Bee Input Requirements

Christophe Alias

Bee (current) Input, I

- Schedules are specified by means of `#pragma`s
- Arbitrary multi-dimensional affine schedules:

```
#pragma contract a

for(i=0; i<N; i++)
#pragma schedule[i][0]
  a[i] = 0;

for(i=0; i<N; i++)
#pragma schedule[i][1]
  output[i] = a[i] + 1;
```

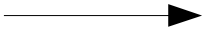
```
for(i=0; i<N; i++)
  {
    a = 0;
    output[i] = a + 1;
  }
```

Loop fusion

Bee (current) Input, II

- Schedules are specified by means of **#pragma**s
- Arbitrary multi-dimensional affine schedules:

```
for(i=1; i<N; i++)
  {
#pragma schedule[2*i]
    a[i] = b[i-1]*2;
#pragma schedule[2*i+1]
    b[i] = a[i] + 3;
#pragma schedule[2*i+2]
    c[i] = b[i] + c[i-1];
#pragma schedule[2*i+3]
    d[i] = c[i] + 2;
  }
```

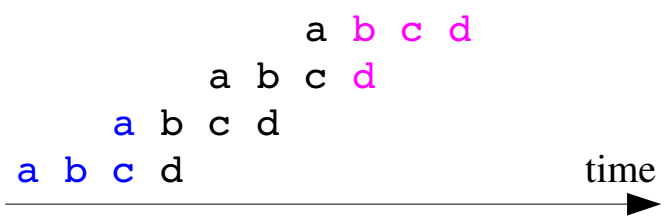


```
a = b * 2;
b = a + 3;
c = b + c;
a = b * 2;
for(i = 5; i <= N-2; i++)
  {
    if(i%2 == 1)
      {
        d[(i - 3) / 2] = c + 2;
        b = a + 3;
      }
    if(i%2 == 0)
      {
        c = b + c;
        a = b * 2;
      }
  }
d[N-2] = c + 2;
b = a + 3;
c = b + c;
d[N-1] = c + 2;
```

Software pipelining, II=2

Live-in = b[0] and c[0]

Live-out = d



Bee Input Requirements

- Arrays to contract
- For each assignment:
  - Iteration domain
    => Obtained with a simple top-down traversal
  - (multi-dimensional affine) schedule
    => default = sequential schedule
    = iteration vector interleaved with constants indicating the position in the loop body
  - List of arrays read + array written
- Matrix representation of indexes and domains

```
for(i=0; i<N; i++)
  {
    x[i] = b[i] / a[i][i];
    for(j=i+1; j<N; j++)
      b[j] = b[j] – a[j][i] * x[i];
  }
```