SEJITS: Implementing a Specializer

Berkeley Parlab

Derrick Coetzee, Shoaib Kamil, Jeffrey Morlan, Armando Fox, David Patterson, …


Derrick Coetzee

Photo unavail-able

Shoaib Kamil


Jeffrey Morlan

Walking through an example with the stencil kernel SEJITS specializer

Stencil example: input

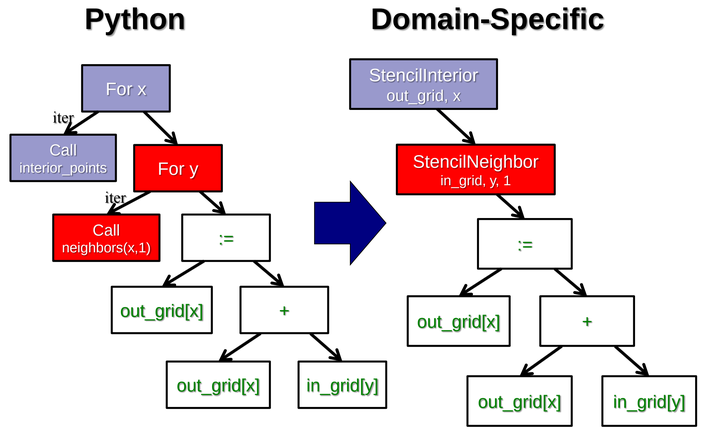
```
from stencil_kernel import *

class ExampleKernel(StencilKernel):
    def kernel(self, in_grid, out_grid):
        for x in out_grid.interior_points():
            for y in in_grid.neighbors(x, 1):
                out_grid[x] = out_grid[x] + in_grid[y]

in_grid = StencilGrid([5,5])
in_grid.data = numpy.ones([5,5])
out_grid = StencilGrid([5,5])
ExampleKernel().kernel(in_grid, out_grid)
```

```
1 1 1     0 0 0
1 1 1  →  0 4 0
1 1 1     0 0 0
```

1. Python AST → Domain AST

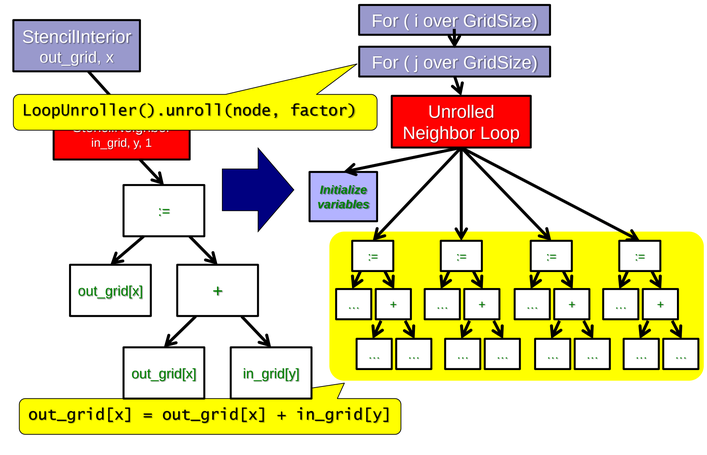


- Rejects ASTs that don't match domain format
- If specializer fails, fall back on running as pure Python
- Gives clear error messages in both cases

Opportunities for improvement:

- New specializers for new domains

2. Optimize Domain AST

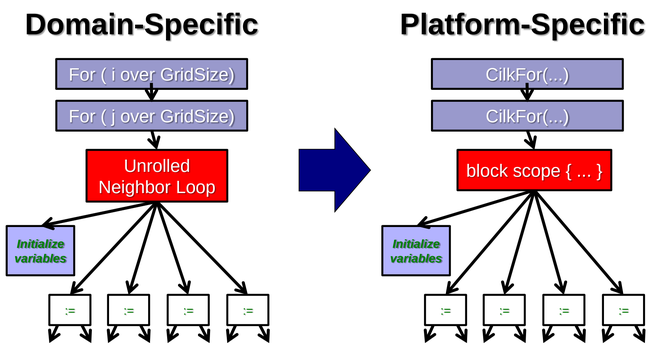


- Optimizations are platform-independent
- Optimizations can exploit domain knowledge

Opportunities for improvement:

- New domain-specific optimizations

3. Bind platform

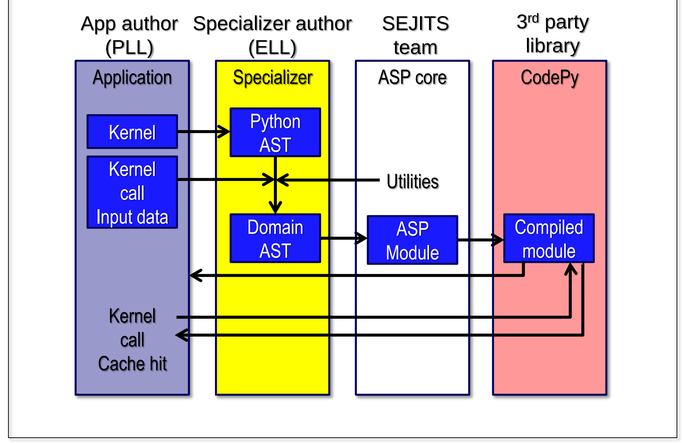


- Autotuning and other platform-specific optimizations
- Optimizations can be reused across "similar" platforms (e.g., emit a generic SIMD representation, let compiler map to final instruction set)

Opportunities for improvement:

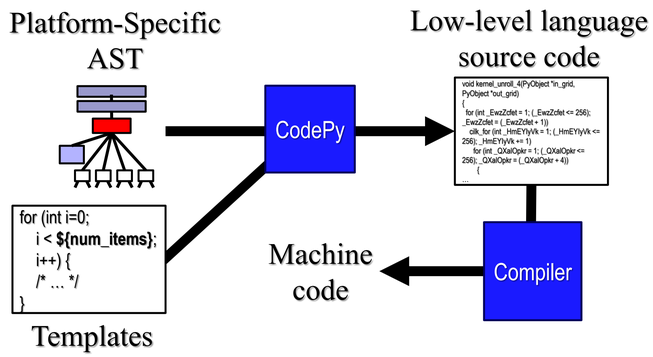
- Target new platforms (multicore, GPU, cloud)
- New platform-specific optimizations

Summary



Try SEJITS yourself at:
http://aspsejits.pbworks.com

4. Emit code



Opportunities for improvement:

- New code generation and compiler tools
  - Easy way to deploy a new code generation tool to a wide audience without application programmers having to learn a new language