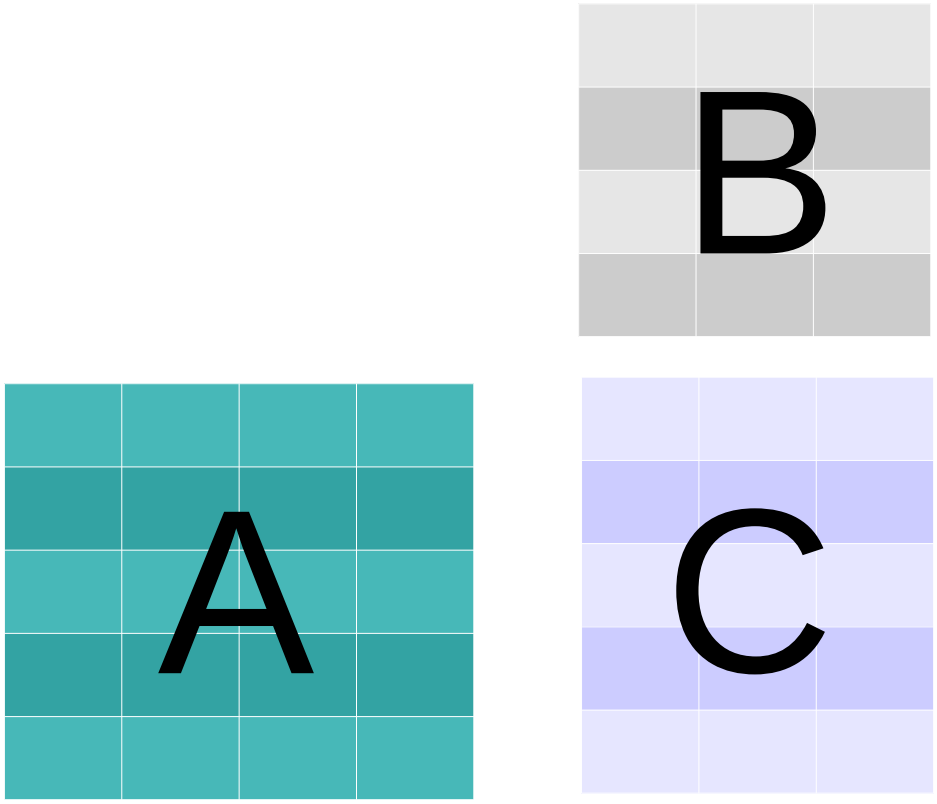
PaRSEC PTG DSL
Local Iterators

Running Example: GEMM



ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY OF TENNESSEE KNOXVILLE

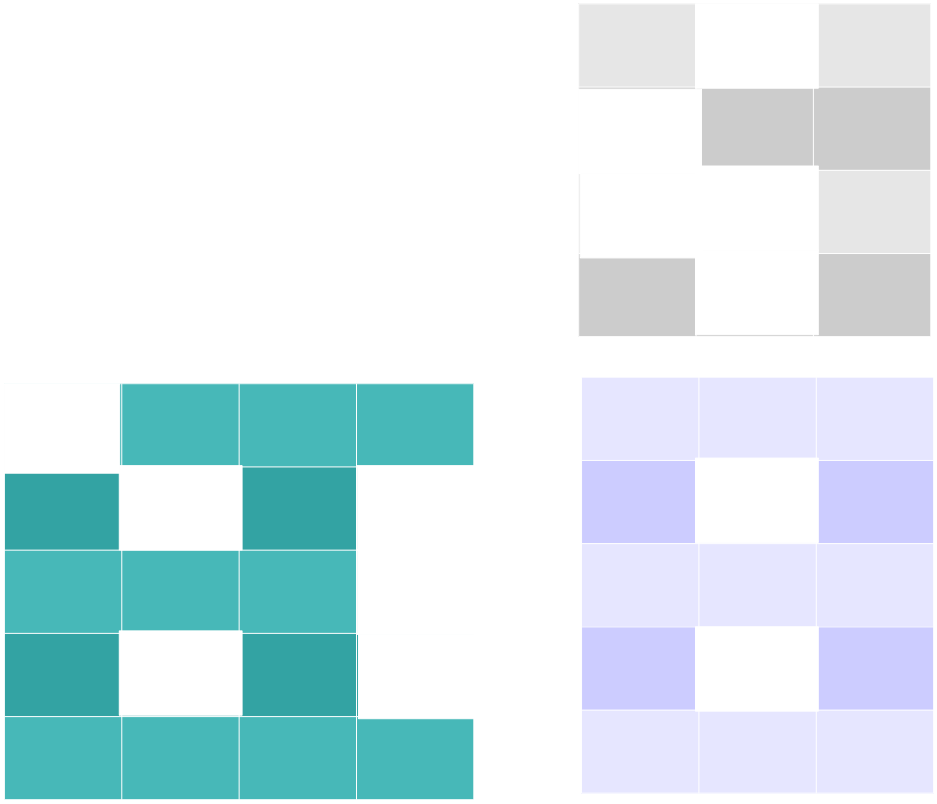
Parameterized Task Graph

```
GEMM(m, n, k)
  m = 0 .. C.mt-1
  n = 0 .. C.nt-1
  k = 0 .. A.nt-1

READ A <- A READ_A(m, k)
READ B <- B READ_B(k, n)
RW   C <- k == 0  ? C READ_C(m, n)
                  : C GEMM(m, n, k-1)
       -> k == A.nt-1 ? C WRITE_C(m, n)
                      : C GEMM(m, n, k+1)

BODY
  blas_gemm(A, B, C);
END
```

```
READ_A(m, k)
 m = 0 .. A.mt-1
 k = 0 .. A.nt-1

 :A(m, k)

READ A <- A(m, k)
  -> A GEMM(m, 0..A.nt-1, k)

BODY
  /* nothing */
END
```

INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY OF TENNESSEE KNOXVILLE

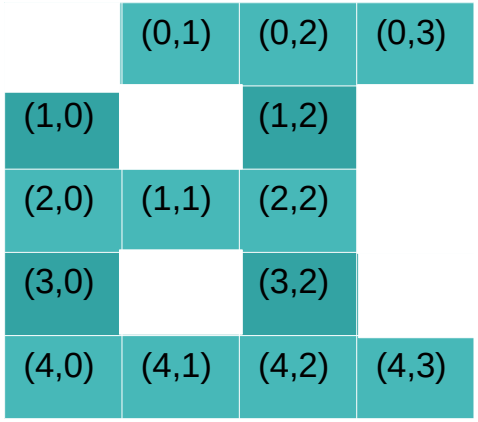
Example: Block Sparse GEMM



ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY OF TENNESSEE KNOXVILLE

PTG: Local iterators

```
READ_A(m, k)
 m = [mi = 0 .. %{ return nb_nnz_rows(A)-1;%}]
        %{ return nnz_row(A, mi);%}
 k = [ki = 0 .. %{ return nb_nnz_tile(A, m)-1; %}]
        %{ return nnz_col_tile(A, m, ki); %}


 :A(m, k)

READ A <- A(m, k)
       -> A GEMM(m, ???, k)

BODY
  /* nothing */
END
```



nb_nnz_rows(A) = 4; nnz_row(A, i) = i
nb_nnz_tile(A, 0) = 3;
nnz_col_tile(A, 0, 0) = 1; nnz_col_tile(A, 0, 1) = 2;
nnz_col_tile(A, 0, 2) = 3;
nb_nnz_tile(A, 1) = 2;
nnz_col_tile(A, 1, 0) = 0; nnz_col_tile(A, 1, 1) = 2;
nb_nnz_tile(A, 2) = 3;
nnz_col_tile(A, 0, 0) = 0; nnz_col_tile(A, 0, 1) = 1;
nnz_col_tile(A, 0, 2) = 2;
...

ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY OF TENNESSEE KNOXVILLE

PTG: Local iterators

```
READ_A(m, k)
 m = [mi = 0 .. %{ return nb_nnz_rows(A)-1;%}]
        %{ return nnz_row(A, mi);%}
 k = [ki = 0 .. %{ return nb_nnz_tile(A, m)-1; %}]
        %{ return nnz_col_tile(A, m, ki); %}


 :A(m, k)

READ A <- A(m, k)
       -> [ni = 0 .. %{ return nb_gemm_n(m, k); %}]
                  A GEMM(m, %{ return gemm_n(m, k, ni); %}, n)

BODY
  /* nothing */
END
```

INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY OF TENNESSEE KNOXVILLE

PTG: local iterators

```
GEMM(m, n, k)
  m = [mi = 0 .. %{ return nb_nnz_rows(C)-1; %}] %{ return nnz_row(C, mi) %}
  n = [ni = 0 .. %{ return nb_nnz_cols(C, m)-1; %}]
                                          %{ return nnz_tile(C, m, ni); %}
  k = [ki = 0 .. %{ return nb_gemm_k(m, n)-1; %} %{ return gemm_k(m, n, ki); %}
  ki = %{ return gemm_index_k(m, n, k); %}
  firstk = %{ return gemm_k(m, n, 0); %}
  lastk =  %{ return gemm_k(m, n, nb_gemm_k(m, n)-1); %}
  prevk =  %{ return gemm_k(m, n, ki-1); %}
  nextk =  %{ return gemm_k(m, n, ki+1); %}

READ A <- A READ_A(m, k)
READ B <- B READ_B(k, n)
RW   C <- k == firstk  ? C READ_C(m, n)
                       : C GEMM(m, n, prevk)
       -> k == lastk ? C WRITE_C(m, n)
                     : C GEMM(m, n, nextk)

BODY
  blas_gemm(A, B, C);
END
```

ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY OF TENNESSEE KNOXVILLE

PTG: local iterators

```
TASK(p1, p2)
  p1 = [p1i = CST1 .. CST2,
        p2i = %{ return f1(p1i); %} .. %{ return f2(p1i); %}]
              %{ return f3(p1i, p2i); %}
  p2 = [p2i = %{ return f4(p1); %} .. %{ return f5(p1); %}]
              %{ return f6(p1, p2i); %}

READ A <- [ki = A1 .. B3] test(p1, p2, ki) ?
             [li = A2 .. B3] X TASK(f1(p1, p2, ki, li), f2(p1, p2, ki, li)) :
             [ji = A3 .. B3] Y TASK(f1(p1, p2, ki, ji), f2(p1, p2, ki, li))
       -> A TASK( [mi = A4 .. B4] %{return f(p1, p2, mi); %},
                  [mi = A5 .. B5] %{return f(p1, p2, mi); %})
```

Local iterators can be used:

- in any execution space / range definition (e.g. , in front of an expression.
  - Scope of local named variables is the expression
  - Space defined is the set of values for the expression, for all possible values of the named variables
- In front of a guarded call
  - Scope is all tests and calls after the local iterator; this corresponds to doing a sparse broadcast
- In front of a call
  - Scope is the call only; this corresponds to doing a sparse broadcast

ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY OF TENNESSEE KNOXVILLE

ICL
INNOVATIVE
COMPUTING LABORATORY
THE UNIVERSITY OF
TENNESSEE
KNOXVILLE