Parsing em Haskell

Francisco Soares
xfrancisco.soares@gmail.com

Compilação típica

- Análise léxica
- Análise sintática
- Análise semântica
- Geração de código

Análise léxica

- Divisão da entrada em *tokens* (pedaços) interpretáveis no contexto da aplicação.

```
a = 1
```

- Temos um identificador – `a` –, uma operação de atribuição, e um valor numérico.

Caracteres agrupados em *tokens*

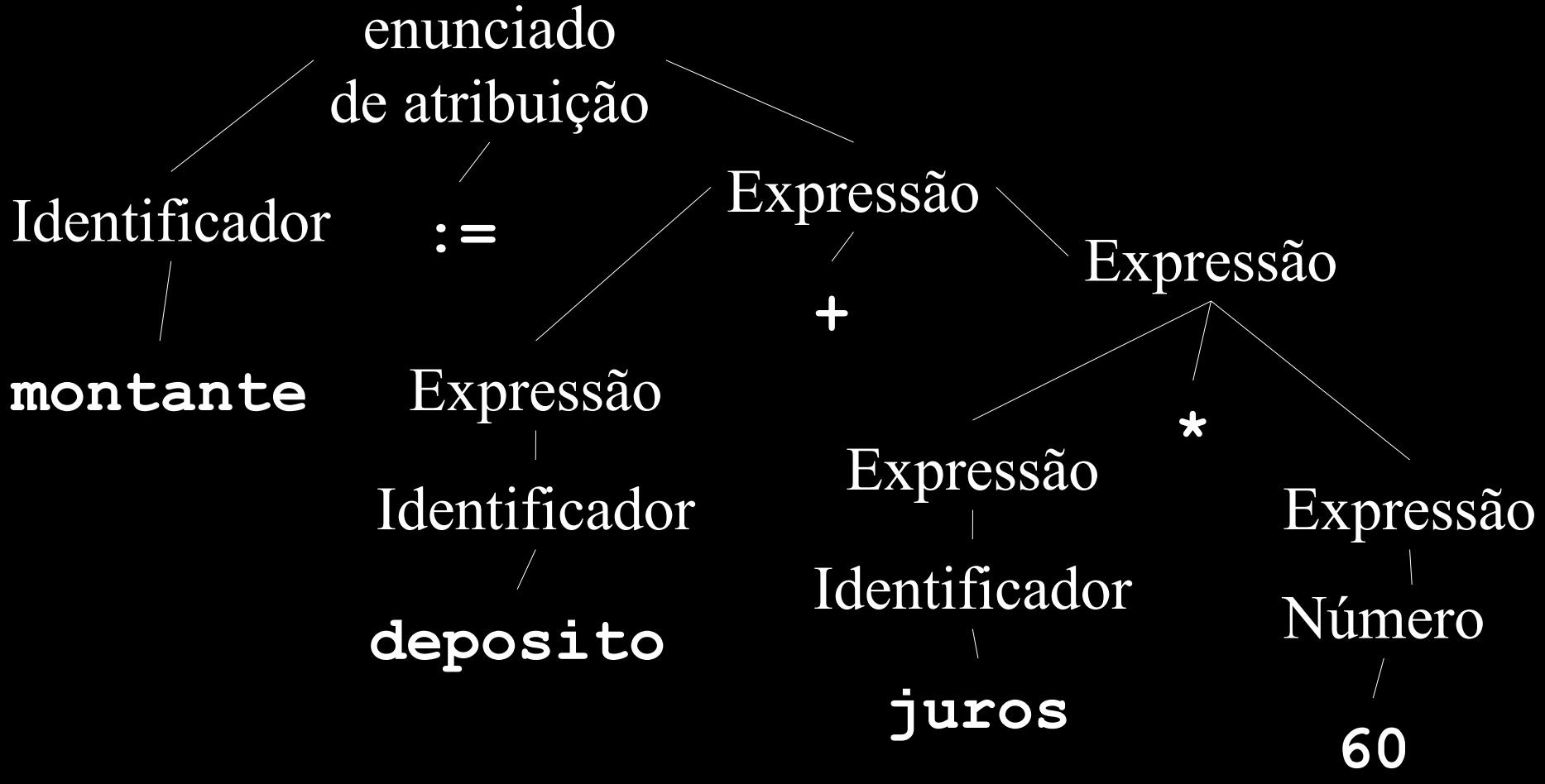
`montante := deposito+juros*60`

1) O identificador **`montante`**
2) O símbolo de atribuição **`:=`**
3) O identificador **`deposito`**
4) O símbolo de adição **`+`**
5) O identificador **`juros`**
6) O símbolo de **`multiplicação`**
7) O número **`60`**

Análise sintática

- Interpretação (*parsing*) dos *tokens* resultado da análise léxica.
- Agrupamento dos *tokens* em frases gramaticais.
- Geralmente, se constrói uma *árvore sintática*.
  - A partir dela, é possível avaliar se a entrada faz sentido.

`montante := deposito+juros*60`



Gramáticas (BNF)

- Geralmente expressas em relação a:
  - Elementos terminais (variáveis, dígitos);
  - Elementos não-terminais (funções, controle);
  - Regras de Produção: define a sintaxe de elementos não-terminais.

Exemplos de ferramentas

- Em Haskell
  - Parsec: léxica e sintática;
  - Alex: léxica;
  - Happy: sintática.
- Em C/C++
  - lex ou flex: léxica;
  - yacc ou bison: sintática.

Obtenção das ferramentas

- Todas as ferramentas mencionadas para Haskell estão presentes na Plataforma Haskell.
  - Usadas com o *Glasgow Haskell Compiler*
- *http://hackage.haskell.org/platform*

Haskell

Download Haskell
Find A Library

About

Why use Haskell?
Language definition
History of Haskell
Future of Haskell
Implementations
The Haskell Platform

Learning

Haskell in 5 steps
Learning Haskell
Haskell Cheatsheet

Haskell is an advanced purely functional programming language. An open source product of more than twenty years of cutting edge research, it allows rapid development of robust, concise, correct software. With strong support for integration with other languages, built-in concurrency and parallelism, debuggers, profilers, rich libraries and an active community, Haskell makes it easier to produce flexible, maintainable high-quality software.

1 Headlines

- *2010:*
  - **Download the Haskell Platform** - the standard development environment for Haskell
  - Try Haskell! -- an interactive, online Haskell interpreter
  - A new **code generator for GHC based on LLVM** has been developed
- *2009:*
  - **Haskell in 2009: Year in Review**
  - Over **1800 Haskell packages** have now been released on Hackage.
  - **GHC 6.12.1**, the flagship Haskell compiler, has been released. Get it now
  - **Haskell 2010**: The Haskell-prime committee has announced a new revision of the Haskell language standard
  - The **Haskell Platform 2009.2.0.2** has been released
  - The **Industrial Haskell Group** has been launched, for commercial users of Haskell.

Get the Haskell Platform

The standard Haskell development environment


Mac


Windows


Linux

Comprehensive
A rich development environment for Haskell programming.

Robust
Stable and widely-used tools and libraries.

Cutting Edge
Advanced features such as easy multicore parallelism and transactional memory.

Learn more...

Problems?

Parsec

- Parte do módulo **`Text.Parsec`** ou **`Text.ParserCombinators.Parsec`**
- *Parsers* produzidos a partir da combinação de outros *parsers*.
- É um tipo de *Monad*.

Exemplo

- Parsing de um arquivo CSV:

Funções *parsec*

- sepBy
- endBy
- fail
- <|>
- <?>
- string
- letter

Happy

- Gerador de parser a partir de gramáticas livres de contexto (*Backus-Naus Form*)
- Geralmente produz *parsers* com melhor desempenho que combinadores de *parsers*.
- Análogo ao *yacc* ou *bison* em C.

Funcionamento do Happy

- Ver código-fonte!

Alex

- Analisador léxico para Haskell
- Pode trabalhar em conjunto com analisador sintático (Happy)
- Análogo ao *lex* ou *flex* em C.

Funcionamento do Alex

- Ver código-fonte!

Avaliando as expressões

- A partir da árvore de expressões produzida, podemos partir para produzir código fonte.
  - Análise semântica;
  - Geração de código.

Partindo para o programa

- Ver código-fonte!

Referências

- Alex Website: http://www.haskell.org/alex/
- Happy Website: http://haskell.cs.yale.edu/happy/
- “Compiladores: Princípios, técnicas e ferramentas” – Aho