Программная среда разработки CLIPS: Назначение и основные возможности. Базовые типы данных и представление фактов.

Общая характеристика среды CLIPS

Среда CLIPS (C Language Integrated Production System) предназначена для построения ЭС и поддерживает три основных способа представления знаний:

- продукционные правила для представления эвристических знаний;
- функции для представления процедурных знаний;
- объектно-ориентированное программирование.

Поддерживаются 6 основных черт ООП:

классы, обработчики сообщений, абстракции, инкапсуляция, наследование и полиморфизм.

Правила могут сопоставляться с объектами и фактами. Приложения могут разрабатываться с использованием только правил, только объектов или их комбинации.

Предусмотрена интеграция с другими средствами:

CLIPS может вызываться из процедурных языков, выполнять свои функции и затем возвращать управление вызвавшей программе.

С другой стороны, процедурный код может быть определен как внешняя функция и вызван из CLIPS.

Первоначально CLIPS поддерживал возможность для представления только правил и фактов.

Начиная с версии 6.0, правила могут сопоставляться с объектами, так же как с фактами.

Более того, объекты могут использоваться без правил путем посылки сообщений и в этом случае отпадает необходимость в машине вывода, если используются только объекты.

Базовые типы данных и представление фактов в системе CLIPS

В CLIPS поддерживаются восемь базовых типов данных:

целые (integer) и *вещественные* (float) числа,

символьные (symbol) и *строковые* (string) данные,

внешний адрес (external-address),

адрес факта (fact-address),

имя экземпляра (instance-name) и *адрес экземпляра* (instance-address).

Целые числа состоят из знака (необязательного для положительных чисел) и последовательности десятичных цифр.

Например: 27; +125; -38.

Вещественные числа содержат мантиссу и необязательный порядок, состоящий из символа "e" и целого числа.

Например: 12.0; -1.59; 237e3; -32.3e-7.

Символьное значение задается продолжающейся до ограничителя последовательностью отображаемых ASCII-символов.

Ограничителями являются:

любой неотображаемый ASCII-символ (пробел, табуляция, возврат каретки, перевод строки),

кавычка, открывающая и закрывающая скобки, амперсанд (&), вертикальная черта (|), знак «меньше» (<) и тильда (~).

Точка с запятой (;) также является ограничителем и используется для указания на начало комментария.

Ограничители не включаются в символьное значение, за исключением символа "<", стоящего в начале значения.

Символьное значение не может начинаться с символа "?" или пары символов "$?", но может содержать их внутри себя.

CLIPS различает символы нижнего и верхнего регистров, поэтому abc и ABC воспринимаются как два разных значения.

Примеры символьных значений:

bad_value; 456-93-039; @+=%.

Строковое значение – это заключенная в кавычки последовательность (в том числе пустая) отображаемых символов.

Чтобы включить кавычки в строковое значение, перед ними необходимо поставить символ "\".

Чтобы вставить в строковое значение символ "\", перед ним необходимо поместить еще один символ "\", т.е. записать два символа "\" подряд.

Примеры строковых значений:

"abc"; "a & b"; "a\"quote"; "fgs\\85".

Внешний адрес – адрес внешней структуры данных, возвращаемый интегрированной в CLIPS функцией (написанной на языках C или Ada).

Этот тип данных создается только как результат вызова функции, его невозможно специфицировать вводом значения.

Отображаемое представление внешнего адреса:

<Pointer-XXXXXX>,

где XXXXXX – внешний адрес.

Адрес факта используется для ссылки на факты.

Факт представляет собой список атомарных значений, на которые можно ссылаться либо позиционно (в упорядоченных фактах), либо по имени (в неупорядоченных фактах).

Отображаемый формат адреса факта:

<Fact-XXX>,

где XXX – индекс факта.

Имя экземпляра используется для ссылки на экземпляры классов.

Экземпляр представляет собой *объект*, являющийся представителем некоторого *класса*.

Объектами в CLIPS по определению являются целые и вещественные числа, символьные и строковые значения, многоместные значения, внешние адреса, адреса фактов или экземпляры определенных пользователем классов (создаваемых с помощью конструкции defclass).

Имя экземпляра представляется заключенным в квадратные скобки символьным типом.

Например: [pump-1]; [foo]; [+++]; [123-890].

Скобки не являются частью имени, а только указывают тип значения.

Адрес экземпляра может быть получен путем связывания значения, возвращаемого функцией `instance-address`, или связывания переменной с экземпляром, сопоставляемым с объектным образцом в левой части правила.

Невозможно специфицировать адрес экземпляра вводом значения.

Отображаемое представление адреса экземпляра в CLIPS:

<Instance-XXX>,

где XXX – имя экземпляра.

На экземпляры определяемых пользователем классов можно ссылаться либо по имени либо по адресу.

Адреса экземпляров должны использоваться, когда критично время решения.

В CLIPS место, занимаемое одним значением базового типа данных, называется *полем* (field).

Все значения базовых типов являются *одноместными* (single-field value).

Последовательность из нуля или более одноместных значений рассматривается как *многоместное значение* (multifield value).

Они отображаются в скобках, причем одноместные значения разделяются пробелами.

Примеры многоместных значений: (a 123); (); (x 3.0 "red" 567).

Одной из основных форм представления информации в CLIPS-системах являются *факты*.

Они используются правилами для вывода новых фактов из имеющихся.

Все текущие факты в CLIPS помещаются в список фактов (fact-list).

По формату представления в CLIPS выделяют два типа фактов: *упорядоченные* и *неупорядоченные*.

Упорядоченный факт состоит из заключенной в скобки последовательности одного или более разделенных пробелами полей, причем первое поле должно быть символьного типа, а остальные могут быть любыми базовыми типами данных.

Первое поле специфицирует отношение, которое применяется к остальным полям факта.

Примеры упорядоченных фактов:

(высота 100); (студент Сидоров); (отец Иван Петр); (однокурсники Иванов Петров Сидоров).

Для работы с фактами используются следующие команды:

assert – добавляет факт в факт-список;

retract – удаляет факт из списка;

modify – модифицирует список;

duplicate – дублирует факт.

Например, команда

(assert (length 150) (width 15) (weight “big”))

добавляет в список фактов три факта, каждый из которых состоит из двух полей.

Эти команды могут исполняться как в режиме командной строки, так и включаться в CLIPS-программы.

Команды retract, modify и duplicate требуют, чтобы факты были идентифицированы с помощью индекса факта (fact-index) либо адреса факта (fact-address).

Индекс факта представляет собой уникальный целочисленный индекс, приписываемый факту всякий раз, когда факт добавляется (или модифицируется).

Индексация фактов начинается с нуля и инкрементируется при каждом новом или измененном факте.

Идентификатор факта (fact-identifier) представляет собой краткую нотацию для отображения факта.

Он состоит из символа “f”, за которым через тире следует индекс факта. Например, f-10 ссылается на факт с индексом 10.

Для задания исходного множества фактов используется конструкция `deffacts`, имеющая следующий синтаксис:

(`deffacts` <имя_группы_фактов> ["<комментарий>"] <факт>*),

где <имя_группы_фактов> – идентификатор символьного типа;

<комментарий> – необязательное поле комментария;

<факт>* – произвольная последовательность фактов, записанных через разделитель.

Пример использования конструкции `deffacts`:

```
(deffacts stud "Студент"
 (student name John)
 (student spec “COMPUTER”)).
```

Факты, определенные конструкцией `deffacts`, добавляются в список фактов всякий раз при выполнении команды reset.

Неупорядоченные факты представляют собой список взаимосвязанных именованных полей, называемых *слотами*.

Наличие имен полей позволяет выполнять доступ к полям по именам, в отличие от упорядоченных фактов, где поля специфицируются своим местоположением в факте.

Существует два типа слотов: одиночные и мультислоты.

Одиночный слот (или просто слот) содержит единственное поле, тогда как мультислот может содержать любое число полей.

Для спецификации состава неупорядоченных фактов (т.е. содержащихся в них слотов) используются *шаблоны*, которые задаются конструкцией `deftemplate`.

Синтаксис конструкции deftemplate определен ниже:

```
(deftemplate <имя шаблона>
["<комментарий>"]
   <определение слота-1>
     .          .          .
   <определение слота-N>)
```

Пример шаблона, содержащего три одиночных слота, представлен ниже:

```
(deftemplate object “Шаблон объекта”
 (slot name)
 (slot location)
 (slot weight))
```

Пример конкретного неупорядоченного факта на основе данного шаблона представлен ниже:

```
(object (name car) (location 100) (weight 600)).
```