

Конспект лекції № 8

Тема № 8. СЕРЕДОВИЩЕ CLIPS. ФУНКЦІОНАЛЬНІ МОЖЛИВОСТІ

Міжпредметні зв'язки: Зв'язок із елементами знань і умінь таких навчальних дисциплін як „Вступ до фаху” та „Інформатика”, «Методологія тестування програмного забезпечення».

Мета лекції: ознайомитися з функціональними можливостями CLIPS.

План лекції

1. Основні елементи мови. Абстракції даних.
2. Подання знань. Об'єктно-орієнтовані можливості CLIPS.

Опорні поняття: ознайомитися з експертною оболонкою CLIPS, охарактеризувати основні переваги та недоліки, розглянути основи функціонування CLIPS.

Інформаційні джерела:

Основна та допоміжна література:

- Федорчук Є.Н. Програмування систем штучного інтелекту. Експертні системи / Є.Н.Федорчук, Вид-во Львівської політехніки, 2012. - 168 с.
- Баклан І.В. Експертні системи. Курс лекцій /Навчальний посібник. - К.: НАУ, 2012. – 132с.
- Експертні системи в міжнародних відносинах / Ю.О.Лунь, А.М. Козел, С.М. Ніколаєв. – Львів: Видавництво Львівської політехніки, 2011. – 196с.

Інтернет ресурси:

- Концепции общей теории информации. Статьи. Наука и техника.[Електрон. ресурс]. – Режим доступу: <http://www.n-t.org/tp/ng/oti.htm>
- Общая теор. информации.[Електрон. ресурс]. – Режим доступу: <http://www.inteltec.ru/publish/articles/textan/ibook.shtml>
- Теория информации. [Електрон. ресурс]. – Режим доступу: <http://www.inftech.webservis.ru/it/information/>

Навчальне обладнання: ТЗН, презентація тощо.

ВИКЛАД МАТЕРІАЛУ ЛЕКЦІЇ

1. Основні елементи мови. Абстракції даних.

Синтаксис мови CLIPS можна розбити на три основних групи елементів, призначених для написання програм:

- примітивні типиданих;
- функції, що використовуються для обробки даних;
- конструктори, призначені для створення таких структур мови, як факти, правила, класи йт.д.

Розглянемо кожну із цих трьох груп більш докладно.

Типиданих

CLIPS підтримує 8 *примітивних типів даних*: **float**, **integer**, **symbol**, **string**, **external-address**, **fact-address**, **instance-name**, **instance-address**.

Для *зберігання* чисельної інформації призначаються типи **float** й **integer**, для *символічної* - **symbol** й **string**.

Число в CLIPS може складатися тільки із символів цифр (0-9), десяткової крапки (.), знака (+ або -) і експонентного символу (e) з відповідним знаком, у випадку подання числа в експонентній формі. Нижче наведені приклади припустимих в CLIPS подань цілих і речовинних типів:

Приклад 1. Подання чисел в CLIPS

Цілі: **237 15 +12-32**

Речовинні: **237e3 15.09 +12.0 -32.3e-7**

Визначення цілого значення можна представити в такий спосіб:

Визначення 1. Подання цілогочисла

**<ціле> ::= [+ | -]<цифра>+
<цифра>::=0|1|2|3|4|5|6|7|8|9**

Речовинне значення має наступний синтаксис:

Визначення 2. Подання речовинного числа

**<речовинне> ::= <ціле>
<експонента> | <ціле> . [експонента] |
<беззнаковое-целое> [експонента] |
<ціле> . <беззнаковое-целое> [експонента] <беззнаковое-целое> ::=
<цифра>+<експонента>: :=e|E
<ціле>**

Якщо послідовність символів не відповідає наведеним вище визначенням цілого або речовинного числа, то дана послідовність сприймається CLIPS як значення типу **symbol**.

Значенням типу **symbol** може бути будь-яка послідовність символів, що починається з будь-якого не керуючого ASCII-символу. Значення типу **symbol** закінчується *обмежником*.

Обмежниками є будь-які невідображувані символи (наприклад, пробіл, символ табуляції або переходу на інший рядок), подвійні лапки, що

відкривають або закриває кругла дужка, символи &, |, < й ~. Крапка з коми (;) є символом початку коментарів і також може обмежувати значення типу symbol. Обмежувачі-символи-обмежники не можуть утримуватися в значенні symbol, за винятком <, що може бути першим символом значення. Значення типу symbol не може починатися із символу ? або \$?, але може містити ці символи. CLIPS є мовою, чутливим до регістра.

Значення типу, **string** являє собою рядок символів, укладену в подвійні лапки. Символ подвійних лапок також може бути включений у рядок. Для цього перед символом " необхідно поставити символ зворотної косої риси (\). Для включення в рядок символу зворотної косої риси необхідно використати два послідовних символи \. Приклади припустимих значень string наведені нижче:

[Продовжити перегляд](#)

2. Подання знань. Об'єктно-орієнтовані можливості CLIPS.

CLIPS підтримує як *евристичну*, так і *процедурну парадигму подання знань*. Обидві ці парадигми описані в даному розділі.

Евристичні знання

Одним з основних методів подання знань в CLIPS є *правила*. Правила використовуються для подання евристик або емпіричних правил, що визначають дії, які необхідно виконати у випадку виникнення деякої ситуації. розробник експертної системи створює набір правил, які, працюючи разом, вирішують поставлену задачу. Правила складаються з *передумов* і *наслідку*. Передумови називаються також *Якщо-частиною* правила або *LHS* правила (left-hand side). Наслідок називається *Т-частиною* правила або *RHS* правила (right-hand side).

Передумови правила являють собою набір *умов* (або *умовних елементів*), які повинні задовольнитися, для того щоб правило виконалося. Передумови правил задовольняються залежно від наявності або відсутності деяких заданих фактів у списку фактів або деяких створених об'єктів, що є екземплярами класів, певних користувачем. Один з найпоширеніших типів умовних виражень в CLIPS - *зразки* (patterns). Зразки складаються з набору *обмежень*, які використовуються для визначення того, чи задовольняє деякий факт або об'єкт умовному елементу. Інакше кажучи, зразок задає деяку маску для фактів або об'єктів. Процес зіставлення зразків фактам або об'єктам називається *зіставленням зразків* (pattern- matching). CLIPS надає механізм, називаний *механізмом логічного висновку* (inference engine), що автоматично зіставляє зразки з поточним списком фактів і певних об'єктів і шукає правила, які застосовні в даний момент.

Наслідок правила представляється набором деяких дій, які потрібно виконати, у випадку якщо правило застосовне до поточної ситуації. Таким чином, дії, задані в наслідку правила, виконуються по команді механізму

логічного висновку, якщо всі передумови правила задоволені. У випадку, якщо в цей момент застосовно більше одного правила, механізм логічного висновку використає поточну *стратегію дозволу конфліктів* (conflict resolution strategy), що визначає, яке саме правило буде виконано. Після цього CLIPS виконує дії, описані внаслідок обраного правила (які можуть вплинути на список застосовних правил), і приступає до вибору наступного правила. Цей процес триває доти, поки список застосовних правил неспорожніє.

У більшості випадків правила CLIPS можна представити у вигляді операторів IF-THEN, використовуваних у процедурних мовах програмування, наприклад, таких як Ada або С. Однак умовні вираження IF-THEN у процедурних мовах перевіряються тільки тоді, коли потік керування програми безпосередньо попадає на дане вираження шляхом послідовного перебору виражень й операторів, що становлять програму. В CLIPS, на відміну від цього, механізм логічного висновку створює й постійно модифікує список правил, умови яких у цей момент задоволені. Ці правила запускаються на виконання механізмом логічного висновку. Із цієї сторони правила схожі на оброблювачі повідомлень, що є присутнім у таких мовах, як, наприклад, Ada або Smalltalk.

Процедурнізнання

Крім евристичної, CLIPS підтримує й *процедурну парадигму подання знань*, використовувану в більшості мов програмування. Конструктори deffunction й defgeneric дозволяють користувачеві визначати нові виконувані конструкції безпосередньо в середовищі CLIPS, що повертають деякі значення або виконують якісь корисні дії. Виклик цих нових функцій нічим не відрізняється від виклику убудованих функцій CLIPS. Оброблювачі повідомлень дозволяють користувачеві визначати поведінку об'єктів, за допомогою завдання тієї або іншої реакції на повідомлення. Функції, родові функції й оброблювачі повідомлень являють собою шматки коду, заданого користувачем і виконуваного, якщо буде потреба, інтерпретатором CLIPS. Крім того, механізм модулів (конструктор defmodule) дозволяє розбивати базу знань CLIPS на окремі змістовні частини.

Функції

Конструктор deffunction дозволяє створювати нові *функції* безпосередньо в CLIPS. Більше ранні версії CLIPS дозволяли використати тільки зовнішні користувацькі функції, написані на якій-небудь мові програмування (найчастіше Сі) і приєднані до середовища CLIPS.

Тіло функції, певної за допомогою конструктора deffunction, являє собою послідовність дій, подібну використовуваній в правій частині правил. Задані користувачем дії виконуються при виклику відповідної функції. Значення, що повертає функцією, є результатом обчислення останньої дії.

Родовіфункції

Родові функції, так само як і звичайні функції, можуть бути створені

безпосередньо в CLIPS.

Спосіб виклику таких функцій також нічим не відрізняється від способу виклику звичайних функцій. Однак родові функції набагато могутніше звичайних, тому що вони здатні перевантажуватися. Завдяки механізму перевантаження родова функція може виконувати різні дії залежно від типу й числа аргументів.

Звичайно родова функція складається з декількох компонентів, названих *методами*. Кожен метод містить різний набір аргументів родової функції.

Наприклад, можна перевантажити системну функцію + (арифметичне додавання) для виконання операції конкатенації двох рядків. Однак після цього функція + усе ще зможе виконувати арифметичне додавання. У даному прикладі в родової функції + існує два методи: перший метод явно визначений користувачем для конкатенації двох рядків, другий являє собою неявний виклик стандартної функції, що виконує арифметичне додавання. Значення, повернуте родовою функцією, є значенням, отриманим у результаті обчислення останньої дії в застосовуваному методі.

[Продовжити перегляд](#)

Контрольні питання:

1. Основні типи елементів, призначених для написання програмв синтаксисі мови CLIPS.
2. Що являє собою значення типу string в CLIPS.
3. Що таке факт в CLIPS.
4. Що таке функції в CLIPS.
5. Що таке вираз в CLIPS.
6. Що таке об'єкт в CLIPS.
7. Що таке модуль в CLIPS.
8. Які основні можливості ОПП ви знаєте?

Укладач: _____ Старух А.І., доцент, к.е.н.
(підпис) (ПІБ, посада, науковий ступінь, вчене звання)