Tópicos em LP

Fabio Mascarenhas – 2018.1

http://www.dcc.ufrj.br/~fabiom/comp2

SmallLua

```
bloco <- stat* ret? -> list
   stat  <- # "while" exp "do" bloco "end" -> nwhile
       / # "local" id '=' exp -> nlocal
       / # "if" exp "then" bloco ("else" bloco)? "end" -> nif
       / # id '=' exp -> nassign
       / # pexp -> nstatexp
ret   <- # "return" exp -> nret
ids   <- id (',' id)* -> list
exps  <- exp (',' exp)* -> list
exp   <- lexp (# {"or"} -> op lexp)* -> chainl
lexp  <- rexp (# {"and"} -> op rexp)* -> chainl
rexp  <- cexp (# rop -> op cexp)* -> chainl
cexp  <- aexp # '..' cexp -> nconcat  / aexp
aexp  <- mexp (# aop -> op mexp)* -> chainl
mexp  <- sexp (# mop -> op sexp)* -> chainl
sexp  <- '-' sexp -> nunm / "not" sexp -> nnot / "false" -> nfalse / "true" -> ntrue
    / number -> nnumber / lmb / "nil" -> nnil / string -> nstring / pexp
lmb   <- # "function" '(' (ids? -> opt) ')' bloco "end" -> nlmb
pexp  <- ('(' exp ')' / # id -> nvar) (# '(' -> ncall (exps? -> opt) ')')* -> chainl
rop   <- {'<'} / {'=='} / {'~='}
aop   <- {'+'} / {'-'}
mop   <- {'*'} / {'/'}

string <- sp { string = [squote] (![squote] .)* [squote] / [dquote] (![dquote] .)* [dquote] }
number <- sp { number = [isdigit]+ (['.'] [isdigit]+)? }
id     <- !kws sp { id = [isidstart] [isidrest]* }
sp     <- [isblank]*

kws    <- "while" / "end" / "if" / "then" / "else" / "local" / "true" / "false" / "nil"
   / "function" / "not" / "and" / "or" / "do" / "return"
```

Tipagem estática

- Lua é uma linguagem *dinamicamente tipada*: o interpretador Lua sabe o tipo de cada valor, e verifica cada operação antes de ser executada, para checar se os tipos dos operandos estão corretos

- Em linguagens estaticamente tipadas, o *compilador* sabe o tipo de cada *termo* (expressão e variável) do programa, e pode verificar se os tipos dos operandos de cada operação estão corretos sem precisar executar o programa

- Um *sistema de tipos* é um sistema lógico que dá suporte à verificação de tipos em uma linguagem estaticamente tipada

Um sistema de tipos simples para SmallLua

- Nossa linguagem SmallLua tem quatro tipos *atômicos*: **number**, **boolean, string** e **nil**

- Funções são valores de primeira classe em SmallLua, então também temos *tipos compostos*: tipos da forma **(t1, ..., tn) -> tr** representam funções de *n* parâmetros de tipos **t1** a **tn**, e tvalor de retorno de tipo **tr**.

- Funções que não retornam nada têm tipo de retorno **nil**

- A PEG abaixo dá a sintaxe dos tipos de SmallLua:

```
type  <- # ({"number"} / {"string"} / {"boolean"} / {"nil"}) -> ntype / tfunc
tfunc <- # '(' (types? -> opt) ')' '->' type -> ntfunc
types <- type (',' type)* -> list
```

Anotações de tipos

- A forma mais simples de implementar tipagem estática é exigindo *anotações de tipos* explícitas

- Em SmallLua, podemos anotar apenas parâmetros e tipos de retorno de funções, já que as variáveis locais podem pegar seus tipos da sua expressão de inicialização

```
lmb   <- # "function" '(' (prms? -> opt) ')' ':' type bloco "end" -> nlmb
prms  <- param (',' param)* -> list
param <- # id ':' type -> nparam
```