4-23-2021

Optimizing Scientific Computations with the Sparse Polyhedral Framework

Anna Rift
Boise State University

Optimizing Scientific Computations with the Sparse Polyhedral Framework

Anna Rift
Advisor: Dr. Catherine Olschanowsky

BOISE STATE UNIVERSITY
COLLEGE OF ENGINEERING
Department of Computer Science

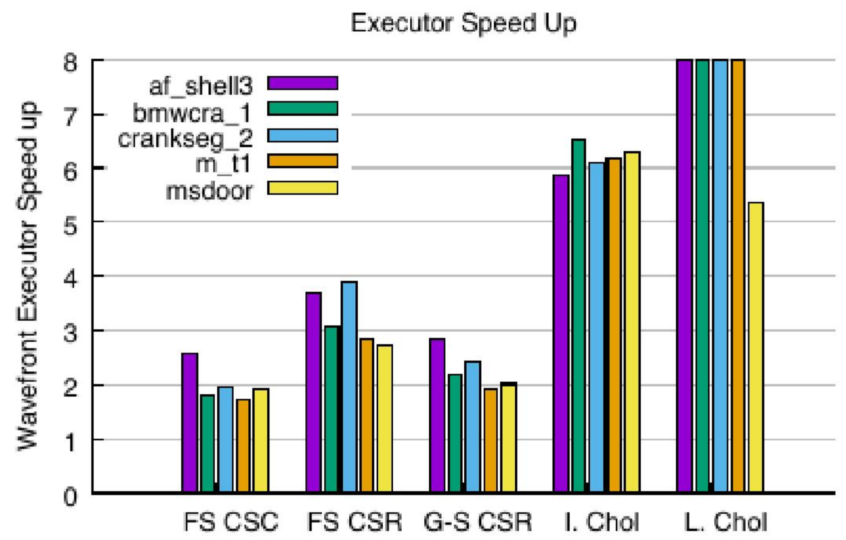
adapt
DATA FLOW OPTIMIZATIONS LAB

1. Problem Statement

- Scientific applications are computationally intensive, requiring expensive HPC resources
- optimizing scientific applications requires a balance of Performance, Productivity, and Portability

2. Motivation

Speedup of executor transformed for wavefront parallelism vs. library serial code.



FS CSC - Forward Solve, Compressed Sparse Column format
FS CSR - Forward Solve, Compressed Sparse Row format
G-S CSR - Gauss Seidel, Compressed Sparse Row format
I. Chol - Incomplete Cholesky
L. Chol - Left Cholesky

Source: Mahdi et. al.

3. The Polyhedral Model

- Represents the iteration of each statement of a computation in a loop nest as lattice points in a polyhedron
- Only supports affine data accesses -- **does not work for sparse computations**

```
for (i = 1; i <= 3; ++i)
  for (j = 1; j <= 3; ++j)
    S1(i, j)
```

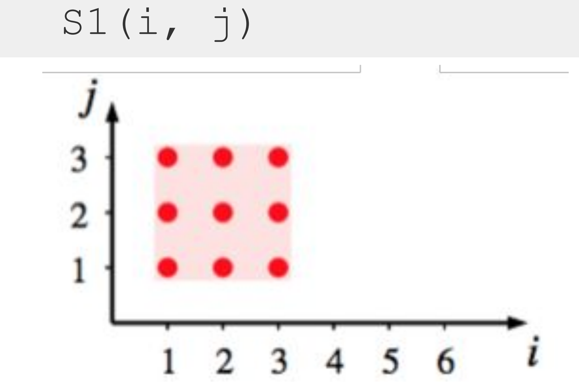
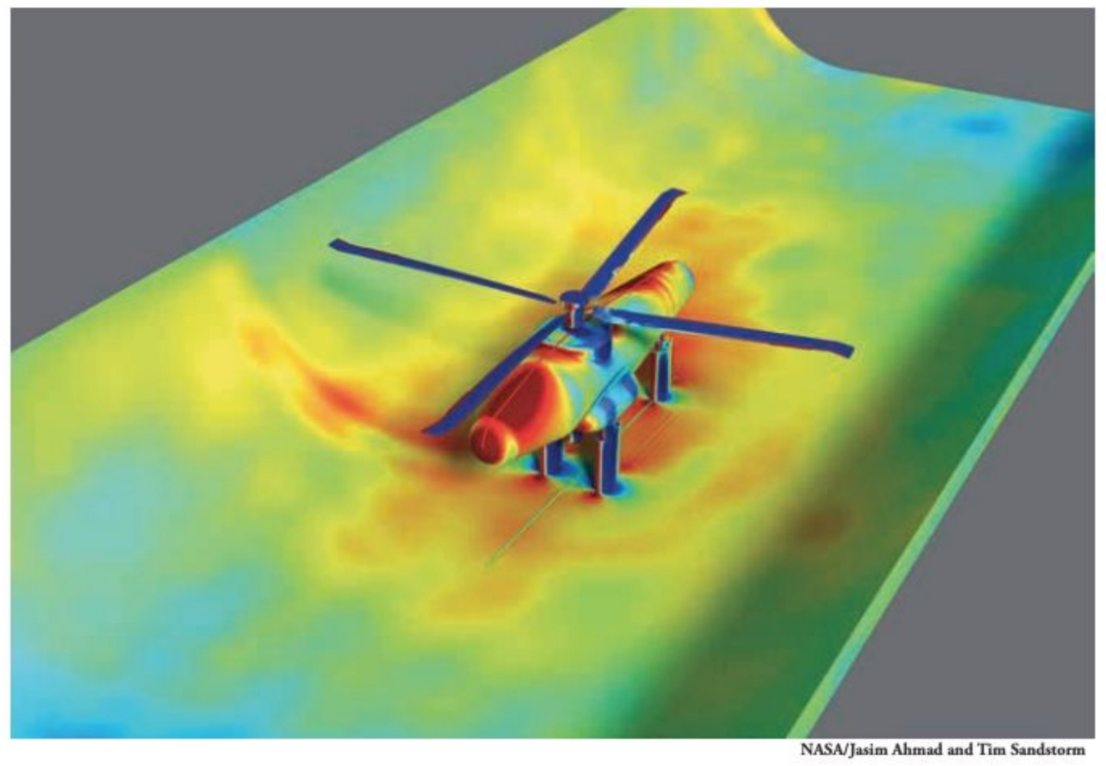


Image source: http://web.cse.ohio-state.edu/~pouchet.2/doc/cc-slides.10.pdf

4. Sparse Data



NASA/Jasim Ahmad and Tim Sandstorm

This colorful image is a Computational Fluid Dynamics simulation of a full-scale UH-60A rotor from a Black Hawk helicopter in the giant 40-by 80-Foot Wind Tunnel at NASA Ames Research Center in Moffett Field, California. Colors represent pressure – red is high pressure and blue is low pressure.

5. Sparse Polyhedral Framework (SPF)

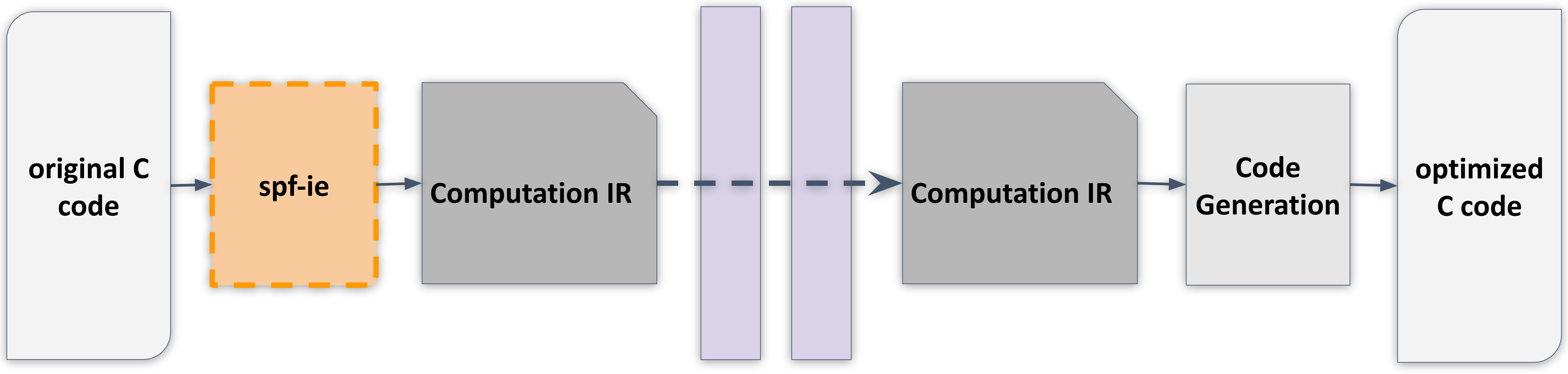
- Extends the polyhedral model
- Provides a mathematical framework for representing and transforming irregular computations (uninterpreted functions)
- Suitable for **non-affine** loop bounds present in irregular applications

```
for (i = 0; i < N; i++)
  for (k = index[i]; k < index[i + 1]; k++)
    product[i] += A[k] * x[col[k]];
```

⇩

$$\{[i,k] : i \geq 0 \ \&\& \ i < N \ \&\& \ k \geq index(i) \ \&\& \ k < index(i+1)\}$$

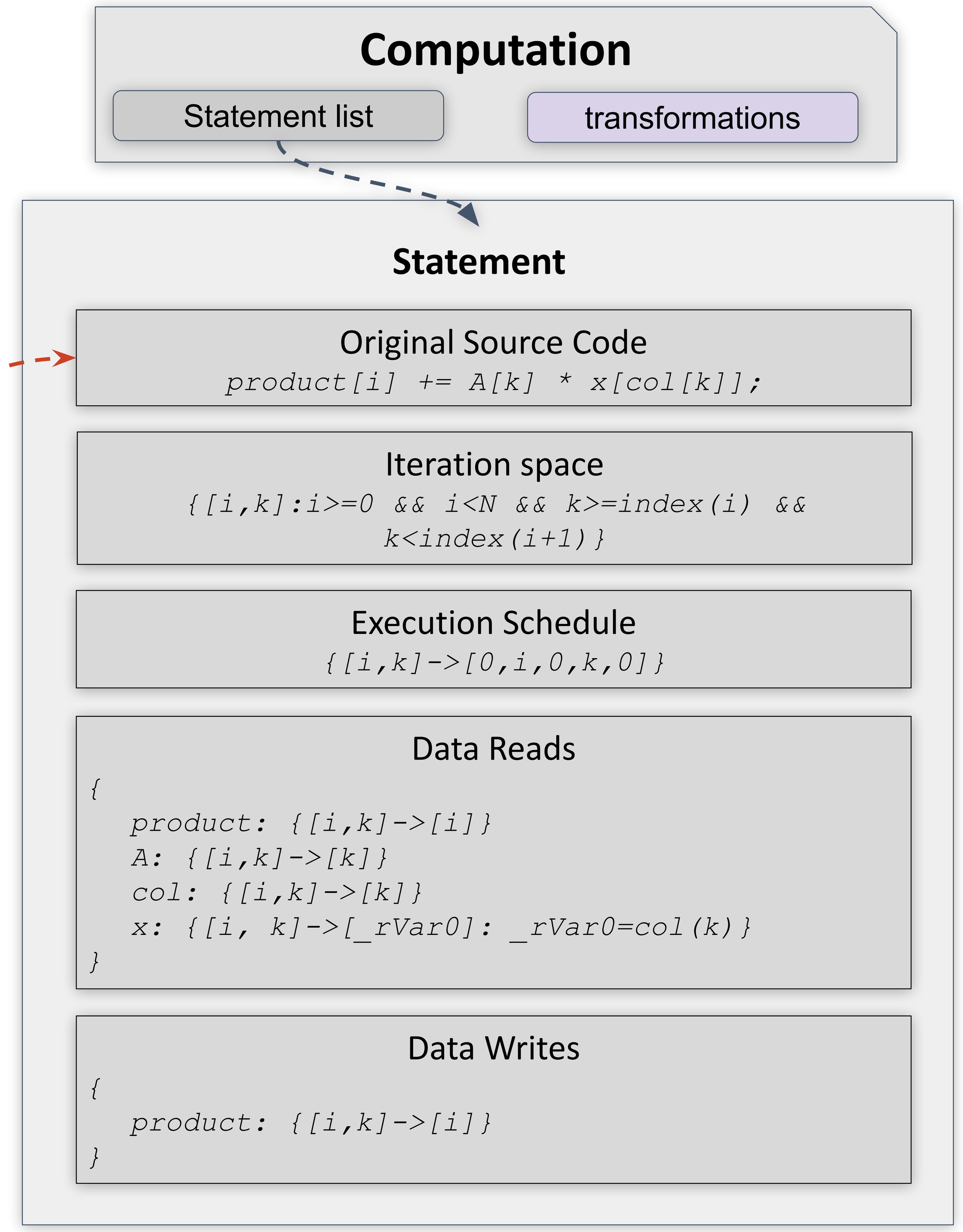
6. Optimization Overview



7. spf-ie

- Can be thought of as the compiler frontend of the project
- Extracts SPF representation of original source code, entering it into the Computation IR
- Implemented as a Clang tool that recursively traverses the abstract syntax tree
- Enforces polyhedral model restrictions on code (no *goto* statements, etc.)

8. Intermediate Representation



9. Future Development

- Currently only have an identity transformation, need to write more
- Algorithmically manipulating data layout to meet execution requirements
- Inlining computations that call others
- Synthesize IR to facilitate conversion from one sparse format to another

10. Acknowledgements

CHiLL-I/E: Ravi Shankar and Tobi Popoola
Boise State's Research Computing Department. 2017. R2: Dell HPC Intel E5v4 (High Performance Computing Cluster). Boise, ID: Boise State University. DOI: 10.18122/B2S41H

11. Collaborators

THE UNIVERSITY OF ARIZONA

THE UNIVERSITY OF UTAH