Livermore loops optimization with OpenMP

Kernel 21:

- optimization manner:
  - changing loops order (bigger loop be outer)
  - loop unrolling (5 times)
  - omp_set_num_threads(THEADS);
  - #pragma omp parallel shared(px, vy, cx) private(k, i, j)
  - collapsing two inner loops
- compiling:

```
gcc -fopenmp ker21.c -m64 -lrt -lc -lm -o ker21
```

- analyzing best number of threads:
- results:

```
mester 2/Parallel algorithms/hws/hw3/common_64
bit$ ./ker21
list size n = 101
number of threads= 2
times of run = 20.
-------------
average serial time: 868.100 microsecond
average paralel time: 385.450 microsecond
-------------
speed up: 2.252 microsecond
```

```
bit$ ./ker21
list size n = 1001
number of threads= 4
times of run = 20.
-------------
average serial time: 14618.700 microsecond
average paralel time: 3597.950 microsecond
-------------
speed up: 4.063 microsecond
```

Kernel 22:

- optimization manner:
  - loop unrolling (4 times)
  - omp_set_num_threads(THEADS);
  - #pragma omp parallel shared(y, u, v, w, x) private(k)

- compiling:

```
gcc -fopenmp ker22.c -m64 -lrt -lc -lm -o ker22
```

- results:

```
mester 2/Parallel algorithms/hws/hw3/common_64
bit$ ./ker22
list size n = 101
number of threads= 2
times of run = 20.
-------------
average serial time: 2.500 microsecond
average paralel time: 4.900 microsecond
-------------
speed up: 0.510 microsecond
```

```
mester 2/Parallel algorithms/hws/hw3/common_64
bit$ ./ker22
list size n = 1001
number of threads= 4
times of run = 20.
-------------
average serial time: 41.000 microsecond
average paralel time: 28.600 microsecond
-------------
speed up: 1.434 microsecond
```

Kernel 23:

- optimization manner:
  - changing loops order (bigger loop be outer)
  - loop unrolling (2 times)
  - omp_set_num_threads(THEADS);
  - #pragma omp parallel private(j)
  - collapsing nested loops with #pragma omp collapse(2)

- compiling:

```
gcc -fopenmp ker23.c -m64 -lrt -lc -lm -o ker23
```

- results:

```
mester 2/Parallel algorithms/hws/hw3/common_64
bit$ ./ker23
list size n = 101
number of threads= 2
times of run = 20.
-------------
average serial time: 32.750 microsecond
average paralel time: 28.200 microsecond
-------------
speed up: 1.161 microsecond
```

```
mester 2/Parallel algorithms/hws/hw3/common_64
bit$ ./ker23
list size n = 1001
number of threads= 4
times of run = 20.
-------------
average serial time: 577.400 microsecond
average paralel time: 272.750 microsecond
-------------
speed up: 2.117 microsecond
```

Kernel 24:

- optimization manner:
  - omp_set_num_threads(THEADS);
  - #pragma omp parallel private(k)
  - #pragma critical for critical section of code
- compiling:

```
gcc -fopenmp ker24.c -m64 -lrt -lc -lm -o ker24
```

- results:

```
mester 2/Parallel algorithms/hws/hw3/common_64
bit$ ./ker24
list size n = 1001
number of threads= 2
times of run = 20.
-------------
average serial time: 9.000 microsecond
average paralel time: 17.900 microsecond
-------------
speed up: 0.503 microsecond
```

```
mester 2/Parallel algorithms/hws/hw3/common_64
bit$ ./ker24
list size n = 10001
number of threads= 4
times of run = 20.
-------------
average serial time: 78.700 microsecond
average paralel time: 40.250 microsecond
-------------
speed up: 1.955 microsecond
```

System configuration:

CPU: core i5 8 th generation
RAM: 8GB
OS: Ubuntu 16
Cache: 1L = 256KB, 2L = 1MB, 3L = 6MB