Verified Lifting of Stencil Computations

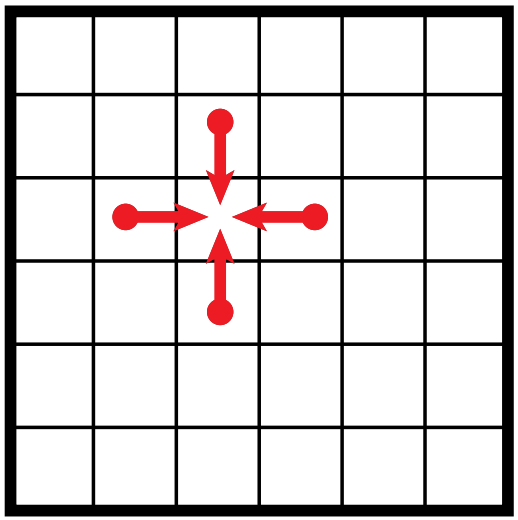
Shoaib Kamil
Adobe Research &
MIT CSAIL

Alvin Cheung
University of Washington

Shachar Itzhaky

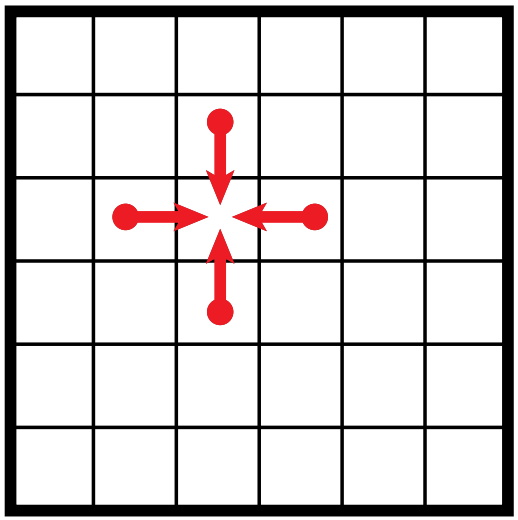
Armando Solar Lezama
MIT CSAIL

Stencil computations update each point in a multidimensional regular array with a function of a subset of its neighbors.



Adobe

$$out(i,j) = in(i,j) + 0.25 \times (in(i+1,j) + in(i-1,j) + in(i,j+1) + in(i,j-1))$$



Adobe

```
do ib=istart,iend,2*ibsize
do i=ib, min(ib+ibsize,iend)
do j=jstart,jend,2
  out(i,j) = in(i,j) + (1.0/4.0)*(in(i+1,j)+in(i-1,j)+in(i,j+1)+in(i,j-1))
  out(i,(j+1)) = in(i,(j+1)) + (1.0/4.0)*(in(i+1,(j+1))+ &
        in(i-1,(j+1))+in(i,(j+1)+1)+in(i,(j+1)-1))
enddo
do i=ib+ibsize, min(ib+2*ibsize,iend)
do j=jstart,jend,2
  out(i,j) = in(i,j) + (1.0/4.0)*(in(i+1,j)+in(i-1,j)+in(i,j+1)+in(i,j-1))
  out(i,(j+1)) = in(i,(j+1)) + (1.0/4.0)*(in(i+1,(j+1))+ &
        in(i-1,(j+1))+in(i,(j+1)+1)+in(i,(j+1)-1))
enddo
enddo
enddo
```

Verified Lifting

Goal: Given a snippet of stencil code, *lift* it into a high-level summary that fully describes the algorithm, without optimizations.

Overview

- Introduction
- Verifying A Summary
- Synthesizing Loop Invariants & Postconditions
- Evaluation
- Summary

Verifying a summary

```
procedure sten(imin,imax,jmin,jmax,a,b)
  real (kind=8), dimension(imin:imax,jmin:jmax) :: a
  real (kind=8), dimension(imin:imax,jmin:jmax) :: b
  do j=jmin,jmax
    t = b(imin, j)
    do i=imin+1,imax
      q = b(i,j)
      a(i,j) = q + t
      t = q
    enddo
  enddo
end procedure
```

$$\forall i,j \in [imin+1, imax] \times [jmin, jmax]$$
$$a(i,j) = b(i-1,j) + b(i,j)$$

Adobe

Verifying a summary

```
do j=jmin,jmax
 t = b(imin, j)
 do i=imin+1,imax
   q = b(i,j)
   a(i,j) = q + t
   t = q
 enddo
enddo
```

Loop invariant I_j

$$\forall i, j' \in [imin + 1, imax] \times [jmin, j)$$
$$a(i, j') = b(i - 1, j') + b(i, j')$$

Loop invariant I_i

$$\forall i, j' \in [imin + 1, imax] \times [jmin, j)$$
$$a(i, j') = b(i - 1, j') + b(i, j')$$
$$\forall i', j' \in [imin + 1, i) \times [j, j]$$
$$a(i', j') = b(i' - 1) + b(i', j')$$

$$\forall i, j \in [imin + 1, imax] \times [jmin, jmax]$$
$$a(i, j) = b(i - 1, j) + b(i, j)$$

Adobe

Verifying a summary

```
do j=jmin,jmax
 t = b(imin, j)
 do i=imin+1,imax
   q = b(i,j)
   a(i,j) = q + t
   t = q
 enddo
enddo
```

Need loop invariants such that:

- $\forall a, b, j, i \ . \ I_j(a, b, jmin)$
- $\forall a, b, j, i \ . \ I_j(a, b, j) \wedge (j > jmax) \rightarrow post(a, b)$
- ...
- $\forall a, b, j, i \ . \ I_i(a, b, j, i) \wedge (i \leq imax) \rightarrow$
  $I_i(a[(i, j) := b(i-1, j) + b(i, j)], b, j, i+1)$

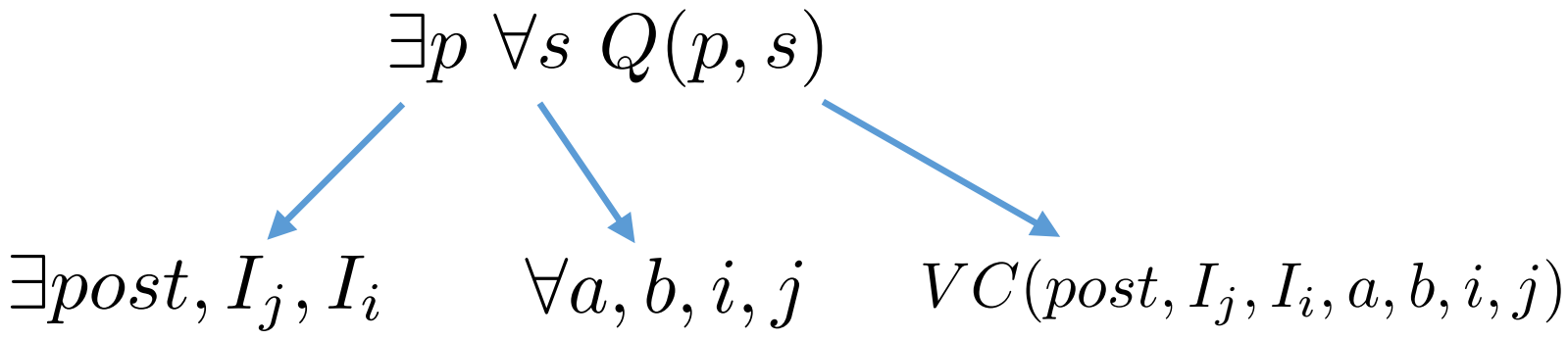
$$\forall i, j \in [imin + 1, imax] \times [jmin, jmax]$$
$$a(i, j) = b(i-1, j) + b(i, j)$$

Adobe

Overview

- Introduction
- Verifying A Summary
- Synthesizing Loop Invariants & Postconditions
- Evaluation
- Summary

Program Synthesis



Adobe

Template Generation

```
 do j=jmin,jmax
  t = b(imin, j)
  do i=imin+1,imax
    q = b(i,j)
    a(i,j) = q + t
    t = q
  enddo
enddo
```

$$jmin, jmax, imin, imax = rand()$$

$$b = \{b_0, b_1, b_2, ...\}$$

$$a = \{..., b_9 + b_{10}, b_{10} + b_{11}, ...\}$$

```
a(i,j) = b(??) + b(??)
```

Adobe

Template Generation

$$\forall \vec{x} \in D.\ out[\vec{x}] = expr(\vec{x})$$

Adobe

$$\exists post, I_j, I_i \quad \forall a, b, i, j \quad VC(post, I_j, I_i, a, b, i, j)$$

$$I_j(a, b, j) \wedge (j > jmax) \rightarrow post(a, b)$$

$$\neg I_j(a, b, j) \vee \neg(j > jmax) \vee post(a, b)$$

$$\forall i, j' \in [imin + 1, imax] \times [jmin, j)$$
$$a(i, j') = b(i - 1, j') + b(i, j')$$

Adobe

∃ ∀ ∃

Adobe

Skolemization

$$\exists x \ \forall y \ \exists z$$

$$\exists x \ \forall y \ f(y)$$

```
if (j<jmin)
  something
else if (j<jmax)
  something_else
else if…
  another_thing
```

Adobe

Partial Skolemization

$$\exists x\ \forall y\ \exists z$$

$$\exists x\ \forall y\ f_p(y) \qquad \exists x\ \forall y\ Q(x, y, p_1) \vee Q(x, y, p_2)$$

Adobe

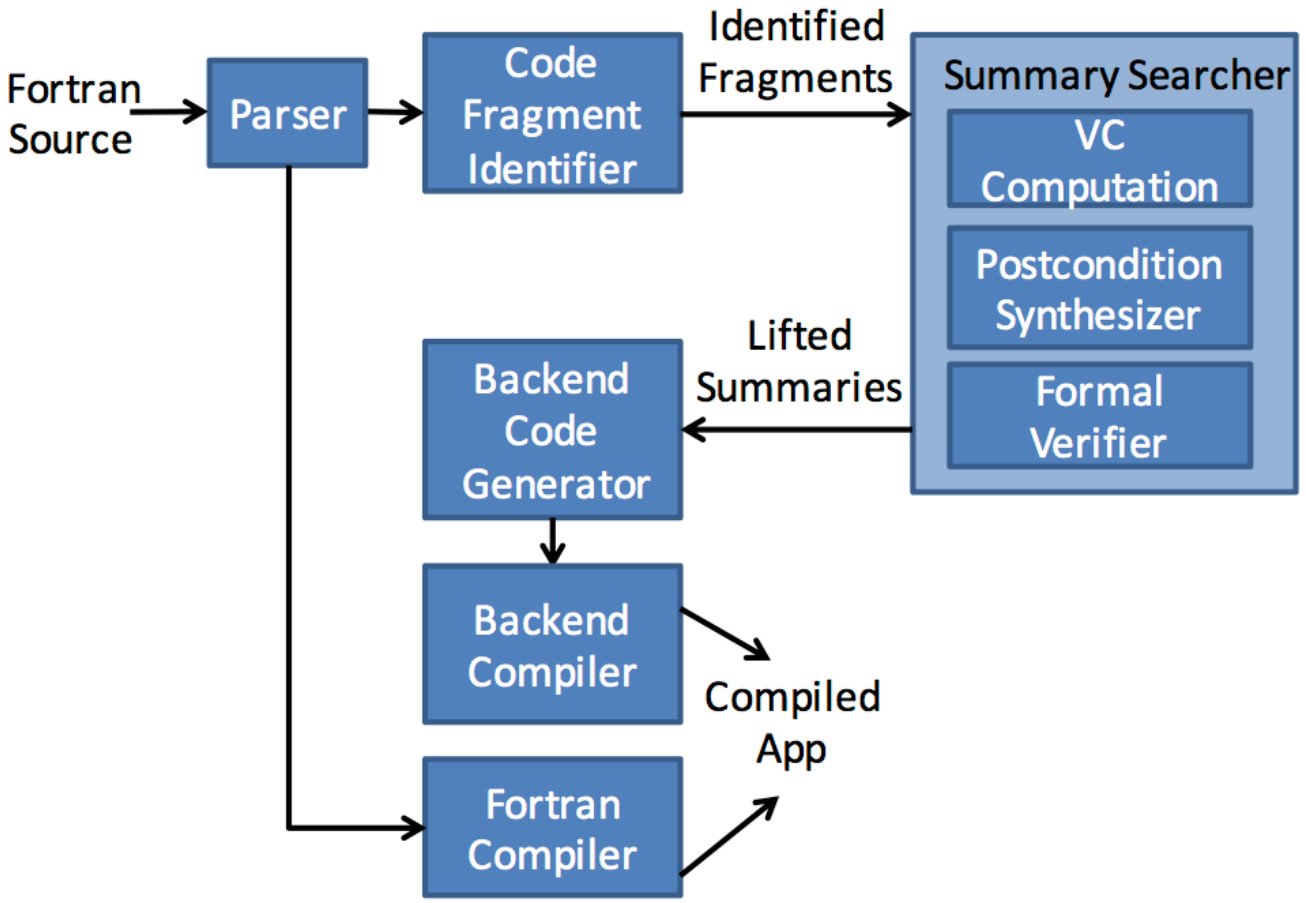
Partial Skolemization

```
boolean Inv_j(ref double[sz] a, ref double[sz] b, int jmin, int jmax,
 int imin, int imax) {

  int[2] _js = {gen(jmin, jmax, imin, imax), gen(jmin, jmax, imin, imax)};

  int[2] _is =  {gen(jmin, jmax, imin, imax), gen(jmin, jmax, imin, imax)};

  for (int ji=0; ji<2; ji++) {
  for (int ii=0; ii<2; ii++) {
    i = _is[ii];
    j = _js[ji];
    if a(i,j) != b(gen_pt(i,j)) + b(gen_pt(i,j)) return false;
  }}
  return true;
}
```

Overview

- Introduction
- Verifying A Summary
- Synthesizing Loop Invariants & Postconditions
- Evaluation
- Summary

STNG: Verified Lifting for Stencils



Adobe

Limitations

- Concentrate on a specific subset of all stencil computations
  - No conditionals or boundary conditions
  - Distinct input and output arrays
  - Incrementing loops only
- Even with careful generation of Sketches, time to solution can be very large

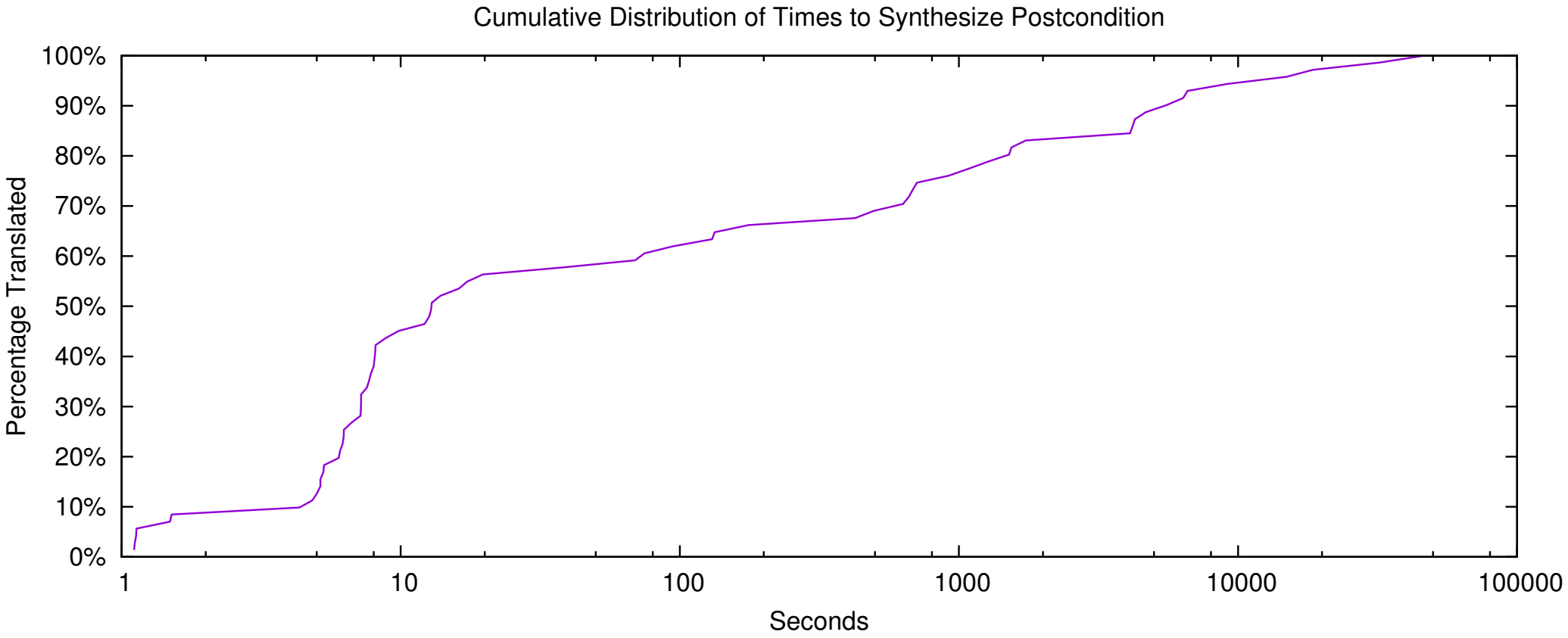
Adobe

Benchmark Codes

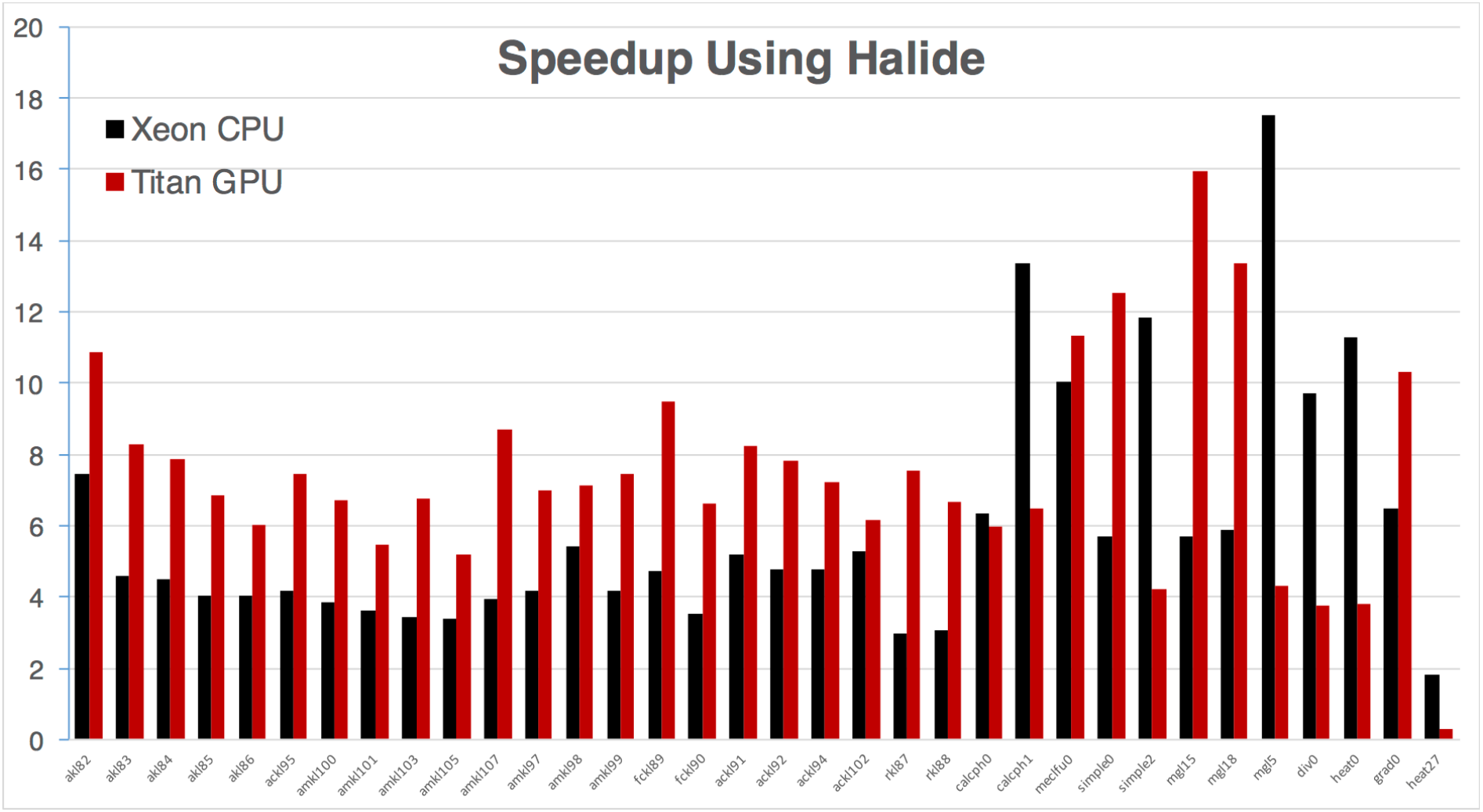
- 4 scientific applications & 2 sets of benchmark codes
- All benchmarks are in (dialects of) Fortran
- Dimensionality from 1D to 6D

	Candidates	Translated	Failed
StencilMark	4	3	1
NAS MG	9	3	5
Cloverleaf	45	40	4
Terra	1	1	0
NFFS-FVM	29	25	1
Challenge	5	5	0
Total	93	77	11

Adobe


Cumulative Distribution of Times to Synthesize Postcondition
Percentage Translated
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
1
10
100
1000
10000
100000
Seconds

Adobe


Speedup Using Halide
Xeon CPU
Titan GPU
20
18
16
14
12
10
8
6
4
2
0
akl82
akl83
akl84
akl85
akl86
ackl95
amkl100
amkl101
amkl103
amkl105
amkl107
amkl97
amkl98
amkl99
fckl89
fckl90
ackl91
ackl92
ackl94
ackl102
rkl87
rkl88
calcph0
calcph1
mecfu0
simple0
simple2
mgl15
mgl18
mgl5
div0
heat0
grad0
heat27

Adobe

Related Work

- Automatic invariant generation
- Compiler techniques (e.g. polyhedral model)
- QBS: Verified lifting of ORM code

Summary

- STNG performs verified lifting on stencil computations by synthesizing postconditions and loop invariants
- Much of the effort is in generating Sketches that limit the search space of possible postconditions/invariants
- Performance results show that the transformed code can leverage high performance DSLs