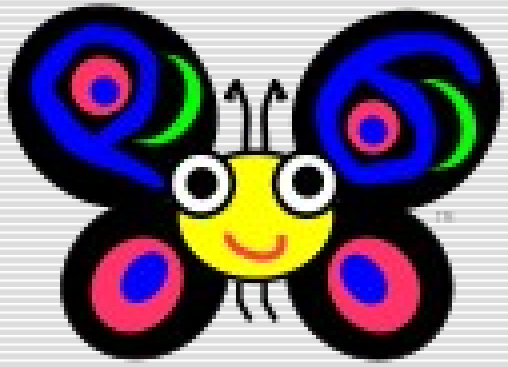
Yet Another Perl6 Interpreter a hack around LuaJIT

(45')



François Perrad

francois.perrad@gadz.org

TvmJIT vs LuaJIT

```
$ luajit
> print(6*7)
42

$ tvmjit
> (!call print (!mul 6 7))
42
```

A hack around LuaJIT

- **T**able **V**irtual **M**achine **J**ust **I**n **T**ime
- TP (**T**able **P**rocessing) language uses the S-expression syntax
- Lua 5.1 C/API, ABI compatible
  - LPeg (Parsing Expression Grammar)
  - lua-linenoise
- MIT License
- Started with LuaJIT 2.0.0, now aligned with LuaJIT 2.0.2
- GitHub hosted (+ Travis CI)
- TAP test suite

TP as Intermediate Language

- Easy to generate
  - Stringification of Op tree
- Allow any char almost everywhere
  - Sigils
  - UTF-8
- Easy to hack
  - **`!line`** annotation
  - **`!assign`** as expression instead of statement

A Lua translator included

- Required by JIT bytecode utilities

```
$ tvmjit -bl 42.tp
-- BYTECODE --
0001    KSHORT   0   6
0002    KSHORT   1   7
0003    GGET     2   0      ; "print"
0004    MULVV    3   0   1
0005    CALL     2   1   2
0006    RET0     0   1
```

- 1.3 KLoC
  - lexer with LPeg
  - recursive-descent parser in plain Lua
- 19 Kb of bytecode (in a C file)

p6jit

- Perl6 backend
- Linked with libtvmjit
- Native libraries (p6str, p6num)
- NQP.lua bytecode included
  - 3 KLoC
- Artistic License v2
- GitHub hosted (+ Travis CI)

Have fun

```
$ ./p6jit
> say(6*7);
42
```

Enjoy AST

```
$ ./p6jit --ast
> say(6*7);
- TVM::AST::Block(blocktype => "immediate")
  - TVM::AST::Stmts()
  - TVM::AST::Stmts()
    - TVM::AST::Op(op => "call", name => "&say")
      - TVM::AST::Op(op => "op", name => "&infix:<*>")
        - TVM::AST::NVal(value => 6)
        - TVM::AST::NVal(value => 7)
```

Enjoy Opcode

```
$ ./p6jit --op
> say(6*7);

(!do
(!line 1)(!call &say ((!mul 6 7))))
```

Have fun

```
$ ./p6jit --tp
> (!call &say ((!mul 6 7)))
42
```

Language gap : calling convention

```
$ ./p6jit --op
> f(:z(2), 3, :y(1), 4)
(!call &f (3 4 "z": 2 "y": 1))
```

- named/positional mixed in a table

Language gap : MOP

```
$ ./p6jit --op
> class Foo is Bar {}
(!let p6class (!index _P6PKG "NQP::Metamodel::ClassHO
(!let _PKG (!callmeth p6class new ("Foo")))
(!assign (!index _P6PKG "Foo") _PKG)
(!call (!index p6class "add_parent")
          (_PKG (!index _P6PKG "Bar")))
(!call (!index p6class "add_parent")
          (_PKG (!index _P6PKG "Any")))
```

- add_parent
- add_attribute
- add_method

Runtime Classes

- Num, num
- Str, str
- Bool, bool
- Nil, nil
- Array
- Hash
- Mu, Any, Cool

Runtime Classes

```
$ ./p6jit
> say((1).WHAT)
num

$ ./p6jit --op
> say((1).WHAT)
(!do
(!line 1)(!call &say (
(!line 1)(!callmeth 1 WHAT ())))))
```

Pending stuff

- Grammar
- Multimethod dispatch
- Lazy evaluation
  - with Lua coroutine
- Async I/O
  - see luvit

Next Step

- an «official» announcement
- a NQP backend
  - with perl6 community

Others recent good news

- NQP on JVM bootstrapped
- MoarVM (Metamodel On A Runtime Virtual Machine)

Bibliography / Webography

- www.luajit.org
- www.lua.org
- Sortie de LuaJIT 2.0.0 @linuxfr
- Slides LuaJIT @OSDC.fr 2011

- http://lists.parrot.org/pipermail/parrot-dev/2013-February/007345.html