```
Script started on 2021-10-07 13:41:34-04:00
                                                                              x center = 1.16,
[TERM="xterm-256color" TTY="/dev/pts/1" COLUMNS="80"
                                                                              y_center = -0.10;
LINES="24"]
]0;bjk47@remotel2:
                                                                        MPE_XGraph graph;
~/Desktop/proj04/mandel/chunks[01;32mbjk47@remotel2[00
m:[01;34m~/Desktop/proj04/mandel/chunks[00m$ cat
                                                                        // initialize new varibales
                                                                        int * myChunk;
                                                                        int* totalChunks:
/* Compute/draw mandelbrot set using MPI/MPE commands,
paralellized in chunks
                                                                        int numProcesses, start, stop = -1;
                                                                        int count, size;
* Written Winter, 1998, W. David Laverell.
                                                                        double startTime, endTime, chunkSize2 = 0;
* Main edited by Brea Koenes on 10/7/21
                                                                        MPI_Init(&argc,&argv);
                                                                        MPI_Comm_rank(MPI_COMM_WORLD, &id);
* Refactored Winter 2002, Joel Adams.
                                                                        MPI_Comm_size(MPI_COMM_WORLD,
*/
                                                                     &numProcesses);
                                                                        // allocate memory
#include <stdio.h>
#include <stdbool.h>
                                                                        myChunk = (int
#include <math.h>
                                                                     *)malloc(WINDOW_HEIGHT*WINDOW_WIDTH*sizeof(int));
#include <sys/time.h>
                                                                        totalChunks = (int
#include <mpi.h>
                                                                      *)malloc(WINDOW_HEIGHT*WINDOW_WIDTH*sizeof(int));
#include <mpe.h>
#include "display.h"
                                                                        // beain time
#include <stdlib.h>
                                                                        startTime = MPI_Wtime();
/* compute the Mandelbrot-set function for a given
                                                                        // calculate chunk size
                                                                        int chunkSize1 = WINDOW WIDTH/numProcesses;
* point in the complex plane.
                                                                        int remainder = WINDOW_WIDTH % numProcesses;
* Receive: doubles x and y,
                                                                        // if remainder == 0 or remainder > id, go ahead
       complex c.
* Modify: doubles ans x and ans y.
                                                                        // otherwise spread chunk size among rest of processes
* POST: ans_x and ans_y contain the results of the
                                                                     and reduce chunk size by 1
                                                                        if ((remainder == 0) || (remainder != 0 && id < remainder))
mandelbrot-set
      function for x, y, and c.
                                                                               start = id * chunkSize1;
void compute(double x, double y, double c_real, double
                                                                               stop = start + chunkSize1;
c_imag,
                                                                       } else {
        double *ans_x, double *ans_y)
                                                                          int chunkSize2 = chunkSize1 + 1;
                                                                               start = (remainder * chunkSize1) + (chunkSize2 *
{
     *ans_x = x*x - y*y + c_real;
                                                                     (id - remainder));
     *ans y = 2*x*y + c imag;
                                                                               stop = start + chunkSize2;
}
                                                                        // if not chunkSize2, size uses chunkSize1
/* compute the 'distance' between x and y.
                                                                        // otherwise size uses chunksize2
* Receive: doubles x and y.
                                                                        if (!chunkSize2) {
* Return: x^2 + y^2.
                                                                               size = chunkSize1 * WINDOW_HEIGHT;
*/
                                                                        } else {
                                                                               size = chunkSize2 * WINDOW_HEIGHT;
double distance(double x, double y)
     return x*x + y*y;
                                                                        for (iy = start; iy < stop; iy++)
}
                                                                         for (ix = 0; ix < WINDOW_WIDTH; ix++)
int main(int argc, char* argv[])
                                                                           c_real = (ix - 400) * SPACING - x_center;
c_imag = (iy - 400) * SPACING - y_center;
  const int WINDOW HEIGHT = 900;
  const int WINDOW WIDTH = 1200;
  const double SPACING = 0.003;
                                                                           x = y = 0.0;
                                                                           n = 0;
               = 0,
  int
         n
         ix
               = 0.
                                                                           while (n < 50 \&\& distance(x, y) < 4.0)
              = 0,
         button = 0,
                                                                             compute(x, y, c_real, c_imag, &x, &y);
         id
              = 0;
                                                                             n++;
                = 0.0,
  double x
              = 0.0,
                                                                           if (n < 50) {
         c real = 0.0,
                                                                             myChunk[ix + (iy-start)*WINDOW WIDTH] = 0;
         c_{imag} = 0.0,
                                                                           } else {
```

```
myChunk[ix + (iy-start)*WINDOW WIDTH] = 1;
                                                                    Click in the window to continue...
     count++;
                                                                    Total time: 0.720765
                                                                    X connection to :10.0 broken (explicit kill or server
  // gather chunks into an array
                                                                    shutdown).
  MPI_Gather(myChunk, size, MPI_INT, totalChunks, size,
        MPI_INT, 0, MPI_COMM_WORLD);
                                                                    make: *** [Makefile:23: run] Error 1
                                                                    10;bjk47@remotel2:
  // if id == 0 open graphics
                                                                    ~/Desktop/proj04/mandel/chunks[01;32mbjk47@remotel2[00
                                                                    m:[01;34m~/Desktop/proj04/mandel/chunks[00m$ cd ...
  if (id == 0) {
                                                                    10;bjk47@remotel2:
    count = 0;
    MPE_Open_graphics( &graph, MPI_COMM_WORLD,
                                                                    ~/Desktop/proj04/mandel[01;32mbjk47@remotel2[00m:[01;3
               getDisplay(),
                                                                    4m~/Desktop/proj04/mandel[00m$ cd slices
                                                                    10;bjk47@remotel2:
               WINDOW WIDTH, WINDOW HEIGHT, 0);
                                                                    ~/Desktop/proj04/mandel/slices[01;32mbjk47@remotel2[00
    for (ix = 0; ix < WINDOW_WIDTH; ix++)
                                                                    m:[01;34m~/Desktop/proj04/mandel/slices[00m$ cat sli
                                                                    mandel.c
      for (iy = 0; iy < WINDOW_HEIGHT; iy++)
                                                                    /* Compute/draw mandelbrot set using MPI/MPE commands,
                                                                    paralellized in slices
        // draw graphics
        if (totalChunks[ix + (iy*WINDOW_WIDTH)])
                                                                     * Written Winter, 1998, W. David Laverell.
          MPE Draw point(graph, ix, iy, MPE BLACK);
                                                                     * Main edited by Brea Koenes on 10/7/21
        else
                                                                     * Refactored Winter 2002, Joel Adams.
                                                                     */
        { MPE Draw point(graph, ix, iy, MPE PINK);
                                                                    #include <stdio.h>
         count++;
                                                                    #include <stdbool.h>
      }
                                                                    #include <math.h>
    // calculate time
                                                                    #include <sys/time.h>
     endTime = MPI_Wtime() - startTime;
                                                                    #include <mpi.h>
                                                                    #include <mpe.h>
    // print to console
                                                                    #include "display.h"
    printf("\nClick in the window to continue...\n");
                                                                    #include <stdlib.h>
     printf("\nTotal time: %f\n", endTime);
                                                                    /* compute the Mandelbrot-set function for a given
     MPE_Get_mouse_press( graph, &ix, &iy, &button );
                                                                      point in the complex plane.
    MPE_Close_graphics(&graph);
                                                                     * Receive: doubles x and y,
  free(myChunk);
                                                                           complex c.
  free(totalChunks);
                                                                     * Modify: doubles ans_x and ans_y.
                                                                     * POST: ans_x and ans_y contain the results of the
  MPI Finalize();
                                                                    mandelbrot-set
  return 0;
                                                                          function for x, y, and c.
                                                                     */
                                                                    void compute(double x, double y, double c_real, double
]0;bjk47@remotel2:
                                                                    c_imag,
~/Desktop/proj04/mandel/chunks[01;32mbjk47@remotel2[00
                                                                            double *ans_x, double *ans_y)
m:[01;34m~/Desktop/proj04/mandel/chunks[00m$ make
make: 'mandel' is up to date.
                                                                         *ans_x = x*x - y*y + c_real;
]0;bjk47@remotel2:
                                                                         *ans_y = 2*x*y + c_imag;
~/Desktop/proj04/mandel/chunks[01;32mbjk47@remotel2[00
m:[01;34m~/Desktop/proj04/mandel/chunks[00m$ makecat
mandel.cd [Kmake run NP=16[K
                                                                    /* compute the 'distance' between x and y.
mpirun.mpich -np 1 ./mandel
                                                                     * Receive: doubles x and y.
Click in the window to continue...
                                                                     * Return: x^2 + y^2.
Total time: 0.175760
                                                                    double distance(double x, double y)
X connection to :10.0 broken (explicit kill or server
shutdown).
                                                                         return x*x + y*y;
                                                                    }
make: *** [Makefile:23: run] Error 1
]0;bjk47@remotel2:
~/Desktop/proj04/mandel/chunks[01;32mbjk47@remotel2[00
                                                                    int main(int argc, char* argv[])
m:[01;34m~/Desktop/proj04/mandel/chunks[00m$ make run
                                                                      const int WINDOW HEIGHT = 900;
                                                                      const int WINDOW_WIDTH = 1200;
mpirun.mpich -np 16 ./mandel
```

}

```
const double SPACING = 0.003;
                                                                          for (iy = 0; iy < WINDOW HEIGHT; iv++)
               = 0.
         n
              = 0,
                                                                             // draw graphics
        ix
              = 0,
                                                                             if (totalSlices[ix + (iy*WINDOW WIDTH)])
        iy
        button = 0,
                                                                               MPE_Draw_point(graph, ix, iy, MPE_BLACK);
        id
             = 0;
               = 0.0.
  double x
              = 0.0,
        c_real = 0.0
                                                                             { MPE_Draw_point(graph, ix, iy, MPE_PINK);
        c_{imag} = 0.0,
        x_center = 1.16
        y_center = -0.10;
                                                                         // calculate time
                                                                         endTime = MPI Wtime() - startTime;
  MPE XGraph graph;
  // initialize new varibales
                                                                         //print to console
  int * mySlices;
                                                                         printf("\nClick in the window to continue...\n");
                                                                         printf("\nTotal time: %f\n", endTime);
  int* totalSlices;
  int numProcesses = -1:
  double startTime, endTime = 0;
                                                                         MPE_Get_mouse_press( graph, &ix, &iy, &button );
                                                                         MPE_Close_graphics(&graph);
  MPI_Init(&argc,&argv);
  MPI Comm rank(MPI COMM WORLD, &id):
                                                                      free(mvSlices):
  MPI_Comm_size(MPI_COMM_WORLD,
                                                                      free(totalSlices);
&numProcesses):
                                                                      MPI Finalize();
  // allocate memory
                                                                      return 0;
  mySlices = (int
*)malloc(WINDOW HEIGHT*WINDOW WIDTH*sizeof(int));
  totalSlices = (int
                                                                    ]0;bjk47@remotel2:
*)malloc(WINDOW HEIGHT*WINDOW WIDTH*sizeof(int));
                                                                    ~/Desktop/proj04/mandel/slices[01;32mbjk47@remotel2[00
                                                                    m:[01;34m~/Desktop/proj04/mandel/slices[00m$ c\ make
                                                                    mpicc.mpich -DMPE GRAPHICS -c -Wall -ansi -pedantic
  // begin time
  startTime = MPI_Wtime();
                                                                    -std=c99 ./mandel.c
                                                                    mpicc.mpich ./mandel.o display.o -o ./mandel -lmpe -IX11 -lm
  for (iy = id; iy < WINDOW_HEIGHT; iy+= numProcesses)
                                                                    10;bik47@remotel2:
                                                                    ~/Desktop/proj04/mandel/slices[01;32mbjk47@remotel2[00
    for (ix = 0; ix < WINDOW WIDTH; ix++)
                                                                    m:[01;34m~/Desktop/proj04/mandel/slices[00m$ makecat
                                                                    mandel.c[3Pd slices[4P..make run NP=16[K
    {
     c_real = (ix - 400) * SPACING - x_center;
c_imag = (iy - 400) * SPACING - y_center;
                                                                    mpirun.mpich -np 1 ./mandel
                                                                    Click in the window to continue...
      x = y = 0.0;
      n = 0:
                                                                    Total time: 0.167805
                                                                    X connection to :10.0 broken (explicit kill or server
      while (n < 50 \&\& distance(x, y) < 4.0)
                                                                    shutdown).
                                                                    make: *** [Makefile:23: run] Error 1
       compute(x, y, c real, c imag, &x, &y);
                                                                    10;bjk47@remotel2:
       n++;
                                                                    ~/Desktop/proj04/mandel/slices[01;32mbjk47@remotel2[00
                                                                    m:[01;34m~/Desktop/proj04/mandel/slices[00m$ make run
      if (n < 50) {
       mySlices[ix + (iy*WINDOW WIDTH)] = 0;
                                                                    NP=1[Kcat mandel.c[8Pmake run NP=16
                                                                    mpirun.mpich -np 16 ./mandel
      } else {
       mySlices[ix + (iy*WINDOW_WIDTH)] = 1;
                                                                    Click in the window to continue...
   }
                                                                    Total time: 1.378152
  // reduce slices into array
                                                                    X connection to :10.0 broken (explicit kill or server
  MPI_Reduce(mySlices, totalSlices,
                                                                    shutdown).
WINDOW HEIGHT*WINDOW WIDTH,
        MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD);
                                                                    make: *** [Makefile:23: run] Error 1
                                                                    10:bjk47@remotel2:
  // if id == 0 open graphics
                                                                    ~/Desktop/proj04/mandel/slices[01;32mbjk47@remotel2[00
  if (id == 0) {
                                                                    m:[01;34m~/Desktop/proj04/mandel/slices[00m$ cd ...
    MPE_Open_graphics( &graph, MPI_COMM_WORLD,
                                                                    10:bjk47@remotel2:
                                                                    ~/Desktop/proj04/mandel[01;32mbjk47@remotel2[00m:[01;3
               getDisplay().
                                                                    4m~/Desktop/proj04/mandel[00m$ cd MW
               WINDOW WIDTH, WINDOW HEIGHT, 0);
    for (ix = 0; ix < WINDOW_WIDTH; ix++)
```

```
10;bjk47@remotel2:
~/Desktop/proj04/mandel/MW[01;32mbjk47@remotel2[00m:[
                                                                      // initialize new varibales
01:34m~/Desktop/proj04/mandel/MW[00m$ cat mandel.c
                                                                      int * myMW;
/* Compute/draw mandelbrot set using MPI/MPE commands,
                                                                      int numProcesses, next;
paralellized using a master-worker pattern
                                                                      double startTime, endTime = 0;
* Written Winter, 1998, W. David Laverell.
                                                                      MPI_Init(&argc,&argv);
                                                                      MPI_Comm_rank(MPI_COMM_WORLD, &id):
* Main edited by Brea Koenes on 10/7/21
                                                                      MPI Comm size(MPI COMM WORLD,
                                                                    &numProcesses);
* Refactored Winter 2002, Joel Adams.
                                                                      MPI_Status status;
*/
                                                                      // allocate memory
#include <stdio.h>
                                                                      m_VMW = (int
#include <stdbool.h>
                                                                    *)malloc(WINDOW_HEIGHT*WINDOW_WIDTH*sizeof(int));
#include <math.h>
#include <sys/time.h>
                                                                      // begin time
#include <mpi.h>
                                                                      startTime = MPI Wtime();
#include <mpe.h>
#include "display.h"
                                                                     // if id is 0, perform process using the default approach
#include <stdlib.h>
                                                                     if (id == 0) {
                                                                       MPE_Open_graphics( &graph, MPI_COMM_WORLD,
/* compute the Mandelbrot-set function for a given
                                                                                                 getDisplay(),
  point in the complex plane.
                                                                                                 -1. -1.
                                                                                                 WINDOW_WIDTH,
* Receive: doubles x and y,
                                                                    WINDOW HEIGHT, 0);
       complex c.
* Modify: doubles ans_x and ans_y.
                                                                      for (int i = 0; i < WINDOW HEIGHT; i++)
* POST: ans_x and ans_y contain the results of the
                                                                        for (ix = 0; ix < WINDOW WIDTH; ix++)
mandelbrot-set
      function for x, y, and c.
                                                                        {
                                                                          c real = (ix - 400) * SPACING - x center;
                                                                          c_imag = (iy - 400) * SPACING - y_center;
void compute(double x, double y, double c_real, double
c_imag,
        double *ans_x, double *ans_y)
                                                                          x = y = 0.0;
                                                                          n = 0;
     *ans_x = x*x - y*y + c_real;
                                                                          while (n < 50 \&\& distance(x, y) < 4.0)
     *ans_y = 2*x*y + c_imag;
}
                                                                            compute(x, y, c_real, c_imag, &x, &y);
/* compute the 'distance' between x and y.
                                                                            n++;
* Receive: doubles x and y.
                                                                          if (n < 50) {
* Return: x^2 + y^2.
                                                                            myMW[ix] = 1;
                                                                            MPE_Draw_point(graph, ix, iy, MPE_PINK);
double distance(double x, double y)
                                                                          } else {
                                                                            mvMW[ix] = 0;
     return x*x + y*y;
                                                                            MPE_Draw_point(graph, ix, iy, MPE_BLACK);
}
                                                                        // if id is not 0, calculate using the master-worker
int main(int argc, char* argv[])
                                                                    approach
                                                                        if (id != 0) {
  const int WINDOW HEIGHT = 900;
                                                                          // send to worker
  const int WINDOW WIDTH = 1200;
                                                                               MPI_Send(myMW, WINDOW_WIDTH, MPI_INT,
  const double SPACING = 0.003;
                                                                    0, iy, MPI_COMM_WORLD);
                                                                               // wait to recieve from worker
                                                                               MPI_Recv(&iy, 1, MPI_INT, 0, MPI_ANY_TAG,
  int
         n
               = 0,
              = 0,
                                                                    MPI_COMM_WORLD, &status);
         ix
         iy
              = 0,
                                                                        }else{
         button = 0,
         id
             = 0;
  double x
                = 0.0,
                                                                        // if status.MPI_TAG equals 0 or the processes are
               = 0.0,
                                                                    greater than 1, break loop
                                                                        if(status.MPI_TAG == 0 || (id == 0 && numProcesses >
         c real = 0.0,
         c_{imag} = 0.0,
                                                                    1)) {
         x center = 1.16,
                                                                               break;
         y_center = -0.10;
                                                                        }
                                                                      }
  MPE_XGraph graph;
```

```
// if it is not process 1, calculate using the master-worker
approach
  if (numProcesses != 1) {
      for (next = numProcesses; next <
WINDOW_HEIGHT+(numProcesses-1); next++) {
          // wait to recieve from worker
MPI_Recv(myMW, WINDOW_WIDTH, MPI_INT, MPI_ANY_SOURCE, MPI_ANY_TAG,
                          MPI_COMM_WORLD, &status);
                    if (next >= WINDOW HEIGHT) {
                         // send next to worker
                              MPI_Send(&next, 1, MPI_INT,
status.MPI_SOURCE, 0, MPI_COMM_WORLD);
                    } else {
                         // send next to worker
                              MPI_Send(&next, 1, MPI_INT,
status.MPI_SOURCE, 1, MPI_COMM_WORLD);
          for (int x = 0; x < WINDOW_WIDTH; x++) {
          if (myMW[x]==1) {
                    MPE_Draw_point(graph, x,
status.MPI_TAG, MPE_PINK);
          } else {
                    MPE_Draw_point(graph, x,
status.MPI_TAG, MPE_BLACK);
       }
     // calculate time
     endTime = MPI_Wtime() - startTime;
     // print to console
     printf("\nClick in the window to continue...\n");
     printf("\nTotal time: %f\n", endTime);
     MPE_Get_mouse_press( graph, &ix, &iy, &button );
     MPE Close graphics(&graph);
  free(myMW);
  MPI_Finalize();
  return 0;
}
]0;bjk47@remotel2:
~/Desktop/proj04/mandel/MW[01;32mbjk47@remotel2[00m:[ 01;34m~/Desktop/proj04/mandel/MW[00m$ c make
make: 'mandel' is up to date.
10;bjk47@remotel2:
~/Desktop/proj04/mandel/MW[01;32mbjk47@remotel2[00m:[ 01;34m~/Desktop/proj04/mandel/MW[00m$ makecat
mandel.c[7Pd MW..make run NP=16
mpirun.mpich -np 1 ./mandel
Click in the window to continue...
Total time: 0.118968
X connection to :10.0 broken (explicit kill or server
shutdown).
Script done on 2021-10-07 13:43:27-04:00
[COMMAND_EXIT_CODE="130"]
```