Грамматики и DCG в Prolog

Сергей Геннадьевич Синица
КубГУ, 2014
sin@kubsu.ru

Формальная грамматика

Формальная грамматика – это набор правил для строк на некотором формальном языке.

Правила определяют переписывание строк и стартовый символ, с которого начинается переписывание. Таким образом, грамматику можно представлять себе как генератор языка. Грамматику также используют для распознавания – создание функций, которые определяют, принадлежит ли строка данному языку или является грамматически некорректной. Для такого распознавания строятся автоматы.

Парсинг – это процесс распознавания строки на естественном языке с помощью разбиения её на множество символов и анализа каждого на соответствие грамматики языка.

Формальная грамматика

Пример:
S -> aSb
S -> ba
$\{a^n bab^n | n>=0\}$

Грамматика – это набор (N,E,P,S)
 Где N – нетерминальные символы (никакой из них не будет появляться в словах языка)
 E – терминальные символы
 P – множество продуционных правил
 $(E \cup N)^*(E \cup N)^* \rightarrow (E \cup N)^*$
 S – стартовый символ.

Пример

$N=\{S,B\}$

$E=\{a,b,c\}$

$S \rightarrow aBSc$

$S \rightarrow abc$

$Ba \rightarrow aB$

$Bb \rightarrow bb$

$h(G)=\{a^n b^n c^n \mid n>=1\}$

Пример

$N=\{S,B\}$
$E=\{a,b,c\}$
$S \rightarrow aBSc$
$S \rightarrow abc$
$Ba \rightarrow aB$
$Bb \rightarrow bb$
$h(G)=\{a^n b^n c^n \mid n>=1\}$

Типы грамматик

Type0 – неограниченные грамматики

Type1 – контекстно-зависимые

Type2 – контекстно-свободные

Type3– регулярные

DCG

Definite clause grammar (DCG) – это способ представления грамматики естественного или формального языка в языке логического программирования, таком как пролог.

Правила представляются с помощью логики первого порядка. Правила можно представлять как аксиомы, тогда корректность выражения и тот факт, что оно имеет некоторое дерево парсинга, можно представить как теоремы которые следуют из этих аксиом.

Таким образом распознавание и парсинг выражений в языке становится обычным доказательством утверждений, таки как утверждений логического программирования.

1972

Alan Colmerauver
Philipe Roussel
Fernando Pereira
David Warner

Пример

```
sentence --> noun_phrase, verb_phrase.
noun_phrase --> det, noun.
verb_phrase --> verb,noun_phrase.
det --> [the].
det --> [a].
noun --> [cat].
noun --> [bat].
verb --> [eats].

?- sentence(X, Y).
```

Пример

```
sentence(S1,S3):- noun_phrase(S1,S2),
verb_phrase(S2,S3).

noun_phrase(S1,S3):- det(S1,S2),
noun(S2,S3).

verb_phrase(S1,S3):-
verb(S1,S2),noun_phrase(S2,S3).

det([the|X],X).

det([a|X],X).

noun([cat|X],X).

noun([bat|X],X).

verb([eats|X],X).
```

Пример

```
s --> symbols(Sem,a),symbols(Sem,b),symbols(Sem,c).
symbols(end,_) --> [].
symbols(s(Sem),S) --> [S],symbols(Sem,S).

?- s(X, Y).
```

Пример

```
sentence --> pronoun(subject),verb_phrase.
verb_phrase --> verb,pronoun(object).
pronoun(subject) --> [he].
pronoun(subject) --> [she].
pronoun(object) --> [him].
pronoun(object) --> [her].
verb --> [likes].

?- sentence(X, Y).
```

Пример на русском

```
предложение -->
местоимение(именительный),глагольная_фраза.

глагольная_фраза --> глагол,местоимение(родительный).

местоимение(именительный) --> [он].

местоимение(именительный) --> [она].

местоимение(родительный) --> [его].

местоимение(родительный) --> [ее].

глагол --> [любит].


?- предложение(X, Y).
```

Пример

```
sentence(s(NP,VP)) --> noun_phrase(NP), verb_phrase(VP).
noun_phrase(np(D,N)) --> det(D), noun(N).
verb_phrase(vp(V,NP)) --> verb(V),noun_phrase(NP).
det(d(the)) --> [the].
det(d(a)) --> [a].
noun(n(cat)) --> [cat].
noun(n(bat)) --> [bat].
verb(v(eats)) --> [eats].

?-sentence(Parse_tree,[the,bat,eats,a,cat],[]).
```

Пример

```
integer(I) --> digit(D0), digits(D), {number_chars(I, [D0|D])}.

digits([D|T]) --> digit(D),!, digits(T).

digits([]) --> [].

digit(D) -->[D], {code_type(D,digit)}.% если символ D имеет тип digit


?-phrase(integer(X), ”42 times”, Rest).

X = 42,

Rest = [32, 116, 105, 109, 101, 115].
```

DCG

Тело грамматического правила может состоять из трех типов термов:

1. Составной терм, интерпретируемый как ссылка на грамматическое правило
2. Код между фигурными скобками, интерпретируемый как обычный код prolog
3. Список, интерпретируемый как последовательность литералов

Набор грамматических правил вызывается в SWI-prolog встроенными предикатами.

```
Phrase(+RuleSet,+InputList).

Phrase(+RuleSet,+InputList,-Rest).% Rest унифицируется с оставшимися токенами, если…
```

Простая грамматика для английского языка

```
s --> np,vp.
np --> pn.
np --> d,n,rel. % определитель, существительное, отношение
vp --> tv,np. % транзитивный глагол и существительное
vp --> iv.  % нетранзитивный глагол
rel --> [].
rel --> rpn,vp. % относительное местомение и глагольная фраза
pn --> [PN],{pn(PN)}.
pn(mary).
pn(herry).
rpn --> [RPN],{rpn(RPN)}.
rpn(that).
rpn(which).
rpn(who).
iv --> [IV],{iv(IV)}.
iv(runs).
iv(sits).
```

```
d --> [DET],{d(DET)}.
d(a).
d(the).
n --> [N],{n(N)}.
n(book).
n(girl).
n(boy).
tv --> [TV],{tv(TV)}.
tv(gives).
tv(reads).

?- s([the,boy,who,sits,reads,a,book],[]).
true
?- phrase(s, [the,boy,who,sits,reads,a,book]).
true
?- s([herry,reads],[]).
false.
```

Задание 9 а)

С помощью DCG построить контекстно-зависимую грамматику на русском языке.

Разностные списки и DCG

В процессе распознавания в прологе фразы представляются в виде разностных списков терминальных символов. Каждая фраза оформлена в виде двух списков и представляет собой разность между ними.

Разностные списки и DCG

У нас есть робот, принимающий команды «вверх» и «вниз» (up, down). Робот воспринимает фразы в виде последовательности таких команд. Up или down образуют элементарный шаг. Любая последовательность элементарных шагов образует движение. Пусть move – это движение, а step – это шаг. Таким образом, любое движение move состоит либо из одного шага, либо из любого шага, за которым следует любое движение. Это можно выразить с помощью следующей грамматики:

move --> step.

move --> step,move.

step --> [up].

step --> [down].

Разностные списки и DCG

```
?- move([up,up,down],[]).
true .

?- move([up,up,left],[]).
false.

?- move([up,X,up],[]).
X = up ;
X = down ;
false.

?- listing(move).
move(A, B) :-
        step(A, B).
move(A, C) :-
        step(A, B),
        move(B, C).

true.
```

```
move --> step.
move --> step,move.
step --> [up].
step --> [down].
```

Разностные списки и DCG

Грамматики DCG являются синтаксическим сахаром для записи обычных предикатов пролога. При компиляции преобразуются в предикаты автоматически.

```
move(List,Rest):-
    step(List,Rest).
move(List1,Rest):-
    step(List1,List2),
    move(List2,Rest).
step([up|Rest],Rest).
step([down|Rest],Rest).
```

Предикат `move` истинен, если разность между списками List и Rest представляет собой допустимое движение робота.

Разностные списки и DCG

Примеры, когда предикат move истинен:

```
move([up,down],[]).
move([up,down,a,b,c],[a,b,c]).
move([up,down,up],[down,up]).
```

Разностные списки используются потому, что с помощью них эффективно реализуется конкатенация списков.
Подробнее см. книгу И. Братко

Сосредоточимся на предикате:

```
move(List1,Rest):-
  step(List1,List2),
  move(List2,Rest).
```

Разность списка в List1, Rest означает движение, если
1) Разность между списками List1 и List2 соответствует шагу
2) А разность между List2 и Rest соответствует движению

Пара (List1,Rest) представляет собой конкатенацию списков, оформленных в виде пар (List1,List2) и (List2,Rest).

Алгоритм преобразования DCG грамматики в форму работы с разностными списками

n --> n1,n2,… nn.

если все компоненты n1,n2.. этого правила представляют собой нетерминальные символы, то оно транслируется в

```
n(List1,Rest):-
n1(List1,List2),
…
nn(Listn,Rest)
```

Если какой-либо из компонентов является терминальным символом

(в квадратных скобках по правилам DCG), то он обрабатывается иначе.

Он не появляется в виде одной из целей предложений, а непосредственно вставляется в соответствующий список.

n --> n1,[t2],n3,[t4].

```
n(List1,Rest):-
  n1(List1,[t2|List3])
  n3(List3,[t4|Rest]).
```

Обработка ЕЯ, идеоматический интерфейс в мире блоков

Идиома – это некоторая фраза, с помощью которой человек общается с программой.

Представление:

on(a,b).

on(b,c).

on(c,table).

Идиомы:

{please} place <put> {block} X on <onto> {block} Y,W on Z...

I want <would like> X on Y, W on Z,...

I want <would like> you to put <please> ...

Can <could> <would> you {please} put <place> X on Y ...

Задание 9 б)

Для мира блоков сформулировать идиомы:

«what is block X sitting on?»,

«which blocks are on the table?»,

«put all blocks in single pile.»,

«put the block on top of X on top of block Y (or on the table).»

Задание 9 в)

Для экспертной системы 3 сформулировать не менее 5 идиом вида:

«Какие животные умеют летать?»,

«Кто кормит детенышей молоком?» и т.д.

Строки из базы знаний по возможности не должны дублироваться в грамматике.

=/=

Конец