**Лістинг програми ПРГ2**

Файл CalculateUtils.java

|  |  |
| --- | --- |
|  | /\*\* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \* \* |
|  | \* Програмування паралельний комп'ютерних систем \* |
|  | \* Курсова робота. ПРГ2. Бібліотека MPI \* |
|  | \* \* |
|  | \* Завдання: A = B(MO\*MK)\*a + min(Z)\*E\*MR \* |
|  | \* \* |
|  | \* Файл: CalculateUtils.java \* |
|  | \* Автор: Кузьменко Володимир \* |
|  | \* Група: ІО-21 \* |
|  | \* Дата: 23.04.15 \* |
|  | \* \* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \*/ |
|  | public class CalculateUtils { |
|  |  |
|  | public static Vector inputVector(int value) { |
|  | Vector vector = new Vector(Executor.N); |
|  | for (int i = 0; i < vector.size(); i++) { |
|  | vector.set(i, value); |
|  | } |
|  | return vector; |
|  | } |
|  |  |
|  | public static Matrix inputMatrix(int value) { |
|  | Matrix matrix = new Matrix(Executor.N); |
|  | for (int i = 0; i < matrix.size(); i++) { |
|  | for (int j = 0; j < matrix.size(); j++) { |
|  | matrix.set(i, j, value); |
|  | } |
|  | } |
|  | return matrix; |
|  | } |
|  |  |
|  | public static void outputVector(Vector vector) { |
|  | if (vector.size() <= 40) { |
|  | System.out.print(vector.toString()); |
|  | } |
|  | } |
|  | public static void outputMatrix(Matrix matrix) { |
|  | if (matrix.size() <= 40) { |
|  | System.out.print(matrix.toString()); |
|  | } |
|  | } |
|  | public static Vector add(final Vector left, final Vector right) { |
|  |  |
|  | Vector result = new Vector(left.size()); |
|  | for (int i = 0; i < left.size(); i++) { |
|  | result.set(i, left.get(i) + right.get(i)); |
|  | } |
|  | return result; |
|  | } |
|  | public static Vector mult(final int left, final Vector right) { |
|  | Vector result = new Vector(right.size()); |
|  | for (int i = 0; i < right.size(); i++) { |
|  | result.set(i, left \* right.get(i)); |
|  | } |
|  | return result; |
|  | } |
|  | public static Matrix mult(final Matrix left, final Matrix right) { |
|  |  |
|  | Matrix result = new Matrix(right.size()); |
|  | result = result.copy(0, left.size()); |
|  | for (int i = 0; i < left.size(); i++) { |
|  | for (int j = 0; j < right.size(); j++) { |
|  | result.set(i, j, 0); |
|  | for (int y = 0; y < right.size(); y++) { |
|  | result.set(i, j, |
|  | result.get(i, j) + left.get(i, y) \* right.get(y, j)); |
|  | } |
|  | } |
|  | } |
|  | return result; |
|  | } |
|  | public static Vector mult(final Vector left, final Matrix right) { |
|  |  |
|  | Vector result = new Vector(right.size()); |
|  | for (int i = 0; i < right.size(); i++) { |
|  | result.set(i, 0); |
|  | for (int j = 0; j < left.size(); j++) { |
|  | result.set(i, result.get(i) + left.get(j) \* right.get(i, j)); |
|  | } |
|  | } |
|  | return result; |
|  | } |
|  | public static int min(final Vector vector) { |
|  | int min = Integer.MAX\_VALUE; |
|  | for (int i = 0; i < vector.size(); i++) { |
|  | if (vector.get(i) < min) { |
|  | min = vector.get(i); |
|  | } |
|  | } |
|  | return min; |
|  | } |
|  | } |

Файл Executor.java:

|  |  |
| --- | --- |
|  | import mpi.\*; |
|  | /\*\* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \* \* |
|  | \* Програмування паралельний комп'ютерних систем \* |
|  | \* Курсова робота. ПРГ2. Бібліотека MPI \* |
|  | \* \* |
|  | \* Завдання: A = B(MO\*MK)\*a + min(Z)\*E\*MR \* |
|  | \* \* |
|  | \* Файл: Executor.java \* |
|  | \* Автор: Кузьменко Володимир \* |
|  | \* Група: ІО-21 \* |
|  | \* Дата: 23.04.15 \* |
|  | \* \* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \*/ |
|  |  |
|  | public class Executor { |
|  | public static int N; |
|  | public static int P; |
|  | public static int H; |
|  | public static int k; |
|  |  |
|  | public static void main(String[] args) throws Exception { |
|  | MPI.Init(args); |
|  | System.out.println("Task " + MPI.COMM\_WORLD.Rank() + " started"); |
|  | P = Integer.parseInt(args[1]); |
|  | N = Integer.parseInt(args[3]); |
|  | H = N / P; |
|  | k = P / 2; |
|  | long start; |
|  | long[] timework = new long[1]; |
|  | long[] buf = new long[1]; |
|  | TaskPool pool = new TaskPool(); |
|  | start = System.currentTimeMillis(); |
|  | if (MPI.COMM\_WORLD.Size() == 1) { |
|  | pool.singleThreadTask(); |
|  | } else { |
|  | pool.leftTaskGroup(); |
|  | pool.middleTaskGroup(); |
|  | pool.rightTaskGroup(); |
|  | } |
|  | buf[0] = System.currentTimeMillis() - start; |
|  | MPI.COMM\_WORLD.Reduce(buf, 0, timework, 0, 1, MPI.LONG, MPI.MAX, 0); |
|  | System.out.println("Task " + MPI.COMM\_WORLD.Rank() + " finished"); |
|  | if (MPI.COMM\_WORLD.Rank() == 0) { |
|  | System.out.println("Time work: " + timework[0]); |
|  | } |
|  |  |
|  | MPI.Finalize(); |
|  |  |
|  | } |
|  | } |

Файл Matrix.java:

|  |  |
| --- | --- |
|  | import java.io.Serializable; |
|  |  |
|  | /\*\* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \* \* |
|  | \* Програмування паралельний комп'ютерних систем \* |
|  | \* Курсова робота. ПРГ2. Бібліотека MPI \* |
|  | \* \* |
|  | \* Завдання: A = B(MO\*MK)\*a + min(Z)\*E\*MR \* |
|  | \* \* |
|  | \* Файл: Matrix.java \* |
|  | \* Автор: Кузьменко Володимир \* |
|  | \* Група: ІО-21 \* |
|  | \* Дата: 23.04.15 \* |
|  | \* \* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \*/ |
|  | public class Matrix implements Serializable { |
|  | private static final long serialVersionUID = 1L; |
|  | private Vector[] array; |
|  |  |
|  | public Matrix(int n) { |
|  | array = new Vector[n]; |
|  | for (int i = 0; i < array.length; i++) { |
|  | array[i] = new Vector(n); |
|  | } |
|  | } |
|  | public void set(int n, int m, int val) { |
|  | array[n].set(m, val); |
|  | } |
|  | public int get(int n, int m) { |
|  | return array[n].get(m); |
|  | } |
|  | public Vector get(int index) { |
|  | return array[index]; |
|  | } |
|  |  |
|  | public int size() { |
|  | return array.length; |
|  | } |
|  | public String toString() { |
|  | String res = ""; |
|  | for (int i = 0; i < array.length; i++) { |
|  | res += array[i].toString(); |
|  | } |
|  | return res; |
|  | } |
|  | public Matrix copy(int start, int size) { |
|  | Matrix result = new Matrix(size); |
|  | for (int i = 0; i < result.size(); i++) { |
|  | result.array[i] = array[i + start].copy(0, Executor.N); |
|  | } |
|  | return result; |
|  | } |
|  | } |

Файл MessageBox.java:

|  |  |
| --- | --- |
|  | import java.io.Serializable; |
|  | import java.util.ArrayList; |
|  | /\*\* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \* \* |
|  | \* Програмування паралельний комп'ютерних систем \* |
|  | \* Курсова робота. ПРГ2. Бібліотека MPI \* |
|  | \* \* |
|  | \* Завдання: A = B(MO\*MK)\*a + min(Z)\*E\*MR \* |
|  | \* \* |
|  | \* Файл: MessageBox.java \* |
|  | \* Автор: Кузьменко Володимир \* |
|  | \* Група: ІО-21 \* |
|  | \* Дата: 23.04.15 \* |
|  | \* \* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \*/ |
|  | public class MessageBox implements Serializable { |
|  | private static final long serialVersionUID = 1L; |
|  | private ArrayList<Matrix> matrixs = new ArrayList<Matrix>(3); |
|  | private ArrayList<Vector> vectors = new ArrayList<Vector>(3); |
|  | private ArrayList<Integer> values = new ArrayList<>(2); |
|  |  |
|  | public void addMatrix(Matrix matrix) { |
|  | matrixs.add(matrix); |
|  |  |
|  | } |
|  |  |
|  | public Matrix setMatrix(int key, Matrix matrix) { |
|  | return matrixs.set(key, matrix); |
|  | } |
|  |  |
|  | public void AddVector(Vector vector) { |
|  | vectors.add(vector); |
|  | } |
|  |  |
|  | public Vector setVector(int key, Vector vector) { |
|  | return vectors.set(key, vector); |
|  | } |
|  |  |
|  | public void AddValue(int value) { |
|  | values.add(value); |
|  | } |
|  |  |
|  | public int setValue(int key, int value) { |
|  | return values.set(key, value); |
|  | } |
|  |  |
|  | public Matrix getMatrix(int key) { |
|  | return matrixs.get(key); |
|  | } |
|  |  |
|  | public Vector getVector(int key) { |
|  | return vectors.get(key); |
|  | } |
|  |  |
|  | public int getValue(int key) { |
|  | return values.get(key); |
|  | } |
|  | } |

Файл MessageKeys.java:

|  |  |
| --- | --- |
|  | /\*\* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \* \* |
|  | \* Програмування паралельний комп'ютерних систем \* |
|  | \* Курсова робота. ПРГ2. Бібліотека MPI \* |
|  | \* \* |
|  | \* Завдання: A = B(MO\*MK)\*a + min(Z)\*E\*MR \* |
|  | \* \* |
|  | \* Файл: MessageKeys.java \* |
|  | \* Автор: Кузьменко Володимир \* |
|  | \* Група: ІО-21 \* |
|  | \* Дата: 23.04.15 \* |
|  | \* \* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \*/ |
|  | public class MessageKeys { |
|  | static int SEND\_RECIVE\_B\_MK\_MR\_alfa = 1; |
|  | static int SEND\_RECIVE\_Z\_E\_MO = 2; |
|  | static int SEND\_RECIVE\_RESULT\_A = 3; |
|  | static int SEND\_RECIVE\_B\_MK\_MR\_alfa\_TO\_k=4; |
|  | static int SEND\_RECIVE\_MO\_Z\_E\_TO\_P\_1=5; |
|  | static int SEND\_RECIVE\_A\_TO\_0 = 6; |
|  | } |

Файл TaskPool.java:

|  |  |
| --- | --- |
|  | import mpi.MPI; |
|  | import mpi.Request; |
|  | /\*\* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \* \* |
|  | \* Програмування паралельний комп'ютерних систем \* |
|  | \* Курсова робота. ПРГ2. Бібліотека MPI \* |
|  | \* \* |
|  | \* Завдання: A = B(MO\*MK)\*a + min(Z)\*E\*MR \* |
|  | \* \* |
|  | \* Файл: TaskPool.java \* |
|  | \* Автор: Кузьменко Володимир \* |
|  | \* Група: ІО-21 \* |
|  | \* Дата: 23.04.15 \* |
|  | \* \* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \*/ |
|  | public class TaskPool { |
|  | private int P = MPI.COMM\_WORLD.Size(); |
|  | private int H = Executor.H; |
|  | private int k = P / 2; |
|  | private int rank; |
|  | int leftRank; |
|  | int rightRank; |
|  | int minRank = (k - 1) / 2; |
|  | int endFirsRowRank = k - 1; |
|  | int middleRightRank = P - 1; |
|  | private MessageKeys KEY; |
|  | boolean notFourCore = true; |
|  | boolean notTwoCore = true; |
|  |  |
|  | private Vector B, E, Z, A; |
|  | private Matrix MK, MR, MO; |
|  | private int alfa; |
|  | private int localMin = Integer.MAX\_VALUE; |
|  |  |
|  | private MessageBox[] box = new MessageBox[1]; |
|  | private MessageBox message = new MessageBox(); |
|  |  |
|  | public TaskPool() { |
|  | rank = MPI.COMM\_WORLD.Rank(); |
|  | rightRank = rank + 1; |
|  | leftRank = rank - 1; |
|  |  |
|  | if (P == 4) { |
|  | minRank = 1; |
|  | endFirsRowRank = 1; |
|  | notFourCore = false; |
|  | } |
|  |  |
|  | if (P == 2) { |
|  | notTwoCore = false; |
|  | endFirsRowRank = 1; |
|  | minRank = 1; |
|  |  |
|  | } |
|  |  |
|  | } |
|  |  |
|  | public void singleThreadTask() { |
|  | Vector B, E, Z, A; |
|  | Matrix MK, MR, MO; |
|  | int alfa = 1; |
|  | E = CalculateUtils.inputVector(1); |
|  | Z = CalculateUtils.inputVector(10); |
|  | Z.set(7, 1); |
|  | MO = CalculateUtils.inputMatrix(1); |
|  | B = CalculateUtils.inputVector(1); |
|  | MR = CalculateUtils.inputMatrix(1); |
|  | MK = CalculateUtils.inputMatrix(1); |
|  |  |
|  | A = CalculateUtils.add( |
|  | CalculateUtils.mult(alfa, |
|  | CalculateUtils.mult(B, CalculateUtils.mult(MK, MO))), |
|  | CalculateUtils.mult(CalculateUtils.min(Z), |
|  | CalculateUtils.mult(E, MR))); |
|  |  |
|  | CalculateUtils.outputVector(A); |
|  |  |
|  | } |
|  |  |
|  | @SuppressWarnings("static-access") |
|  | public void leftTaskGroup() { |
|  | if (rank == 0 || (rank == k && notTwoCore)) { |
|  |  |
|  | if (rank == 0) { |
|  | // 1. Якщо rank = 0, то ввести B, MK, MR, α. |
|  | alfa = 1; |
|  |  |
|  | B = CalculateUtils.inputVector(1); |
|  | MR = CalculateUtils.inputMatrix(1); |
|  | MK = CalculateUtils.inputMatrix(1); |
|  |  |
|  | message.AddValue(alfa); |
|  | message.AddVector(B); |
|  | if (notTwoCore) { |
|  | message.addMatrix(MK.copy(MK.size() / 2, MK.size() / 2)); |
|  | message.addMatrix(MR.copy(MR.size() / 2, MR.size() / 2)); |
|  | MR = MR.copy(0, MK.size() / 2); |
|  | MK = MK.copy(0, MK.size() / 2); |
|  | box[0] = message; |
|  | // 2. Якщо rank = 0, то передати B, MK(k\*H), MR(k\*H), α. |
|  | // Задачі знизу. |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, k, |
|  | KEY.SEND\_RECIVE\_B\_MK\_MR\_alfa\_TO\_k); |
|  | } else { |
|  | message.addMatrix(MK.copy(H, MR.size() - H)); |
|  | message.addMatrix(MR.copy(H, MR.size() - H)); |
|  | box[0] = message; |
|  | } |
|  |  |
|  | message.setMatrix(0, MK.copy(H, MR.size() - H)); |
|  | message.setMatrix(1, MR.copy(H, MR.size() - H)); |
|  | // 4. Передати B, MK(k\*H)-H, MR(k\*H)-H, α задачі справа. |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, rightRank, |
|  | KEY.SEND\_RECIVE\_B\_MK\_MR\_alfa); |
|  |  |
|  | box[0] = null; |
|  | } |
|  | if (rank == k && notTwoCore) { |
|  | // 3. Якщо rank = k, то прийняти B, MK(k\*H), MR(k\*H), α, від |
|  | // задачі зверху. |
|  | MPI.COMM\_WORLD.Recv(box, 0, 1, MPI.OBJECT, 0, |
|  | KEY.SEND\_RECIVE\_B\_MK\_MR\_alfa\_TO\_k); |
|  | MK = box[0].getMatrix(0); |
|  | MR = box[0].getMatrix(1); |
|  |  |
|  | box[0].setMatrix(0, MK.copy(H, MK.size() - H)); |
|  | box[0].setMatrix(1, MR.copy(H, MR.size() - H)); |
|  | // 4. Передати B, MK(k\*H)-H, MR(k\*H)-H, α задачі справа. |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, rightRank, |
|  | KEY.SEND\_RECIVE\_B\_MK\_MR\_alfa); |
|  |  |
|  | alfa = box[0].getValue(0); |
|  | B = box[0].getVector(0); |
|  | box[0] = null; |
|  |  |
|  | } |
|  |
|  | MK = MK.copy(0, H); |
|  | MR = MR.copy(0, H); |
|  | // 5. Прийняти MO, E, ZH від задачі справа |
|  | MPI.COMM\_WORLD.Recv(box, 0, 1, MPI.OBJECT, rightRank, |
|  | KEY.SEND\_RECIVE\_Z\_E\_MO); |
|  | E = box[0].getVector(0); |
|  | Z = box[0].getVector(1); |
|  | calcMinButtonTask(); |
|  | MO = box[0].getMatrix(0); |
|  | box[0] = null; |
|  | Request r = MPI.COMM\_WORLD.Irecv(box, 0, 1, MPI.OBJECT, rightRank, |
|  | KEY.SEND\_RECIVE\_RESULT\_A); |
|  | // 14. Обчислити MAH = B(MO MKH)α + minZ∙E∙MRH |
|  | A = CalculateUtils.add(CalculateUtils.mult(alfa, |
|  | CalculateUtils.mult(B, CalculateUtils.mult(MK, MO))), |
|  | CalculateUtils.mult(localMin, CalculateUtils.mult(E, MR))); |
|  |  |
|  | // 15. Отримати MAH\*k-H від задачі справа |
|  | r.Wait(); |
|  | A.merge(box[0].getVector(0)); |
|  | if (rank == k && notTwoCore) { |
|  | box[0].setVector(0, A); |
|  | // 16. Якщо rank = k, то передати MAH\*k задачі зверху |
|  | MPI.COMM\_WORLD.Send(box, 0, 1, MPI.OBJECT, 0, |
|  | KEY.SEND\_RECIVE\_A\_TO\_0); |
|  | } |
|  | if (rank == 0) { |
|  | if (notTwoCore) { |
|  | box[0] = null; |
|  | // 17. Якщо rank = 0, то Отримати MAH\*k від задачі знизу |
|  | MPI.COMM\_WORLD.Recv(box, 0, 1, MPI.OBJECT, k, |
|  | KEY.SEND\_RECIVE\_A\_TO\_0); |
|  | A.merge(box[0].getVector(0)); |
|  | } |
|  | // 18. Якщо rank = 0, то Вивести MA |
|  | CalculateUtils.outputVector(A); |
|  | } |
|  |  |
|  | } |
|  |
|  | } |
|  | @SuppressWarnings("static-access") |
|  | public void middleTaskGroup() { |
|  | if (rank != 0 && rank != k && rank != endFirsRowRank && rank != P - 1) { |
|  | Request[] inputRequest = new Request[2]; |
|  | // 1. Прийняти B, MK(MK.size())\*H MR, α від задачі зліва. |
|  | inputRequest[0] = MPI.COMM\_WORLD.Irecv(box, 0, 1, MPI.OBJECT, |
|  | leftRank, KEY.SEND\_RECIVE\_B\_MK\_MR\_alfa); |
|  | inputRequest[0].Wait(); |
|  | MK = box[0].getMatrix(0); |
|  | MR = box[0].getMatrix(1); |
|  | box[0].setMatrix(0, MK.copy(H, MK.size() - H)); |
|  | box[0].setMatrix(1, MR.copy(H, MR.size() - H)); |
|  | // 2. Передати B, MK(MK.size()-1)\*H, MR, α задачі справа. |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, rightRank, |
|  | KEY.SEND\_RECIVE\_B\_MK\_MR\_alfa); |
|  | B = box[0].getVector(0); |
|  | alfa = box[0].getValue(0); |
|  | box[0] = null; |
|  | // 3. Прийняти MO, Z(Z.size())\*H, E від задачі справа. |
|  | inputRequest[1] = MPI.COMM\_WORLD.Irecv(box, 0, 1, MPI.OBJECT, |
|  | rightRank, KEY.SEND\_RECIVE\_Z\_E\_MO); |
|  | inputRequest[1].Wait(); |
|  | Z = box[0].getVector(1); |
|  | box[0].setVector(1, Z.copy(H, Z.size() - H)); |
|  | // 4. Передати MO, Z(Z.size()-1)\*H, E задачі зліва. |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, leftRank, |
|  | KEY.SEND\_RECIVE\_Z\_E\_MO); |
|  | Z = Z.copy(0, H); |
|  | calcMinButtonTask(); |
|  | E = box[0].getVector(0); |
|  | MO = box[0].getMatrix(0); |
|  | MK = MK.copy(0, H); |
|  | MR = MR.copy(0, H); |
|  |  |
|  | Request r = MPI.COMM\_WORLD.Irecv(box, 0, 1, MPI.OBJECT, rightRank, |
|  | KEY.SEND\_RECIVE\_RESULT\_A); |
|  | // 22. Обчислити MAH = B(MO MKH)α + minZ∙E∙MRH |
|  | A = CalculateUtils.add(CalculateUtils.mult(alfa, |
|  | CalculateUtils.mult(B, CalculateUtils.mult(MK, MO))), |
|  | CalculateUtils.mult(localMin, CalculateUtils.mult(E, MR))); |
|  |  |
|  | // 23. Отримати MA (MA.size()-1)\*H від задачі справа |
|  | r.Wait(); |
|  | A.merge(box[0].getVector(0)); |
|  | box[0].setVector(0, A); |
|  | // 24. Передати MA (MA.size())\*H задачі зліва |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, leftRank, |
|  | KEY.SEND\_RECIVE\_RESULT\_A); |
|  |  |
|  | } |
|  | } |
|  |  |
|  | @SuppressWarnings("static-access") |
|  | public void rightTaskGroup() { |
|  | if (rank == P - 1 || (rank == endFirsRowRank && notTwoCore)) { |
|  |  |
|  | if (rank == endFirsRowRank) { |
|  | // 1. Якщо rank = P/2-1, то ввести MO, Z, E. |
|  | E = CalculateUtils.inputVector(1); |
|  | Z = CalculateUtils.inputVector(10); |
|  | Z.set(7, 1); |
|  | MO = CalculateUtils.inputMatrix(1); |
|  | message.addMatrix(MO); |
|  | message.AddVector(E); |
|  | if (notTwoCore) { |
|  | message.AddVector(Z.copy(k \* H, Z.size() / 2)); |
|  | box[0] = message; |
|  | // 2. Якщо rank = P/2-1, то передати MO, Z(k\*H), E задачі |
|  | // знизу. |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, P - 1, |
|  | KEY.SEND\_RECIVE\_MO\_Z\_E\_TO\_P\_1); |
|  | box[0].setVector(1, Z.copy(H, Z.size() / 2 - H)); |
|  | message = box[0]; |
|  | } else { |
|  | message.AddVector(Z.copy(H, Z.size() / 2)); |
|  | } |
|  | } |
|  |  |
|  | box[0] = null; |
|  | // 3. Прийняти B, MKH, MRH, α від задачі справа. |
|  | MPI.COMM\_WORLD.Recv(box, 0, 1, MPI.OBJECT, leftRank, |
|  | KEY.SEND\_RECIVE\_B\_MK\_MR\_alfa); |
|  |  |
|  | B = box[0].getVector(0); |
|  | MK = box[0].getMatrix(0); |
|  | MR = box[0].getMatrix(1); |
|  | alfa = box[0].getValue(0); |
|  |  |
|  | if (rank == P - 1 && notTwoCore) { |
|  | // 4. Якщо rank = P-1, то прийняти MO, Z(k\*H), E від задачі |
|  | // зверху. |
|  | MPI.COMM\_WORLD.Recv(box, 0, 1, MPI.OBJECT, k - 1, |
|  | KEY.SEND\_RECIVE\_MO\_Z\_E\_TO\_P\_1); |
|  | MO = box[0].getMatrix(0); |
|  | E = box[0].getVector(0); |
|  | Z = box[0].getVector(1); |
|  | box[0].setVector(1, Z.copy(H, Z.size() - H)); |
|  | message = box[0]; |
|  | } |
|  |  |
|  | box[0] = message; |
|  | // 5. Передати MO, Z(k\*H)-H, E задачі зліва. |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, leftRank, |
|  | KEY.SEND\_RECIVE\_Z\_E\_MO); |
|  |  |
|  | Z = Z.copy(0, H); |
|  | calcMinButtonTask(); |
|  | // 14. Обчислити MAH = B(MO MKH)α + minZ∙E∙MRH |
|  | A = CalculateUtils.add(CalculateUtils.mult(alfa, |
|  | CalculateUtils.mult(B, CalculateUtils.mult(MK, MO))), |
|  | CalculateUtils.mult(localMin, CalculateUtils.mult(E, MR))); |
|  |  |
|  | box[0] = null; |
|  | box[0] = new MessageBox(); |
|  | box[0].AddVector(A); |
|  | // 15. Передати MAH\*k-H від задачі зліва |
|  | MPI.COMM\_WORLD.Isend(box, 0, 1, MPI.OBJECT, leftRank, |
|  | KEY.SEND\_RECIVE\_RESULT\_A); |
|  | } |
|  | } |
|  | private void calcMinButtonTask() { |
|  |  |
|  | if (rank > endFirsRowRank) { |
|  | // нижній рядок |
|  | int min[] = new int[] { CalculateUtils.min(Z) }; |
|  | // передати localMinZ задачі зверху |
|  | MPI.COMM\_WORLD.Send(min, 0, 1, MPI.INT, rank - k, 103); |
|  | min = new int[1]; |
|  | MPI.COMM\_WORLD.Recv(min, 0, 1, MPI.INT, rank - k, 130); |
|  | localMin = min[0]; |
|  | } else { |
|  |  |
|  | int min[] = new int[1]; |
|  | if (notTwoCore) { |
|  | // прийняти minZ від задачі знизу |
|  | MPI.COMM\_WORLD.Recv(min, 0, 1, MPI.INT, rank + k, 103); |
|  | } else { |
|  | min[0] = Integer.MAX\_VALUE; |
|  | } |
|  | // Обчислити localMinZ = min(ZH) |
|  | // обчислити localMinZ = min(minZ, localMinZ) |
|  |  |
|  | localMin = Math.min(min[0], CalculateUtils.min(Z)); |
|  | if (rank > minRank) { |
|  | int[] res = new int[1]; |
|  | int buf = localMin; |
|  | if (rank != endFirsRowRank) { |
|  | // отримати minZ від задачі з cправа |
|  | MPI.COMM\_WORLD.Recv(res, 0, 1, MPI.INT, rank + 1, 104); |
|  | // minZ = (localMinZ, minZ) |
|  | buf = Math.min(localMin, res[0]); |
|  | } |
|  | res[0] = buf; |
|  | // передати localMinZ задачі зліва |
|  | MPI.COMM\_WORLD.Send(res, 0, 1, MPI.INT, rank - 1, 104); |
|  | } else { |
|  | if (rank != minRank) { |
|  |  |
|  | int[] res = new int[1]; |
|  | res[0] = localMin; |
|  | if (rank != 0) { |
|  | // отримати minZ від задачі зліва |
|  | MPI.COMM\_WORLD.Recv(res, 0, 1, MPI.INT, rank - 1, 104); |
|  | localMin = Math.min(localMin, res[0]); |
|  | res[0] = localMin; |
|  | } |
|  | // передати localMinZ задачі з права |
|  | MPI.COMM\_WORLD.Send(res, 0, 1, MPI.INT, rank + 1, 104); |
|  | } |
|  | } |
|  | if (rank == minRank) { |
|  | int res[] = new int[1]; |
|  | if (notFourCore && notTwoCore) { |
|  | // 14. Якщо rank = (k-1)/2, то Отримати minZright від задачі |
|  | // справа |
|  | MPI.COMM\_WORLD.Recv(res, 0, 1, MPI.INT, rank + 1, 104); |
|  | // 15. Якщо rank = (k-1)/2, то обчислити minZ = (minZ, |
|  | // minZleft, minZright) |
|  | this.localMin = Math.min(this.localMin, res[0]); |
|  | res = new int[1]; |
|  | } |
|  | MPI.COMM\_WORLD.Recv(res, 0, 1, MPI.INT, rank - 1, 104); |
|  |  |
|  | this.localMin = Math.min(this.localMin, res[0]); |
|  | // знайшли мінімум |
|  | res[0] = this.localMin; |
|  | // 16. Якщо rank = (k-1)/2, то передати minZ задачі зліва |
|  | MPI.COMM\_WORLD.Send(res, 0, 1, MPI.INT, rank - 1, 122); |
|  | if (notFourCore && notTwoCore) |
|  | // 16. Якщо rank = (k-1)/2, то передати minZ задачі справа |
|  | MPI.COMM\_WORLD.Send(res, 0, 1, MPI.INT, rank + 1, 122); |
|  | } else if (rank > 0 && rank < minRank) { |
|  | min = new int[1]; |
|  | // 16. Якщо rank = (k-1)/2, то передати minZ задачі справа |
|  | MPI.COMM\_WORLD.Recv(min, 0, 1, MPI.INT, rank + 1, 122); |
|  | this.localMin = min[0]; |
|  | // 16. Якщо rank = (k-1)/2, то передати minZ задачі зліва |
|  | MPI.COMM\_WORLD.Send(min, 0, 1, MPI.INT, rank - 1, 122); |
|  | } else if (rank != 0 && rank != endFirsRowRank) { |
|  | min = new int[1]; |
|  | // 16. Якщо rank = (k-1)/2, то передати minZ задачі зліва |
|  | MPI.COMM\_WORLD.Recv(min, 0, 1, MPI.INT, rank - 1, 122); |
|  | this.localMin = min[0]; |
|  | // 16. Якщо rank = (k-1)/2, то передати minZ задачі справа |
|  | MPI.COMM\_WORLD.Send(min, 0, 1, MPI.INT, rank + 1, 122); |
|  | } else if (rank == 0) { |
|  | min = new int[1]; |
|  | // 16. Якщо rank = (k-1)/2, то передати minZ задачі справа |
|  | MPI.COMM\_WORLD.Recv(min, 0, 1, MPI.INT, rank + 1, 122); |
|  | this.localMin = min[0]; |
|  | } else { |
|  | min = new int[1]; |
|  | // 16. Якщо rank = (k-1)/2, то передати minZ задачі зліва |
|  | MPI.COMM\_WORLD.Recv(min, 0, 1, MPI.INT, rank - 1, 122); |
|  | localMin = min[0]; |
|  | } |
|  | if (notTwoCore) { |
|  | min[0] = localMin; |
|  | // 19. Якщо rank >k, то передати minZ від задачі знизу |
|  | MPI.COMM\_WORLD.Send(min, 0, 1, MPI.INT, rank + k, 130); |
|  | } |
|  | } |
|  | } |
|  | } |

Файл Vector.java:

|  |  |
| --- | --- |
|  | import java.io.Serializable; |
|  |  |
|  | /\*\* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \* \* |
|  | \* Програмування паралельний комп'ютерних систем \* |
|  | \* Курсова робота. ПРГ2. Бібліотека MPI \* |
|  | \* \* |
|  | \* Завдання: A = B(MO\*MK)\*a + min(Z)\*E\*MR \* |
|  | \* \* |
|  | \* Файл: Vector.java \* |
|  | \* Автор: Кузьменко Володимир \* |
|  | \* Група: ІО-21 \* |
|  | \* Дата: 23.04.15 \* |
|  | \* \* |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | \*/ |
|  | public class Vector implements Serializable { |
|  | private static final long serialVersionUID = 1L; |
|  | private int[] array; |
|  |  |
|  | public Vector(int n) { |
|  | array = new int[n]; |
|  | } |
|  | public void set(int index, int value) { |
|  | array[index] = value; |
|  | } |
|  | public int get(int index) { |
|  | return array[index]; |
|  | } |
|  | public int size() { |
|  | return array.length; |
|  | } |
|  | public String toString() { |
|  | String res = ""; |
|  | for (int i = 0; i < array.length; i++) { |
|  | res += " " + array[i]; |
|  | } |
|  | res += "\n"; |
|  | return res; |
|  | } |
|  | public Vector copy(int start, int size) { |
|  | Vector result = new Vector(size); |
|  | for (int i = 0; i < result.size(); i++) { |
|  | result.array[i] = this.array[i + start]; |
|  | } |
|  | return result; |
|  | } |
|  | public void merge(Vector vector) { |
|  | int[] buf = array; |
|  | array = new int[buf.length + vector.size()]; |
|  | System.arraycopy(buf, 0, array, 0, buf.length); |
|  | System.arraycopy(vector.array, 0, this.array, buf.length, |
|  | vector.array.length); |
|  | } |
|  | public void reverse() { |
|  | for (int i = 0; i < array.length / 2; i++) { |
|  | int buf = array[i]; |
|  | array[i] = array[i + array.length / 2]; |
|  | array[array.length / 2 + i] = buf; |
|  | } |
|  | } |
|  | } |