CS 5379 Project 1: Calculating the sum of rows in a matrix.

The technological idea of this project is to use MPI (Message Passing Interface) to significantly overlap communication and computation in a parallel matrix row sum computation application. The initial code provided uses two processes to calculate row sums for a 100x100 matrix. Process pid 0 generates data for rows 0-49 and calculates row sums for rows 50-99, while Process pid 1 receives input for rows 0-49 and calculates row sums for rows 0-49. Main task is to efficiently overlap computation and transmit data between the two processes and improve overall performance.

Blocking communication:

MPI_Send(buf, count, datatype, dest, tag, comm)
MPI_Recv(buf, count, datatype, source, tag, comm, status)

Non-Blocking communication:

MPI_Isend(buf, count, datatype, dest, tag, comm, request)
MPI_Irecv (buf, count, datatype, source, tag, comm, request)
MPI_Wait(request, status)

Process 0 is responsible for generating data for rows 0-49 of the matrix and initiating communication with Process 1. It generates data for rows 0-49 using the **generate_data** function. To overlap communication with computation, Process 0 initiates a non-blocking send operation (**MPI_Isend**) to send data for rows 50-99 to Process 1 while it computes row sums for rows 50-99 in parallel.
Process 1 receives data for rows 0-49 from Process 0 using a non-blocking receive operation (**MPI_Irecv**). While waiting for data to arrive, Process 1 computes row sums for rows 0-49 in parallel, effectively overlapping computation with communication. Once data for rows 0-49 arrives, Process 1 completes the receive operation and sends back the computed row sums for rows 0-49 to Process 0 using **MPI_Send**.
After receiving data for rows 50-99 from Process 1 and completing the computation for rows 50-99, Process 0 prints the results of row sums for all 100 rows.

The usage of non-blocking MPI communication procedures (MPI_Isend and MPI_Irecv) is essential to achieving overlap between communication and processing. These non-blocking calls minimize idle time spent by individual processes and increase overall performance by allowing computation to continue while data is being sent.

In conclusion, the technical idea efficiently overlaps communication and computation using non-blocking MPI communication calls to divide the workload between two processes and maximize matrix row sum calculation.