakulteta za elektrotehniko

Nada Lavrač
Inštitut Jožef Stefan

UDK 681.3.06. Prolog: 001.4

Prolog je novejši programski jezik, ki se vse bolj uveljavlja tudi pri nas. Zaradi velikih razlik v terminologiji tako v svetu kot pri nas je potrebno poenotiti terminologijo, ki se uporablja v logičnem programiranju ter programiranju v prologu. Osnova za delo sta slovarja v (Kononenko 87) in (Kononenko, Lavrač 87). V prispevku smo podali razlike med terminologijama matematične logike in logičnega programiranja ter med prologom in ostalimi programskimi jeziki. Poskusili smo odpraviti tudi nekatere terminološke nesporazume. Podani so tudi prevodi nekaterih izrazov iz angleščine v slovenščino, za katere smo se zedinili na Fakulteti za elektrotehniko in Institutu Jožef Stefan v Ljubljani.

ABSTRACT

Lately, Prolog became a very popular language in the computer programming community. As there is a lot of diversity in the terminology we point out some of the main problems and try to eliminate them by defining the most useful terms. The basis for this paper is a glossary in (Kononenko & Lavrač 87). We present differences between mathematical logic and logic programming, between Prolog and other programming languages and show how these differences affect the terminology. We also point out some misunderstandings regarding the usual terminology and mention some synonyms used in the field of logic programming.

1. RAZLIKE MED MATEMATIČNO LOGIKO IN LOGIČNIM PROGRAMIRANJEM

Prolog (Clocksin, Mellish 81) je najbolj razširjen jezik logičnega programiranja. Ime "PROLOG" je dejansko kratica za 'PROgramming in LOGic'. Ker prolog temelji na matematični logiki, je iz nje privzeta tudi terminologija, kar je v določenih primerih pripeljalo do dvoumnosti.

1. MATHEMATICAL LOGIC AND LOGIC PROGRAMMING

Prolog (Clocksin & Mellish 81) is the most widespread language of the logic programming. PROLOG is in fact an abbreviation for PROgramming in LOGic. As it is based on mathematical logic the similar terminology is adopted which in some cases leads to ambiguities.

Osnovna razlika med stavki v logiki in stavki v prologu je ta, da imajo stavki v prologu tudi proceduralen in ne le deklarativen pomen. Pogosto se uporablja izraz 'predikat' kot sinonim za 'proceduro'. Procedura je del programa (množica stavkov), ki definira predikat z določenim številom argumentov. Definicija je kljub deklarativnemu značaju proceduralna, zato je izraz 'predikat' zavajajoč. V logičnem programiranju zadostuje uporaba izrazov 'procedura', 'vgrajena procedura' ipd. namesto izrazov 'predikat', 'vgrajeni predikat', ipd.

Pogosto se v literaturi mešata izraza 'prilagajanje' in 'unifikacija'. Prilagajanje v prologu je proceduralna implementacija unifikacije iz matematične logike, ki ponavadi zaradi učinkovitosti deluje drugače kot unifikacija. Neopredeljena spremenljivka se lahko prilagodi z izrazom, v katerem sama nastopa, ne more pa se z njim unificirati.

Preglavice povzročajo tudi izrazi 'atom', 'literal' in 'cilj'. Atom je v matematični logiki kratica za atomarno formulo, medtem ko v prologu pomeni vrsto preprostih nedeljivih podatkovnih konstruktov, ki so konstante različne od števil. Tako ustreza atom v prologu kostanti v matematični logiki. Atom v matematični logiki ustreza strukturi v prologu, ki predstavlja klic procedure, če se nahaja v telesu stavka (v pogojnem delu stavka), ali pa sklep, če se nahaja v glavi stavka (v sklepnem delu stavka).

Klicu procedure v telesu prologovega stavka pravimo cilj. Nekateri avtorji (npr. Sterling, Shapiro 86) pravijo glavi stavka tudi cilj. Sintaktično so glava stavka in cilji v telesu stavka pozitivni (nenegirani) literali. Literal je sintaktični konstrukt sestavljen iz imena predikata in seznama argumentov v oklepaju, ločenih z vejicami.

Funktor v matematični logiki določa funkcijo, medtem ko v prologu funkcij ni. Funktor v prologu omogoča konstrukcijo struktur z imenom in mestnostjo (številom argumentov) danega funktorja. Strukture so podatkovni objekti v prologu in nimajo veliko skupnega s funkcijami v matematiki.

Nerodno je, da sta se pri programiranju v

The main difference between logical statements and Prolog clauses is that Prolog clauses have also the procedural meaning while logical statements have only the declarative meaning. The term 'predicate' is often used as a synonym for a 'procedure'. A procedure is a part of a program (a sequence of clauses) that defines a predicate with the given name and arity. Although the definition may be viewed declaratively it is in fact procedural (the order of clauses and the order of goals in clauses are important). The term 'predicate' is therefore misleading. For programming purposes it is thus more convenient to use terms 'procedure', 'built-in procedure', etc. instead of 'predicate', 'built-in predicate', etc.

In the literature is often made no explicit distinction between 'unification' and 'matching'. Matching in Prolog is a procedural implementation of the unification from mathematical logic and usualy differs from it (for the efficiency reasons). It allows an uninstantiated variable to be matched with a term in which it itself appears while it cannot be unified with such a term (this problem is usually referred to as the 'occurs check' problem).

There are difficulties with terms 'atom', 'literal' and 'goal'. In mathematical logic an atom is an abbreviation for 'atomic formula' while in Prolog an atom is a simple data structure, i.e. a constant that is not a number. Therefore an atom in Prolog corresponds to a constant in mathematical logic. An atom in mathematical logic corresponds to a Prolog structure representing a procedure call if it appears in a body of a clause (a condition part) or a conclusion if it appears in the head of a clause (a conclusion part).

A procedure call in a body of a clause is usually called a goal. Some authors (e.g. Sterling & Shapiro 86) call the head of a clause also a goal. Syntactically the head of a clause and goals in the body of a clause are positive (nonnegated) literals. A literal is constructed from a predicate name and a list of arguments enclosed in brackets and separated by commas. A positive literal is a synonym for atomic formula. It is better to use a term 'literal' because of the previously mentioned definition of an atom in Prolog.

prologu v angleščini udomačila izraza 'bound variable' in 'free variable' za spremenljivko, ki ima oziroma nima vrednosti. V matematični logiki se namreč ta dva izraza uporabljata za kvantificirano (vezano) in nekvantificirano (nevezano) spremenljivko. Zato je bolje, da se namesto 'bound' in 'free' uporabljata izraza 'instantiated' in 'uninstantiated'. Možni prevodi so instancirana (neinstancirana), prilagojena (neprilagojena) ali opredeljena (neopredeljena). Zadnji prevod je najustreznejši, saj se vrednost spremenljivke v prologu lahko med izvajanjem bolj ali manj opredeli.

2. RAZLIKE MED PROLOGOM IN OSTALIMI PROGRAMSKIMI JEZIKI

Za razliko od ostalih razširjenih programskih jezikov, ki imajo proceduralni značaj, je prolog je deklarativni jezik. Proceduralni pomen programa v prologu je definiran z načinom izvajanja prologovega interpreterja. Vsekakor je v angleščini pravilneje uporabljati izraz 'execution' za izvajanje prologovega programa namesto ustaljenega izraza 'computation' (računanje), saj je izvajanje prologovega programa dejansko sklepanje (inference) na osnovi pravil in dejstev. Zato računalnikom vse pogosteje pravijo tudi 'inference machine' (stroj za sklepanje) namesto 'computer'.

Med izvajanjem prologovega programa spremenljivke dobijo svojo vrednost s procesom prilagajanja. Pravimo, da vrednost spremenljivke postane (bolj) določena oziroma opredeljena. Ne more pa se vrednost spremenljivke spremeniti na neko povsem drugo vrednost, razen pri avtomatskem vračanju. Pri avtomatskem vračanju vrednost lahko postane manj opredeljena. Za razliko od prologa pa se vrednosti spremenljivk v proceduralnih jezikih lahko spreminjajo. Zato ima izraz 'vrednost spremenljivke' v prologu nekoliko drugačen pomen.

Prireditev v standardnih proceduralnih jezikih pomeni spremembo vrednosti spremenljivke na neko določeno vrednost. V prologu take prireditve ni. Izraz 'prireditev' se uporablja v prologu za vgrajeno proceduro 'is', ki se uporablja za izračun vrednosti aritmetičnega izraza, in prilagoditev izračunane vrednosti s spremenljivko ali konstanto, če je to možno.

In mathematical logic a functor determines a function while in Prolog are no functions. Functors in Prolog are used to construct compound data structures with the given name and arity. Structures in Prolog have not much in common with mathematical functions.

It is awkward that the term 'bound variable' is used for a variable taht has a value and 'free variable' for a variable that doesn't have a value. In mathematical logic bound variable represents a quantified and free variable represents an unquantified variable. In Prolog all variables are universally quantified (except in questions). Therefore it is better to use terms 'instantiated' and 'uninstantiated variable'.

2. PROLOG AND OTHER PROGRAMMING LANGUAGES

Prolog is a declarative language and it largely differs from other popular programming languages which are procedural. The procedural meaning of a Prolog program is defined with the way how it is executed by the Prolog interpreter. It is better to use the term 'execution' than 'computation'. Execution of a Prolog program can be viewed as inference and a computer can be also called an inference machine.

During the execution of a Prolog program variables get their values by matching. We say that the value of a variable becomes (more) specified or determined. While backtracking the value may become less specified. A variable is therefore more or less instantiated. The value of a variable cannot be changed as in other programming languages. Thus in Prolog the term 'the value of a variable' has a slightly different interpretation than in other programming languages.

An assignment in standard procedural languages causes the change of the value of a variable to a certain new value. There is no such assignment in Prolog. The term 'assignment' is used in Prolog for the built-in procedure 'is' which computes the value of the arithmetical expression on the right-hand side and matches the result with the argument on the left-hand side, if this is possible.

Pri programiranju niz (string) predstavlja ponavadi niz znakov med dvema enojnima narekovajima zgoraj. V prologu je niz seznam celih števil, ki ustrezajo ASCII kodam znakov, in ga lahko namesto v standardni notaciji seznama napišemo tudi kot niz ustreznih znakov med dvojnima narekovajima zgoraj. Niz znakov med enojnima narekovajima zgoraj pa v prologu predstavlja atom.

3. DVOUMNOSTI V TERMINOLOGIJI

Prioriteta operatorjev je v prologu definirana z celimi števili (ponavadi od 1 do 1200). Čim manjše je število, tem močneje veže operator in obratno. Ta nekoliko neobičajna definicija dostikrat pripelje do nerazumevanja.

V literaturi se pogosto navajata 'green cut' (zeleni rez) in 'red cut' (rdeči rez) (glej npr. Bratko 86). Brez zelenega reza bi procedura pravilno delovala, le nekoliko počasnejše bi bilo izvajanje zaradi nepotrebnega vračanja. Rdečega reza pa iz procedure ne smemo odstraniti, ker bi bilo izvajanje napačno. Mnogi avtorji napačno ugotavljajo, da zeleni rez ne spremeni deklarativnega pomena medtem ko rdeči rez spremeni deklarativni pomen procedure.

Rez je kontrolni konstrukt, ki nima deklarativnega pomena in zato tudi ne more spremeniti deklarativnega pomena. Vpliva pa na proceduralni pomen dane procedure, saj se zaporedje izvajanja z dodajanjem reza spremeni. Torej tako zeleni kot rdeči rez spremenita proceduralni pomen procedure, le da zeleni rez lahko odstranimo, ne da bi spremenili proceduralno pravilnost procedure, medtem ko z odstranitvijo rdečega reza postane procedura nepravilna.

4. SINONIMI

V seznamu sinonimov so ustreznejši izrazi na desni strani:

atomarna formula - pozitivni literal

objekt - izraz

podatkovna struktura - izraz

In programming the term 'string' usually represents a string of characters enclosed in single quotes. In Prolog a string is represented with a string of characters enclosed in double quotes and is equivalent to a list of integers that correspond to ASCII codes of characters in the string. In Prolog a string of characters enclosed in single quotes is an atom.

3. AMBIGUITIES IN TERMINOLOGY

The precedence of an operator is defined as a strength with which an operator binds its arguments. It is labeled with an integer (usually between 1 and 1200). The greater the label is the weaker the operator binds its arguments and vice versa. This somehow unusual definition often leads to misunderstandings.

In the literature we often meet 'green cut' and 'red cut' (e.g. Bratko 86). Without green cut the procedure would still perform correctly although less efficiently due to unnecessary backtracking. The red cut must not be removed because the execution of the procedure would be incorrect. Many authors incorrectly state that the green cut does not affect the declarative meaning while the red cut does. Cut is a procedural construct with no declarative meaning and thus cannot affect the declarative meaning. It certanly affects the procedural meaning of a procedure because the execution is changed. Therefore both green and red cuts affect the procedural meaning of a procedure. The green cut can be removed without affecting the procedural correctness of the procedure while when the red cut is removed the procedure becomes incorrect.

4. SYNONYMS

In the following list of synonyms the terms at the right-hand side should be preferred.

atomic formula - (positive) literal

bound variable - instantiated variable

predefinirana procedura - vgrajena procedura

sistemska procedura - vgrajena procedura

struktura - sestavljeni izraz

term - izraz

vgrajeni predikat - vgrajena procedura

5. USTALJENI PREVODI NEKATERIH IZRAZOV

Za naslednje prevode izrazov smo se zedinili na Fakulteti za elektrotehniko in Inštitutu Jožef Stefan v Ljubljani:

cut - rez

instance - primer

instance of a term - primer izraza

instantiated - opredeljen

matching - prilagajanje

parent goal - nadrejeni cilj

term - izraz

ZAHVALA

Slovarja v (Kononenko 87) in (Kononenko, Lavrač 87), ki sta osnova tega dela, sta rezultat skupnega prizadevanja ob pomoči mnogih sodelavcev. Zahvaljujeva se Tatjani Janc in Alenu Varšku za pripombe na rokopis.

built-in predicate - built-in procedure

compound term - structure

computation - execution

control predicate - control procedure

data structure - term

evaluable predicate - built-in procedure

free variable - uninstantiated variable

joint variable - shared variable

object - term

predefined operator - built-in operator

predefined procedure - built-in procedure

priority - precedence

system procedure - built-in procedure

stopping condition - boundary condition

unbound variable - uninstantiated variable

AKNOWLEDGEMENTS

Many colleagues helped us in the preparation of a glossary in (Kononenko & Lavrač 87) which is the basis for this paper. We thank Tatjana Janc and Alen Varšek for their remarks on a manuscript.

LITERATURA - REFERENCES

Bratko, I. (1986) Prolog programming for artificial intelligence, Addison-Wesley.

Clocksin, W.F., Mellish, F.G. (1981) Programming in prolog, Springer-Verlag.

Kononenko, I. (1987) Uvod v prolog in zbirka nalog z rešitvami, skripta, Fakulteta za elektrotehniko, Ljubljana.

Kononenko, I., Lavrač, N. (1987) Prolog through examples - A practical programming guide, Sigma Press.

Sterling, L., Shapiro, E. (1986) The art of prolog - Advanced programming techniques, MIT Press.