Progress report in Pen programming language

June 4th, 2022

@raviqqe

Agenda

- Progress report
    - Heap reuse in record updates
- Next plans

Progress report

Heap reuse in record updates

- Programs reuse memory blocks if record updates are performed on unique references.
- Part of the Perceus reference counting algorithm
- Limited to records with more than one field
    - One-field records are planned to be unboxed.
- Similar to `Arc::make_mut()` in Rust

```
type foo {
  bar number
  baz number
}

f = \(x foo) foo {
  # Reuse a memory block originally allocated for x if its reference is unique.
  foo{...x, bar: 42}
}
```

Benchmark

- Map insertion
- Element count: 100,000
- ~20% improvement in time

Before:

```
> time ./app
./app  0.84s user 0.05s system 94% cpu 0.945 total

> valgrind ./app
total heap usage: 2,414,147 allocs, 2,414,074 frees, 262,931,512 bytes allocated
```

After:

```
> time ./app
./app  1.09s user 0.03s system 95% cpu 1.168 total

> valgrind ./app
total heap usage: 2,446,293 allocs, 2,446,220 frees, 279,647,432 bytes allocated
```

Further improvements

- Still so many allocations in the benchmark...
- Pen is missing some basic optimization.
    - e.g. lambda lifting

Others

- The standard `Sql` package
- FFI improvements
  - Conversion between lists in Pen and async streams in Rust
  - Type casting of `any` type values
- Attribute changes in LLVM
  - e.g. `unnamed_addr`, `nounwind`, etc.

Next plans

- Reference counting optimization
  - Relaxed atomic reference counting with sync bit #468
  - Unboxing small records #671
- Proper C calling convention in FFI #444
  - Compiling to MLIR?

Summary

- Progress
  - Heap reuse in record updates
- Next plans
  - Relaxed atomic reference counting
  - Record unboxing

Benchmark (old)

- Map insertion (with list iteration)
- Element count: 100,000

Before:

```
> time ./app
./app  0.94s user 0.08s system 95% cpu 1.064 total

> valgrind ./app
total heap usage: 3,414,149 allocs, 3,414,074 frees, 290,931,552 bytes allocated
```

After:

```
> time ./app
./app  1.10s user 0.05s system 95% cpu 1.202 total

> valgrind ./app
total heap usage: 3,446,295 allocs, 3,446,220 frees, 307,647,472 bytes allocated
```