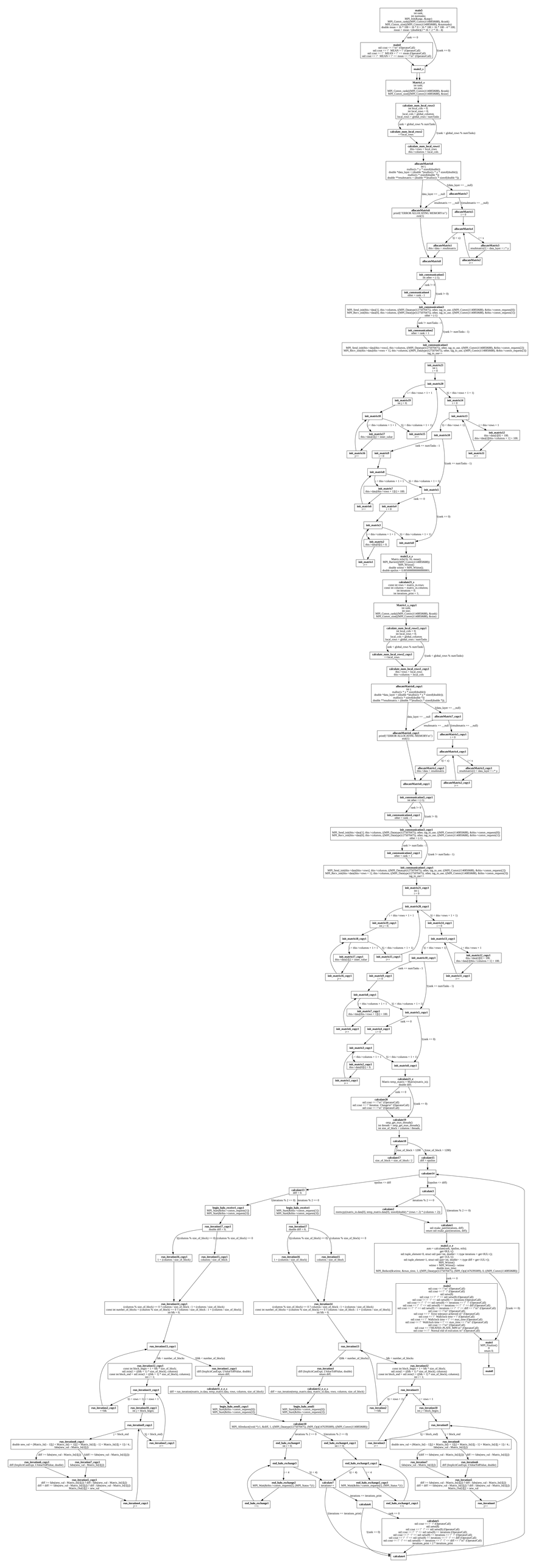
main5
int argc
int numtasks
MPI_Init(&argc, &argv)
MPI_Comm_rank(MPI_Comm(1140950688), &rank)
MPI_Comm_size(MPI_Comm(1140950688), &numtasks)
double mean = 35.4 * 100 + 35.4 * 34 * 100 + 35.4 * 66 / 4 * 100;
mean = mean / (double)(2.4 * 35 / 2 * 35 / 4);
rank == 0
main4
std::cout << "" MEAN = " << mean << "" << std::endl;
std::cout << "" MEAN = " << mean << "" << std::endl;
rank != 0
main3
Matrix3_x
int rank;
int size;
MPI_Comm_rank(MPI_Comm(1140950688), &rank);
MPI_Comm_size(MPI_Comm(1140950688), &size);
calculate_num_local_rows3
int local_rem = 0;
int local_rows = 0;
local_rem = global_rows % numTasks;
local_rows = global_rows / numTasks;
rank < global_rows % numTasks
calculate_num_local_rows2
++local_rows;
rank >= global_rows % numTasks
calculate_num_local_rows3
int local_rem = 0;
int local_rows = 0;
local_rem = global_rows % numTasks;
local_rows = global_rows / numTasks;
rank < global_rows % numTasks
calculate_num_local_rows2
++local_rows;
rank >= global_rows % numTasks
calculate_num_local_rows1
this->rows = local_rows
this->columns = local_cols
allocateMatrix0
int i;
double **matrix = (double **)malloc(rows * sizeof(double*));
matrix[0] = (double *)malloc(rows * columns * sizeof(double));
double **resultmatrix = (double **)malloc(rows * sizeof(double *));
data_layer == __null
allocateMatrix7
resultmatrix == __null
resultmatrix == __null
allocateMatrix6
printf("ERROR ALLOCATING MEMORY\n");
exit(1);
allocateMatrix5
i < 0
i++
allocateMatrix4
i * x
'i' < x
allocateMatrix1
this->data = resultmatrix
allocateMatrix3
resultmatrix[i] = data_layer + i * y
allocateMatrix0
allocateMatrix2
init_communication5
int other = i+1;
init_communication4
other = rank + 1;
rank != 0
init_communication3
MPI_Send_init(this->data[i], this->columns, (MPI_Datatype)1275070473, other, tag_to_use, (MPI_Comm)1140950688, &this->comm_request[i]);
MPI_Recv_init(this->data[i], this->columns, (MPI_Datatype)1275070473, other, tag_to_use, (MPI_Comm)1140950688, &this->comm_request[i+1]);
other = i-1;
rank != numTasks - 1
init_communication2
other = rank + 1
rank == numTasks - 1
init_communication1
MPI_Send_init(this->data[this->rows], this->columns, (MPI_Datatype)1275070473, other, tag_to_use, (MPI_Comm)1140950688, &this->comm_request[2];
MPI_Recv_init(this->data[this->rows + 1], this->columns, (MPI_Datatype)1275070473, other, tag_to_use, (MPI_Comm)1140950688, &this->comm_request[3];
tag_to_use++
init_matrix21
int i;
i < 2
i++
init_matrix20
i < this->rows + 1 + 1
'i' < this->rows + 1 + 1
init_matrix19
int j = 0;
init_matrix14
i + 0
init_matrix18
j < this->columns + 1 + 1
'j' < this->columns + 1 + 1
init_matrix15
j++
init_matrix10
init_matrix13
j < this->rows + 1
'i' < this->rows + 1
init_matrix17
this->data[i][j] = inner_value
init_matrix12
this->data[i][j] = 100;
this->data[this->rows + 1][j + 100;
rank == numTasks - 1
init_matrix9
i++
init_matrix11
init_matrix8
int i;
i < 0
i++
init_matrix6
this->data[this->rows + 1][j] = 100;
init_matrix5
rank == 0
init_matrix4
i + 0
init_matrix3
j < this->columns + 1 + 1
'j' < this->columns + 1 + 1
init_matrix2
this->data[i][j] = 0;
init_matrix0
init_matrix1
i++
matrix3_x_y
Matrix *matrix = new(
MPI_Barrier((MPI_Comm)1140950688);
MPI_Wtime();
double epsilon = 0.0456000000000000001;
calculate21_x
const int rows = matrix_in.rows;
const int columns = matrix_in.columns;
int iteration = 0;
int iteration_print = 1;
Matrix3_x_copy1
int rank;
int size;
MPI_Comm_rank((MPI_Comm)1140950688, &rank);
MPI_Comm_size((MPI_Comm)1140950688, &size);
calculate_num_local_rows3_copy1
int local_rem = 0;
int local_rows = 0;
local_rem = global_rows % numTasks;
local_rows = global_rows / numTasks;
rank < global_rows % numTasks
calculate_num_local_rows2_copy1
++local_rows;
rank >= global_rows % numTasks
calculate_num_local_rows1_copy1
this->rows = local_rows
this->columns = local_cols
allocateMatrix0_copy1
int i;
double **matrix = (double **)malloc(rows * sizeof(double*));
matrix[0] = (double *)malloc(rows * columns * sizeof(double));
double **resultmatrix = (double **)malloc(rows * sizeof(double *));
data_layer == __null
allocateMatrix7_copy1
resultmatrix == __null
resultmatrix == __null
allocateMatrix6_copy1
printf("ERROR ALLOCATING MEMORY\n");
exit(1);
allocateMatrix5_copy1
i < 0
i++
allocateMatrix4_copy1
i * x
'i' < x
allocateMatrix1_copy1
this->data = resultmatrix
allocateMatrix3_copy1
resultmatrix[i] = data_layer + i * y
allocateMatrix0_copy1
allocateMatrix2_copy1
init_communication5_copy1
int other = i+1;
init_communication4_copy1
other = rank + 1;
rank != 0
init_communication3_copy1
MPI_Send_init(this->data[i], this->columns, (MPI_Datatype)1275070473, other, tag_to_use, (MPI_Comm)1140950688, &this->comm_request[0]);
MPI_Recv_init(this->data[i], this->columns, (MPI_Datatype)1275070473, other, tag_to_use, (MPI_Comm)1140950688, &this->comm_request[1];
other = i-1;
rank != numTasks - 1
init_communication2_copy1
other = rank + 1
rank == numTasks - 1
init_communication1_copy1
MPI_Send_init(this->data[this->rows], this->columns, (MPI_Datatype)1275070473, other, tag_to_use, (MPI_Comm)1140950688, &this->comm_request[2];
MPI_Recv_init(this->data[this->rows + 1], this->columns, (MPI_Datatype)1275070473, other, tag_to_use, (MPI_Comm)1140950688, &this->comm_request[3];
tag_to_use++
init_matrix21_copy1
int i;
i < 2
i++
init_matrix20_copy1
i < this->rows + 1 + 1
'i' < this->rows + 1 + 1
init_matrix19_copy1
int j = 0;
init_matrix14_copy1
i + 0
init_matrix18_copy1
j < this->columns + 1 + 1
'j' < this->columns + 1 + 1
init_matrix15_copy1
j++
init_matrix10_copy1
init_matrix13_copy1
j < this->rows + 1
'i' < this->rows + 1
init_matrix17_copy1
this->data[i][j] = inner_value
init_matrix12_copy1
this->data[i][j] = 100;
this->data[this->rows + 1][j + 100;
rank == numTasks - 1
init_matrix9_copy1
i++
init_matrix11_copy1
init_matrix8_copy1
int i;
i < 0
i++
init_matrix6_copy1
this->data[this->rows + 1][j] = 100;
init_matrix5_copy1
rank == 0
init_matrix4_copy1
i + 0
init_matrix3_copy1
j < this->columns + 1 + 1
'j' < this->columns + 1 + 1
init_matrix2_copy1
this->data[i][j] = 0;
init_matrix0_copy1
init_matrix1_copy1
i++
calculate21_x
Matrix temp_calculate21_matrix_ini(
double diff;
rank == 0
calculate20
std::cout << "" a" (OperatorCall)
std::cout << "" Iteration Changes" (OperatorCall)
std::cout << "" in" (OperatorCall)
rank != 0
calculate19
omp_get_wtick_threads()
int threads = omp_get_max_threads();
int size_of_block = columns / threads;
calculate18
size_of_block > 1200
'size_of_block < 1200
calculate17
size_of_block = size_of_block / 2
calculate16
diff = epsilon
calculate14
epsilon <= diff
epsilon >= diff)
calculate13
diff = 0.0
Iterations % 2 == 0
Iterations % 2 == 1
calculate3
iterations % 2 == 0
iterations % 2 == 1
begin_halo_receive1_copy1
MPI_Start(&this->comm_request[1]);
MPI_Start(&this->comm_request[2]);
begin_halo_receive1
MPI_Start(&this->comm_request[2]);
MPI_Start(&this->comm_request[3]);
calculate2
memcpy(matrix_in.data[0], temp_matrix.data[0], sizeof(double) * (rows + 2) * (columns + 2));
calculate1
std::make_pair(iterations, diff)
run_iteration17_copy1
double diff = 0;
run_iteration17
double diff = 0;
this->column % size_of_block) == 0
this->column % size_of_block) == 0
this->column % size_of_block) == 0
this->column % size_of_block) == 0
run_iteration16_copy1
i = (columns / size_of_block)
run_iteration15_copy1
columns / size_of_block
run_iteration16
i = (columns / size_of_block)
run_iteration15
columns / size_of_block
matrix3_x_y
auto = matrix3_x_y(matrix_in.rows, matrix_in.columns, mfs);
get<0>(R1) = 0;
std::tuple_element<0, struct std::pair<int, double> >::type iterations = get<0>(L+);
get<0>(R1) = iterations;
std::pair<int, double> const &L = std::pair<int, double>(iterations, diff);
MPI_Wtime()
write_out_time = write_out_time + write
double max_time;
MPI_Reduce(&wtime, &max_time, 1, (MPI_Datatype)1275070473, (MPI_Op)1476395009, 0, (MPI_Comm)1140950688);
run_iteration14_copy1
((column % size_of_block) == 0 || column % size_of_block) == 1) * ((column / size_of_block) const int number_of_blocks = (column % size_of_block) == 0 ? column / size_of_block : (column / size_of_block);
int blk = 0;
run_iteration14
((column % size_of_block) == 0 || column % size_of_block) == 1) * ((column / size_of_block) const int number_of_blocks = (column % size_of_block) == 0 ? column / size_of_block : (column / size_of_block);
int blk = 0;
rank == 0
matrix3
std::cout << "Cols" (OperatorCall)
std::cout << "" (OperatorCall)
std::setw(6)
std::cout << "" << "" << std::setw(6) (OperatorCall)
std::cout << "" << "" << std::setw(6) << std::setw(6) (OperatorCall)
std::cout << "" << "" << std::setw(6) << iterations << "" << "" << diff (OperatorCall)
std::cout << "" << "" << std::setw(6) << iterations << "" << "" << diff (OperatorCall)
std::cout << "" Error tolerance achieved (or diverged)." (OperatorCall)
std::cout << "" Total time = " << max_time (OperatorCall)
std::cout << "" Walltick time = " << max_time (OperatorCall)
std::cout << "" Walltick time = " << max_time << "" (OperatorCall)
std::cout << "" Total time = " << max_time << "" (OperatorCall)
std::cout << "" THEATED_PLATE_MPI-vC" (OperatorCall)
std::cout << "" Normal end of execution.\n" (OperatorCall)
rank != 0
matrix2
MPI_Finalize()
return 0;
main4
run_iteration13_copy1
((column % size_of_block) == 0 || column % size_of_block) == 1) * ((column / size_of_block) const int number_of_blocks = (column % size_of_block) == 0 ? column / size_of_block : (column / size_of_block);
int blk = 0;
run_iteration13
((column % size_of_block) == 0 || column % size_of_block) == 1) * ((column / size_of_block) const int number_of_blocks = (column % size_of_block) == 0 ? column / size_of_block : (column / size_of_block);
int blk = 0;
run_iteration12_copy1
const int block_end = std::min(((blk + 1) * size_of_block), columns);
const int block_end = std::min((((blk + 1) * size_of_block), columns);
int i = 1;
run_iteration1_copy1
diff (implicitConv)(up_t)(bar)(foo)(value, double)
return diff;
run_iteration12
const int block_end = std::min(((blk + 1) * size_of_block), columns);
const int block_end = std::min((((blk + 1) * size_of_block), columns);
int i = 1;
run_iteration1
diff (implicitConv)(up_t)(bar)(foo)(value, double)
return diff;
run_iteration11_copy1
blk < number_of_blocks
blk < number_of_blocks
run_iteration11
blk < number_of_blocks
blk < number_of_blocks
run_iteration2_copy1
i = (i + 1)
i < rows + 1
calculate12_x_y_copy1
diff += run_iteration(temp_matrix.data, temp_matrix.data, rows, columns, size_of_block)
calculate12_x_y
diff += run_iteration(temp_matrix.data, matrix_in.data, rows, columns, size_of_block)
run_iteration2
i = (i + 1)
i < rows + 1
run_iteration10
i = (block_begin)
run_iteration1_copy1
i = block_end
i < block_end
run_iteration0_copy1
double new_val = (Matrix_h[i - 1][j] + Matrix_h[i + 1][j] + Matrix_h[i][j - 1] + Matrix_h[i][j + 1]) / 4;
fabs(new_val - Matrix_h[i][j])
diff += fabs(new_val - Matrix_h[i][j]);
Matrix_h[i][j] = new_val;
run_iteration3_copy1
diff (implicitConv)(up_t)(bar)(foo)(value, double)
run_iteration3_copy1
fabs(new_val - Matrix_h[i][j])
run_iteration1_copy1
diff += fabs(new_val - Matrix_h[i][j]);
diff = diff >= fabs(new_val - Matrix_h[i][j]) ? diff : fabs(new_val - Matrix_h[i][j]);
Matrix_h[i][j] = new_val;
run_iteration1_copy1
i++
run_iteration0
i = (block_begin)
run_iteration1
i = block_end
i < block_end
run_iteration0
double new_val = (Matrix_h[i - 1][j] + Matrix_h[i + 1][j] + Matrix_h[i][j - 1] + Matrix_h[i][j + 1]) / 4;
fabs(new_val - Matrix_h[i][j])
diff += fabs(new_val - Matrix_h[i][j]);
run_iteration3
diff (implicitConv)(up_t)(bar)(foo)(value, double)
run_iteration3
fabs(new_val - Matrix_h[i][j])
run_iteration1
diff += fabs(new_val - Matrix_h[i][j]);
diff = diff >= fabs(new_val - Matrix_h[i][j]) ? diff : fabs(new_val - Matrix_h[i][j]);
Matrix_h[i][j] = new_val;
run_iteration1
i++
calculate10
MPI_Allreduce((void *)i, &diff, 1, (MPI_Datatype)1275070473, (MPI_Op)1476395009, ((MPI_Comm)1140950688));
iterations % 2 == 0
iterations % 2 == 1
red_halo_exchange2
red_halo_exchange2_copy1
red_halo_exchange3
i < 4
i < 4
red_halo_exchange3_copy1
i < 4
i < 4
red_halo_exchange4
MPI_Wait(&this->comm_request[i], (MPI_Status *)1)
calculate9
iterations < 1
red_halo_exchange4_copy1
MPI_Wait(&this->comm_request[i], (MPI_Status *)1)
red_halo_exchange5
red_halo_exchange5_copy1
calculate4
iterations += iterations_print
calculate4
iterations += iterations_print
red_halo_exchange6
red_halo_exchange6_copy1
rank == 0
calculate5
std::cout << "" a" (OperatorCall)
std::cout << "" Iteration Changes" (OperatorCall)
std::cout << "" in" (OperatorCall)
std::cout << "" << "" << std::setw(6) << iterations << "" << "" (OperatorCall)
std::cout << "" << "" << std::setw(6) << iterations << "" << "" (OperatorCall)
std::cout << "" << "" << std::setw(6) << iterations << "" << "" << diff << "" (OperatorCall)
iterations_print = 2 * iterations_print
rank != 0
calculate6