DaFlEx

(slide credits - M. Nussbaumer)

Array/Scalar Fission

- **LoopToMap could not parallelize following loop automatically**

```
ZQADJ = ...        //some write
... = ZQADJ        //some read

do JK = 1, KLEV
   do JL = 1, KFDIA
      ZQADJ = ZQX(JL, JK, 1) * ZQTM
      t_q(JL,JK) = t_q(JL,JK) + ZQADJ
   enddo
enddo

ZQADJ = ...        //some write
... = ZQADJ        //some read
```

```
ZQADJ_0 = ...      //some write
... = ZQADJ_0      //some read

do JK = 1, KLEV
   do JL = 1, KFDIA
      ZQADJ_1 = ZQX(JL, JK, 1) * ZQTM
      t_q(JL,JK) = t_q(JL,JK) + ZQADJ_1
   enddo
enddo

ZQADJ_2 = ...      //some write
... = ZQADJ_2      //some read
```

- **Needs to determine**
  - if loop reads from definition of ZQADJ from outside the loop
  - if definition of ZQADJ is read from outside of the loop
- **Scalar Fission makes new variable for each assignment to make ZQADJ loop-local → Used to parallelize almost 30 loops in cloudsc**

→Generalize this concept to arrays

Approximate writes

- **Need to determine if array is definitiely overwritten**

```
tmp = np.zeros((N,))
for i in range(N):
    tmp[i] = 1
```

write set for tmp: [0:N:1] → tmp is definitely overwritten after loop

- **Currently DaCe can over-approximate accesses**

```
tmp = np.zeros((N,))
for i in range(0,N,3):
    tmp[i] = 1
```

write set for tmp: [0:N:1]

- **New pass that under-approximates accesses**

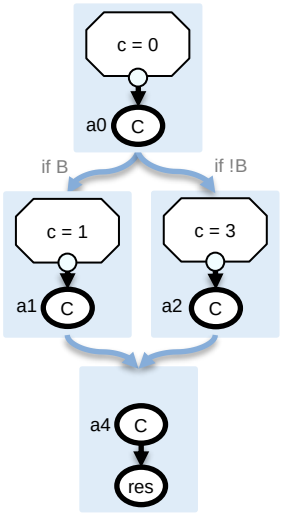
```
tmp = np.zeros((N,))
for i in range(N):
    if b:
        tmp[i] = 1
```

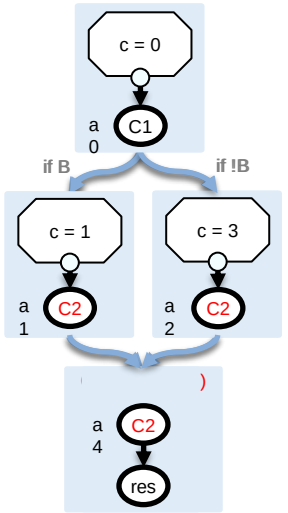
over-approximated write set for tmp: [0:N:1]

under-approximated write set for tmp: []

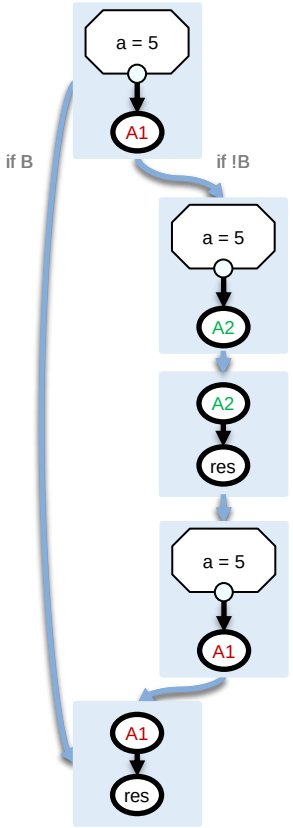
- **Can be used later for other transformations**

More Fissioning

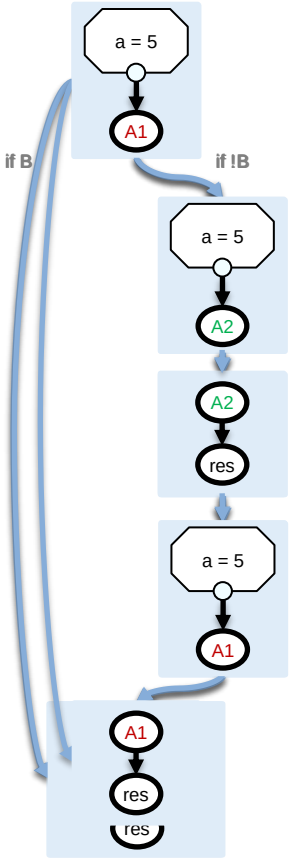




Yet more Fisioning



SSA



Results

- Without Array Fission: No speedup
- Parallel speedups of “Array Fission“-versions are similar to each other:
  - *>2.7x on 4 cores*
  - *>4.2x on 8 cores*
- The speedups come close to the manually parallelized version of CLOUDSC2