More Synchronization II

`ordered` Directive

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Some Loops Are “Almost” Parallel

```
B[0] = 1.0;
for (i=1; i< N; i++) {
      A[i] = f(C[i]); //assume f is expensive
      B[i] = B[i-1] +i*A[i];
      D[i] = B[i]*B[i];
}
```

The `ordered` Clause

- Original concept of "DoAcross" loops
- Can be used inside a parallel loop

```
#pragma omp ordered
 Code-block
```

- Makes the block of code wait for previous iteration to finish its *ordered* block
- How will you fix the code from the previous page?

```
B[0] = 1.0;
for (i=1; i< N; i++) {
   A[i] = f(C[i]); // assume f is expensive
   B[i] = B[i-1] +i*A[i];
   D[i] = B[i]*B[i];
}
```

Using **ordered** Directive and **ordered** clause

```
B[0] = 1.0;
#pragma omp parallel for ordered
for (i=1; i< N; i++) {
  A[i] = f(C[i]); //assume f is expensive
#pragma omp ordered
  { B[i] = B[i-1] +i*A[i]; }
  D[i] = B[i]*B[i];
}
```

Computation of f and A[i] happen in parallel

- Note that all iterations use old value of B[i] for computing B
- Old: from before the for loop
- So, it's ok to do those in parallel

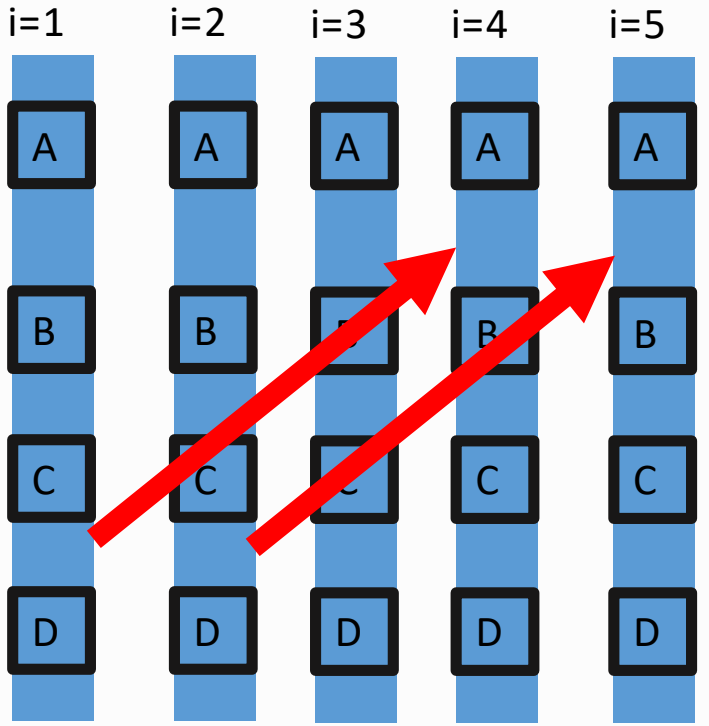
Computation of B[i]s are serialized

Computation of D[i]s are parallel
Especially useful if it were expensive:
e.g., D[i] = g(B[i]);

`depends` clause with `ordered` directive

- Allows you to specify a dependence in a more general and precise way
  - E.g. Execute the following statement in the current iteration i after another specific statement in iteration i-3

```
#pragma omp parallel for ordered(1)
for (i=1; i< N; i++) {
  codeBlock_A
  #pragma omp ordered depend(sink:i-3)
  {codeBlock_B}
  codeBlock_C
  #pragma omp ordered depend(source)
  codeBlock_D
}
```



depends clause with ordered directive: 2

- Dependences across deeper loop nests can be specified too.

```
#pragma omp parallel
#pragma omp for ordered(2)
for (i=1; i<N; i++) {
   for (j=1; j<M; j++)  {
       A[i][j] = foo(i, j);
       #pragma omp ordered depend(sink: i-1,j) depend(sink: i,j-1)
       B[i][j] = bar(A[i][j], B[i-1][j], B[i][j-1]);
       #pragma omp ordered depend(source)
       C[i][j] = baz(B[i][j]);
   }
}
```