Pontifícia Universidade Católica
do Rio de Janeiro

Lua & Pallene

Roberto Ierusalimschy

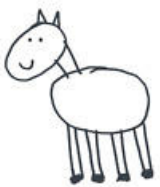
DConf'22

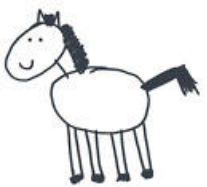
HOW TO:
DRAW A HORSE

BY VAN OKTOP

① DRAW 2 CIRCLES

② DRAW THE LEGS



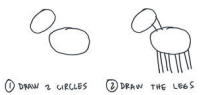


③ DRAW THE FACE

④ DRAW THE HAIR

HOW TO:
DRAW A HORSE

BY VAN OKTOP



1 DRAW 2 CIRCLES

2 DRAW THE LEGS

3 DRAW THE FACE

4 DRAW THE HAIR

5 ADD SMALL DETAILS.

Lua is a language with lots of small details.
(collaboration with L. H. de Figueiredo and W. Celes)

Pallene is a language still with few details, designed to be used as a system language for Lua in a scripting architecture.
(collaboration with H. Gualandi)

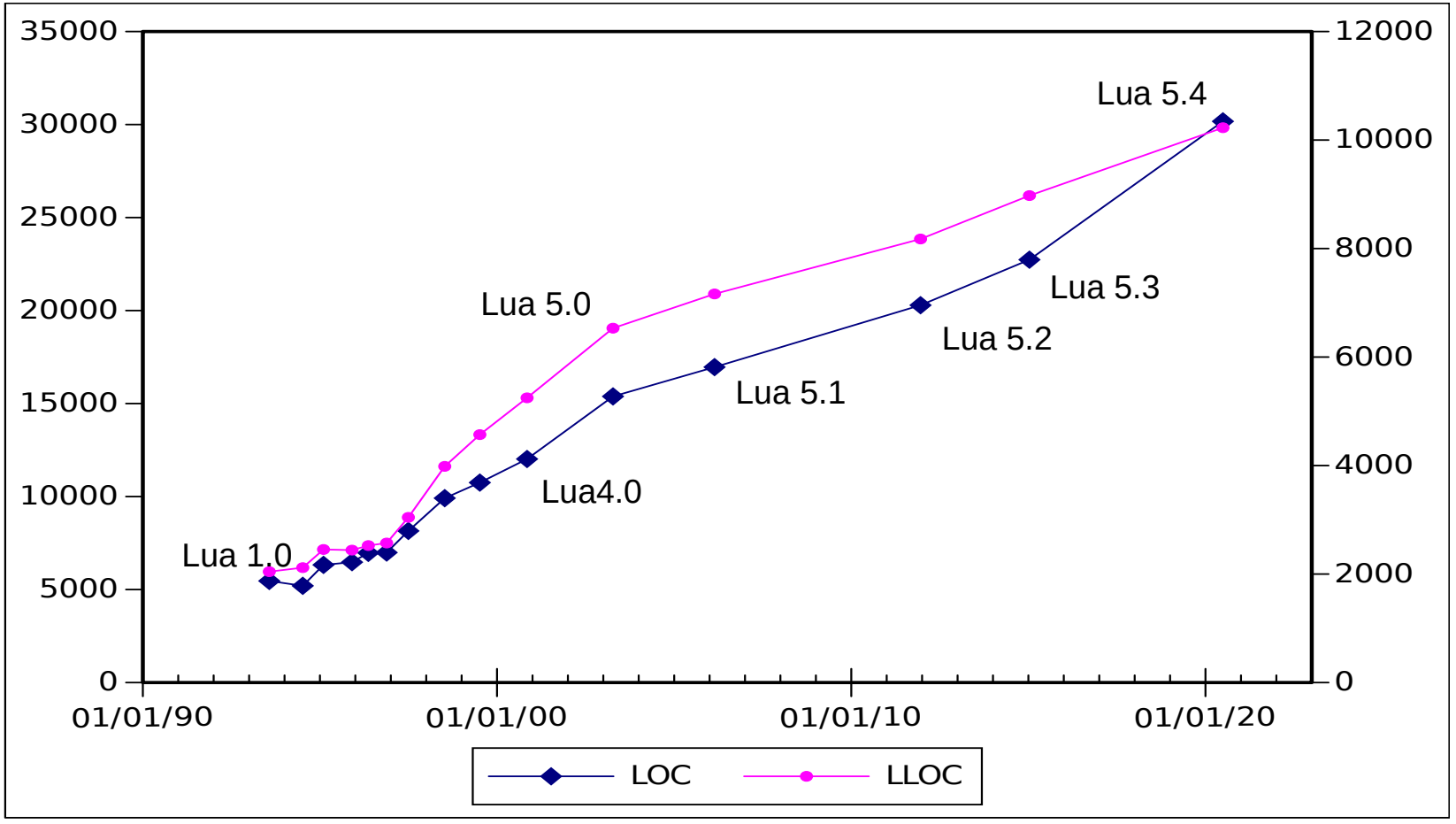
Why Lua?

- Yet another scripting language
- With quite specific goals:
  - Portable
  - Small
  - Simple
  - Emphasis on scripting

Portability

- Runs in virtually any platform with ANSI C.
- Runs inside OS kernels. (e.g., NetBSD)
- Runs directly on the bare metal, without an OS. (e.g., NodeMCU ESP8266)

Size

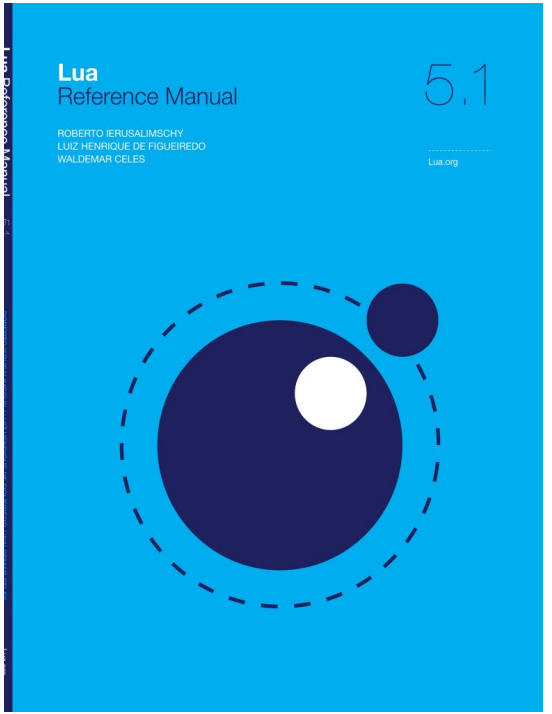

35000
30000
25000
20000
15000
10000
5000
0
12000
10000
8000
6000
4000
2000
0
01/01/90
01/01/00
01/01/10
01/01/20
Lua 1.0
Lua4.0
Lua 5.0
Lua 5.1
Lua 5.2
Lua 5.3
Lua 5.4
LOC
LLOC

Simplicity

Reference manual with ~100 pages

Documents the language, the C API, and the standard libraries.

(SPINE)



Simplicity

- Few but powerful mechanisms
- *Tables*
- Closures
- Coroutines

Tables

- Associative arrays
- Any value as key
- Arrays are tables with integer keys
- Records are tables with literal keys
- Modules are tables, objects are tables

Arrays as Tables

Implementation ensures that tables used as arrays are stored as arrays.

Sparse arrays come for free.

Closures

– First–class anonymous functions with lexical scope
– Widely used in regular Lua code
– Modules are tables populated with anonymous functions
– Exception handling through *protected calls*

```
try {
  <block/throw>
}
catch (err) {
  <exception code>
}
```

(Lambda the ultimate block constructor)

```
local ok, err = pcall(function ()
  <block/error>
end)
if not ok then
  <exception code>
end
```

Coroutines

– Equivalent to cooperative multithreading
– Equivalent to one–shot continuations (`call/1cc`)
– Covers most uses of full continuations with a fraction of the complexity

Scripting

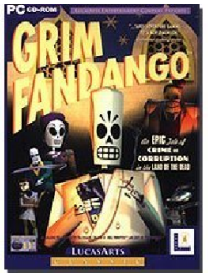
- Scripting language x dynamic language
- Program written in two languages: a *scripting* language and a *system* language
- System language implements the hard parts: algorithms, data structures
- Scripting language connects those parts

Lua and Scripting

- Lua is implemented as a library.
- Lua has been designed for scripting.
- Good for *embedding* and *extending.*
- Embedded in C/C++, Java, Fortan, C#, Perl, Ruby, Python, etc.

Embedded Systems

Samsung (TVs), Cisco (routers), Logitech (keyboards), Volvo (panels), Mercedes (panels), Olivetti (printers), Océ (printers), Ginga (TV middleware), Verison (set–top boxes), Texas Instruments (calculators), Sierra Wireless (M2M), NodeMCU (IoT), Technicolor (gateways), …

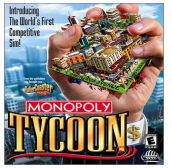

PC CD-ROM
GRIM FANDANGO
LucasArts


BUZZ!
Quiz TV


BRUTAL LEGEND


Tap Tap
Revenge


BLITZKRIEG


Introducing The World's First Competitive Sim!
MONOPOLY
TYCOON


DRIVER
SAN FRANCISCO

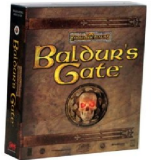

ROBLOX


Games for Windows
PC CD
COMPANY of HEROES
THQ


amped

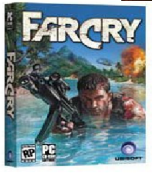

XBOX 360
HULK


PlayStation.2
DESTROY ALL HUMANS!


Baldur's Gate


RAIL SIMULATOR


WORLD of WARCRAFT


FARCRY

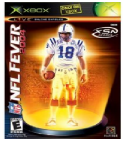

IMPOSSIBLE CREATURES


STALKER


Games for Windows
PC DVD
Lord of the Rings
Online

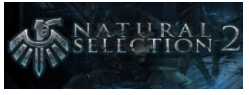

REQUIEM
MEMENTO MORI


Blossom


NFL FEVER
18

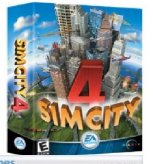

soul ride
Snowboarding Simulator


DINER DASH 2


NATURAL SELECTION 2


ALLODS
ONLINE


XBOX
XSN
Links


4
SIMCITY


Games for Windows
PC DVD
UNIVERSE AT WAR
SEGA


ANGRY BIRDS
PLAY


SpellForce


NAPOLEON
TOTAL WAR
16
SEGA


Games for Windows
PC DVD
SUPREME COMMANDER


BATTLE ZONE
98 REDUX

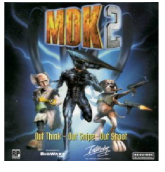

XBOX 360
TOPSPIN 2


XBOX 360
SONIC
THE HEDGEHOG


Games for Windows
PC DVD
THE WITCHER
"90 OUT OF 100" PC GAMER
PC
ATARI


XBOX 360
LIVE
CRACKDOWN


CRIMSON
TITAN PIRATES


MDK2

Games for Windows
PC DVD
CRYSIS

Games for Windows
PC DVD
SUPREME COMMANDER
Forged Alliance

HOMEWORLD 2

PlayStation.2
PSYCHONAUTS

Bubble
Ball

THE INCREDIBLES

There

Games for Windows
DVD
DUNGEONS

PlayStation.2
RAVE

Games for Windows
DVD
DUNGEONS
kalypso

STAR WARS
EMPIRE AT WAR

PlayStation.
Home

the Guild 2

ESCAPE FROM
MONKEY ISLAND

EMPIRE
TOTAL WAR
16
SEGA

PC DVD
CIVILIZATION

plus+

XBOX 360
LIVE
CRACKDOWN

HEROES V

PC DVD
DAWN of WAR
DARK CRUSADE

YOU ARE
EMPTY

XBOX 360
LIVE
FABLE II

Scripting and Performance

“If it's slow, rewrite it in C”

Easier said than done…

- Data mismatch
- Language mismatch

Data Mismatch

- Data has to be transferred between the two languages.
- Data has to be converted between the two languages.
- This process can kill any gains in performance due to a faster language.

Language Mismatch

- Big differences between Lua and C.
- It can be expensive to convert code from Lua to C.
- It can be hard to predict whether it is worth converting.

What about JITs?

- Don't change the language.
- Can achieve quite good performance.
- Hard to implement, port, and maintain.
- Optimization killers.

  Programmers can go to great lengths to appease a JIT. At what point does it become a good idea to switch to a typed language?

Pallene: a Companion Language

- Pallene has been designed to act as a system counter–part of Lua in a scripting architecture.
- To reduce language mismatch, it is a typed subset of Lua.
- To reduce data mismatch, it operates directly on Lua data.
- Scripting + gradual typing.

```
local function addqueen (N:integer, a:{integer}, i:integer)
  if i > N then
    printsolution(N, a)
  else
    for c = 1, N do
      if isplaceok(a, i, c) then
        a[i] = c
        addqueen(N, a, i + 1)
      end
    end
  end
end
```

Some Design Principles

- Same selling points from Lua.
- Very simple type system.
- Good on the borders.
- Gradual guarantee.
- Simple AOT compiler.

Selling points

Portable, Small size, Simple, Emphasis on scripting

Simple type system

The primary goal of the type system is to help the compiler, in particular to allow *unboxing*!

Everything else can be handled with any, the dynamic type. Or, better yet, kept in Lua.

Good on the borders

Real programs are seldom fully translated. Only the performance–critical parts need optimizations.

The change of one single function from Lua to Pallene should not worsen the performance.

Gradual Guarantee

Pallene functions should have the same semantics of their translation to Lua (by removing type annotations), except for type errors.

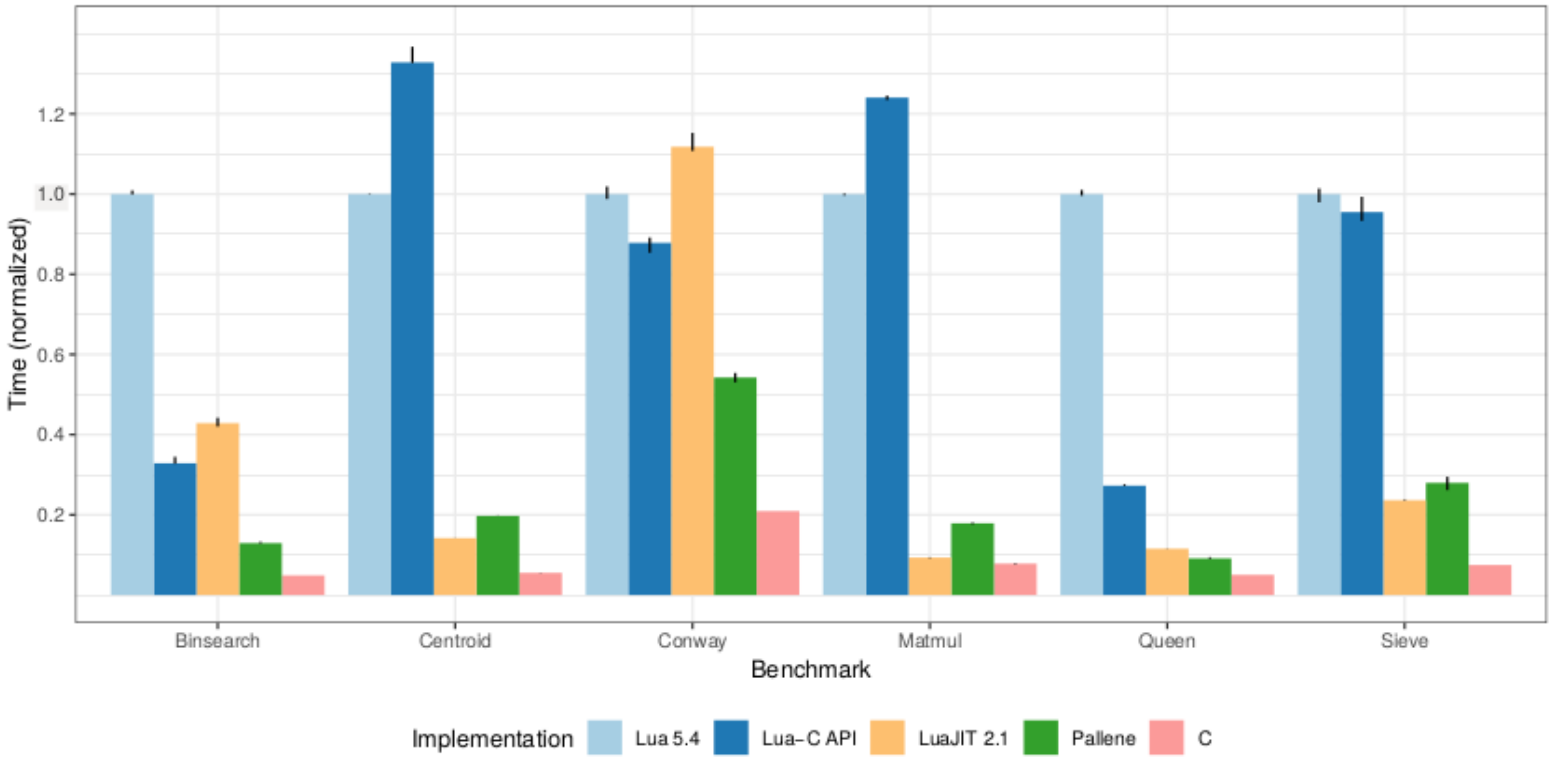
Hard to ensure to the letter in a real language like Lua. Simple solution is to remove features from Pallene, which makes it less expressive.

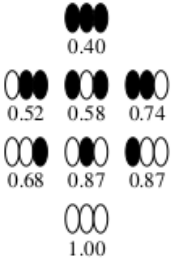
Simple AOT compiler

Generates C code that can be loaded by the standard Lua interpreter, as a C module.

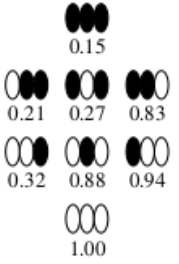
Keeps the same standards of portability as Lua itself.

Simplifies the implementation.

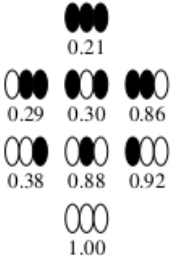

Time (normalized)
1.2
1.0
0.8
0.6
0.4
0.2
Binsearch
Centroid
Conway
Matmul
Queen
Sieve
Benchmark
Implementation
Lua 5.4
Lua−C API
LuaJIT 2.1
Pallene
C



(a) Spectral Norm



(b) Queens



(c) Nbody

(d) Stream Sieve

Conclusions

- Lua is a mature language with strong niches in games and embedded software.
- Lua selling points are simplicity, portability, small size, and emphasis on scripting.
- Performance is an always-present concern for dynamic languages.
- A companion language is an approach for improving the performance of Lua that seems compatible with its selling points.