EECS
Electrical Engineering and Computer Sciences

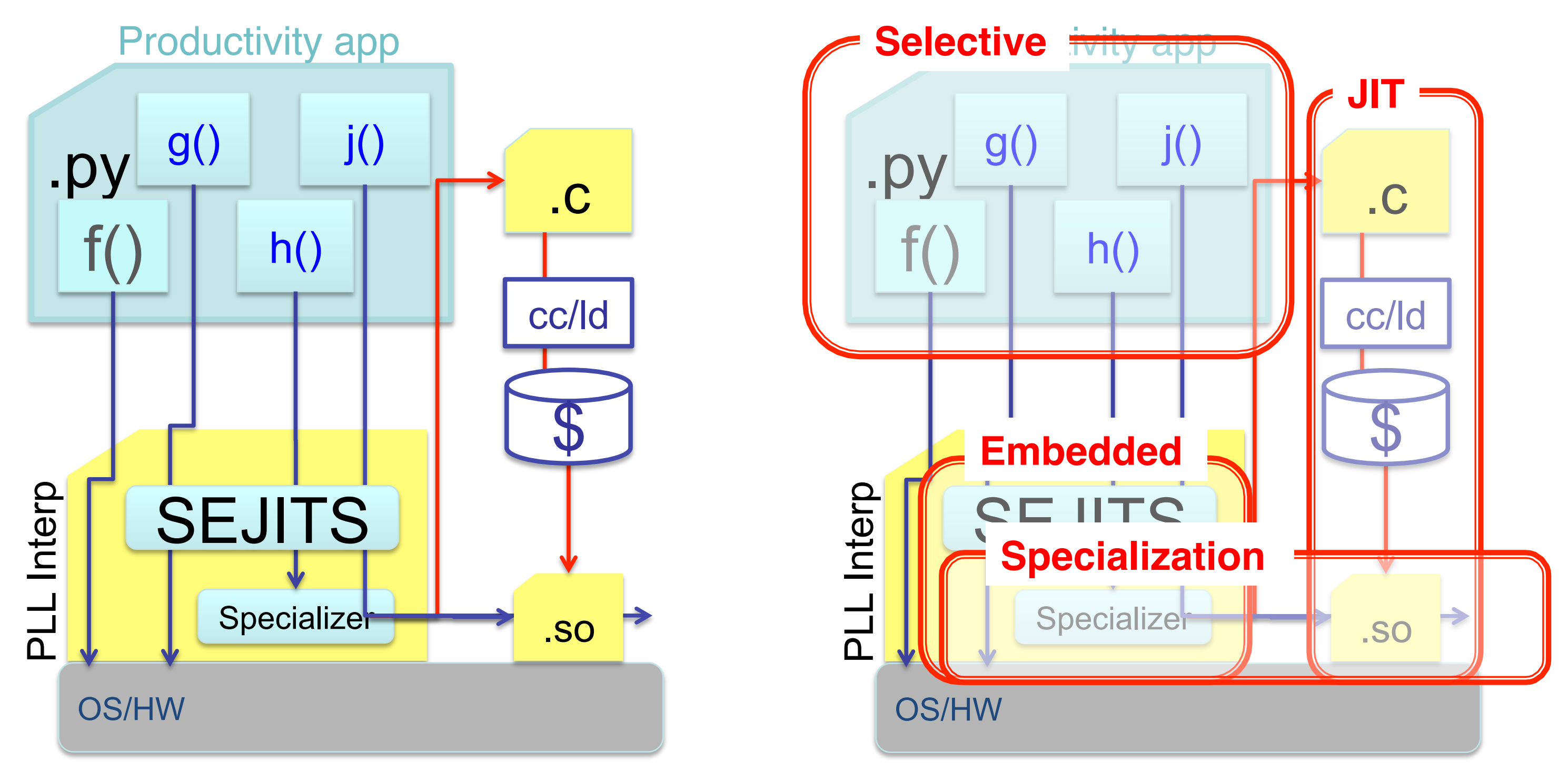
Asp Specializers & Recent Results

Shoaib Kamil, Armando Fox, David Patterson, Katherine Yelick, and many others

BERKELEY PAR LAB

P A R A L L E L   C O M P U T I N G   L A B O R A T O R Y

SEJITS Overview



Specializer == pattern-specific JIT compiler

- *Code templates* hand-authored by efficiency programmers in efficiency language (eg C++)
- *AST transformation* of VHLL code to instantiate templates
- Compile & run specialized code, return results to PLL
- Occurs invisibly to programmer

Asp: A SEJITS Implementation for Python

- Users write their apps in Python
  - Supports code generation in C/C++/CUDA/Cilk+
  - Under rapid development (patches welcome!)
- Public source repo:
  `git://github.com/shoaibkamil/asp.git`
- Wiki:
  `http://github.com/shoaibkamil/asp/wiki`
- Get involved
  - Pre-built VM environment using Vagrant
  - Build your own specializers and help us improve design/usability of Asp

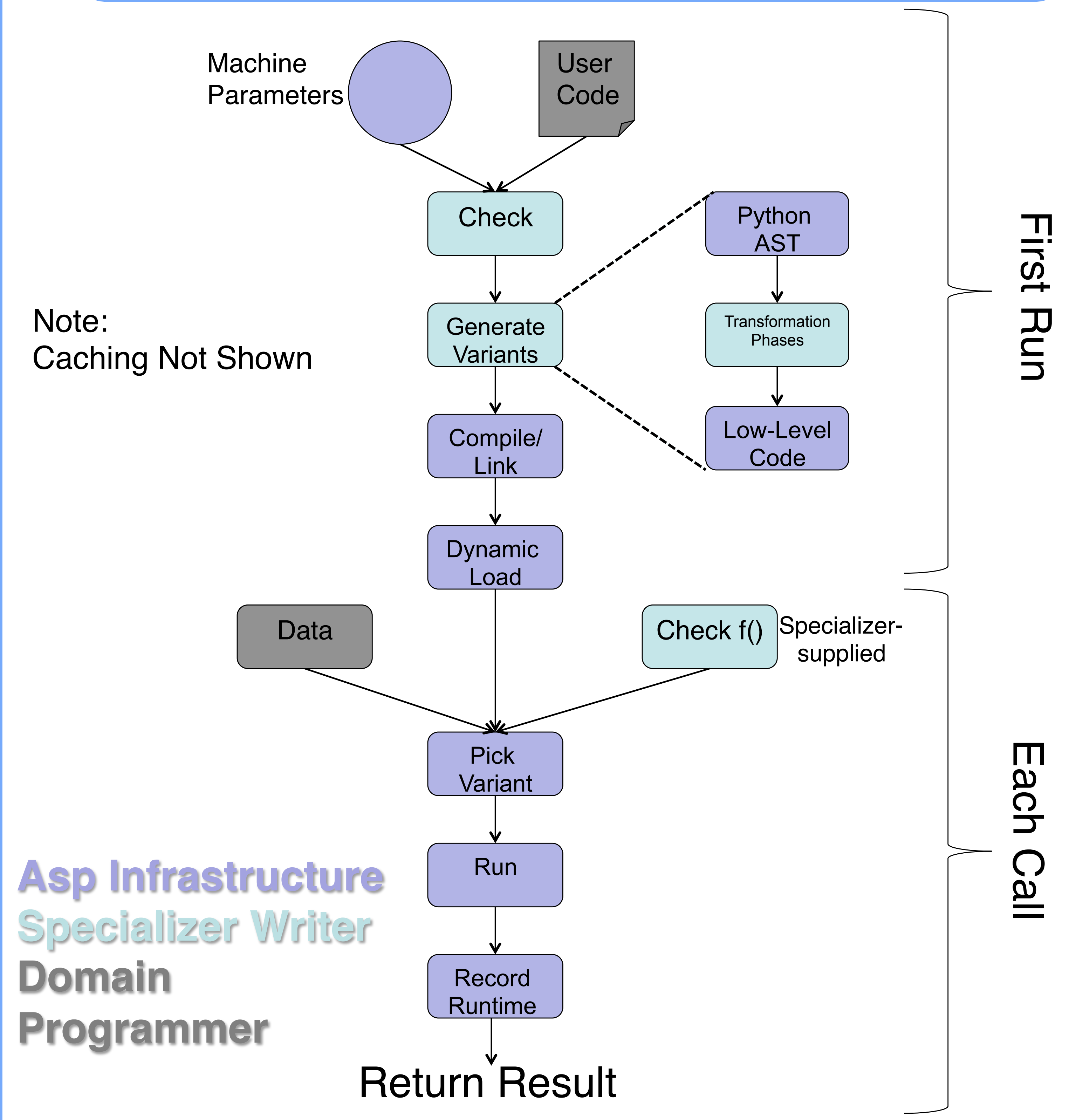
Other Asp/SEJITS Posters

- Implementing a Specializer (Derrick Coetzee)
- Matrix Powers in Asp (Jeffrey Morlan)
- Gaussian Mixture Modeling (Katya Gonina and Henry Cook)
- Using Machine Learning for Auto-tuning Multi-core Stencil (Orianna DeMasi)

Recent Goings-on in Asp

- Multiple specializers in development
  - Two support multiple backends already
  - Productivity programmers adopting specializers
- Summer students will be working on new specializers
- Aspdb, the global db for timings, now running on Google Appengine
- Machine learning already giving insight into recorded specializer run results
- Presenting at SciPy conference in July
  - Developer preview release planned to coincide

Specializer Structure

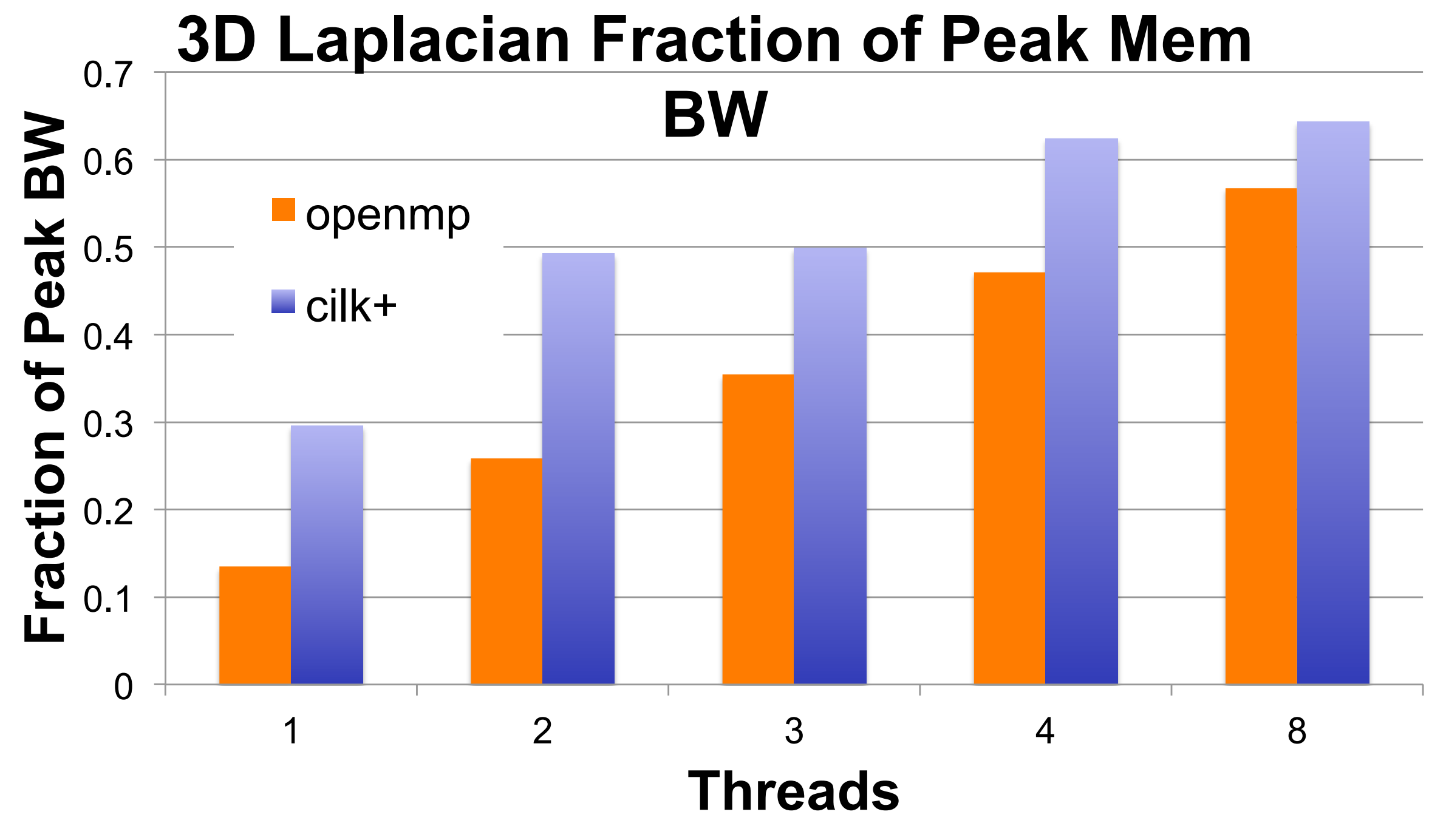


- Many parts handled completely or partially by Asp infrastructure
- Results recorded for entry into global Aspdb (global db as a service) and for future variant selection use
- See Derrick Coetzee's poster for more info on phased transformation

Stencil Specializer Results: Laplacian

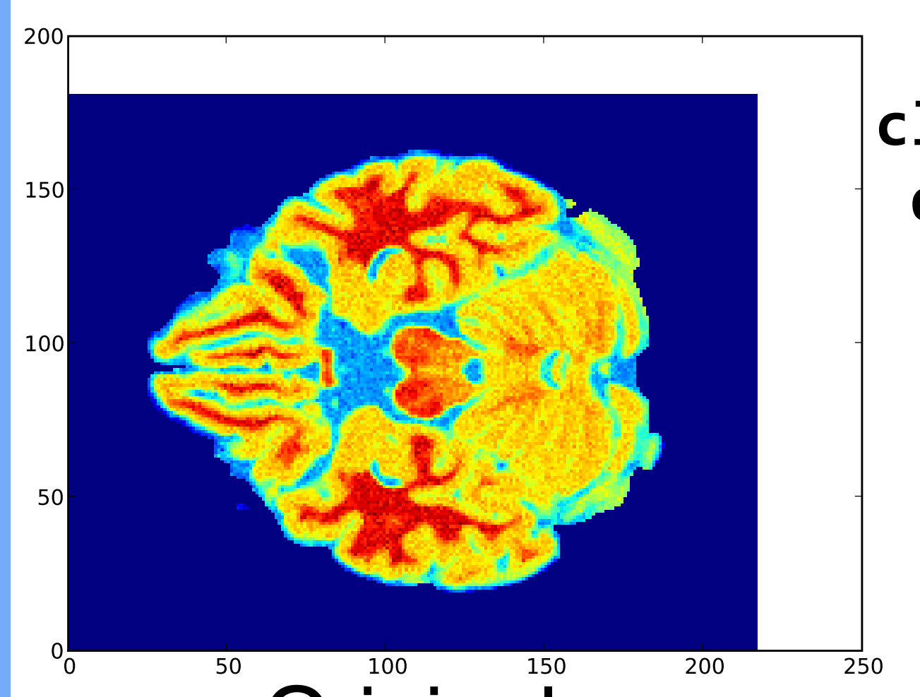
```
class Heat3D(StencilKernel):
  def kernel(self, in_grid, out_grid):
    for x in in_grid.interior_points():
      for y in in_grid.neighbors(x, 1):
        out_grid[x] = out_grid[x] + (1.0/6.0)*in_grid[y]
```

- Standard Laplacian heat equation benchmark
- Specializer outputs both OpenMP and Cilk+
- >65% of peak on Core i7 machine
  - 3 orders of magnitude faster than pure Python
  - Very few optimizations implemented so far
  - Believe we can approach 95+% of peak

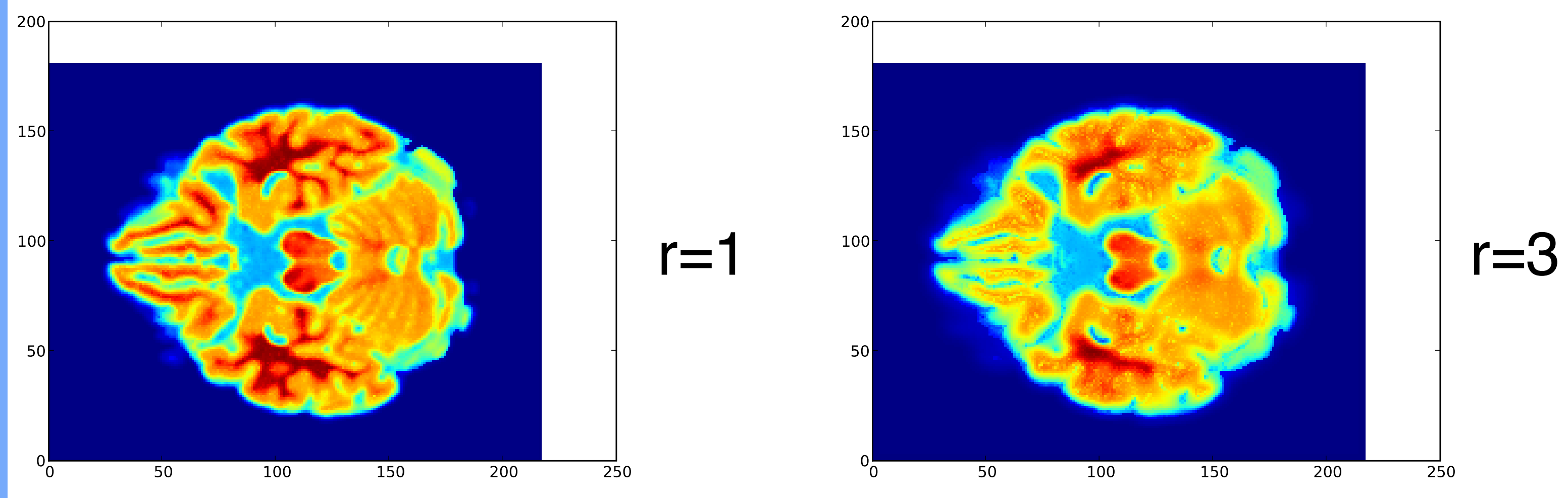


Stencil Specializer App: Brain Imaging

- 3D MRI brain data is noisy
- Bilateral filter applied to bring out varying features
  - Different features appear at different filter radii
- Memory bound at small radii, computation bound at large radii


Original

```
class BilatKernel(StencilKernel):
 def kernel(self,in_img,out_img,filter):
   for x in in_img.interior_points():
     for y in in_img.neighbors(x, r):
       out_img[x] =  out_img[x] +
         filter[abs(int(in_img[x]-
         in_img[y]))%255] * in_img[y]
```



Original imagery courtesy Prof. Owen Carmichael, Dept of Neurology, UCD and the UCD Alzheimer's Center.