LÉPIX

A very small General Purpose Language

THE BIG IDEA

- THE BEST LANGUAGE EVER
  - STRUCTS
    - CONSTRUCTORS, DESTRUCTORS, DETERMINISTIC DESTRUCTION WOO
  - PARALLELISM
    - MASSIVE AMOUNTS OF IT!
    - ALL THE CONCURRENCY
  - FUNCTIONS
    - SO MANY! BUILT IN IMAGE PROCESSING
  - SUPER MULTIDIMENSIONAL ARRAYS

A tiiiny problem…

- Had to work on the project alone
  - Heavy time constraint
  - Aaaahhh

2 weeks, Lots to Do

- No Semantic Analyzer, Lexer/Parser not parsing the language, Segmentation Faults galore, no medicine for nine months, no time
- … Here we go!

The *Better* Idea

- Relax, and take several Chill Pills
  - And still panic
- Focus on implementing a small subset of what was needed, but well

No Structs

- Not for lack of trying!
- Memory safety = gone
  - No constructor/destructor, no automatic memory cleanup (manual new/delete, essentially)

No Parallelism

- Not for lack of Trying
  - Had hand-compiled demo code for parallelism
  - Worked with arrays and other things
  - Couldn't jerry-rig it into the compiler in time
- A bit sad
  - One of the shiniest features

Even no Arrays :(

- At this point, a bit heartbroken
- The syntax, at least, was good
- Slicing
  - The number of arguments in […] = number of shed dimensions
  - Gives C-Like dimension access (z, y, x …)
  - Tossed around by-value

Functions!

- Thankfully, have the most basic functions
  - Parameters by value
    - Mostly because that is all there is!
    - Plans for everything by value with optional reference (&) qualifier
    - Plans for reference analysis
- Overloading selects which function to call properly!
  - Compile-time arity and argument-type based
  - Very strict, no covariance, codegen mangles names

Most lost features still there

- lepixc -s inputfile
  - Invokes the compiler and shows the SemanticAST
  - The semantic AST parses arrays, fixed-sized arrays, parallel blocks, functions
- But lost time struggling with semantic AST for weeks
  - Codegen suffered greatly, even if everything else was well-done

Implementation

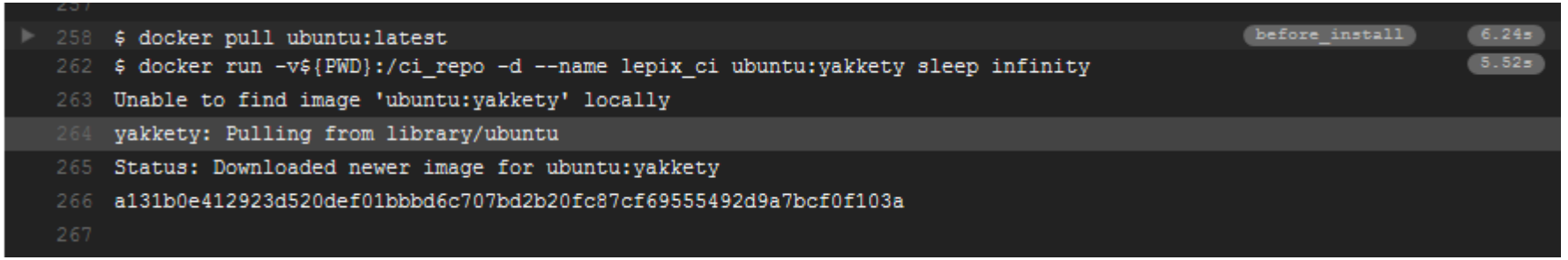
- Problem: Records were initially extremely painful to work with
  - New state that changes one field? Re-vomit all fields and write them all out
- Time Saver: “with” record syntax
  - { record_name with field1 = single_change; }
  - allows for complex records with easy updates

Implementation II – Having Fun

- Might as well get decent at immutability
  - Each function call is entirely self-contained with only dependencies on its arguments
  - Barely any usage of ref

Implementation III – Even More Fun

- Travis CI builds and runs the test suite for every push
  - Useful for knowing when / how things went wrong!
  - A lot of tests failed a lot of the time



Standard OCaml Library?

- Pervasives (the builtins) are sparse
  - Batteries, JaneStreet Core helps with this
  - Some file functions, string manipulation functions not present in version of Ocaml that comes with VM
    - Travis-CI testing required lower level compiler
- Using provided libraries means using OPAM and ocamlbuild
  - Killed the windows build

Things to add in the future

- Structs
  - Needed for proper static language handling
  - Enables IIFEs and captures
- Parallelism
  - Formal implementation and not the handwritten hack that works in only 1 case and breaks everywhere else
- Real multidimensional arrays
  - We used “getelementptr” LLVM instruction for printf calls, is also used with structs/arrays and slicing arrays

Learned Things I

- OCaml is nice
  - Overloading would have been useful
  - Abstract Data Types useful for new things, not employed usefully for regular things
    - string_of_int, string_of_float, String.make 1 ch …
    - Primary motivation for Overloading implementation
  - "Build list then reverse" idiom is a bit annoying
    - Happens everywhere, but alternatives to handling are strange
- Compiler and Ocaml environment do not work well for Not-Linux
  - At least Torvalds is happy?

Learned Things II

- LLVM Binding is somewhat immature
  - Can set custom attributes, but cannot retrieve them (made handlers for native functions difficult)
- Reaching out for help would have been good
  - Understanding the breakdown in communication for teammates would have been better than being upset
  - Bailing not the most desirable option

Demo

- Time to break the compiler!