# HEATED_PLATE_OPENMP
# 2D Steady State Heat Equation Using OpenMP

**HEATED_PLATE_OPENMP**, a C code which uses the OpenMP application program interface by employing an iteration that solves the 2D steady state heat equation in parallel.

The C version of this problem has an feature; while FORTRAN OpenMP programs can easily compute a maximum value in parallel using the "reduction" clause, this is not possible in C. This program shows how the maximum can nonetheless be computed, using an approach that combines private variables and a critical section.

The sequential version of this program needs approximately 18/eps iterations to complete.

The physical region, and the boundary conditions, are suggested by this diagram;

```
                   W = 0
               +------------------+
               |                  |
      W = 100  |                  | W = 100
               |                  |
               +------------------+
                   W = 100
```

The region is covered with a grid of M by N nodes, and an N by N array W is used to record the temperature. The correspondence between array indices and locations in the region is suggested by giving the indices of the four corners:

```
                  I = 0
            [0][0]-------------[0][N-1]
               |                  |
        J = 0  |                  |  J = N-1
               |                  |
          [M-1][0]-----------[M-1][N-1]
                  I = M-1
```

The steady state solution to the discrete heat equation satisfies the following condition at an interior grid point:

```
        W[Central] = (1/4) * ( W[North] + W[South] + W[East] + W[West] )
```

where "Central" is the index of the grid point, "North" is the index of its immediate neighbor to the "north", and so on.

Given an approximate solution of the steady state heat equation, a "better" solution is given by replacing each interior point by the average of its 4 neighbors - in other words, by using the condition as an ASSIGNMENT statement:

```
        W[Central]  <=  (1/4) * ( W[North] + W[South] + W[East] + W[West] )
```

If this process is repeated often enough, the difference between successive estimates of the solution will go to zero.

This program carries out such an iteration, using a tolerance specified by the user, and writes the final estimate of the solution to a file that can be used for graphic processing.

## Licensing:

## Languages:

**HEATED_PLATE_OPENMP** is available in a C version and a C++ version and a FORTRAN90 version.

## Related Data and Programs:

DIJKSTRA_OPENMP, a C code which uses OpenMP to parallelize a simple example of Dijkstra's minimum distance algorithm for graphs.

FFT_OPENMP, a C code which demonstrates the computation of a Fast Fourier Transform in parallel, using OpenMP.

FUNCTIONS_OPENMP, a C code which demonstrates the behavior of a few of the OpenMP library functions.

HEATED_PLATE, a C code which solves the steady (time independent) heat equation in a 2D rectangular region, and is intended as a starting point for implementing an OpenMP parallel version.

heated_plate_openmp_test

HELLO_OPENMP, a C code which prints out "Hello, world!" using the OpenMP parallel programming environment.

IMAGE_DENOISE_OPENMP, a C code which applies simple filtering techniques to remove noise from an image, carrying out the operation in parallel using OpenMP.

JACOBI_OPENMP, a C code which illustrates the use of the OpenMP application program interface to parallelize a Jacobi iteration solving A*x=b.

julia_set_openmp, a C code which produces an image of a Julia set, using OpenMP to carry out the computation in parallel.

MD_OPENMP, a C code which carries out a molecular dynamics simulation using OpenMP.

mpi_test, a library which implements parallel programming in a distributed memory environment, using message passing.

MULTITASK_OPENMP, a C code which demonstrates how to "multitask", that is, to execute several unrelated and distinct tasks simultaneously, using OpenMP for parallel execution.

MXM_OPENMP, a C code which computes a dense matrix product C=A*B, using OpenMP for parallel execution.

openmp_test, C codes which use the OpenMP application program interface for carrying out parallel computations in a shared memory environment.

POISSON_OPENMP, a C code which computes an approximate solution to the Poisson equation in a rectangle, using the Jacobi iteration to solve the linear system, and OpenMP to carry out the Jacobi iteration in parallel.

PRIME_OPENMP, a C code which counts the number of primes between 1 and N, using OpenMP for parallel execution.

PTHREADS, C codes which illustrate the use of the POSIX thread library to carry out parallel program execution.

QUAD_OPENMP, a C code which approximates an integral using a quadrature rule, and carries out the computation in parallel using OpenMP.

RANDOM_OPENMP, a C code which illustrates how a parallel program using OpenMP can generate multiple distinct streams of random numbers.

SATISFY_OPENMP, a C code which demonstrates, for a particular circuit, an exhaustive search for solutions of the circuit satisfiability problem, using OpenMP for parallel execution.

SCHEDULE_OPENMP, a C code which demonstrates the default, static, and dynamic methods of "scheduling" loop iterations in OpenMP to avoid work imbalance.

SGEFA_OPENMP, a C code which reimplements the SGEFA/SGESL linear algebra routines from LINPACK for use with OpenMP.

ZIGGURAT_OPENMP, a C code which demonstrates how the ZIGGURAT library can be used to generate random numbers in an OpenMP parallel program.

## Reference:

1. Peter Arbenz, Wesley Petersen,
   Introduction to Parallel Computing - A practical guide with examples in C,
   Oxford University Press,
   ISBN: 0-19-851576-6,
   LC: QA76.58.P47.
2. Rohit Chandra, Leonardo Dagum, Dave Kohr, Dror Maydan, Jeff McDonald, Ramesh Menon,
   Parallel Programming in OpenMP,
   Morgan Kaufmann, 2001,
   ISBN: 1-55860-671-8,
   LC: QA76.642.P32.
3. Barbara Chapman, Gabriele Jost, Ruud vanderPas, David Kuck,
   Using OpenMP: Portable Shared Memory Parallel Processing,
   MIT Press, 2007,
   ISBN13: 978-0262533027,
   LC: QA76.642.C49.
4. OpenMP Architecture Review Board,
   OpenMP Application Program Interface,
   Version 3.0,
   May 2008.

## Source code:

- heated_plate_openmp.c, the source code;
- heated_plate_openmp.sh, compiles the source code;

---

*Last revised on 01 August 2020.*