
Noções de LISP

(LISP Processing)

Luciano Soler

lsoler@inf.ufrgs.br

Por quê LISP?

- Poucas construções permitem fazer muito
 - Permite ilustrar bem os princípios da recursão
 - Estruturas de dados complexas podem ser expressas de forma simples
 - Não dá margem a “encher de lingüiça”
 - Uma função freqüentemente pode ser verificada visualmente
-

Lisp: Nota histórica

- John McCarthy vinha trabalhando há anos em uma linguagem que fosse, como provado por Turing, equivalente à sua máquina. Em 1960, dando aula no MIT demonstrou que a função “eval” era capaz de simular a máquina de Turing, resultado teórico de grande valor.
-

Lisp: Nota histórica

- Um dos alunos de John McCarthy, Steve Russel comentou: “sendo verdade este teorema, basta implementar “eval” e teremos a Máquina de Turing”, ao que McCarthy contestou “Não confunda teoria com prática, este é um resultado teórico, para ter valor prático tem de percorrer um longo caminho”.
-

Lisp: Nota histórica

- Russel não se satisfez. Implementou “eval” e algumas outras funções em Máquina IBM704, apresentou seu trabalho e assim nasceu Lisp, linguagem fruto do espírito prático de aluno com grande conhecimento teórico. Guarda ainda hoje lembranças do passado:
 - car: “contents of address register;
 - cdr: “contents of decrement register;
 - Símbolos do assembler do IBM704.
-

Características do LISP

- Manipulação de informação simbólica
 - Versão inicial do Lisp era pouco prática (sem iteração)
 - Muitas versões e dialetos: Franz, Mac, Inter, Common (praticamente o padrão) e Scheme (variante “enxuta”)
-

Diferenças entre LISP e outras linguagens

- Programas e Dados têm a mesma forma
 - É trivial escrever um programa que escreve um programa
 - Não se trabalha com variáveis armazenadas (embora LISP suporte variáveis)
 - Em “C++” podemos implementar uma função para inserir um elemento X numa variável L do “tipo” lista
 - Em LISP, escreve-se uma função que, dados um elemento X e uma lista L, retorna uma lista igual a L com X inserido
-

Razões para usar LISP

- Estruturas de dados são facilmente manipuladas em LISP.
 - Por exemplo:
 - A pilha é a própria lista;
 - Existem primitivas para ler, juntar novo elemento na pilha, etc.
 - Árvores são implementadas como listas de listas, sendo fácil percorrê-las e modificá-las.
-

Sintaxe de Lisp

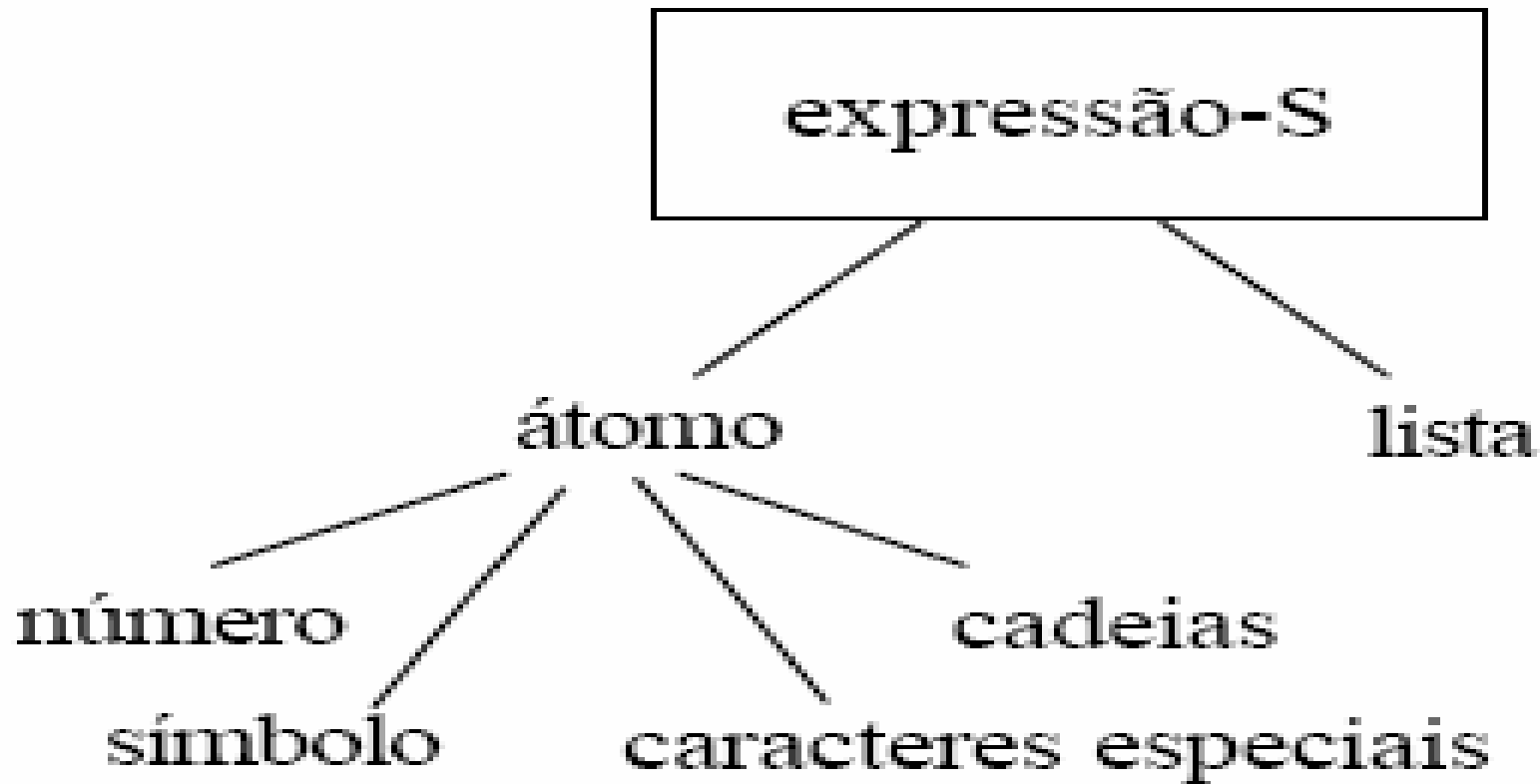
- Vocabulário de Lisp:
 - Átomos: elementos indivisíveis, podendo ser:
 - Números; ex: 1, 13, 15, -.35, etc.
 - Identificadores: seqüências de letras e números; ex: Lisa, Jane1, fibo, etc.
 - Identificadores reservados: +, -, /, *, car, cdr, etc.
 - Delimitadores: (,)
-

Usando LISP

- LISP é freqüentemente implementada por um interpretador
 - Usuário entra com uma expressão
 - Interpretador avalia expressão e imprime o resultado
 - Exemplo:
 - > (+ 3 4 5 6)
18
 - > (+ (+ 3 4) (+ (+ 4 5) 6))
22
-

Sintaxe de Lisp

- Linguagem Lisp:



Lisp Puro

- Lisp Puro contem as seguintes funções primitivas:
 - ❑ Car primeiro elemento de uma lista;
 - ❑ Cdr: o que sobra de uma lista tirando o 1º elemento
 - ❑ Cons: constroi lista dado um elemento e uma lista;
 - ❑ Eql: retorna T se os dois elementos que se seguem são iguais, NIL no caso contrario;
 - ❑ Atom: retorna T se elemento que o segue é atômico, NIL em caso contrário.
-

Semantica operacional de Lisp Puro

Seletores: car, cdr

- ❑ > (car '(a d f))
- ❑ > A
- ❑ > (cdr '(a s d f g))
- ❑ > (s d f g)

Construtor:

- ❑ > (cons 'a '(s d f g))
- ❑ > (a s d f g)

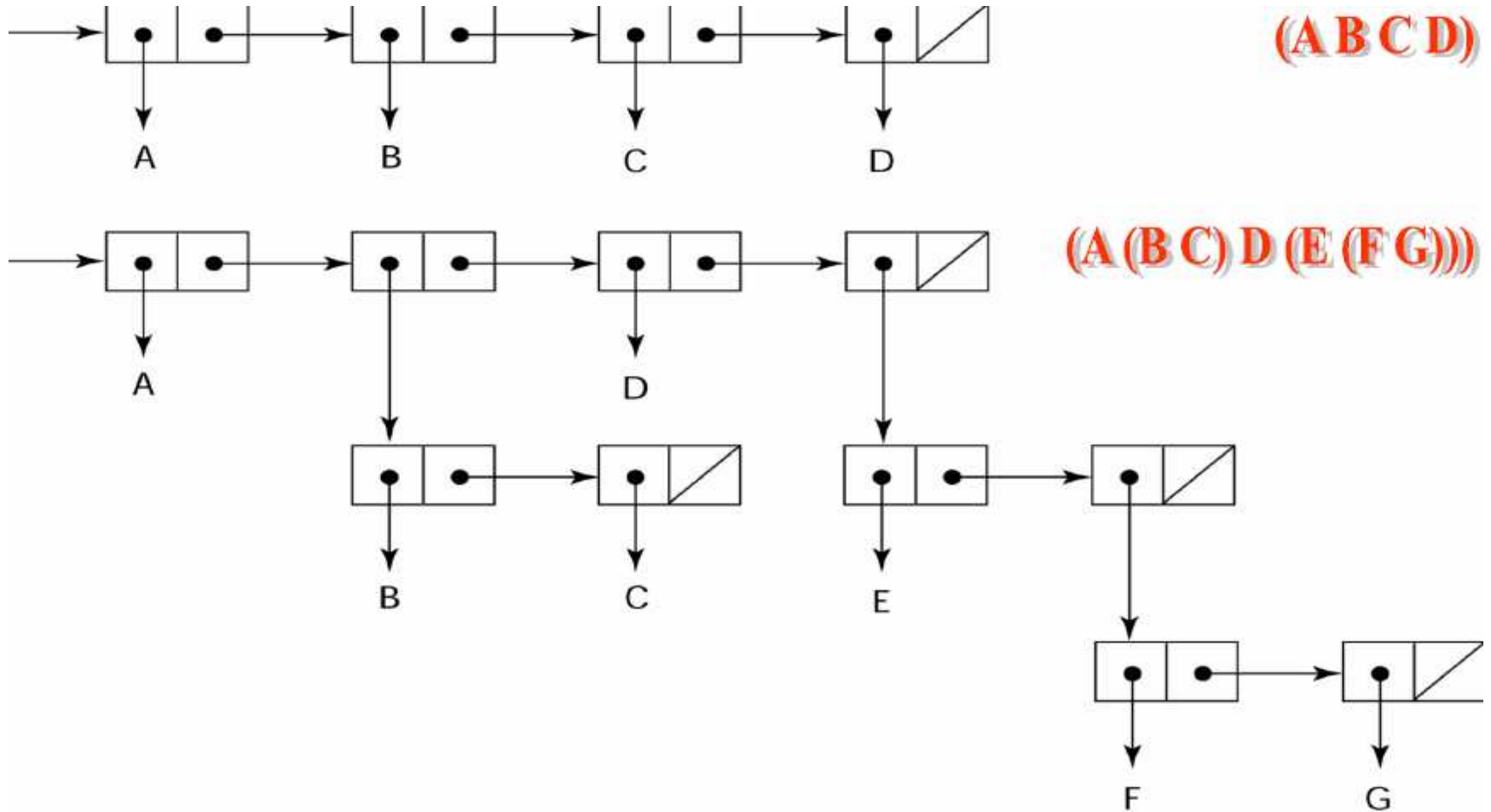
Predicado atômico:

- ❑ > (atom 'jane)
- ❑ > T
- ❑ > (atom '(a s d f g))
- ❑ > NIL

Predicado igualitário

- ❑ > (eql 'casa 'casa)
 - ❑ > T
 - ❑ > (eql 10 20)
 - ❑ > NIL
-

Representação interna de listas



Exemplos de novas funções

- Encontra o segundo elemento de uma lista:

```
(defun segundo (lista)
  (cadr lista))
```

- Calcula o fatorial de um número:

```
(defun fat (n)
  (cond (( > n 0) ('Numero negativo não tem fatorial'))
        (( = n 0) 1)
        (( = n 1) 1)
        (T (fat (- n 1)))))
```

Maquinas Lisp



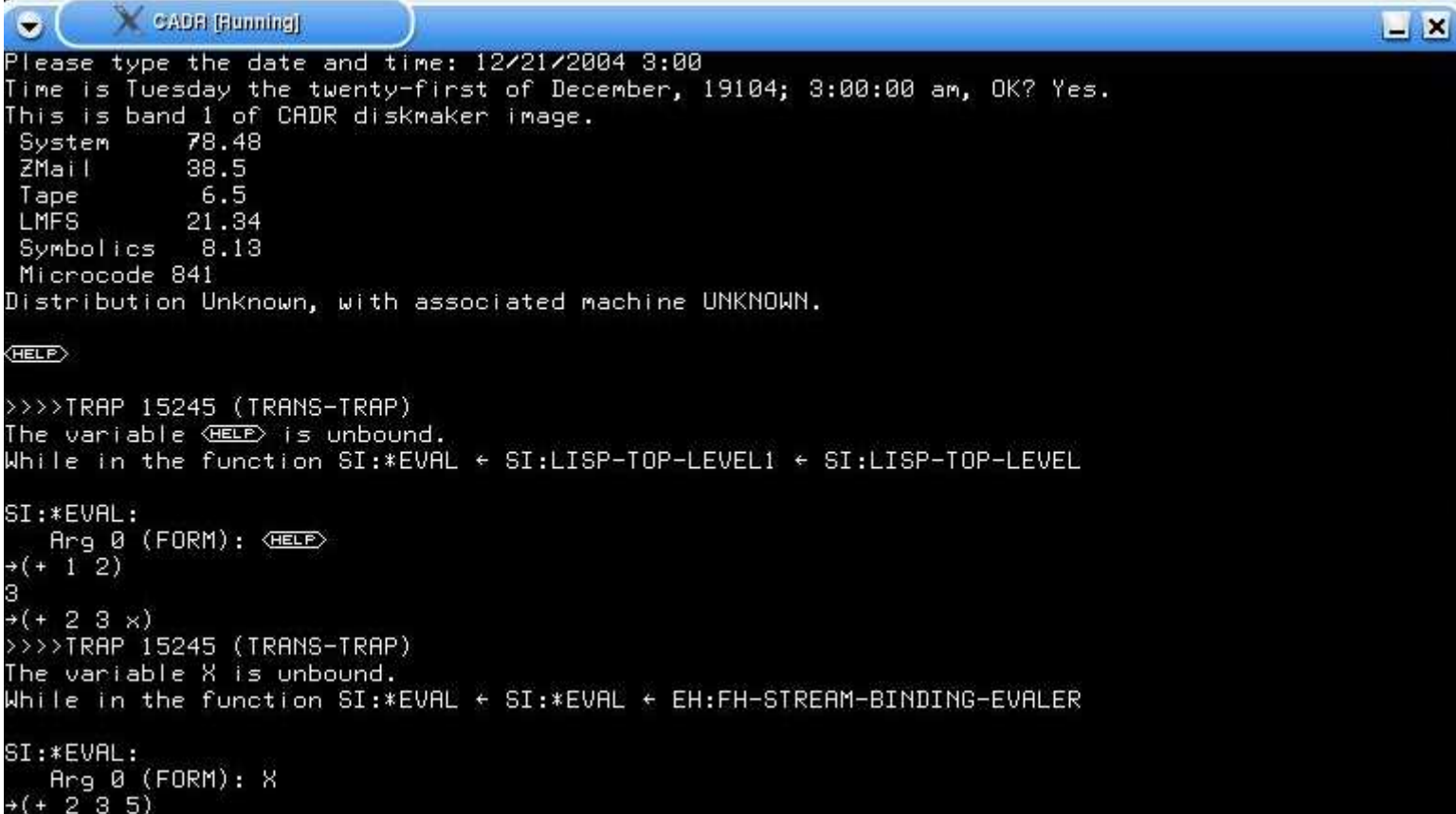
Symbolics LGP



Emulador CADR

- Segunda geração de máquinas Lisp
 - A.I. Memo 528
 - Microprocessador de 32 bits
 - Software ainda inacabado
 - Fornece mais recursos que o CLISP
-

Exemplo de sessão com o CADR



```
CADR [Running]
Please type the date and time: 12/21/2004 3:00
Time is Tuesday the twenty-first of December, 19104; 3:00:00 am, OK? Yes.
This is band 1 of CADR diskmaker image.
System      78.48
ZMail       38.5
Tape        6.5
LMFS        21.34
Symbolics   8.13
Microcode 841
Distribution Unknown, with associated machine UNKNOWN.

<HELP>

>>>>TRAP 15245 (TRANS-TRAP)
The variable <HELP> is unbound.
While in the function SI:*EVAL + SI:LISP-TOP-LEVEL1 + SI:LISP-TOP-LEVEL

SI:*EVAL:
  Arg 0 (FORM): <HELP>
→(+ 1 2)
3
→(+ 2 3 x)
>>>>TRAP 15245 (TRANS-TRAP)
The variable X is unbound.
While in the function SI:*EVAL + SI:*EVAL + EH:FH-STREAM-BINDING-EVALER

SI:*EVAL:
  Arg 0 (FORM): X
→(+ 2 3 5)
```

Conclusão

- Deficiente quando se trata de problemas com muitas variáveis (Ex: Contas de Banco).
 - Vantajosa quando se precisa de alto nível de abstração.
 - Não dependência das operações de atribuição.
 - A avaliação independente da ordem torna as linguagens funcionais as mais indicadas para a programação de computadores maciçamente paralelos.
-

Bibliografia

- HWANG, Kai; DEGROOT, Doug; Parallel Processing for Supercomputers and Artificial Intelligence. McGraw-Hill Publishing, 1989.
 - PAINE, Peter; Lisp Machine supplies and information, <http://www.abstractscience.freeseve.co.uk/symbolics/index.html> . Ultimo acesso Dez/2004.
 - RINO, Lucia; Notas de aula – Lisp; <http://www.dc.ufscar.br/~lucia/> . Ultimo acesso Dez/2004.
 - SUSSMAN, Gerald J.; STEELE, Guy L.; Design of Lisp Based Processors, AI Memo No. 514, MIT. March 1979.
 - Lisp, <http://www.din.uem.br/~ia/ferramen/lisp/> . Ultimo acesso Dez/2004.
 - PARKER, Brad; CADR simulator; <http://www.heeltoe.com/retro/cadr/> Ultimo acesso Dez/2004.
-