Iterative Stencil Computations in Ruby on GPUs

Matthias Springer    Peter Wauligmann    Hidehiko Masuhara
Department of Mathematical and Computing Science, Tokyo Institute of Technology

What is Ikra?

- RubyGem for array-based GPU computing
- Compiles Ruby code to C++/CUDA program
- Current focus: Iterative scientific computations
- Parallel map, reduce, stencil, new
- Data types inside parallel/host sections:
  - primitive (int, float, bool, nil), array (read only), zipped
  - object (partial support, incl. method calls),
  - union type (combination of above ones)

Kernel Fusion in Loops via Symb. Execution

```
result = Ikra.host_section do
  arr = Array.pnew(10) do |i|
    idx[i] += 1
  end
  while arr.reduce(0) < 100
    arr = arr.map do |i|
      v[i] += sum
      i[i] += sum
    end
  end
  puts "Result is #{result.to_a}"
end
```

Design Decisions

- **Modularity**: Build complex programs from multiple parallel sections using object-oriented programming
- **Kernel Fusion**: Combine parallel sections into single GPU kernel, delay execution to the latest possible point
- **Host Section**: Avoid switching between Ruby interpreter and generated C++ program

Ikra API: Example

```
result = Ikra.host_section do
  arr = Array.pnew(10) do |i|
    idx[i] += 1
  end
  while arr.reduce(0) < 100
    arr = arr.map do |i|
      v[i] += sum
      i[i] += sum
    end
  end
  puts "Result is #{result.to_a}"
end
```

Symbolic exec. in Ruby interpreter: returns a `command` (contains all information for code generation + execution)

```
a = Array.pnew(dimensions: [10, 6]) do |idx|
end
b = a.pstencil([-1, 1], [0, 0], 0) do |v, i|
  value_sum = v[0] + v[1]
  index_sum = i[0] + i[1]
  value_sum + index_sum
end
```

Code Generation

- C++ type for polymorphic expressions: union type struct
- Method call with polymorphic receivers: switch stmt.
- Parallel section: Data structure for command data
- Kernel launch: Generated only for run, [], end of section
- Future work: Data sharing between multiple parallel sections (avoid redundant computation). Escape analysis to detect if it is safe to reuse the same memory location

Code generation example:

```
struct union t {
  union { int int; /* ... */ void *pointer; } data;
  int class_id;
}
```

High-level overview with kernel launches only

Type Inference on Host Section AST in SSA Form.
The type of a parallel section is the result of its evaluation in the Ruby interpreter.