CS 220: Introduction to Parallel Computing

Distributed Sorting

Lecture 21

Parallelizing Odd-Even Sort

- We can implement our parallel sort using:

  MPI_Send

  MPI_Recv

- These are also available as a single function:

  MPI_Sendrecv

- **Another hint**: to reduce the complexity of your code, you can write a function to swap values between ranks
  - int sort(int *val_p, neighbor)

Odd-Even Transposition Sort

- Consists of two phases: **odd** and **even**
- In the even phases, each odd-subscripted element is compared with its “left” neighbor
  - If they’re out of order, they’re swapped
- In the odd phases, each odd-subscripted element is compared with its “right” neighbor
  - If they’re out of order, they’re swapped
- Repeat until sorted

An Example

- Suppose the list is:

```
[5,   9,   4,   3]
 0    1    2    3  /* (subscripts) */
```

- Even phase:
  - Compare (5, 9) and (4, 3)
  - 4 and 3 are out of order. Swap them!

An Example

- The list:

```
[5,    9,    3,    4]
 0     1     2     3   /* (subscripts) */
```

- Odd phase:
  - Compare (9, 3)
- The list:

```
[5,    3,    9,    4]
 0     1     2     3   /* (subscripts) */
```

An Example

- The list:

```
[5,   3,   9,   4]
 0    1    2    3  /* (subscripts) */
```

- Even phase:
  - Compare (5, 3) and (9, 4)
  - Both pairs are out of order. Swap them!
- The list:

```
[3,   5,   4,   9]
 0    1    2    3  /* (subscripts) */
```

An Example

- The list:

```
[3,   5,   4,   9]
 0    1    2    3  /* (subscripts) */
```

- Odd phase:
  - Compare (5, 4). Swap!

-

```
[3,   4,   5,   9]
 0    1    2    3  /* (subscripts) */
```