The Racket Virtual Machine as an application of CS 4400
Some Racket Applications

Hacker News

Game Content

Practical Typography

Arc

DC

Pollen

Racket

Racket

Racket

Telescope Controller

Synthesized Program

Homework

DrRacket

Rosette

Beginner Student

Racket

Racket
Virtual Machines

... 
Racket Libraries & Programs 
Racket Virtual Machine 
C 
Operating System 
Memory Hierarchy 
Instruction Set Architecture 
Hardware
Virtual Machines

- ... 
- Racket Libraries & Programs 
- Racket Virtual Machine 
- C 
- Operating System 
- Memory Hierarchy 
- Instruction Set Architecture 
- Hardware
Virtual Machines

... 
Racket Libraries & Programs 
Racket Virtual Machine 
C 
Operating System 
Memory Hierarchy 
Instruction Set Architecture 
Hardware
Virtual Machines

- ... 
- Racket Libraries & Programs
- Racket Virtual Machine
- C
- Operating System
- Memory Hierarchy
- Instruction Set Architecture
- Hardware
Racket Libraries & Programs

- DrRacket
- Slideshow
- GUI
- `raco exe`
- web server

Racket Virtual Machine

- interpreter
- primitives
- JIT compiler
- FFI
- garbage collector
C

The Racket runtime system is implemented in C

src/racket
Representing Numbers

- Representing fixnums: \texttt{SCHEME\_INTP} and \texttt{SCHEME\_TYPE}

- ADD for fixnums

- \texttt{SCHEME\_RATIONAL\_FROM\_FLOAT} for inexact->exact
x86-64 Machine Model

Just-in-Time (JIT) compiler:

- ARITH_ADD

jitarith.c
Representing Control Flow

Just-in-Time (JIT) compiler:

- "unbox" implementation

- list_ref_code implementation

See also github.com/mflatt/jit-demo
Representing Procedures

- unsafe/ffi

  \[
  \text{(define \texttt{atoi}}\ \text{(get-ffi-obj} \texttt{"atoi"}
  \text{#f}
  \text{fun string \text{->} int))}
  \]

- backtrace

- continuations
Arrays

• `array-ref`  
   `ffi/unsafe.rkt`
Structures

- Scheme_Object
- Scheme_Bignum
- Scheme_Small_Bignum
- Scheme_IR_Local
Optimization

- `scheme_application_type` case in `scheme_do_eval`  
  `eval.c`

- `XFORM_ASSERT_NO_CONVERSION` and `fd_write_string` vs. `fd_write_string_s slower`  
  `port.c`
More on Optimization

- Branch-prediction interaction in `scheme_generate_non_tail_call`
Memory Hierarchy, Locality, Caches

- `repair_heap`'s fused loops for the `SIZE_CLASS_SMALL_PAGE` case

```c
newgc.c
```
Linking

The **unsafe/ffi** functions work by dynamically loading shared libraries

- **ffi-lib** uses `dlopen`
- **get-ffi-obj** uses `dlsym`  
  
  `rktio/rktio_dll.c`
ELF and Relocation

*raco exe* creates an executable by

- copying a stub binary that links to the Racket runtime system
- adding a new ELF section to hold bytecode for the Racket source

```
collects/compiler/private/elf.rkt
```
Processes

Racket runs `/bin/uname` to get the result of

```scheme
(system-type 'machine)
```

`string.c`
More on Processes

The *subprocess* execs an arbitrary program

Implementation uses *fork* and *execve*, and *waitpid*

```c
rktio/rktio_process.c
```
File Descriptors

Racket’s I/O uses file descriptors directly

• `fd_get_string_slow`
Signals

- **SIGINT** handler in **main**
  - main.c

- **SIGCHLD** handler related to **subprocess**
  - rktio/rktio_process.c
Virtual Memory

Garbage collector allocates pages using `mmap`

Write permission is disabled to implement a `write barrier` for generational collection

Handler calls `designate_modified_gc` in `newgc.c`
Dynamic Memory Allocation

- `allocate`

`newgc.c`
More on Memory Allocation

- `do_malloc` uses a free list

- Segmented allocation
Garbage Collection

- Bootstrap with conservative collector
  sgc.c
- Convert C code to cooperate with precise GC
- Production GC is fairly complex
  newgc.c

See also github.com/mflatt/jit-demo
Network Programming

- DNS

net/dns.rkt
More Network Programming

- Web server
- `raco pkg`
- Git checkout `net/git-checkout.rkt`
Concurrency

• File and network reads are multiplexed internally

• `getaddrinfo_in_thread`   `rktio/rktio_network.c`
Synchronization

• Mutex at Racket-thread level protects hash tables (e.g., hash_table_count)

• GC keeps a list of threads for cooperation on macOS
...And More

What topics crucial to Racket weren't covered in CS 4400?

• Programming and data structures
• Interpreters and compilers
• Databases
• GUIs and graphics
• Rules and strategies for portability