Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

**Лабораторная работа №2 по курсу**

**«Операционные системы»**

Группа: М8О-215Б-23

Студент: Самарский Я.В.

Преподаватель: Миронов Е.С.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 14.11.24

Москва, 2024

**Постановка задачи**

**Вариант 5.**

Составить программу на языке Си, обрабатывающую данные в многопоточном режиме. При обработки использовать стандартные средства создания потоков операционной системы (Windows/Unix). Ограничение максимального количества потоков, работающих в один момент времени, должно быть задано ключом запуска вашей программы.

Отсортировать массив целых чисел при помощи четно-нечетной сортировки Бетчера

**Общий метод и алгоритм решения**

Использованные системные вызовы:

* pthread\_create(pthread\_t \*thread, const pthread\_attr\_t \*attr, void \*(\*start\_routine) (void \*), void \*arg); – Создает новый поток, возвращает 0 при успехе
* pthread\_join(pthread\_t thread, void \*\*retval); - Ожидает завершения указанного потока. Блокирует вызывающий поток до завершения целевого потока

Ключевые особенности:

1. Размер массива должен быть степенью двойки

2. Количество потоков также должно быть степенью двойки

3. Использует многопоточность через POSIX threads (pthread)

Работа:

1. Инициализация:

- Принимает два параметра: размер массива и максимальное число потоков

- Создает случайный массив заданного размера

2. Четно-нечетная сортировка:

oddEvenMergeSort:

- Разделяет массив на две части

- Рекурсивно сортирует левую половину

- Рекурсивно сортирует правую половину

- После этого объединяет части используя bitonicMerge

oddEvenMerge:

- Рекурсивно сливает массив, увеличивая промежуток между сортируемыми элементами в два раза

- Рекурсивно обрабатывает получившиеся подпоследовательности

- Использует оптимизацию: для маленьких подмассивов (< 1024) работает без создания новых потоков

3. Управление потоками oddEvenMergeSort рекурсивно удваивает число потоков до заданного количества

4. Проверка результатов:

- После сортировки проверяется корректность (каждый следующий элемент должен быть больше предыдущего)

- Измеряется время выполнения сортировки

**Замеры эффективности**

Замеры проводились для 5 разных длин массивов. Количество потоков было от 1 до 64. Массив заполняется случайными значениями. Для каждого количества потоков проводилось 7 замеров, убирались минимальный и максимальный результат и бралось среднее значение времени.

На моем процессоре доступно 6 физических ядер и 12 логических ядер.

График со всеми замерами. Оси с логарифмическими шкалами

График ускорения (массив из 215 элементов):

График эффективности (массив из 215 элементов):

Ускорение показывает во сколько раз применение параллельного алгоритма уменьшает время решения задачи по сравнению с последовательным алгоритмом. Ускорение определяется величиной SN=T1/TN, где Т1 - время выполнения на одном потоке, TN - время выполнения на N потоках.

Эффективность - величина EN = SN/N, где SN - ускорение, N - количество используемых потоков.

**Выводы:**

Параллельная сортировка имеет смысл на достаточно больших данных, где время на создание потоков компенсируется сэкономленным временем на сортировку. Число потоков больше кол-ва логических ядер также не даёт прироста производительности, а наоборот увеличивает время сортировки из-за того, что созданные потоки не могут выполняться одновременно.

**Код программы**

main.cpp

|  |
| --- |
| #include <iostream>  #include <vector>  #include <chrono>  #include "sort.h"  int main(int argc, char\* argv[]) {  if (argc != 3) {  std::cerr << "Usage: " << argv[0] << " <array\_size> <threads\_count>\n";  return 1;  }  int arraySize = std::atoi(argv[1]);  int threads = std::atoi(argv[2]);  // Размер массива должен быть степенью 2  if ((arraySize & (arraySize - 1)) != 0) {  std::cerr << "Array size must be a power of 2\n";  return 1;  }  // Кол-во потоков должно быть степенью 2  if ((threads & (threads - 1)) != 0) {  std::cerr << "Threads count must be a power of 2\n";  return 1;  }  int powerOfParallelism = 1;  while (1 << powerOfParallelism != threads)  powerOfParallelism++;  std::vector<int> originalArray = createRandomValuesVector(arraySize);  std::vector<int> oeSorted = originalArray;  std::cout << "Starting sorting array with length: " << arraySize << "\n";  std::cout << "Max threads: " << threads << std::endl;  auto start = std::chrono::high\_resolution\_clock::now();  oddEvenMergeSort(oeSorted, 0, oeSorted.size(), powerOfParallelism);  auto end = std::chrono::high\_resolution\_clock::now();  std::chrono::duration<double> duration = end - start;  std::cout << "Time taken: " << duration.count() << " seconds\n";  for (int i = 1; i < arraySize; i++) {  if (oeSorted[i] < oeSorted[i-1]) {  std::cout << "Sorting failed!\n";  return 0;  }  }  std::cout << "Sorting successful!\n";  return 0;  } |

sort.cpp

|  |
| --- |
| #include "sort.h"  void \*parallelSort(void \*u\_arg) {  auto \*arg = static\_cast<ParallelSortArg \*>(u\_arg);  oddEvenMergeSort(\*arg->vector, arg->left, arg->right, arg->powerOfParallelism);  return nullptr;  }  void oddEvenMergeSort(std::vector<int> &a, int startIndex, int length, int powerOfParallelism = 0) {  if (length <= 1)  return;  int halfLength = length / 2;  if (powerOfParallelism > 0) {  ParallelSortArg parallelArg{  &a,  startIndex,  halfLength,  powerOfParallelism - 1  };  pthread\_t thread;  pthread\_create(&thread, nullptr, parallelSort, &parallelArg);  oddEvenMergeSort(a, startIndex + halfLength, halfLength, powerOfParallelism - 1);  pthread\_join(thread, nullptr);  } else {  oddEvenMergeSort(a, startIndex, halfLength);  oddEvenMergeSort(a, startIndex + halfLength, halfLength);  }  oddEvenMerge(a, startIndex, length, 1);  }  void oddEvenMerge(std::vector<int> &a, int startIndex, int length, int step) {  int doubleStep = step \* 2;  if (doubleStep < length) {  oddEvenMerge(a, startIndex, length, doubleStep);  oddEvenMerge(a, startIndex + step, length, doubleStep);  for (int i = startIndex + step; i + step < startIndex + length; i += doubleStep) {  compareAndExchange(a, i, i + step);  }  } else {  compareAndExchange(a, startIndex, startIndex + step);  }  }  void compareAndExchange(std::vector<int> &vector, int aIndex, int bIndex) {  if (vector[aIndex] > vector[bIndex]) {  std::swap(vector[aIndex], vector[bIndex]);  }  }  std::vector<int> createRandomValuesVector(size\_t size) {  std::vector<int> array(size);  for (size\_t i = 0; i < size; i++) {  array[i] = std::rand() % 1024;  }  return array;  } |

sort.h

|  |
| --- |
| #pragma once  #include <iostream>  #include <vector>  void oddEvenMergeSort(std::vector<int> &a, int startIndex, int length, int powerOfParallelism);  void oddEvenMerge(std::vector<int> &a, int startIndex, int length, int step);  void compareAndExchange(std::vector<int> &vector, int aIndex, int bIndex);  std::vector<int> createRandomValuesVector(size\_t size);  struct ParallelSortArg {  std::vector<int> \*vector;  int left;  int right;  int powerOfParallelism;  }; |

**Протокол работы программы**

user@DESKTOP-KC5QDB8:~/projects/mai\_os/lab2$ ./cmake-build-release/mai\_os 32768 4

Starting sorting array with length: 32768

Max threads: 4

Time taken: 0.00694654 seconds

Sorting successful!

**Strace:**

$ strace -f ./cmake-build-release/mai\_os 32768 4

user@DESKTOP-KC5QDB8:~/projects/mai\_os/lab2$ strace -f ./cmake-build-release/mai\_os 32768 4

execve("./cmake-build-release/mai\_os", ["./cmake-build-release/mai\_os", "32768", "4"], 0x7ffed1df5f98 /\* 32 vars \*/) = 0

brk(NULL) = 0x556459f65000

arch\_prctl(0x3001 /\* ARCH\_??? \*/, 0x7ffcbc1cd2a0) = -1 EINVAL (Invalid argument)

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fb522e84000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=18463, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 18463, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fb522e7f000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libstdc++.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=2260296, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 2275520, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fb522c53000

mprotect(0x7fb522ced000, 1576960, PROT\_NONE) = 0

mmap(0x7fb522ced000, 1118208, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x9a000) = 0x7fb522ced000

mmap(0x7fb522dfe000, 454656, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1ab000) = 0x7fb522dfe000

mmap(0x7fb522e6e000, 57344, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x21a000) = 0x7fb522e6e000

mmap(0x7fb522e7c000, 10432, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fb522e7c000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libgcc\_s.so.1", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=125488, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 127720, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fb522c33000

mmap(0x7fb522c36000, 94208, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x3000) = 0x7fb522c36000

mmap(0x7fb522c4d000, 16384, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1a000) = 0x7fb522c4d000

mmap(0x7fb522c51000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1d000) = 0x7fb522c51000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68, 896) = 68

newfstatat(3, "", {st\_mode=S\_IFREG|0755, st\_size=2220400, ...}, AT\_EMPTY\_PATH) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

mmap(NULL, 2264656, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fb522a0a000

mprotect(0x7fb522a32000, 2023424, PROT\_NONE) = 0

mmap(0x7fb522a32000, 1658880, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7fb522a32000

mmap(0x7fb522bc7000, 360448, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1bd000) = 0x7fb522bc7000

mmap(0x7fb522c20000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x215000) = 0x7fb522c20000

mmap(0x7fb522c26000, 52816, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fb522c26000

close(3) = 0

openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libm.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st\_mode=S\_IFREG|0644, st\_size=940560, ...}, AT\_EMPTY\_PATH) = 0

mmap(NULL, 942344, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fb522923000

mmap(0x7fb522931000, 507904, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe000) = 0x7fb522931000

mmap(0x7fb5229ad000, 372736, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x8a000) = 0x7fb5229ad000

mmap(0x7fb522a08000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0xe4000) = 0x7fb522a08000

close(3) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fb522921000

arch\_prctl(ARCH\_SET\_FS, 0x7fb5229223c0) = 0

set\_tid\_address(0x7fb522922690) = 45891

set\_robust\_list(0x7fb5229226a0, 24) = 0

rseq(0x7fb522922d60, 0x20, 0, 0x53053053) = 0

mprotect(0x7fb522c20000, 16384, PROT\_READ) = 0

mprotect(0x7fb522a08000, 4096, PROT\_READ) = 0

mprotect(0x7fb522c51000, 4096, PROT\_READ) = 0

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fb52291f000

mprotect(0x7fb522e6e000, 45056, PROT\_READ) = 0

mprotect(0x5564585c0000, 4096, PROT\_READ) = 0

mprotect(0x7fb522ebe000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, {rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY}) = 0

munmap(0x7fb522e7f000, 18463) = 0

getrandom("\xd4\x7d\x12\xbe\x41\xc5\xac\x2a", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x556459f65000

brk(0x556459f86000) = 0x556459f86000

futex(0x7fb522e7c77c, FUTEX\_WAKE\_PRIVATE, 2147483647) = 0

mmap(NULL, 135168, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fb5228fe000

mmap(NULL, 135168, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fb5228dd000

newfstatat(1, "", {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0x3), ...}, AT\_EMPTY\_PATH) = 0

write(1, "Starting sorting array with leng"..., 42Starting sorting array with length: 32768

) = 42

write(1, "Max threads: 4\n", 15Max threads: 4

) = 15

rt\_sigaction(SIGRT\_1, {sa\_handler=0x7fb522a9b870, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_ONSTACK|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7fb522a4c520}, NULL, 8) = 0

rt\_sigprocmask(SIG\_UNBLOCK, [RTMIN RT\_1], NULL, 8) = 0

mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0) = 0x7fb5220dc000

mprotect(0x7fb5220dd000, 8388608, PROT\_READ|PROT\_WRITE) = 0

rt\_sigprocmask(SIG\_BLOCK, ~[], [QUIT], 8) = 0

clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7fb5228dc910, parent\_tid=0x7fb5228dc910, exit\_signal=0, stack=0x7fb5220dc000, stack\_size=0x7fff00, tls=0x7fb5228dc640}strace: Process 45892 attached

=> {parent\_tid=[45892]}, 88) = 45892

[pid 45892] rseq(0x7fb5228dcfe0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 45891] rt\_sigprocmask(SIG\_SETMASK, [QUIT], <unfinished ...>

[pid 45892] <... rseq resumed>) = 0

[pid 45891] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 45892] set\_robust\_list(0x7fb5228dc920, 24 <unfinished ...>

[pid 45891] mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0 <unfinished ...>

[pid 45892] <... set\_robust\_list resumed>) = 0

[pid 45891] <... mmap resumed>) = 0x7fb5218db000

[pid 45892] rt\_sigprocmask(SIG\_SETMASK, [QUIT], <unfinished ...>

[pid 45891] mprotect(0x7fb5218dc000, 8388608, PROT\_READ|PROT\_WRITE <unfinished ...>

[pid 45892] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 45891] <... mprotect resumed>) = 0

[pid 45892] mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0 <unfinished ...>

[pid 45891] rt\_sigprocmask(SIG\_BLOCK, ~[], <unfinished ...>

[pid 45892] <... mmap resumed>) = 0x7fb5210da000

[pid 45891] <... rt\_sigprocmask resumed>[QUIT], 8) = 0

[pid 45892] mprotect(0x7fb5210db000, 8388608, PROT\_READ|PROT\_WRITE <unfinished ...>

[pid 45891] clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7fb5220db910, parent\_tid=0x7fb5220db910, exit\_signal=0, stack=0x7fb5218db000, stack\_size=0x7fff00, tls=0x7fb5220db640} <unfinished ...>

[pid 45892] <... mprotect resumed>) = 0

strace: Process 45893 attached

[pid 45892] mmap(NULL, 134217728, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_NORESERVE, -1, 0 <unfinished ...>

[pid 45891] <... clone3 resumed> => {parent\_tid=[45893]}, 88) = 45893

[pid 45893] rseq(0x7fb5220dbfe0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 45891] rt\_sigprocmask(SIG\_SETMASK, [QUIT], <unfinished ...>

[pid 45892] <... mmap resumed>) = 0x7fb5190da000

[pid 45891] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 45893] <... rseq resumed>) = 0

[pid 45892] munmap(0x7fb5190da000, 49438720 <unfinished ...>

[pid 45893] set\_robust\_list(0x7fb5220db920, 24 <unfinished ...>

[pid 45892] <... munmap resumed>) = 0

[pid 45893] <... set\_robust\_list resumed>) = 0

[pid 45892] munmap(0x7fb520000000, 17670144 <unfinished ...>

[pid 45893] rt\_sigprocmask(SIG\_SETMASK, [QUIT], <unfinished ...>

[pid 45892] <... munmap resumed>) = 0

[pid 45893] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 45892] mprotect(0x7fb51c000000, 135168, PROT\_READ|PROT\_WRITE) = 0

[pid 45892] rt\_sigprocmask(SIG\_BLOCK, ~[], [QUIT], 8) = 0

[pid 45892] clone3({flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7fb5218da910, parent\_tid=0x7fb5218da910, exit\_signal=0, stack=0x7fb5210da000, stack\_size=0x7fff00, tls=0x7fb5218da640}strace: Process 45894 attached

<unfinished ...>

[pid 45894] rseq(0x7fb5218dafe0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 45892] <... clone3 resumed> => {parent\_tid=[45894]}, 88) = 45894

[pid 45894] <... rseq resumed>) = 0

[pid 45892] rt\_sigprocmask(SIG\_SETMASK, [QUIT], <unfinished ...>

[pid 45894] set\_robust\_list(0x7fb5218da920, 24 <unfinished ...>

[pid 45892] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 45894] <... set\_robust\_list resumed>) = 0

[pid 45894] rt\_sigprocmask(SIG\_SETMASK, [QUIT], NULL, 8) = 0

[pid 45891] futex(0x7fb5220db910, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 45893, NULL, FUTEX\_BITSET\_MATCH\_ANY <unfinished ...>

[pid 45893] rt\_sigprocmask(SIG\_BLOCK, ~[RT\_1], NULL, 8) = 0

[pid 45893] madvise(0x7fb5218db000, 8368128, MADV\_DONTNEED <unfinished ...>

[pid 45892] futex(0x7fb5218da910, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 45894, NULL, FUTEX\_BITSET\_MATCH\_ANY <unfinished ...>

[pid 45893] <... madvise resumed>) = 0

[pid 45893] exit(0) = ?

[pid 45891] <... futex resumed>) = 0

[pid 45893] +++ exited with 0 +++

[pid 45894] rt\_sigprocmask(SIG\_BLOCK, ~[RT\_1], NULL, 8) = 0

[pid 45894] madvise(0x7fb5210da000, 8368128, MADV\_DONTNEED) = 0

[pid 45894] exit(0) = ?

[pid 45894] +++ exited with 0 +++

[pid 45892] <... futex resumed>) = 0

[pid 45891] futex(0x7fb5228dc910, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 45892, NULL, FUTEX\_BITSET\_MATCH\_ANY <unfinished ...>

[pid 45892] rt\_sigprocmask(SIG\_BLOCK, ~[RT\_1], NULL, 8) = 0

[pid 45892] madvise(0x7fb5220dc000, 8368128, MADV\_DONTNEED) = 0

[pid 45892] exit(0) = ?

[pid 45891] <... futex resumed>) = 0

[pid 45892] +++ exited with 0 +++

write(1, "Time taken: 0.0126096 seconds\n", 30Time taken: 0.0126096 seconds

) = 30

write(1, "Sorting successful!\n", 20Sorting successful!

) = 20

munmap(0x7fb5228dd000, 135168) = 0

munmap(0x7fb5228fe000, 135168) = 0

exit\_group(0) = ?

+++ exited with 0 +++