

Лисп-системы и Лисп-машины.

Лекция 15.

Специальности : 230105, 010501

Становление Лиспа и исследования по искусственному интеллекту.

- Обработка списков и искусственный интеллект.**
- Возникновение Лиспа и его ранние реализации.**
- Лисп-системы и формирование стандарта Коммон Лиспа.**
- Visual Lisp и AutoCAD.**
- Лисп-машины.**
- Перспективы функционального программирования.**

Обработка списков и искусственный интеллект.

1956 г. – А. Ньюэлл, Дж. Шоу и Г. Саймон разрабатывают язык IPL. IPL представлял собой язык обработки списков для вычислительной машины Johnniac фирмы Rand Corporation и использовался программой вывода теорем логики Логик-теоретик. Благодаря IPL родилась идея хранения данных в памяти в виде списочных структур вне зависимости от их физического расположения.

1958 г. – первая реализация интерпретатора Лиспа Джоном Маккарти на ЭВМ IBM 704 в Массачусетском технологическом институте (MIT). Для символьных цепочек IPL были предложены : скобочная форма представления, иерархическая списочная запись, а также точечная нотация. С использованием понятия пустого списка (NIL) удалось определить основные алгебраические действия по обработке списков, а также примитивные функции и предикаты анализа, конструирования и сравнения списков : CAR, CDR, CONS, ATOM и EQ. Из лямбда-исчисления Черча для определения функций было заимствовано лямбда-выражение.

Ранние реализации Лиспа.

1960 г. – Lisp 1 как первая реализация среды программирования. Реализован на PDP-1 (Programmed Data Processor) – первой ЭВМ, ориентированной на работу в интерактивном режиме. Lisp 1 был первой интегрированной средой с возможностью написания, тестирования и отладки программ без выхода из интерпретатора.

1960 г. – статья Маккарти в журнале “Communications of ACM”, в которой синтаксис и семантика Лиспа определялись в виде алгебраического формализма при помощи самого Лиспа.

1962 г. – Lisp 1.5 – первая полностью описанная в литературе (издательство MIT Press, США) Лисп-система. Способствовала широкому распространению Лиспа за пределами США и дальнейшему развитию Лисп-культуры.

Развитие Лисп-систем.

1962-64 гг. – MIT, проект MAC (Machine Aided Cognition) по разработке операционных систем и языков программирования для использования ЭВМ в диалоговом режиме. Как результат в 1964 году на ЭВМ PDP-6 на базе Lisp 1.5 был реализован Маклисп. Достоинства : эффективный списочный код программ, транслятор с оптимизацией кода, расширенный набор функций и типов данных. Недостатки : узкая исследовательская специализация, отсутствие должной документации.

1966 г. – BBN-Лисп на ЭВМ PDP-10 как подсистема операционной системы TENEX, фирма “Bolt, Beranek and Newman Inc.” (BBN). Специально для поддержки больших лисповских программ в PDP-10 были введены механизмы для работы с разделением времени : страничная организация памяти, команды для работы со стеком и расширен набор регистров для переключения от одной задачи к другой.

Формирование стандарта Коммон Лиспа.

1972 г. – начало развития Interlisp'a (BBN и Xerox). В состав интегрированной среды у Interlisp-D впервые был введен графический пользовательский интерфейс, включающий многооконное взаимодействие, средства выбора из меню и “мышь”, а также ориентированный на использование экрана инспектор структур данных. Недостатки системы : закрытость, отсутствие замыканий, иерархических типов данных и поддержки объектов.

1985 г. – Коммон Лисп (университет Карнеги-Меллона) как развитие Interlisp'a. Коммон Лисп содержит в себе важнейшие черты современных Лисп-систем : возможности определения типов, императивные управляющие структуры, макросы, функционалы, замыкания, пространства имен, последовательности, средства управления потоками, синтаксический интерпретатор и транслятор. В Коммон Лиспе определены базовые механизмы для реализации ООП. Для обеспечения создания различных сред для различных целей среда, инструментальные средства и пользовательский интерфейс не затронуты стандартизацией.

muLISP.

В настоящее время muLISP является наиболее популярным и наиболее удачным диалектом языка LISP. Создан фирмой Soft Warehouse Inc (США) [1]. Существуют несколько реализаций : muLISP81, muLISP83, muLISP85, muLISP87, а также muLISP90.

muLISP87 - система, имеющая порядка 400 встроенных функций. Предусмотрен встроенный текстовый редактор, интерфейс с Ассемблером, имеются исходные тексты системы объектно-ориентированного программирования Flavors. По синтаксису пакет muLISP не совместим со стандартом Common LISP даже при включении в него поставляемой фирмой библиотеки функций Common LISP. muLISP позволяет эффективно решать задачи искусственного интеллекта, а также нетрадиционные для Лиспа задачи : разработка оконного интерфейса, обмен программами на языке C++.

muLISP90.

Система программирования muLISP-90 является "маленьким LISPом", который работает на IBM PC (или на HP 95LX palmtop), используя операционную систему MS-DOS версии 2.1 или более поздние.

muLISP90 включает пакет для совместимости с Common Lisp, содержащий более 450 специальных форм, макросов, функций и управляющих переменных Common Lisp.

muLISP90 включает экранный редактор, отладчик (debugger), оконную систему, интерпретатор и компилятор. Среди многочисленных примеров программ имеется DOCTOR ("Eliza-подобная" программа). Система времени выполнения (run-time system) позволяет создавать небольшие EXE- или COM-файлы. Она использует компактное внутреннее представление кода, обеспечивающее минимизацию памяти и увеличение скорости выполнения. Ядро системы занимает около 50К.

Visual LISP и AutoCAD.

Наряду с `tuLISP`ом другим весьма распространенным на сегодняшний день диалектом Лиспа является `AutoLISP`, на котором написано математическое обеспечение САПР `AutoCAD`. Символьная обработка и методы ООП наилучшим образом подходят для обработки рисунков и чертежей. По синтаксису и соглашениям `AutoLISP` в наибольшей степени близок Коммон Лиспу. `AutoLISP` относится к интерпретируемым языкам.

Для работы с `AutoLISP` в `AutoCAD 2000` появился новый интерфейс, называемый `Visual LISP`. `Visual LISP` – мощная интегрированная среда разработки программ на языке `AutoLISP`. Наряду с лисповским “ядром” в ее состав входят ряд дополнительных функций, отражающих специфику `AutoCAD`. Сюда относится организация ввода данных различного типа (`GETANGLE` – ввод угла, `GETPOINT` – ввод точки), геометрические вычисления (расстояние между точками и т.п.), преобразования систем координат, доступ к примитивам `AutoCAD`.

Трассировка в Visual LISP.

В целях облегчения отладки программ в Visual LISP включены функции трассировки TRACE и UNTRACE.

TRACE устанавливает признак трассировки для заданных функций.

Синтаксис : (trace <функция 1> <функция 2> ...).

TRACE не вычисляет свой аргумент. После ввода директивы TRACE интерпретатор будет распечатывать значения аргументов каждого вызова <функции i> перед ее вычислением и результат, полученный после окончания вычисления каждого ее вызова.

(UNTRACE <функция 1> <функция 2> ...) убирает признак трассировки для заданных функций.

Создание Лисп-машин.

Целью проектирования Лисп-машин была разработка их в виде ПЭВМ, которые можно было бы использовать как для исследований по искусственному интеллекту, так и для различных промышленных и коммерческих приложений. Разработке и распространению Лисп-машин в 1980-е годы помешала необходимость переноса программного обеспечения большого объема из дорогой среды больших ЭВМ.

Преимущества Лисп-машин :

- Большой объем основной памяти;
- Ориентация аппаратных средств для вычислений на Лиспе;
- Проверка типов на уровне аппаратуры (tagged architecture);
- Возможность использования на Лисп-машине интегрированной программной среды.

Производители Лисп-машин.

Развитие микроэлектроники в 1970-е гг. сделало возможным создание ориентированных на язык процессоров и ПЭВМ.

1978 г. – изготовление интегральной схемы LSI “Lisp on a chip”.

1979 г. – создание первой Лисп-машины в Японии (Kurokawa).

1984 г. – первый коммерческий прототип Лисп-машины Alpha фирмы Fujitsu.

Середина 1980-х – начало производства Лисп-машин в США.
Производители :

- Lisp Machine Inc. (LMI) Symbolics Inc – их машины использовали близкие к современным диалекты Зеталиспа.

- Xerox – представляет направление Интерлиспа.

Кроме того, Лисп-машины выпускались фирмой Texas Instruments (Explorer).

Miranda.

В 1985 г. Дэвид Тёрнер объединил в одном языке Miranda многие важные особенности функциональных языков (функции высшего порядка, ленивые вычисления, алгебраические типы данных, параметрический полиморфизм, сопоставление с образцом). Введены удобные конструкторы списков (ZF-выражения), позаимствованные из другого языка Тёрнера KRC. Факториал на Миранде можно выразить следующим образом :

fac n	или, используя сопоставления :
= 1, if n = 0	fac 0 = 1
= n * fac (n - 1), otherwise	fac n = n * fac (n-1)

И с использованием ZF-выражений :

fac n = product [1..n]

Тёрнер запатентовал свой язык (Miranda™ - торговая марка Research Software, Ltd), что замедлило его развитие и распространение. На основе Миранды было создано много языков, в том числе и популярные ныне Haskell и Clean.

Проект ЭВМ пятого поколения.

В 1981 г. в Японии начинаются работы по проекту "вычислительных систем следующего поколения", в основе которых лежат декларативные языки, в первую очередь Пролог. Появляются реализации Пролога для всех распространенных компьютеров. Начинаются исследования по параллельным логическим языкам, таким как Parlog, Concurrent Prolog, GHC.

К концу 1980-х интерес к декларативным языкам снижается в связи с широким распространением ПК. Недостаточные вычислительные ресурсы ПК в совокупности с требованием дружелюбности интерфейса приводят к широкому распространению объектно-ориентированного программирования. Успело сложиться мнение, что декларативные языки "принципиально неэффективны".

Тем не менее, технические задачи были решены – созданы мощные параллельные компьютеры и соответствующие им языки программирования.

Современный этап развития декларативных языков.

Рост производительности ПК при снижении их стоимости во второй половине 1990-х гг. с одновременным ростом сложности решаемых задач и совершенствованием техники реализации компиляторов и интерпретаторов привели к значительному повышению эффективности решения задач на ПЭВМ именно с помощью декларативных языков.

Так, применяя методы глобального анализа программ, создатели систем Aquarius Prolog [3] и Parma [4] смогли вплотную приблизиться к лучшим компиляторам императивных языков. Маленькой сенсацией стал компилятор для функционального языка Sisal [5, 6], ориентированного на численные расчёты, созданный в американском ядерном исследовательском центре (Lawrence Livermore National Laboratory). Он превзошёл не только Си, но и Фортран, бывший в этой области вне конкуренции.

Другие направления исследований в области декларативных языков в последнее десятилетие.

- Повышение "чистоты" языков, то есть устранение из них недекларативных средств.
- Создание специализированных средств для эффективного решения определённых классов задач. Это, прежде всего, языки "программирования в ограничениях". Эти языки содержат встроенные "решатели уравнений" специфических видов, например, линейные уравнения/неравенства и уравнения принадлежности в конечных областях. Такие средства позволяют практически решать многие задачи, которые нереально решить общими методами.
- Синтез различных стилей программирования. В частности объединение функционального и логического программирования и даже объединение их с объектно-ориентированным программированием. Важный принцип этих исследований – найти простую теоретическую основу для такого синтеза.

Литература.

1. **muLisp-85. - Reference Manual. - Software house, Inc., 1985. - 137 с.**
2. **Дехтяренко И.А. Декларативное программирование // <http://www.softcraft.ru/paradigm/dp/index.shtml>**
3. **Van Roy P. Can Logic Programming Execute as Fast as Imperative Programming ? // <http://www.info.ucl.ac.be/people/PVR/Peter.thesis/Peter.thesis.html>**
4. **Taylor A. LIPS on a MIPS: Results from a Prolog Compiler for a RISC. In ICLP, 1990.**
5. **Feo J. T et al. A Report on the SISAL Language Project. 1990.**
6. **Cann D. Retire FORTRAN? A debate rekindled. CACM 35(8), 1992.**
7. **Хювенен Э., Сеппянен Й. Мир Лиспа. Т.2. – М.:Мир, 1990. С. 242-249, 272-296.**