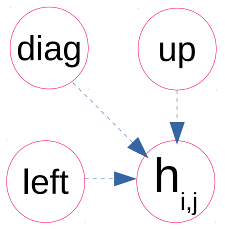
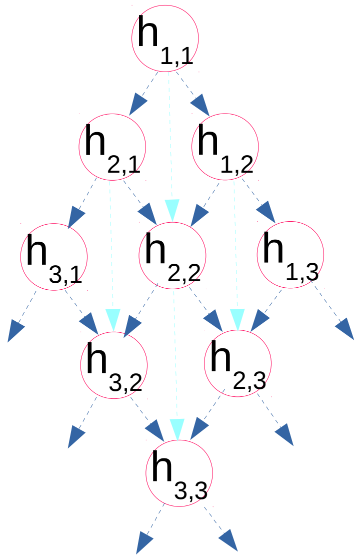
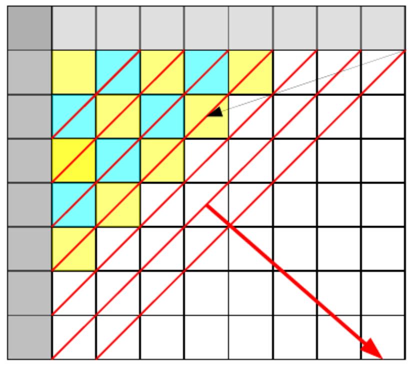
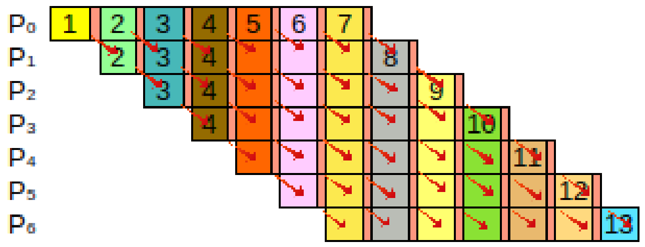
Smith-Waterman Parallelization – Q & A









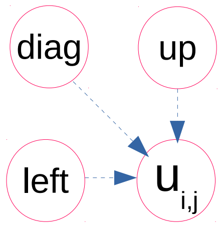
HPC

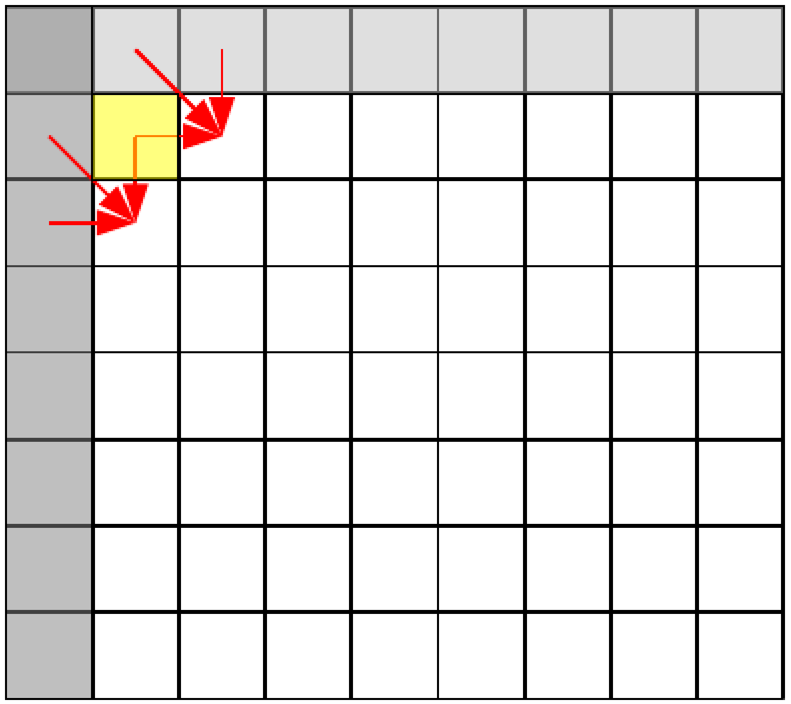
Support Material

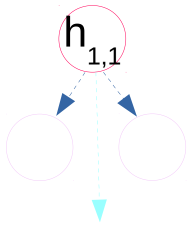
Computer Architecture Department

Departament d'Arquitectura de Computadors
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Smith-Waterman: Dependences

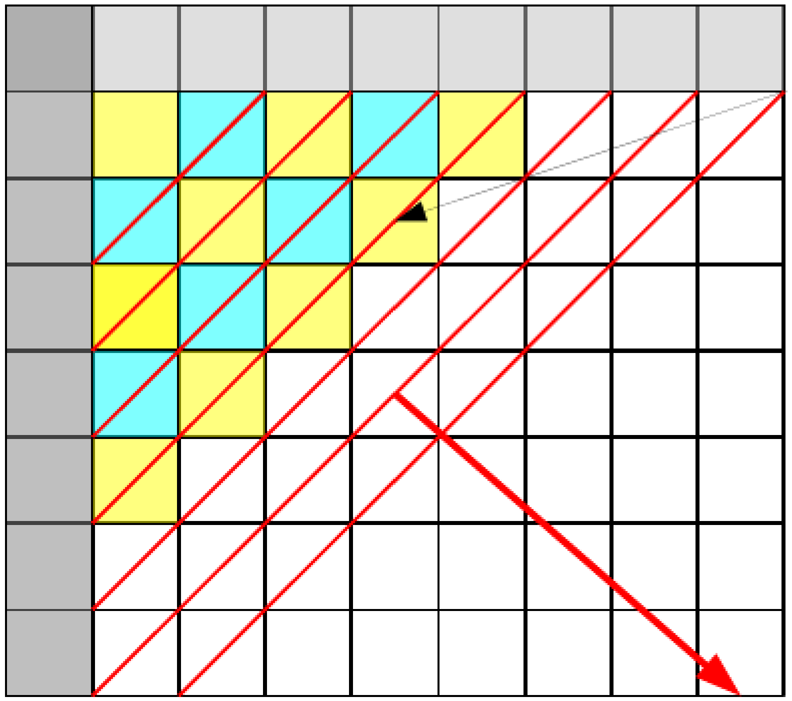




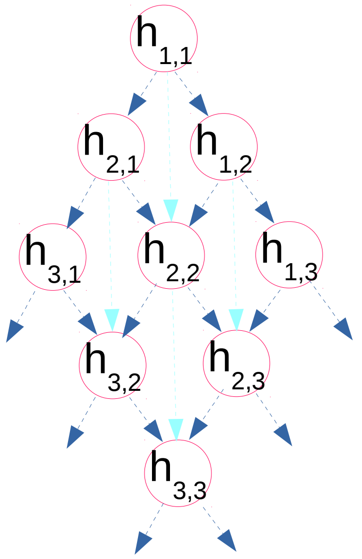


Departament d'Arquitectura de Computadors
UNIVERSITAT POLITÈCNICA DE CATALUNYA

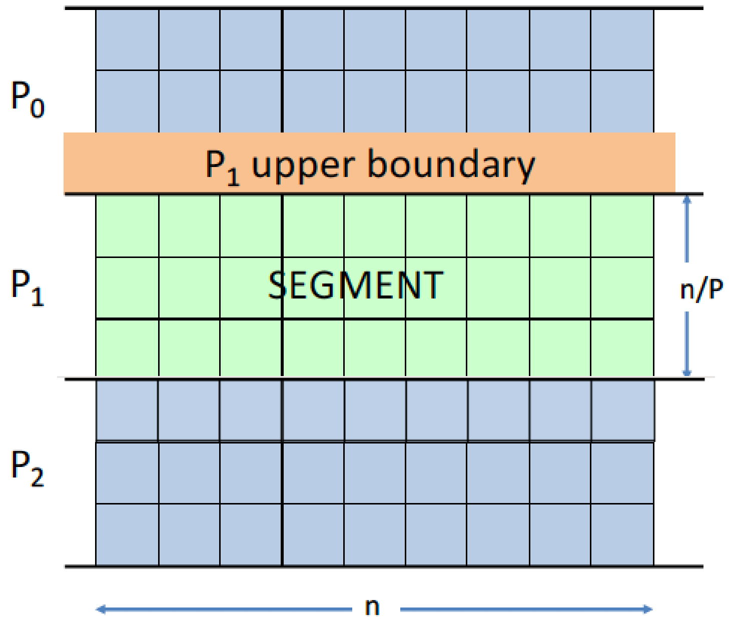
Smith-Waterman: Wavefront Parallelism



The number of **cells that can be computed in parallel** proceeds like a **wavefront** through the matrix

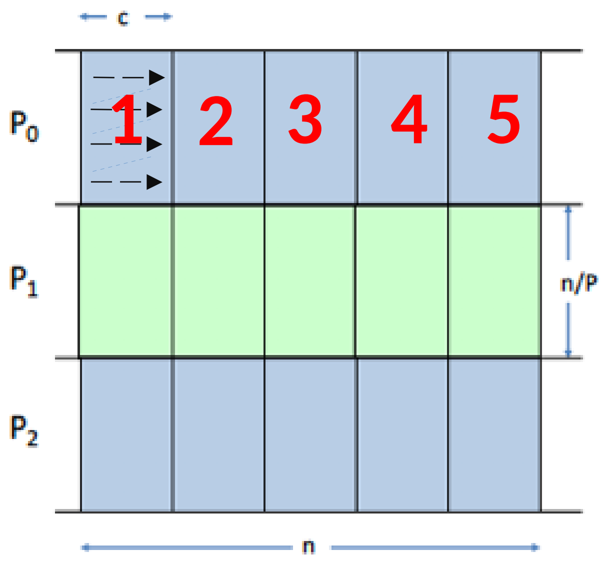


Smith-Waterman: Task Definition by Blocking



- Row decomposition, each processor with n^2/P elements of matrix `h` (segment)
- Upper boundaries with n elements
- No lower boundary
- **Task definition**:
  a task computes a block of `n/P x c` consecutive iterations of `i` and `j` loops

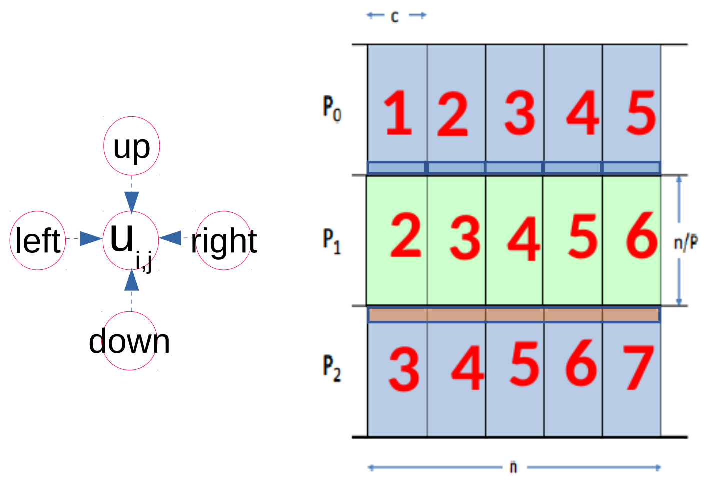
Elements are traversed within each block:

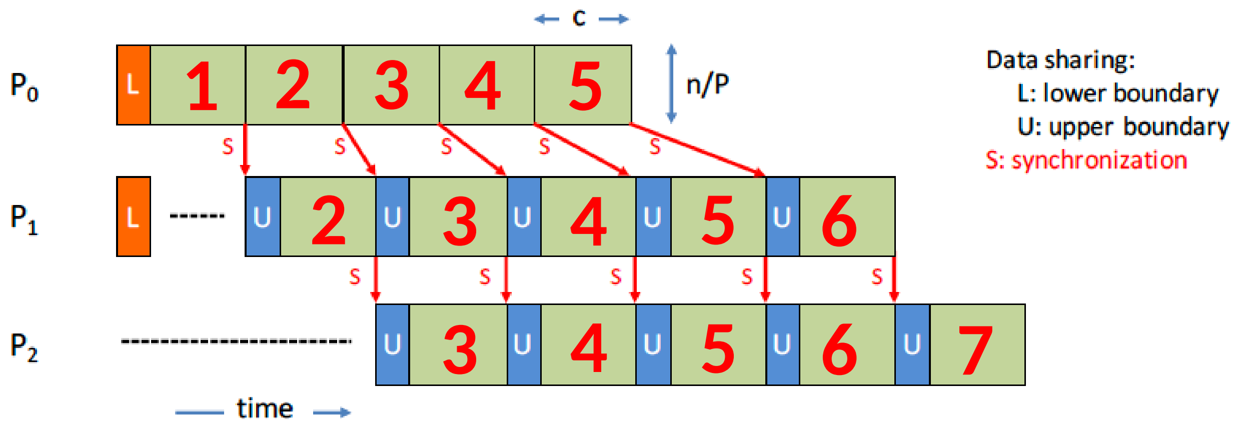


Each processor executes several tasks sequentially

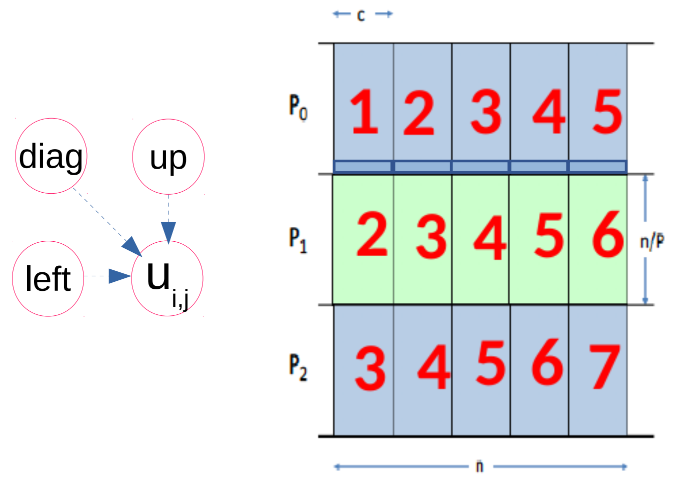
Gauss-Seidel vs Smith-Waterman: Parallel Execution with Blocking

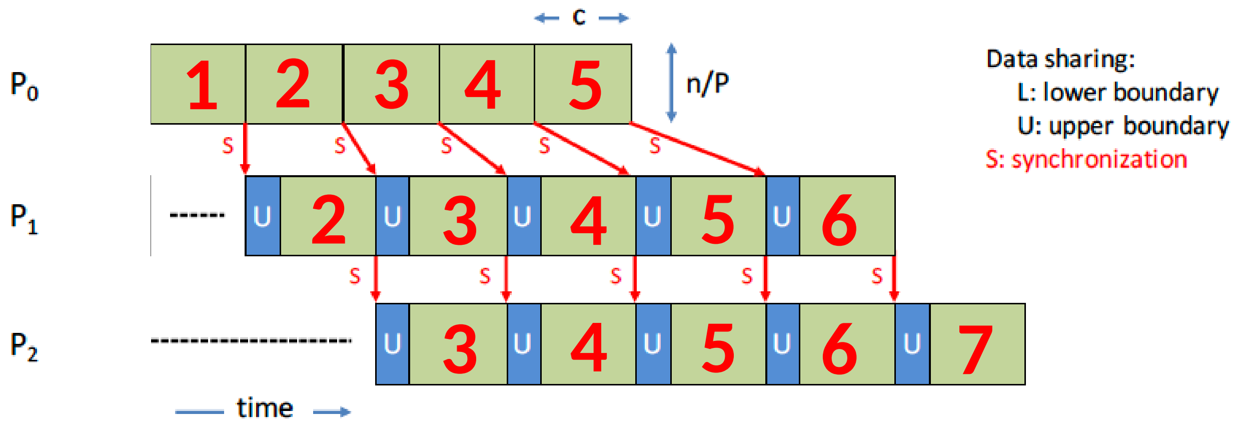
Gauss-Seidel

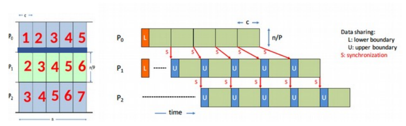




Smith-Waterman







Smith-Waterman: Questions & Answers

- Q: the compiler said that COMM_WORLD was undeclared.

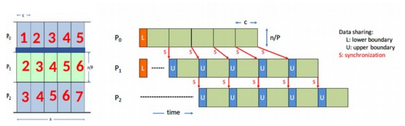
  A: The correct symbol is **MPI**_COMM_WORLD
  If **MPI**_COMM_WORLD or other MPI symbols are not defined it means you forgot mpi.h:
  #include <mpi.h>

- Q: Compiler complains min is undefined

  A: #define min(a,b) (((a)<(b)) ? (a) : (b))

- Q: *Set a process to be the special Master process.*
  Remember that process 0 always exist.
  Do I have to set process 0 as the master?

  A: Yes, set processor 0 as the master.



Smith-Waterman: Questions & Answers

- Q: *Have the master be the only one which reads the input files and then broadcast or distribute the information to the worker processes*.
  How?

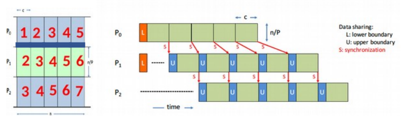
  A: ...
  Make sure all processes have variables declared
  and allocate memory once they know how much they need!

```
Declaration of Variables /* All processes */
...
if (!rank) { /* Only master process */
    read_all_inputs();
}
...
/* All processes */
broadcast dimensions from master to all workers

pointer_variable = malloc(...);
```

  Make sure all processes allocate memory using the correct dimension for each sequence or matrix sim.

  broadcast / scatter the input sequences and matrix sim from master to workers.

  In any communication: **Use the correct data type!** `short` *vs* `int`



Smith-Waterman: Questions & Answers

- Q: What do we have to use as recvbuff when doing Scatter for s1 and h in the worker processors?

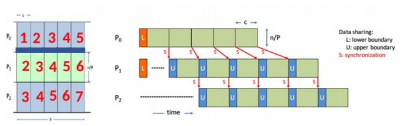
  A: s1l (l for local) and ... [what about h ???].

  However, (see item 5 in the statement of the lab) do we need to send h initially?
  No! It'll be overwritten! → No need to send initial data since each process can initialize with 0's.

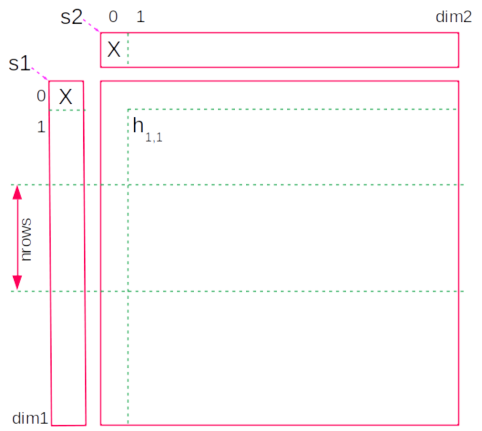
  Note: in the Matrix Multiplication example in the Distributed Memory Systems slides where variable A (which provides the initial address of matrix A) is used both as send buffer (parameter 1) and receive buffer (parameter 4) can be taken for the overall idea, but in general `MPI_Scatter` needs a different buffer for receiving:

  ```
  sizeToBeSent = n * n/mpiSize;

  MPI_Scatter(A, sizeToBeSent, MPI_DOUBLE, Alocal, sizeToBeSent, MPI_DOUBLE, 0,
              MPI_COMM_WORLD)
  ```

  (*) We should use `MPI_Scatterv` to handle any sequence length and number of processe, but you can start with a sequence length evenly divisible by the number of threads and, therefore, use a fixed size for all. Once everything works for the simple case, improve your code to make it general.



Smith-Waterman: Questions & Answers



- Q: Anything else to take into account when sending the sequences to the worker processors?

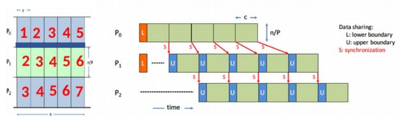
  A: Notice that the 1st position in each array which stores the sequence is not used.

```
nrows = getRowCount(dim1, rank, nprocs);

MPI_Scatter( s1+1,       nrows, MPI_SHORT,
             s1local+1, nrows, MPI_SHORT, 0, MPI_COMM_WORLD)
```

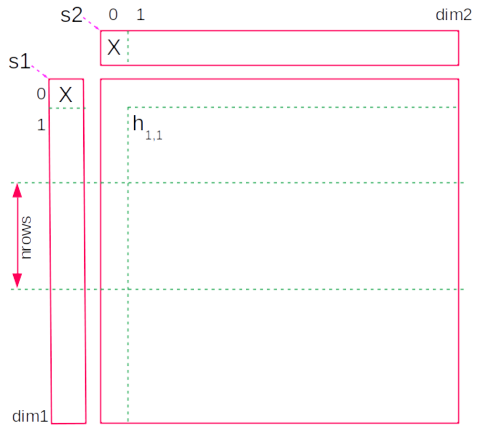
Also, note that s1 and s2 contain elements of datatype `short`.

(*) We should use `MPI_Scatterv` to handle any sequence length and number of processes. However, you can start with a sequence length evenly divisible by the number of threads and, therefore, use a fixed size for all. Once everything works for the simple case, improve your code to make it general.

```
   s1+1: use pointer arithmetic to point to the second element in the array

         since the 1st position is not used.

             ___________________      ____
            |       |      |                 |
            |s1[0]  |s1[1]|        ....      |
            |______|_____|_____       ____|
             ▲       ▲

            s1      s1+1
```



Smith-Waterman: Questions & Answers

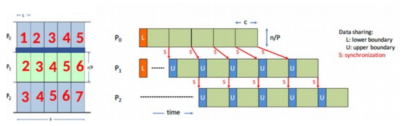


Q: How can we send sequence 2 to the worker processors?

Sequence 2 in array s2 needs to be broadcast.

2 possibilities:

```
MPI_Bcast(s2+1, dim2, MPI_SHORT, 0, MPI_COMM_WORLD);

   or

MPI_Bcast(s2, dim2+1, MPI_SHORT, 0, MPI_COMM_WORLD);
```



Smith-Waterman: Questions & Answers

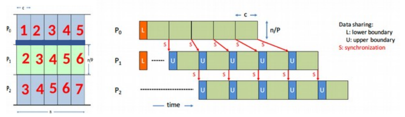
- Q: What else can we learn from that example?

  A: Understand well the Matrix Multiplication example in the Distributed Memory Systems slides (also 2nd problem in June 2018 Final Exam).

  - Original **i loop** traversing rows **changed**
    from `for (int i=0; i<MATSIZE; i++)`
    to → `for (i=0; i<nlocal; i++)`
    where:
    ```
    n_local = getRowCount(n, mpiRank, mpiSize);
    ```

  - Only allocate the space needed:
    ```
    n_local = getRowCount(n, mpiRank, mpiSize);
    n_sq    = n * n;
    n_sq2 = n * n_local;
    A = (double *) malloc(sizeof(double) * (mpiRank ? n_sq2 : n_sq));
    ```

  Make sure all processes have variables declared and allocate memory!



Smith-Waterman: Questions & Answers

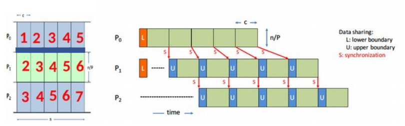
- Q: `getRowCount(i, rank, size);` does not work. Why?

  A:
  ```
  int getRowCount(int rowsTotal, int mpiRank, int mpiSize)
  { ...;
    return(...);
  }
  nrows = getRowCount(...); /* We need to retrieve and store the return value. */
  ```

- Q: `MPI_Bcast(&s2, ... )` does not work.

  A:
  ```
  short *s1, *s2;  /* Pointers to arrays ... */
  s2 = (short *) malloc (...);
       /* s2 stores the initial address of the array */
  ...
  MPI_Bcast(s2, ...
  ```

  Note that `s2` is a pointer and keeps the initial address where the 2nd sequence is stored in memory.
  Thus,
  we don’t have to write `&s2` (to get the address of s2)
  but `s2` (to get the content of s2).



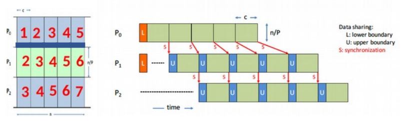
Smith-Waterman: Questions & Answers

- Q: we want to ask you if there is a way to send several data by only one MPI_Bcast to send the sequence and the dim at the same time.

  A: This would be possible if data is stored in consecutive memory positions and have the same data type. Or you pack several values in a more complex data structure.
  However: Advice for this assignment, just do one MPI call for each piece of data you want to communicate.
- Q: And the sequences: is it correct to send them to all the processes or would be better to send only the corresponding part to each one?

  A: Ideally, just send to processes what they need. And allocate in each process only the required space.
- Q: is there a significant difference doing first the mpi initialization then the reading of the inputs (and broadcast) or it is the same that read first all with one process and then create the mpi init? Just considering if the data will be shared as the score matrix.

  A: MPI_Init has to be called before any MPI primitive is used.
  All processes execute the same program. Work can be done in several ways. But, as an exercise, it’s good you try to have only one of them, the master, do the initializations and then share the information with the rest of the processes via messages.
  But make sure all processes have variables declared and allocate memory!

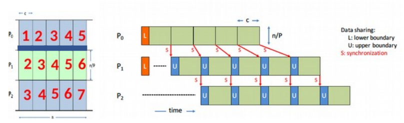


Smith-Waterman: Questions & Answers

- Q: Something else to remember?

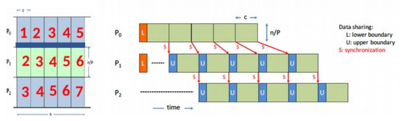
- A:
  - All processes have to issue the MPI collective communication primitive in the same way, not only the master.
  - **You have to call `malloc` in each process**;
    and only after the process has received the information about the size.
  - Broadcast dimensions first;
    then, have the processes allocate memory in view of these dimensions;
    and then send the contents of the two sequences and matrix sim.
  - When you call printf, remember to end the string with a `\n` so that the buffer is flushed to the standard output:

```
printf(" Process %d: nrows=%d\n", rank, nrows);
```



Smith-Waterman: Learning from the errors

```
Declaration of Variables /* All processes */
...
if (!rank) { /* Only master process */
    ...
    double stamp;
    read_all_inputs();
    ...
    MPI_Bcast(...);
}
...
/* All processes */
```



Smith-Waterman: Learning from the errors

Only the master knows dim2!

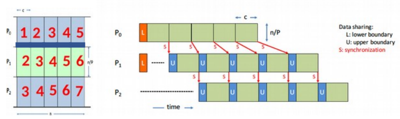
```
...
MPI_Bcast(s2, dim2+1, MPI_SHORT, 0, MPI_COMM_WORLD);
MPI_Bcast(&dim2, 1, MPI_INT, 0, MPI_COMM_WORLD);
...
```

Broadcast `dim2` to all processes!
and then …
Allocate space
and then …
Broadcast `s2`

...

-
  ```
  MPI_Bcast(&dim2, 1, MPI_INT, 0, MPI_COMM_WORLD);
  s2 = (short *) malloc (sizeof (short) * (dim2 + 1)))
  MPI_Bcast(s2, dim2+1, MPI_SHORT, 0, MPI_COMM_WORLD);
  ```

...



Smith-Waterman: Learning from the errors

```
for (int jj = 1; jj <= dim2; jj += BS)
  {
   if (rank != 0)
     MPI_Recv(&h[0][[1+jj*BS], BS, MPI_DOUBLE, rank-1, 0, MPI_COMM_WORLD);
   for (i = 1; i <= nrows; i++)
    { for (j = jj; j < min(jj+BS,dim2+1); j++)
      { ...
        if (max > Max)
          { Max = max;   xMax = i;   yMax = j; }
        if(rank != nprocs-1){
          MPI_Send(&h[i][1+jj*BS], BS, MPI_DOUBLE, rank+1, 0, MPI_COMM_WORLD);
        }
      }
   }  /* End of i loop */
  } /* End of jj loop */
```

jj already has the index of the first column in the current block!

Matrix h does not contain values of type double but int.

Wrong for the last block if dim2%BS != 0

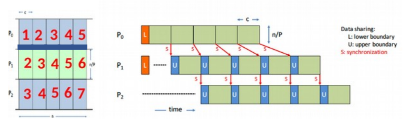
In the general case, the last communication can require a value smaller than BS to avoid overwritting other matrix positions.

min(BS,dim2-jj+1)

Only send the last row in the block!

→ MPI_Send(&h[nrows][jj], ...

Departament d'Arquitectura de Computadors
UNIVERSITAT POLITÈCNICA DE CATALUNYA



Smith-Waterman: Questions & Answers

- Q: Any additional advice?

  A: Test, test and test!

  Do simple unitary tests checking every change in order to fully understand everything and in order to make sure that everything is correct.
  When things work separately then integrate them into the whole code and test the whole code.

- Q: Any additional advice?

  A: Read the whole statement of the laboratory before doing each part to have a general idea. Inspect other examples in the tutorial, the exams and elsewhere.

- Q: Any additional advice?

  A: Check if we allocated all vectors and matrices in both the master and the workers.

  Check if the data types are correct.

  Check if the dimensions are correct.

- Q: Any additional advice?

  A: Explain your code to others. Explaining it you can find your errors.