Курс лекции по дисциплине YaPZ7302 Языки представления знаний

Лекция 5. Родовые функции

Цель: Изучение родовых функций.

План:

1. Понятие «родовые функции»
2. Методы. Явный и неявный методы
3. Синтаксис конструктора defgeneric
4. Синтаксис конструктора defmethod
5. Заголовок родовой функции
6. Индексы методов
7. Ограничения параметров метода
8. Пример. Избыточные ограничения типов
9. Пример. Использование ограничения запросом с двумя параметрами
10. Пример. Перегрузка системной функции +
11. Групповой параметр

Помимо функций CLIPS предоставляет еще один механизм, поддерживающий процедурную парадигму представления знаний, — *родовые функции.* Родовые функции подобны функциям, созданным с помощью конструктора deffunction. Они также могут использоваться для определения нового процедурного кода в CLIPS и могут быть вызваны как любые другие функции. Однако, в отличие от простых, родовые функции являются более мощным средством обработки данных, т. к. они способны выполнять различные наборы действий в зависимости от числа и типа полученных в данный момент аргументов. Родовые функции подобны перегруженным операторам или функциям языка C++. Например, функция + может выполнять как операцию конкатенации строк, так и простое арифметическое сложение чисел. Родовые функции определяются с помощью конструкторов defgeneric и defmethod, которые подробно будут описаны В данной главе. Родовые функции обычно состоят из нескольких компонентов, называемых *методами.* Каждый метод

определяет последовательность действий, выполняющих обработку различных наборов аргументов. Родовая функция, которая имеет более одного метода, называется *перегруженной.*

Родовые функции могут содержать как системные методы, так и методы, определенные пользователем. Например, перегруженная функция + состоит из двух методов:

- *неявный* метод, являющийся системной функцией, которая обрабатывает арифметическое сложение;
- *явный* (определенный пользователем) обработчик сложения строк.

CLIPS не позволяет использовать функции, созданные с помощью конструктора deffunction, в качестве методов родовых функций. Конструкторы deffunction предоставляют возможность добавления в CLIPS новых функций без использования концепции перегрузки. Родовая функция, имеющая только один метод, по своему поведению идентична функции, созданной с помощью конструктора deffunction.

В большинстве случаев методы родовых функций не вызываются напрямую, несмотря на то, что CLIPS предоставляет такую возможность с помощью функции call-specific-method. CLIPS самостоятельно распознает вызов родовой функции и использует аргументы для поиска и запуска соответствующего метода. Этот процесс называется *родовым связыванием.*

Родовая функция состоит из *заголовка* (подобного предварительному объявлению функции) и *нескольких методов* (число которых теоретически может быть равным нулю). Заголовок родовой функции может быть либо явно определен пользователем, либо не явно объявлен определением метода. Объявление метода состоит из 6 элементов:

- имя (которое отображает, к какой основной функции относится метод);
- необязательный индекс;
- необязательные комментарии;
- набор ограничений для параметров;

- необязательный групповой параметр для обработки переменного числа аргументов;
- последовательность действий или выражений, которые будут выполнены в заданном порядке в момент вызова метода.

Ограничения параметров используются в процессе родового связывания для определения применимости метода к некоторому набору аргументов. Для создания заголовка родовой функции служит конструктор defgeneric, а для создания каждого нового метода родовой функции — конструктор defmethod.

Определение. Синтаксис конструктора defgeneric

(defgeneric <имя-функции>
[комментарии])

Определение. Синтаксис конструктора defmethod

(defmethod <имя-функции>
[<индекс>]
[<комментарии>]
(<ограничения-параметра>*
[<групповой-параметр>])
<действие>*)
<ограничения-параметров> ::= <простая-переменная> |
(<простая-переменная>
<ограничение-по-типу> *
[<ограничение-по-запросу>])
<групповой-параметр> ::= <составная-переменная> |
(<составная-переменная>
<ограничение-по-типу>*
[<ограничение-по-запросу >])
<ограничение-по-типу> ::= <имя-класса>
<ограничение-по-запросу> ::= <глобальная-переменная> |
<вызов-функции>

Родовая функция должна быть либо явно объявлена конструктором defgeneric, либо одним из своих методов, до того как она будет вызвана из другой функции, правила или обработчика сообщения. Исключение составляют рекурсивные родовые функции.

Заголовок родовой функции

Родовая функция однозначно определяется по своему имени. В случае использования родовой функции в правиле, другой функции, обработчике сообщения или до объявления первого метода родовой функции необходимо явное создание заголовка родовой функции (с помощью конструктора defgeneric). В других случаях объявление первого метода родовой функции само не явно создает заголовок. Например, в случае если две родовые функции имеют методы, которые взаимно вызывают друг друга (взаимная рекурсия родовых функций), то необходимо явное объявление заголовков.

Индексы методов

Метод родовой функции однозначно определяется либо по имени и индексу, либо по имени и ограничениям параметров. Каждому методу родовой функции назначается целый индекс, уникальный в группе всех методов этой функции. В случае если определен новый метод, который точно совпадает по ограничениям параметров и имени с другим методом этой функции, CLIPS автоматически заменит им существующий метод. Однако малейшее несовпадение в ограничениях параметров приведет к созданию нового метода. Если нужно заменить некоторый уже существующий метод новым методом с другими ограничениями параметров, то в определении нового метода необходимо явно указать индекс существующего метода. При этом ограничения параметров нового метода должны не совпадать с ограничениями других методов той же функции. Если индекс не задан, CLIPS автоматически назначает индекс, который еще не был использован другими методами родовой функции. Индекс, соответствующий методам родовой функции, можно определить, например, с помощью команды list-defmethods.

Ограничения параметров метода

Каждый параметр метода может быть определен с некоторыми произвольными комплексными *ограничениями* или без них. Ограничения параметров применяются к аргументам родовой функции во время работы программы для определения того, какой именно метод должен принимать эти аргументы. Параметр может иметь два типа ограничений: *ограничение типа* и *ограничение запросом*. Ограничение типа содержит классы аргументов, которые может принимать параметр. Ограничение запросом является определенным пользователем условным выражением, которое должно удовлетвориться для аргументов в момент вызова функции. Совмещение ограничений и их сложность прямо влияет на скорость родового связывания.

Если параметр не имеет ограничений, это означает, что метод может принимать любые значения в качестве данного аргумента. Однако каждый метод родовой функции должен иметь определенные ограничения параметров, которые будут отличать его от других методов той же родовой функции. В противном случае, процесс родового связывания не сможет определить, какой именно метод необходимо вызывать. В случае если процесс родового связывания не смог подобрать соответствующий метод для некоторого набора аргументов, CLIPS сгенерирует ошибку.

Ограничение типа позволяет пользователю определить список типов (классов), один из которых должен соответствовать (или являться суперклассом) аргументу родовой функции. Если в используемой вами конфигурации CLIPS не установлен COOL, то в качестве ограничения типа будут доступны только следующие типы (классы): object, primitive, lexeme, symbol,

STRING, NUMBER, INTEGER, FLOAT, MULTIFIELD, FACT-ADDRESS И EXTERNAL-ADDRESS.

В *гл. 11* все эти системные классы будут описаны подробно. Если COOL установлен, то, помимо перечисленных выше, будут доступны классы

INSTANCE, INSTANCE-ADDRESS, INSTANCE-NAME, USER, INITIAL-OBJECT, а также любой определенный пользователем класс. Родовая

функция, которая использует только первую группу типов в своих методах, будет работать как с установленным COOL, так и без него. Классы, заданные в ограничении типа, должны быть определены до определения приоритета метода . CLIPS не поддерживает избыточность в списке ограничений типов аргументов методов. Например, для представленного ниже метода ограничения типов аргументов избыточны, т. к. класс INTEGER — подкласс NUMBER.

Пример. Избыточные ограничения типов

```
(defmethod foo ((?a INTEGER NUMBER)))
```

Если ограничение типа удовлетворяется для некоторого аргумента, то к нему будет применено ограничение запросом (если оно задано). Ограничение запросом должно быть либо глобальной переменной, либо вызовом функции. CLIPS вычисляет заданное выражение, и если полученный результат не равен FALSE — ограничение полагается удовлетворенным.

Так как ограничения запросом вычисляются каждый раз при поиске соответствующего метода, они не могут использоваться для произведения какого-нибудь побочного действия, потому что вычисляемое ограничение может принадлежать методу, неподходящему к данной конкретной ситуации.

Поскольку все ограничения просматриваются слева направо, запрос с несколькими параметрами должен быть записан после ограничений типов всех используемых параметров. Этим правилом обеспечивается условие удовлетворения ограничений типов всех необходимых параметров. Например, метод из примера не вычисляет ограничение запросом до тех пор, пока не удовлетворятся два соответствующих ограничения типа.

Пример. Использование ограничения запросом с двумя параметрами

```
(defmethod foo ((?a INTEGER ) (?b INTEGER(> ?a ?b))))
```

Если аргумент удовлетворяет всем своим ограничениям, то считается, что он применим для данного метода. Если все аргументы родовой функции применимы к ограничениям метода, метод полагается применимым для

данного набора аргументов. В случае если существует более одного метода, применимого для некоторого набора аргументов, процесс родового связывания определяет некоторый упорядоченный список этих методов и использует первый метод из этого списка.

В примере первое обращение к родовой функции + вызовет выполнение системной функции + — неявный метод, выполняющий арифметическое сложение. Второй вызов приведет к выполнению явного метода родовой функции, осуществляющего конкатенацию строк, т. к. оба аргумента являются строками. Третий вызов сгенерирует ошибку, поскольку явный метод для конкатенации строк принимает только два аргумента, а неявный метод для арифметического сложения не принимает строковые аргументы вообще.

Пример. Перегрузка системной функции +

```
(defmethod + ( (?a STRING) (?b STRING))
(str-cat ?a ?b) )
(+ 1 2)
(+ "foo" "bar")
(+ "foo" "bar" "woz")
```

Групповой параметр

В зависимости от того, задан ли групповой параметр, метод может принимать точное число параметров или число параметров не меньшее, чем некоторое заданное. Обязательные параметры определяют минимальное число аргументов, которое должно быть передано методу при его вызове. В действиях, выполняемых методом, можно ссылаться на каждый из этих параметров так же, как на обычные переменные, содержащие простые значения. Если был задан групповой параметр, то метод может принимать любое количество аргументов — большее или равное минимальному числу аргументов. Если групповой параметр не задан, то метод может принимать число аргументов, равное числу обязательных параметров. Все аргументы метода, которые не соответствуют обязательным параметрам, группируются в одно значение

составного поля. Ссылаться на это значение можно, указывая символ группового параметра. Для работы с групповым параметром могут использоваться стандартные функции CLIPS, предназначенные для работы с составными полями, такие как length и nth. Определение метода может содержать только один групповой параметр.

Ограничения типом и запросом могут применяться к аргументам, сгруппированным в групповом параметре, аналогично тому, как они употребляются с основными параметрами метода. Такие ограничения задаются для каждого отдельного поля результирующего составного значения (а не для всего значения). Выражение, содержащее групповой символ, может быть применено в запросе.

Дополнительно в запросе может быть использована специальная переменная ?current-argument для ссылки на отдельные аргументы, объединенные групповым символом. Это переменная существует только в ограничении запросом и не имеет значения в теле метода.

Контрольные вопросы:

1. Назначение родовых функций.
2. Синтаксис конструктора defgeneric
3. Пример перегрузки системной функции +.

Литература:

1. А. П. Частиков Т. А. Гаврилова Д. Л.Белов. РАЗРАБОТКА ЭКСПЕРТНЫХ СИСТЕМ. СРЕДА CLIPS. СПб: БХВ-Петербург, 2003

Ключевые слова:

Родовые функции, *defgeneric*, заголовок родовой функции, индексы методов.