Algorithms in Secondary Memory

Amritansh Sharma (000473628)

Pratham Solanki (000474971)

Shabana Salmaan (000471119)

06.01.2019

Table of Contents

Introduction and Environment....3
- Project Goals:....3
- Machine Environment:....3
  - Hardware:....3
  - Software:....3
- Test Data:....3
- Benchmark Metric:....4

Observations on Streams....4
- Notations:....4
- Data Stream:....4
- Buffered Stream:....6
- Buffered Stream with fixed buffer size:....7
- Memory Mapping:....11
- Conclusion:....13
  - syscalls vs libc....13
  - Laying foundations for the next task and defending our approach....13

External multi-way merge sort....14
- Implementation details....14
  - Notations:....14
  - Sort Phase:....14
  - Merge Phase:....14
- Expected behavior:....15
- Experimental observations:....15
- Discussion:....15

Overall conclusion:....16

References:....16

Raw Data....17
- Data Stream....17
  - Reading:....17
  - Writing:....17
- Buffered Stream....18
  - Reading:....18
  - Writing:....18
- Buffered Stream with fixed buffer size....19
  - Reading:....19
  - Writing:....23
- Memory Mapping:....27
  - Reading:....27
  - Writing:....31
- In-Memory Sort:....35
- External Merge Sort:....36

Introduction and Environment

Project Goals:

- Get real-world experience with the performance of external-memory algorithms.
- Explore several different ways to read data from, and write data to secondary memory:
  - Data Stream
  - Buffered Stream
  - Buffered Stream with fixed buffer size
  - Memory Mapping
- Implement an external-memory merge-sort algorithm and examine its performance under different parameters.

Machine Environment:

Hardware:

- Machine: `Lenovo ThinkPad T410`
- Processor: `Dual-Core Intel® Core™ i5-520M (2.40GHz, 3MB SmartCache)`
- Memory: `4GB DDR3 RAM (2GB + 2GB)`

Software:

- Operating System: `elementary OS 0.4.1 Loki (64 bit)`
- Programming Language: `C++`
- Compiler: `g++ (GNU project C and C++ compiler)`
- External Libraries:
  - `Filesystem library (-lstdc++fs)`
  - `C++11 Standard Library Extension (-std=c++11)`

Test Data:

- Each element in the test data is generated using the `rand()` function from the `stdlib` header. A single loop is used to write randomly generated elements to the test file.

- Our smallest test file consists of 10,000 elements and the biggest 1 billion. To be precise, the number of elements while benchmarking are {10000, 100000, 1000000, 2000000, 3000000, 4000000, 5000000, 6000000, 7000000, 8000000, 9000000, 10000000, 100000000, 1000000000}
- All test files, intermediate files and output files are stored as binary files.

Benchmark Metric:

- Execution time is calculated using `std::clock()` function from the `ctime` library. It returns the approximate processor time used by the process since the beginning of an implementation-defined era related to the program's execution.
- We take the difference between two values returned by different calls to `std::clock` and to convert result value to seconds we divide it by `CLOCKS_PER_SEC`.
- Please note that as we are calculating the execution time from the approximate processor time, our reported numbers do not represent the wall clock time. Also, our reported time indicates the time it takes to bring data from disk to memory and does not include the time taken to move the data in-memory i.e. to console etc.
- The precision of `std::clock()` function is around 40 milliseconds.

Observations on Streams

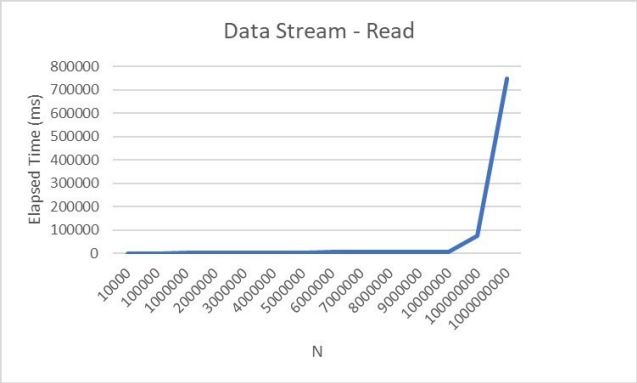
Notations:

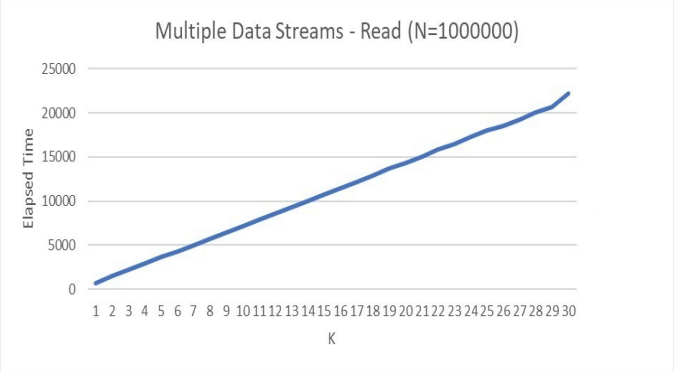
- `N` = number of elements to be read/written
- `B` = Buffer size (number of elements)
- `k` = number of streams
- All cost formulas indicate the number of disk I/O for a single operation (read/write)

Data Stream:

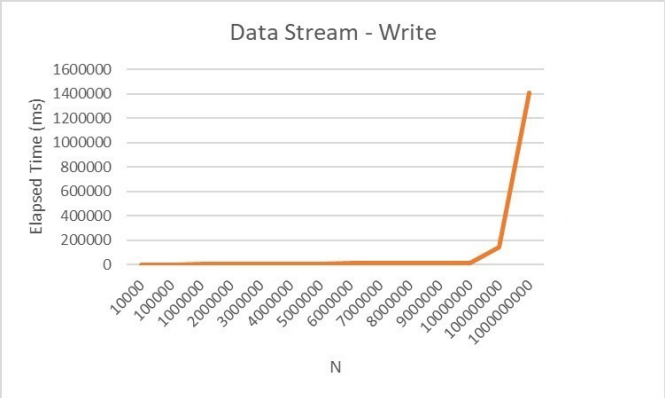
- Motivation: The most basic approach to file I/O is to read/write one element at a time until `eof` using the `syscalls`.
- Headers used:
  - `unistd.h`
  - `fcntl.h`

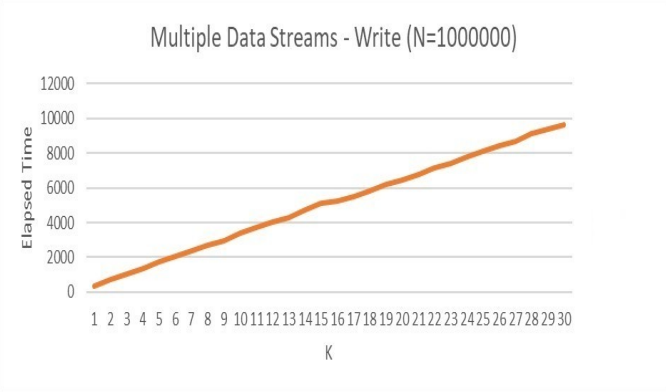
- Data is read/written one element at a time using the `read` and `write` system calls. These are low level, unbuffered and unformatted read and write. They make a direct system call on `UNIX`. The disk is therefore accessed as many times as there are elements to be read/written.
- Cost: $N * k$
- Expected Behavior: The execution time is bound to increase with increasing `N` or `k`.
- Experimental observations:
  - Reading:





  - Writing:





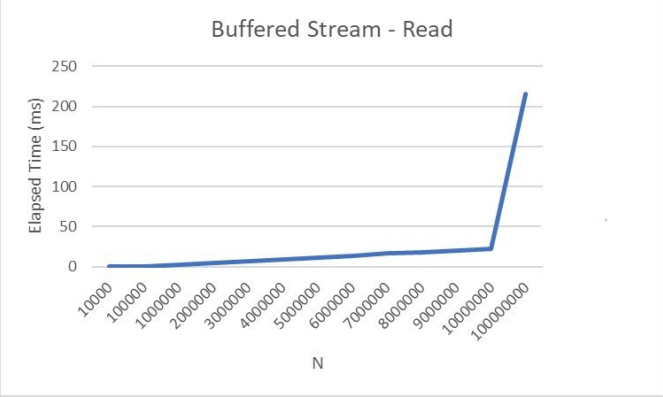
- Discussion:
  - We benchmark our Data Stream implementation for `N` starting from 10000 to 1 billion.
  - As we expected the execution time increases as the file size gets bigger. We also observe that reading is almost two times faster than writing.
  - We also observe that the execution time increases steadily as we increase the number of streams but we notice something peculiar. We observe that for multiple streams writing is

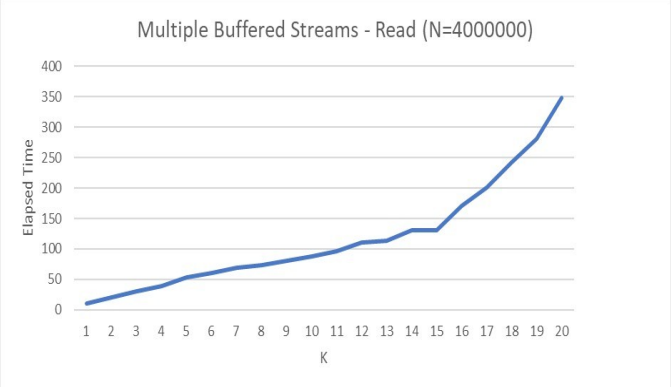
faster than reading which is not the case for a single stream. This is not expected because both hard disks and solid state media read faster than they write as reading only needs to traverse the directory tree and block list down to the data, then read the data, whereas writing needs to perform the same traversal, then write the data, then update some metadata. We research on this topic and find a possible answer:

- Linux keeps a free block list in memory, so while writing finding the next free block is very fast. On the other hand for reading we may have to first look at the metadata, rotate the disk to the respective block and fetch the data.

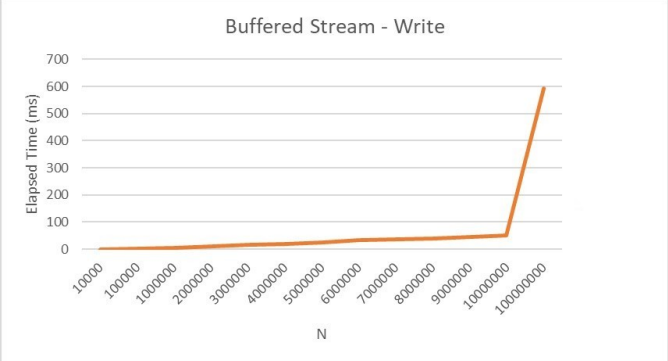
Buffered Stream:

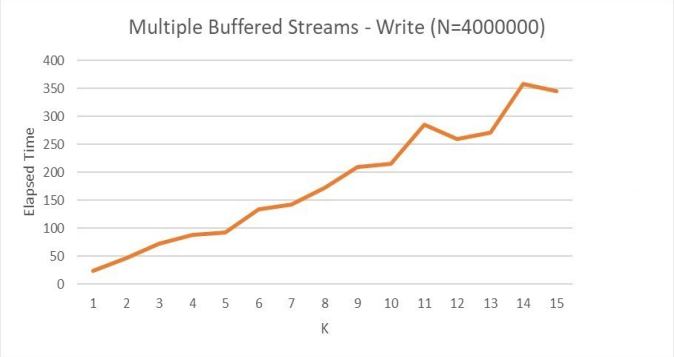
- Motivation: The `syscalls` drop to the kernel for each element and hence we have a performance penalty. Thus we resort to using the buffered read and write provided by `The C standard library (libc)`.
- Headers used:
  - `stdio.h`
- Data is read/written using the `fread` and `fwrite` functions which implement their own buffering mechanism. The default buffer size that `C++` uses is usually 8192 bytes which is equivalent to 2048 elements, but may vary depending upon: file system block size, CPU cache size and cache latency. We will assume the default value of `B` to be 8192 bytes. A disk access is triggered each time the buffer needs to be read/written.
- Cost: $\lceil (N/B) \rceil * k$
- Expected Behavior: The execution time is bound to increase with increasing `N` or `k`. We do not speculate the behavior with respect to `B` as it is constant.
- Experimental observations:
  - Reading:





- Writing:





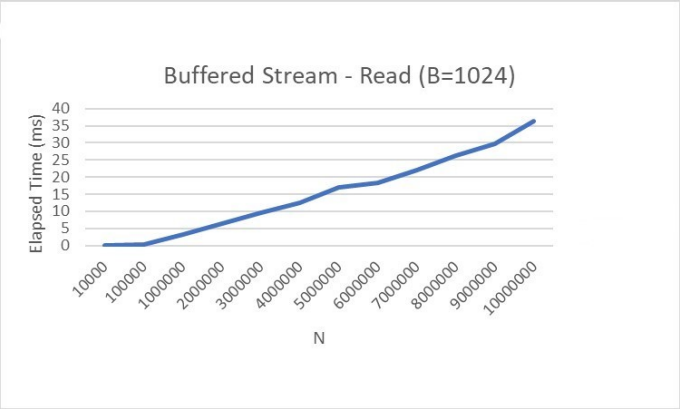
- Discussion:
  - We benchmark our Buffered Stream implementation for `N` starting from 10000 to 1 billion. However, while testing for `N` = 1 billion our program was killed by the OS. We researched as to why this happened and found out the program is probably killed by kernel's `oom killer`. It is the job of the `Linux oom killer` to sacrifice one or more processes in order to free up memory for the system when all else fails.
  - The execution time increases as the file size gets bigger and just as the case with Data Stream, we observe that reading is significantly faster than writing.
  - While benchmarking with multiple streams, our program was killed when `K` exceeded 20 while reading and 15 while writing. But the data we gathered was enough to gain insights. We observe that the execution time increases as we increase the number of streams. However, unlike with Data Stream, here we observe that reading multiple streams is faster than writing. The only plausible reasoning for this is caching.
    - As this is a buffered stream (unlike Data Stream) OS tries to anticipate reads and caches data accordingly. And Reading data that's in cache is very fast.

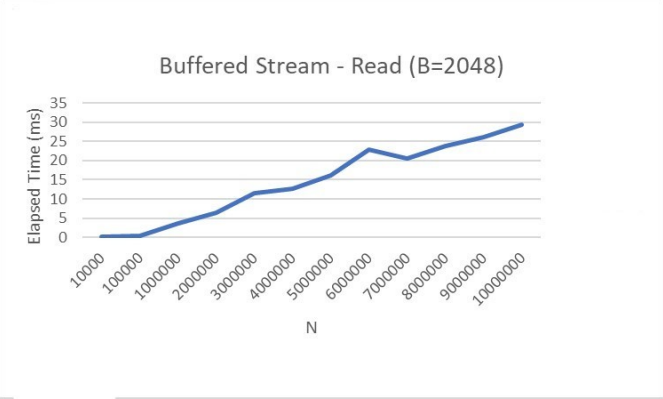
Buffered Stream with fixed buffer size:

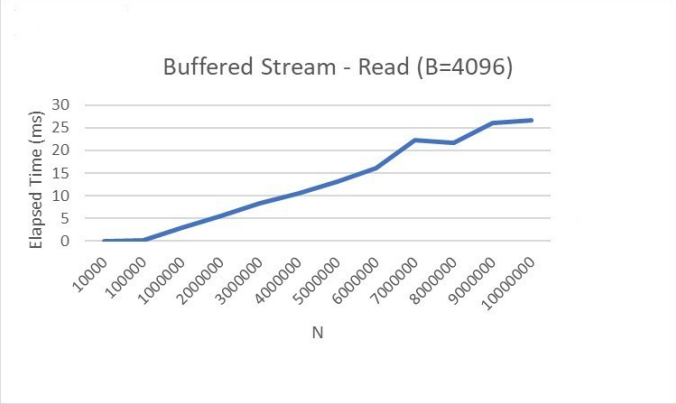
- Motivation: Keeping the concept same as in the previous implementation, we now want to try different buffer sizes and observe how the performance changes.
- Headers used:
  - `stdio.h`
- The differed buffer sizes `(B)` that we experiment on are {1024, 2048, 4096, 8192, 16384, 32768, 65536, 131072, 262144}. However, in its implementation `C++` uses the buffer size of

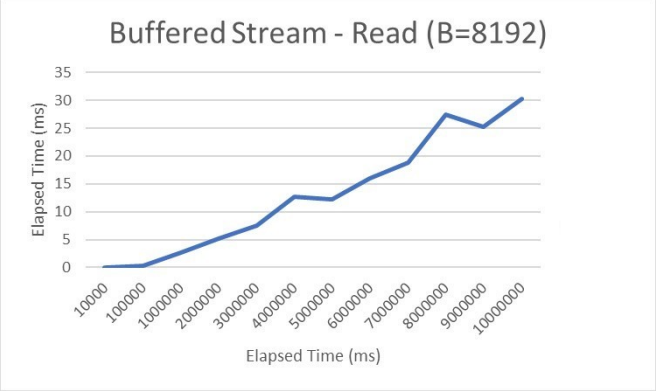
8192 bytes (2048 elements in our case). This gives the granularity independent of what we use in our code. The cost formula remains the same as in the previous implementation.

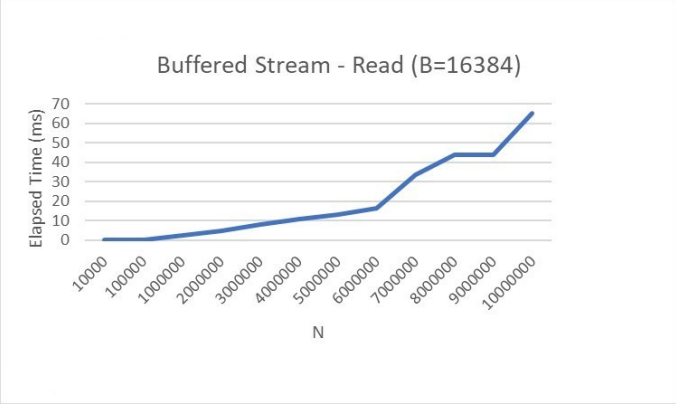
- Cost: $\lceil (N/B) \rceil * k$
- Expected Behavior: The execution time is bound to increase with increasing `N` or `k` and decrease with increasing `B`. However, we expect the performance to be best at the default value of `B (2048)` and expect a decline in performance as we move either way from the default value.
- Experimental observations:
  - Reading:

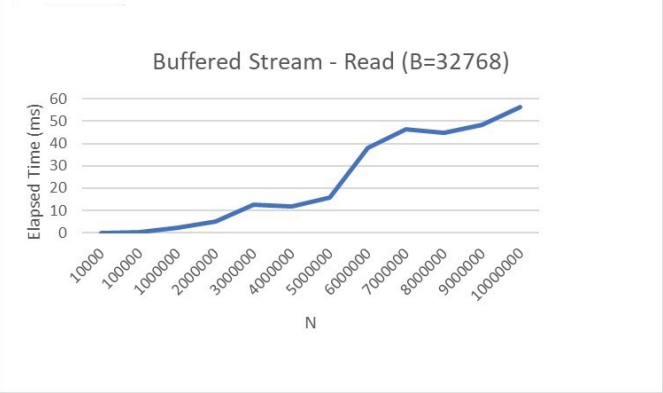


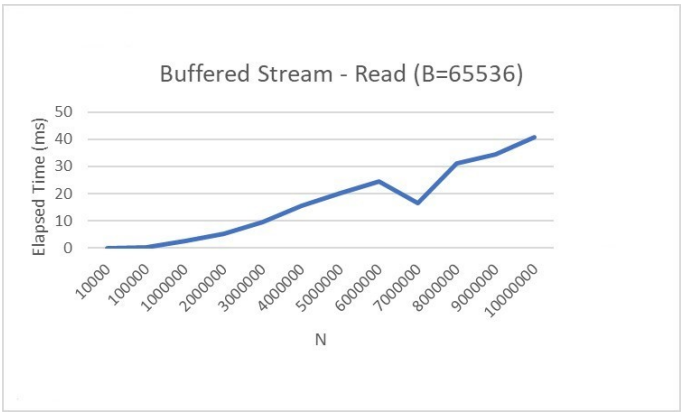


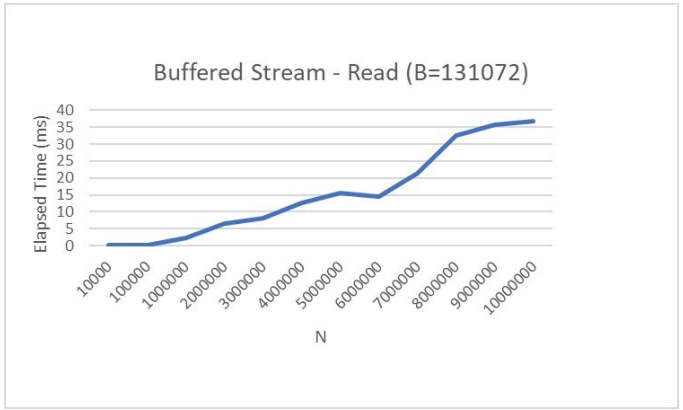


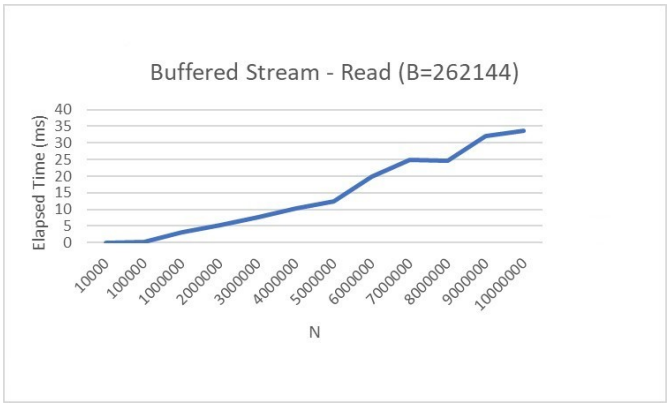


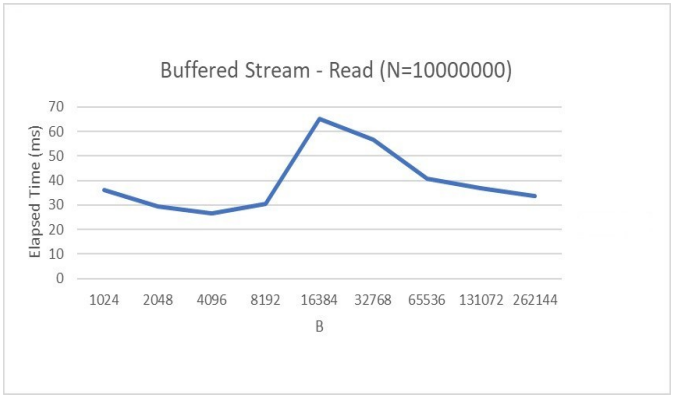




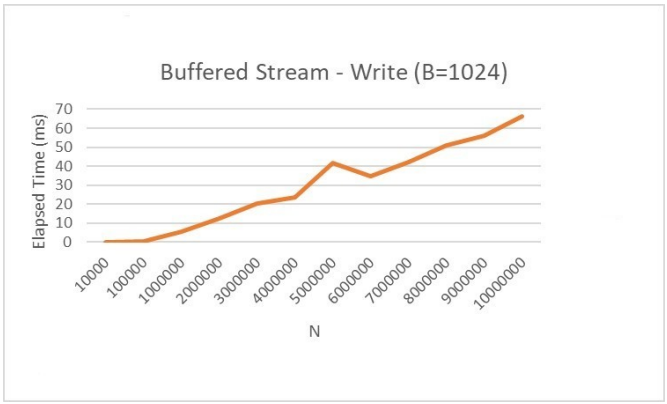

Buffered Stream - Read (B=65536)
Elapsed Time (ms)
0
10
20
30
40
50
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N

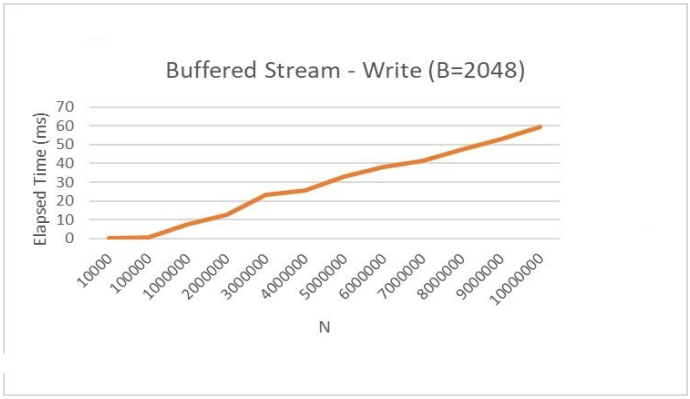

Buffered Stream - Read (B=131072)
Elapsed Time (ms)
0
5
10
15
20
25
30
35
40
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N

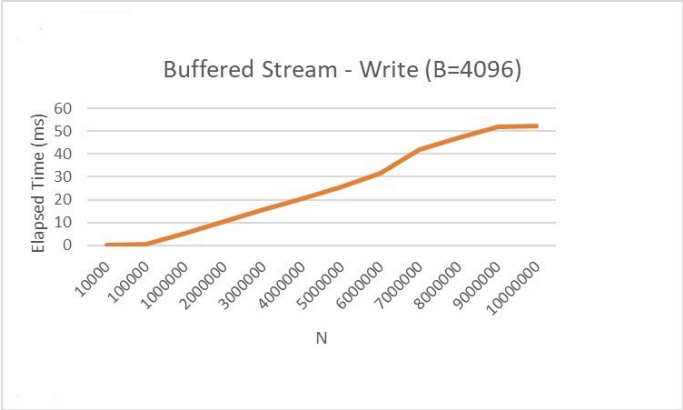

Buffered Stream - Read (B=262144)
Elapsed Time (ms)
0
5
10
15
20
25
30
35
40
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N

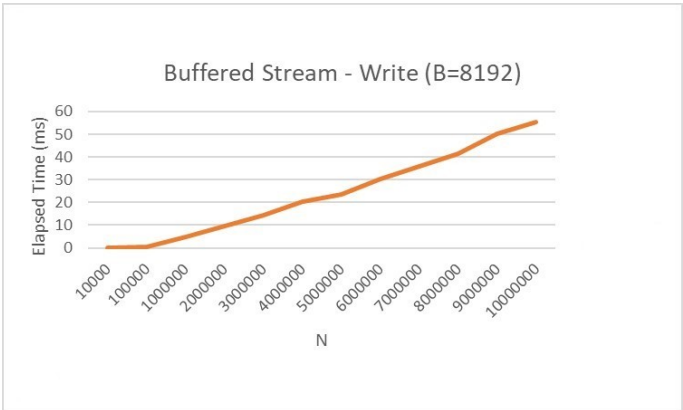

Buffered Stream - Read (N=10000000)
Elapsed Time (ms)
0
10
20
30
40
50
60
70
1024
2048
4096
8192
16384
32768
65536
131072
262144
B

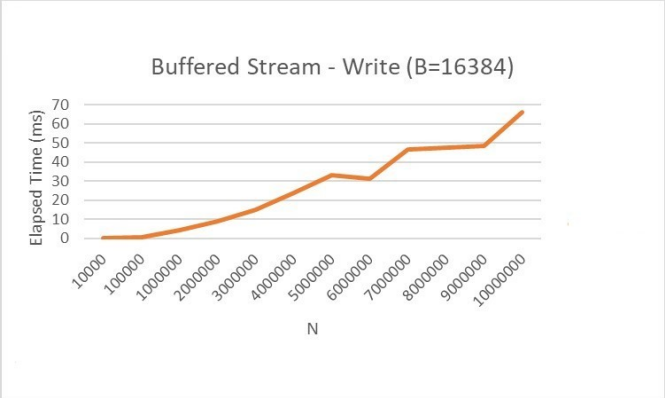
- Writing:

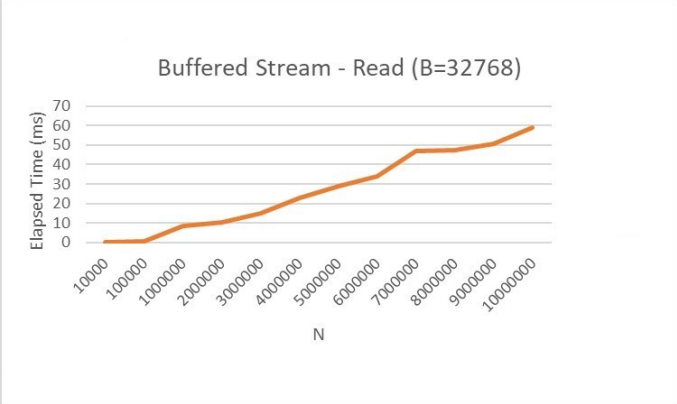

Buffered Stream - Write (B=1024)
Elapsed Time (ms)
0
10
20
30
40
50
60
70
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N

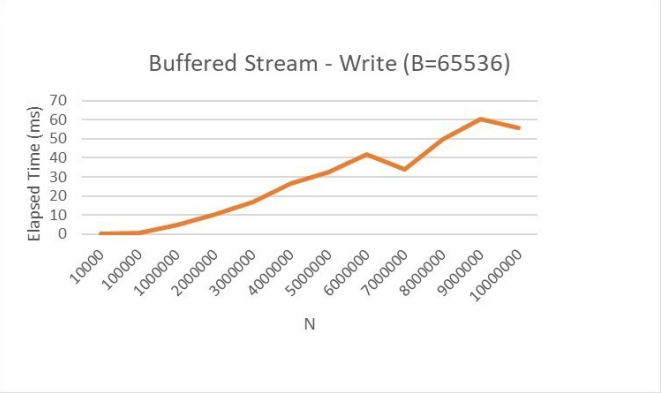

Buffered Stream - Write (B=2048)
Elapsed Time (ms)
0
10
20
30
40
50
60
70
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N

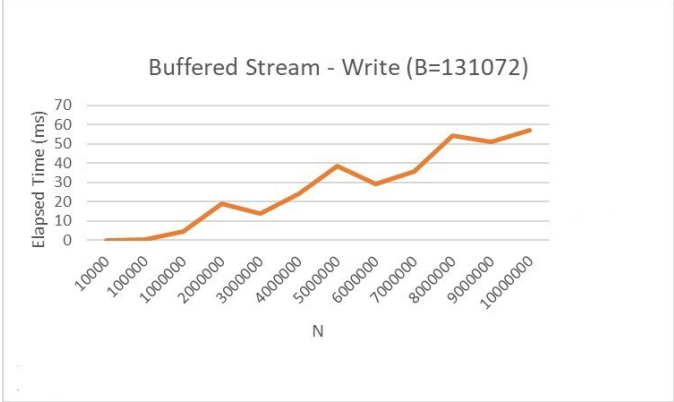

Buffered Stream - Write (B=4096)
Elapsed Time (ms)
60
50
40
30
20
10
0
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N

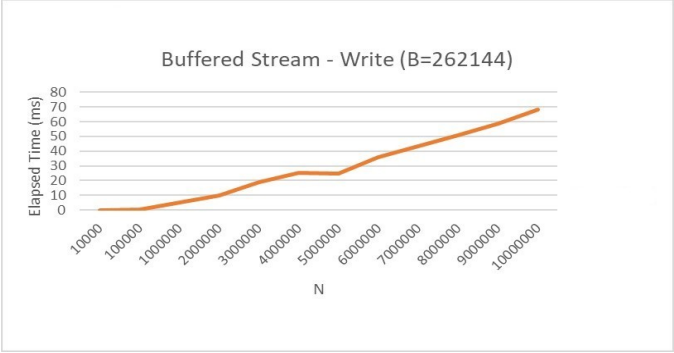

Buffered Stream - Write (B=8192)
Elapsed Time (ms)
60
50
40
30
20
10
0
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N

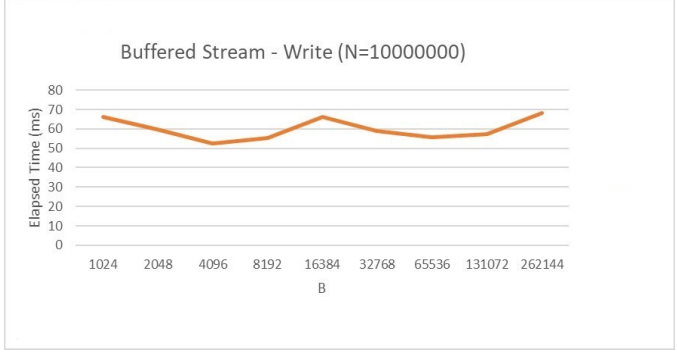

Buffered Stream - Write (B=16384)
Elapsed Time (ms)
70
60
50
40
30
20
10
0
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N


Buffered Stream - Read (B=32768)
Elapsed Time (ms)
70
60
50
40
30
20
10
0
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N


Buffered Stream - Write (B=65536)
Elapsed Time (ms)
70
60
50
40
30
20
10
0
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N


Buffered Stream - Write (B=131072)
Elapsed Time (ms)
70
60
50
40
30
20
10
0
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N


Buffered Stream - Write (B=262144)
Elapsed Time (ms)
80
70
60
50
40
30
20
10
0
10000
100000
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
N

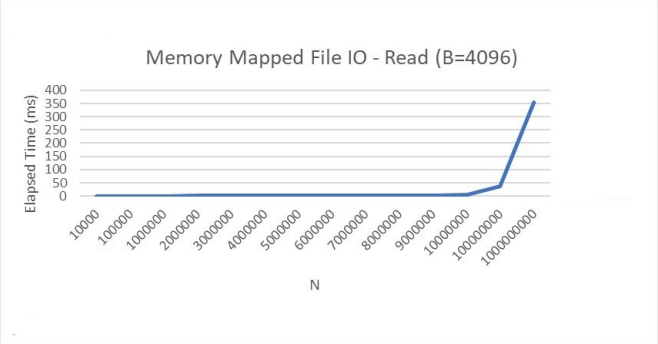

Buffered Stream - Write (N=10000000)
Elapsed Time (ms)
80
70
60
50
40
30
20
10
0
1024
2048
4096
8192
16384
32768
65536
131072
262144
B

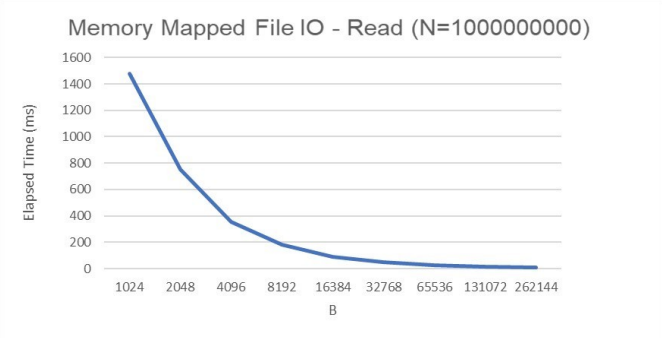
- Discussion:
  - We benchmark our Buffered Stream implementation for `N x B` combinations of parameters. However, while testing for N greater than 10 million our program was killed by the kernel's `oom killer`.
  - The execution time increases as the file size gets bigger and unlike the previous mechanisms there is no significant difference in performance between reading and writing.
  - However, the plots with `N` constant and `B` variable are interesting. We observe that for both reading and writing the best performance is achieved when `B` is 4096 which is not the default buffer size (2048) that `C++` uses.

Memory Mapping:

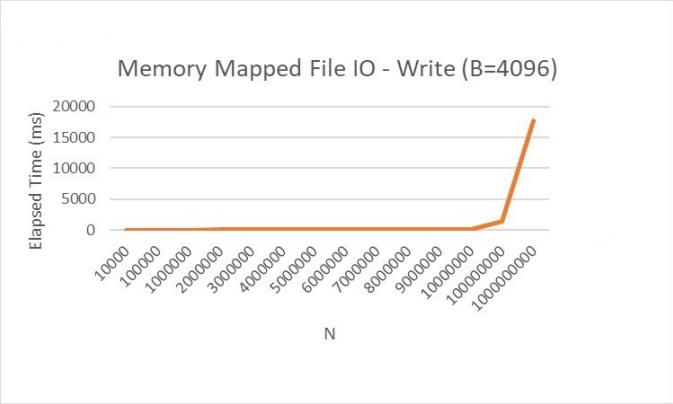
- Motivation: A memory-mapped file is a segment of virtual memory that has been assigned a direct byte-for-byte correlation with some portion of a file. This correlation between the file and the memory space permits applications to treat the mapped portion as if it were primary memory. This reduces the I/O time significantly. [1]
- Concept:
  - Why is accessing memory mapped files fast?
    - The OS kernel routines for I/O, like read or write calls, are still just functions. These functions are written to copy data to/from userspace buffer to a kernel space structure, and then to a device. When we consider that there is a user buffer, an I/O library buffer, a kernel buffer, then a file, the data may potentially go through 3 copies to get between the program and the disk. The I/O routines also have to be robust, and lastly, the syscalls themselves impose a latency (trapping to kernel, context switch, waking process up again). [2]
    - When we memory map a file, we are skipping right through much of that, eliminating buffer copies. By effectively treating the file like a big virtual array, we enable random access without going through the syscall overhead, so we lower the latency per I/O. [2]
- Headers used:
  - `sys/mman.h`
- Read and write is performed by mapping and unmapping a `B` element portion of the file into internal memory through memory mapping. To read/write outside of the mapped portion, the currently mapped portion is unmapped and the next `B` element portion of the file is mapped.
- Cost: $\lceil (N/B) \rceil * k$

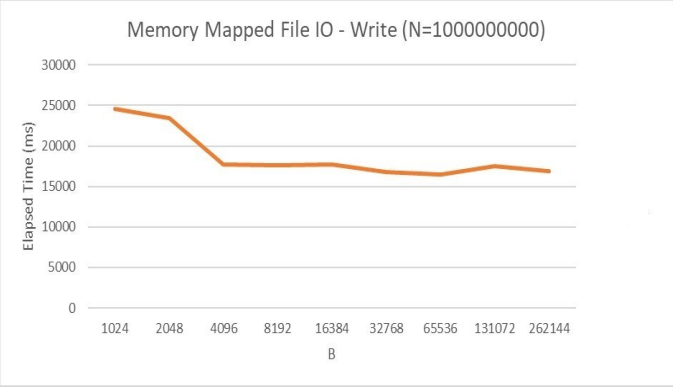
- Expected Behavior: The time taken by the OS to map/unmap a buffer with the respective portion on disk will be a dominating factor in the overall execution time. The bigger the file, the more number of times we will have to map/unmap smaller portions of it. While we increase the buffer size we will have lesser portions to map/unmap. Hence, we expect the execution time to increase as we increase `N` or `k` and decrease as we increase `B`.
- Experimental observations:
  - Reading:





  - Writing:





- Discussion:
  - We benchmark our Memory Mapping implementation for `N x B` combinations of parameters (We have just shown the plots for `B` = 4096 as all other plots follow an identical trend).
  - As we expected the execution time increases as the file size gets bigger. But we observe a significant difference in performance between reading and writing.
  - We observe that for reading, the performance keeps increasing steadily as we increase the buffer size (as clearly evident from the plot), however for writing the respective plot is

unclear. Even though the performance eventually increases with large `B` it does not show a steady decline in elapsed time as we increase `B`.

Conclusion:

- **`syscalls vs libc`**
  - First we address the comparison between an unbuffered stream and a buffered stream. The entirety of benchmark tests are biased towards buffered streams because we are testing sequential read and write. `UNIX System Calls` are extremely powerful but are not meant for sequential file I/O where we read/write in chunks.
  - Non-buffered read/write (using `syscalls`) are effective when we need to read/write a byte of data when it is ready, while buffered streams (using `libc`) are for reading/writing continuous chunks of data.

- **Laying foundations for the next task and defending our approach**
  - External sorting is a term for a class of sorting algorithms that can handle massive amounts of data. External sorting is required when the data being sorted do not fit into the main memory of a computing device (usually RAM) and instead they must reside in the slower external memory (usually a hard drive). [3]
  - It typically uses a two phase sort-merge strategy:
    - In the sorting phase, chunks of data small enough to fit in main memory are read, sorted, and written out to a temporary file.
    - In the merge phase, the sorted sub-files are combined into a single larger file.
  - For our implementation of the external merge sort algorithm we decided upon a hybrid approach:
    - For the sort phase we decided to read and write the required chunks of data using the `fread` and `fwrite` functions from `The C standard library`. Our main motivation behind this is that this way we can compute the entire sort phase just using the buffer size specified. We will iteratively read chunks of data (corresponding to the specified buffer size), sort this buffer in-memory and then dump it onto disk. Note that this is not possible using memory mapping because we cannot sort the same buffer that we read as it is mapped to the file and any changes done will change the file. We will first have to copy the contents of the mapped buffer to another buffer, then sort it and then write it to disk. This will require twice the specified buffer size.

    - For the merge phase we decided to use the `UNIX System Calls`. The concept of an unbuffered stream fits perfectly in this scenario. Again, one of our main motivations behind this decision is that this is the only way to compute the entire merge phase using the buffer size that equals to just the number of sublists that we are merging. We only want a single priority queue that holds one element each from each of the sublists. We write the minimum element directly to the output file and read the next element from the respective sublist. If we were using a buffered stream mechanism or memory mapping instead, we would have to have had another buffer to accumulate the minimum elements and then when it becomes full write it to disk.
- We believe that our proposed external merge sort implementation based on the above specified hybrid approach fits very well.

External multi-way merge sort

Implementation details

Notations:

- `N` = number of elements to be read/written
- `M` = Buffer size (number of elements)
- `d` = number of sublists to merge
- All cost formulas indicate the number of disk I/O

Sort Phase:

- `M` elements are read at a time using `fread` and sorted using `sort()` function from the `algorithm` library. Cost of in-memory sorting is $M*\log_2(M)$. The sorted list is then stored on disk using `fwrite`. In the end we have $\lceil(N/M)\rceil$ sorted files.
- Buffer size: `M`
- Cost: We read and write a buffer of at-most `M` elements $\lceil(N/M)\rceil$ times. Hence, the cost is $\lceil(N/M)\rceil*2$.

Merge Phase:

- We iteratively merge `d` sublists until we only have one list. To do so, we create a `priority queue` of `d` nodes. A node is a structure that contains an integer value and a file reference for that integer. We initialize the priority queue with the first elements from each of the `d` sublists

and their references. We provide a custom `comparator` function that is called each time a node is pushed to the priority queue. With the help of this function the queue is sorted on the element attribute each time there is a push. We then write the minimum element to the merged file and read the next element from the file which it referenced. When all `d` files are empty we move to the next `d` files. This is continued until we have just a single merged file.

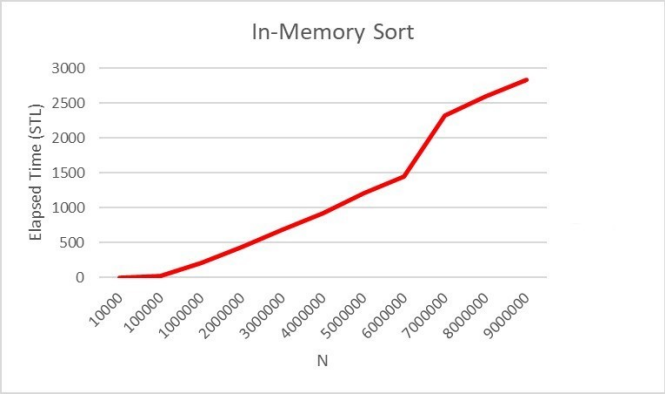
- Buffer size: `d`
- Cost: Let `x` be the number of passes. We merge `d` sublists at a time until we have a single merged list. Hence, $\lceil (N/M) \rceil / d^x = 1$. Thus the number of passes calculate to $\log_d \lceil (N/M) \rceil$.

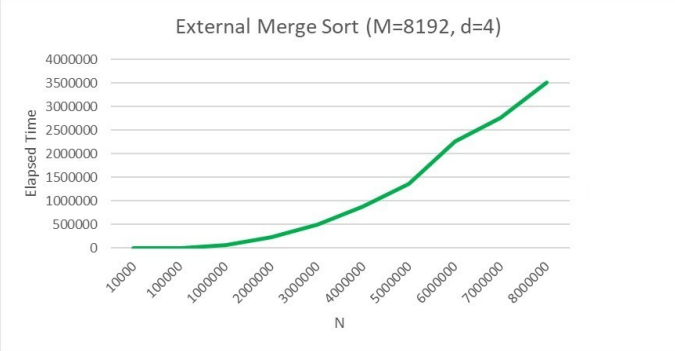
  We read and write `N` elements for each pass. Hence the cost is $\log_d \lceil (N/M) \rceil * N * 2$.

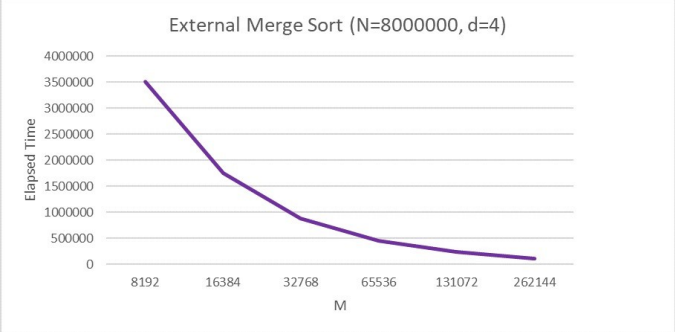
Expected behavior:

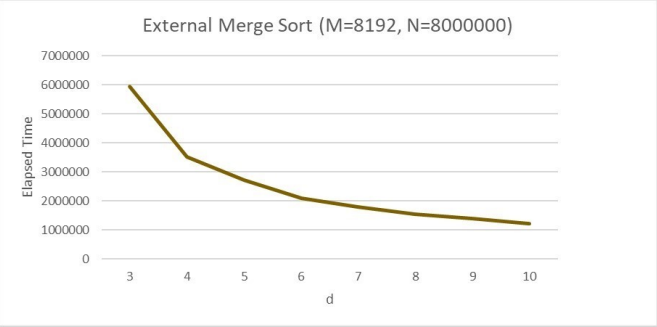
- The execution time is bound to increase with increasing `N` and decrease with increasing `M` or `d`.

Experimental observations:









Discussion:

- We benchmark our External Merge-Sort implementation for `N x M x d` combinations of parameters (We have shown limited plots but the raw data is available at the end of the

document). Our program was `killed` whenever the number of sublists exceeded 1000 so benchmark results are available for those combinations of `N` and `M` for which the number of sublists is not greater than 1000.

- First we benchmark how the in-memory sort performs. We observe that the execution time increases steadily as we increase `N`.
- Next we benchmark the performance of our external merge-sort implementation under various parameters. We clearly understand from the plots that the execution time increases as we increase `N` and that the performance increases steadily as we increase `M` or `d`.
- As expected we notice that the in-memory sort outperforms our external merge-sort implementation significantly.

Overall conclusion:

- We first laid a decent foundation by implementing various I/O mechanisms and benchmarking their respective performances.
  - We started with an unbuffered mechanism and made use of the `UNIX syscalls` which is as low level as you can go. Working with `syscalls` was the most interesting for us because it felt like we were in control and were less dependent on the programming language.
  - Then, we moved to buffered streams. First we explored functions provided by `libc`. We found them to be black-boxes where the programming language implemented its own mechanism. We then explored the `mman` header and worked with memory-mapped files. Memory mapping was a completely new topic for us and we had to dig the internet to properly understand the concept.
- After exploring all four mechanisms, we brainstormed the external merge-sort task and deviced a hybrid approach using multiple mechanisms to implement the task.
- We also got to learn about the `oom killer` and thanks to this project we stumbled upon the `Linux-Kernel Email Archives` where ***Linus Torvalds*** himself talks about `mmap()`[5]

References:

[1] Memory-mapped file. Available at: https://en.wikipedia.org/wiki/Memory-mapped_file

[2] Why is reading from a memory mapped file so fast? Available at: https://stackoverflow.com/questions/26456195/why-is-reading-from-a-memory-mapped-file-so-fast?answertab=oldest#tab-top

[3] External sorting. Available at https://en.wikipedia.org/wiki/External_sorting

[4] mmap for writing sequential log file for speed? Available at: https://stackoverflow.com/questions/35891525/mmap-for-writing-sequential-log-file-for-speed?answertab=oldest#tab-top

[5] Re: mmap/mlock performance versus read. Available at: http://lkml.iu.edu/hypermail/linux/kernel/0004.0/0728.html

Raw Data

Data Stream

Reading:

N	Elapsed Time (ms)
10,000	8.03
100,000	71.23
1,000,000	760.07
2,000,000	1,456.23
3,000,000	2,168.69
4,000,000	2,887.12
5,000,000	3,591.62
6,000,000	4,323.79
7,000,000	5,064.92
8,000,000	5,806.47
9,000,000	6,509.29
10,000,000	7,252.52
100,000,000	74,064.10
1,000,000,000	749,396.00

Writing:

N	Elapsed Time (ms)
10,000	15.72
100,000	136.82
1,000,000	1,438.36

2,000,000	2,808.45
3,000,000	4,213.75
4,000,000	5,569.16
5,000,000	6,945.97
6,000,000	8,313.51
7,000,000	9,694.74
8,000,000	11,194.80
9,000,000	12,551.20
10,000,000	13,934.80
100,000,000	140,915.00
1,000,000,000	1,407,150.00

Buffered Stream

Reading:

N	Elapsed Time (ms)
10,000	0.09
100,000	0.30
1,000,000	2.21
2,000,000	4.38
3,000,000	6.50
4,000,000	8.57
5,000,000	10.95
6,000,000	13.65
7,000,000	16.48
8,000,000	17.98
9,000,000	20.08
10,000,000	22.14
100,000,000	215.41
1,000,000,000	`killed`

Writing:

N	Elapsed Time (ms)

10,000	0.08
100,000	0.51
1,000,000	4.94
2,000,000	10.10
3,000,000	14.89
4,000,000	19.66
5,000,000	25.05
6,000,000	32.85
7,000,000	35.91
8,000,000	40.38
9,000,000	45.14
10,000,000	50.06
100,000,000	593.66
1,000,000,000	`killed`

Buffered Stream with fixed buffer size

Reading:

B	N	Elapsed Time (ms)
1,024	10,000	0.05
1,024	100,000	0.30
1,024	1,000,000	3.16
1,024	2,000,000	6.44
1,024	3,000,000	9.62
1,024	4,000,000	12.47
1,024	5,000,000	16.95
1,024	6,000,000	18.37
1,024	7,000,000	22.15
1,024	8,000,000	26.17
1,024	9,000,000	29.80
1,024	10,000,000	36.28
1,024	100,000,000	`killed`
1,024	1,000,000,000	`killed`

2,048	10,000	0.04
2,048	100,000	0.30
2,048	1,000,000	3.55
2,048	2,000,000	6.30
2,048	3,000,000	11.45
2,048	4,000,000	12.72
2,048	5,000,000	16.15
2,048	6,000,000	22.88
2,048	7,000,000	20.50
2,048	8,000,000	23.75
2,048	9,000,000	26.16
2,048	10,000,000	29.47
2,048	100,000,000	killed
2,048	1,000,000,000	killed
4,096	10,000	0.05
4,096	100,000	0.25
4,096	1,000,000	2.90
4,096	2,000,000	5.58
4,096	3,000,000	8.28
4,096	4,000,000	10.48
4,096	5,000,000	13.07
4,096	6,000,000	16.14
4,096	7,000,000	22.18
4,096	8,000,000	21.67
4,096	9,000,000	26.02
4,096	10,000,000	26.56
4,096	100,000,000	killed
4,096	1,000,000,000	killed
8,192	10,000	0.05
8,192	100,000	0.24
8,192	1,000,000	2.62
8,192	2,000,000	5.17

8,192	3,000,000	7.54
8,192	4,000,000	12.73
8,192	5,000,000	12.20
8,192	6,000,000	15.93
8,192	7,000,000	18.76
8,192	8,000,000	27.42
8,192	9,000,000	25.27
8,192	10,000,000	30.31
8,192	100,000,000	killed
8,192	1,000,000,000	killed
16,384	10,000	0.05
16,384	100,000	0.25
16,384	1,000,000	2.58
16,384	2,000,000	4.88
16,384	3,000,000	8.08
16,384	4,000,000	11.04
16,384	5,000,000	13.19
16,384	6,000,000	16.61
16,384	7,000,000	33.38
16,384	8,000,000	44.00
16,384	9,000,000	43.62
16,384	10,000,000	64.99
16,384	100,000,000	killed
16,384	1,000,000,000	killed
32,768	10,000	0.03
32,768	100,000	0.21
32,768	1,000,000	2.36
32,768	2,000,000	5.16
32,768	3,000,000	12.47
32,768	4,000,000	11.93
32,768	5,000,000	15.79
32,768	6,000,000	38.17

32,768	7,000,000	46.40
32,768	8,000,000	44.73
32,768	9,000,000	48.29
32,768	10,000,000	56.54
32,768	100,000,000	killed
32,768	1,000,000,000	killed
65,536	10,000	0.03
65,536	100,000	0.21
65,536	1,000,000	2.57
65,536	2,000,000	5.33
65,536	3,000,000	9.48
65,536	4,000,000	15.70
65,536	5,000,000	20.24
65,536	6,000,000	24.36
65,536	7,000,000	16.60
65,536	8,000,000	31.04
65,536	9,000,000	34.35
65,536	10,000,000	40.86
65,536	100,000,000	killed
65,536	1,000,000,000	killed
131,072	10,000	0.03
131,072	100,000	0.23
131,072	1,000,000	2.22
131,072	2,000,000	6.59
131,072	3,000,000	8.14
131,072	4,000,000	12.57
131,072	5,000,000	15.50
131,072	6,000,000	14.55
131,072	7,000,000	21.38
131,072	8,000,000	32.45
131,072	9,000,000	35.63
131,072	10,000,000	36.87

131,072	100,000,000	killed
131,072	1,000,000,000	killed
262,144	10,000	0.03
262,144	100,000	0.20
262,144	1,000,000	3.21
262,144	2,000,000	5.15
262,144	3,000,000	7.56
262,144	4,000,000	10.20
262,144	5,000,000	12.42
262,144	6,000,000	19.93
262,144	7,000,000	24.93
262,144	8,000,000	24.63
262,144	9,000,000	32.05
262,144	10,000,000	33.67
262,144	100,000,000	killed
262,144	1,000,000,000	killed

Writing:

B	N	Elapsed Time (ms)
1,024	10,000	0.07
1,024	100,000	0.51
1,024	1,000,000	5.66
1,024	2,000,000	12.24
1,024	3,000,000	20.07
1,024	4,000,000	23.63
1,024	5,000,000	41.79
1,024	6,000,000	34.79
1,024	7,000,000	41.96
1,024	8,000,000	50.77
1,024	9,000,000	55.94
1,024	10,000,000	66.30
1,024	100,000,000	killed
1,024	1,000,000,000	killed

2,048	10,000	0.06
2,048	100,000	0.57
2,048	1,000,000	7.51
2,048	2,000,000	12.75
2,048	3,000,000	23.08
2,048	4,000,000	25.74
2,048	5,000,000	33.22
2,048	6,000,000	38.16
2,048	7,000,000	41.40
2,048	8,000,000	47.28
2,048	9,000,000	52.88
2,048	10,000,000	59.62
2,048	100,000,000	killed
2,048	1,000,000,000	killed
4,096	10,000	0.07
4,096	100,000	0.46
4,096	1,000,000	5.22
4,096	2,000,000	10.66
4,096	3,000,000	15.71
4,096	4,000,000	20.49
4,096	5,000,000	25.69
4,096	6,000,000	31.56
4,096	7,000,000	41.86
4,096	8,000,000	47.20
4,096	9,000,000	51.69
4,096	10,000,000	52.39
4,096	100,000,000	killed
4,096	1,000,000,000	killed
8,192	10,000	0.05
8,192	100,000	0.41
8,192	1,000,000	4.74
8,192	2,000,000	9.56

8,192	3,000,000	14.34
8,192	4,000,000	20.15
8,192	5,000,000	23.67
8,192	6,000,000	30.21
8,192	7,000,000	35.68
8,192	8,000,000	41.53
8,192	9,000,000	50.20
8,192	10,000,000	55.26
8,192	100,000,000	killed
8,192	1,000,000,000	killed
16,384	10,000	0.05
16,384	100,000	0.41
16,384	1,000,000	4.40
16,384	2,000,000	9.19
16,384	3,000,000	15.02
16,384	4,000,000	23.92
16,384	5,000,000	33.35
16,384	6,000,000	31.14
16,384	7,000,000	46.71
16,384	8,000,000	47.58
16,384	9,000,000	48.55
16,384	10,000,000	66.27
16,384	100,000,000	killed
16,384	1,000,000,000	killed
32,768	10,000	0.05
32,768	100,000	0.41
32,768	1,000,000	8.44
32,768	2,000,000	10.37
32,768	3,000,000	15.03
32,768	4,000,000	22.59
32,768	5,000,000	29.00
32,768	6,000,000	34.01

32,768	7,000,000	46.72
32,768	8,000,000	47.31
32,768	9,000,000	50.49
32,768	10,000,000	58.91
32,768	100,000,000	killed
32,768	1,000,000,000	killed
65,536	10,000	0.05
65,536	100,000	0.39
65,536	1,000,000	4.56
65,536	2,000,000	10.35
65,536	3,000,000	16.64
65,536	4,000,000	26.37
65,536	5,000,000	32.74
65,536	6,000,000	41.89
65,536	7,000,000	33.91
65,536	8,000,000	49.79
65,536	9,000,000	60.19
65,536	10,000,000	55.74
65,536	100,000,000	killed
65,536	1,000,000,000	killed
131,072	10,000	0.06
131,072	100,000	0.40
131,072	1,000,000	4.73
131,072	2,000,000	18.80
131,072	3,000,000	13.77
131,072	4,000,000	23.95
131,072	5,000,000	38.61
131,072	6,000,000	29.16
131,072	7,000,000	35.79
131,072	8,000,000	54.29
131,072	9,000,000	50.95
131,072	10,000,000	57.20

131,072	100,000,000	killed
131,072	1,000,000,000	killed
262,144	10,000	0.04
262,144	100,000	0.39
262,144	1,000,000	5.10
262,144	2,000,000	9.66
262,144	3,000,000	19.11
262,144	4,000,000	25.17
262,144	5,000,000	24.76
262,144	6,000,000	35.98
262,144	7,000,000	43.27
262,144	8,000,000	50.58
262,144	9,000,000	58.45
262,144	10,000,000	68.18
262,144	100,000,000	killed
262,144	1,000,000,000	killed

Memory Mapping:

Reading:

B	N	Elapsed Time (ms)
1,024	10,000	0.07
1,024	100,000	0.18
1,024	1,000,000	1.46
1,024	2,000,000	2.94
1,024	3,000,000	4.45
1,024	4,000,000	5.84
1,024	5,000,000	8.20
1,024	6,000,000	8.69
1,024	7,000,000	10.95
1,024	8,000,000	12.01
1,024	9,000,000	13.64
1,024	10,000,000	14.56

1,024	100,000,000	143.90
1,024	1,000,000,000	1,475.66
2,048	10,000	0.04
2,048	100,000	0.12
2,048	1,000,000	0.82
2,048	2,000,000	1.50
2,048	3,000,000	2.21
2,048	4,000,000	2.90
2,048	5,000,000	4.91
2,048	6,000,000	4.24
2,048	7,000,000	5.04
2,048	8,000,000	6.29
2,048	9,000,000	7.29
2,048	10,000,000	7.34
2,048	100,000,000	73.71
2,048	1,000,000,000	752.06
4,096	10,000	0.05
4,096	100,000	0.07
4,096	1,000,000	0.39
4,096	2,000,000	1.21
4,096	3,000,000	1.11
4,096	4,000,000	1.43
4,096	5,000,000	1.84
4,096	6,000,000	2.17
4,096	7,000,000	2.52
4,096	8,000,000	2.91
4,096	9,000,000	3.25
4,096	10,000,000	3.55
4,096	100,000,000	36.83
4,096	1,000,000,000	354.93
8,192	10,000	0.03
8,192	100,000	0.03

8,192	1,000,000	0.21
8,192	2,000,000	0.39
8,192	3,000,000	0.58
8,192	4,000,000	0.76
8,192	5,000,000	0.90
8,192	6,000,000	1.09
8,192	7,000,000	1.25
8,192	8,000,000	1.47
8,192	9,000,000	1.64
8,192	10,000,000	1.87
8,192	100,000,000	17.74
8,192	1,000,000,000	184.53
16,384	10,000	0.03
16,384	100,000	0.03
16,384	1,000,000	0.13
16,384	2,000,000	0.26
16,384	3,000,000	0.31
16,384	4,000,000	0.39
16,384	5,000,000	0.48
16,384	6,000,000	0.57
16,384	7,000,000	0.66
16,384	8,000,000	0.75
16,384	9,000,000	0.88
16,384	10,000,000	0.97
16,384	100,000,000	8.81
16,384	1,000,000,000	89.88
32,768	10,000	0.03
32,768	100,000	0.03
32,768	1,000,000	0.09
32,768	2,000,000	0.13
32,768	3,000,000	0.23
32,768	4,000,000	0.23

32,768	5,000,000	0.28
32,768	6,000,000	0.33
32,768	7,000,000	0.38
32,768	8,000,000	0.48
32,768	9,000,000	0.46
32,768	10,000,000	0.53
32,768	100,000,000	4.99
32,768	1,000,000,000	49.25
65,536	10,000	0.03
65,536	100,000	0.02
65,536	1,000,000	0.06
65,536	2,000,000	0.09
65,536	3,000,000	0.11
65,536	4,000,000	0.14
65,536	5,000,000	0.17
65,536	6,000,000	0.18
65,536	7,000,000	0.21
65,536	8,000,000	0.24
65,536	9,000,000	0.26
65,536	10,000,000	0.31
65,536	100,000,000	2.52
65,536	1,000,000,000	25.26
131,072	10,000	0.03
131,072	100,000	0.02
131,072	1,000,000	0.05
131,072	2,000,000	0.07
131,072	3,000,000	0.07
131,072	4,000,000	0.09
131,072	5,000,000	0.10
131,072	6,000,000	0.11
131,072	7,000,000	0.13
131,072	8,000,000	0.14

131,072	9,000,000	0.15
131,072	10,000,000	0.16
131,072	100,000,000	1.34
131,072	1,000,000,000	13.68
262,144	10,000	0.03
262,144	100,000	0.03
262,144	1,000,000	0.06
262,144	2,000,000	0.05
262,144	3,000,000	0.06
262,144	4,000,000	0.07
262,144	5,000,000	0.08
262,144	6,000,000	0.08
262,144	7,000,000	0.09
262,144	8,000,000	0.08
262,144	9,000,000	0.11
262,144	10,000,000	0.11
262,144	100,000,000	0.75
262,144	1,000,000,000	7.22

Writing:

B	N	Elapsed Time (ms)
1,024	10,000	0.20
1,024	100,000	1.71
1,024	1,000,000	16.92
1,024	2,000,000	33.56
1,024	3,000,000	51.14
1,024	4,000,000	66.90
1,024	5,000,000	89.75
1,024	6,000,000	100.37
1,024	7,000,000	132.15
1,024	8,000,000	142.10
1,024	9,000,000	159.25
1,024	10,000,000	168.36

1,024	100,000,000	1,899.45
1,024	1,000,000,000	24,544.40
2,048	10,000	0.20
2,048	100,000	1.73
2,048	1,000,000	15.96
2,048	2,000,000	31.23
2,048	3,000,000	48.07
2,048	4,000,000	67.50
2,048	5,000,000	79.93
2,048	6,000,000	91.89
2,048	7,000,000	111.21
2,048	8,000,000	128.40
2,048	9,000,000	148.58
2,048	10,000,000	154.50
2,048	100,000,000	1,853.66
2,048	1,000,000,000	23,409.30
4,096	10,000	0.14
4,096	100,000	1.31
4,096	1,000,000	13.24
4,096	2,000,000	27.87
4,096	3,000,000	39.56
4,096	4,000,000	52.98
4,096	5,000,000	66.86
4,096	6,000,000	79.57
4,096	7,000,000	92.91
4,096	8,000,000	105.90
4,096	9,000,000	118.79
4,096	10,000,000	134.92
4,096	100,000,000	1,335.11
4,096	1,000,000,000	17,707.80
8,192	10,000	0.15
8,192	100,000	1.33

8,192	1,000,000	13.06
8,192	2,000,000	26.02
8,192	3,000,000	38.92
8,192	4,000,000	52.03
8,192	5,000,000	64.92
8,192	6,000,000	83.57
8,192	7,000,000	97.24
8,192	8,000,000	103.39
8,192	9,000,000	116.29
8,192	10,000,000	129.47
8,192	100,000,000	1,311.64
8,192	1,000,000,000	17,626.50
16,384	10,000	0.15
16,384	100,000	1.34
16,384	1,000,000	13.19
16,384	2,000,000	26.67
16,384	3,000,000	38.55
16,384	4,000,000	51.17
16,384	5,000,000	64.01
16,384	6,000,000	77.11
16,384	7,000,000	89.82
16,384	8,000,000	102.36
16,384	9,000,000	115.20
16,384	10,000,000	128.27
16,384	100,000,000	1,299.16
16,384	1,000,000,000	17,696.00
32,768	10,000	0.15
32,768	100,000	1.32
32,768	1,000,000	13.18
32,768	2,000,000	25.45
32,768	3,000,000	41.77
32,768	4,000,000	50.93

32,768	5,000,000	63.61
32,768	6,000,000	76.41
32,768	7,000,000	91.35
32,768	8,000,000	101.48
32,768	9,000,000	114.54
32,768	10,000,000	127.03
32,768	100,000,000	1,286.71
32,768	1,000,000,000	16,821.30
65,536	10,000	0.15
65,536	100,000	1.24
65,536	1,000,000	12.64
65,536	2,000,000	25.29
65,536	3,000,000	37.90
65,536	4,000,000	50.28
65,536	5,000,000	62.80
65,536	6,000,000	82.60
65,536	7,000,000	88.85
65,536	8,000,000	100.74
65,536	9,000,000	113.47
65,536	10,000,000	126.74
65,536	100,000,000	1,296.48
65,536	1,000,000,000	16,445.70
131,072	10,000	0.14
131,072	100,000	1.29
131,072	1,000,000	12.62
131,072	2,000,000	25.05
131,072	3,000,000	37.66
131,072	4,000,000	50.52
131,072	5,000,000	63.17
131,072	6,000,000	75.18
131,072	7,000,000	87.84
131,072	8,000,000	100.34

131,072	9,000,000	112.73
131,072	10,000,000	125.14
131,072	100,000,000	1,273.64
131,072	1,000,000,000	17,489.90
262,144	10,000	0.15
262,144	100,000	1.29
262,144	1,000,000	12.52
262,144	2,000,000	25.04
262,144	3,000,000	37.66
262,144	4,000,000	50.40
262,144	5,000,000	62.74
262,144	6,000,000	75.00
262,144	7,000,000	88.29
262,144	8,000,000	100.07
262,144	9,000,000	114.60
262,144	10,000,000	125.88
262,144	100,000,000	1,284.16
262,144	1,000,000,000	16,857.50

In-Memory Sort:

N	Elapsed Time (ms)
10,000	2.08
100,000	17.77
1,000,000	207.68
2,000,000	438.67
3,000,000	687.29
4,000,000	918.77
5,000,000	1,208.85
6,000,000	1,449.34
7,000,000	2,318.82
8,000,000	2,592.01
9,000,000	2,831.41

External Merge Sort:

N	M	d	Elapsed Time (ms)
10,000	1,024	3	88.99
10,000	1,024	4	58.53
10,000	1,024	5	63.94
10,000	1,024	6	43.44
10,000	1,024	7	46.03
10,000	1,024	8	48.17
10,000	1,024	9	51.45
10,000	1,024	10	28.69
10,000	2,048	3	42.05
10,000	2,048	4	46.56
10,000	2,048	5	28.20
10,000	2,048	6	28.48
10,000	2,048	7	28.35
10,000	2,048	8	28.83
10,000	2,048	9	28.48
10,000	2,048	10	28.30
10,000	4,096	3	27.60
10,000	4,096	4	27.33
10,000	4,096	5	27.21
10,000	4,096	6	27.19
10,000	4,096	7	27.01
10,000	4,096	8	34.31
10,000	4,096	9	30.22
10,000	4,096	10	29.60
10,000	8,192	3	27.87
10,000	8,192	4	28.25
10,000	8,192	5	28.05
10,000	8,192	6	27.48
10,000	8,192	7	28.05
10,000	8,192	8	27.74

10,000	8,192	9	27.18
10,000	8,192	10	28.89
10,000	16,384	3	2.01
10,000	16,384	4	2.07
10,000	16,384	5	2.05
10,000	16,384	6	2.20
10,000	16,384	7	2.01
10,000	16,384	8	2.04
10,000	16,384	9	2.00
10,000	16,384	10	2.06
10,000	32,768	3	2.00
10,000	32,768	4	2.08
10,000	32,768	5	2.06
10,000	32,768	6	2.01
10,000	32,768	7	2.05
10,000	32,768	8	2.00
10,000	32,768	9	2.03
10,000	32,768	10	2.00
10,000	65,536	3	2.60
10,000	65,536	4	4.92
10,000	65,536	5	4.89
10,000	65,536	6	2.69
10,000	65,536	7	2.00
10,000	65,536	8	2.02
10,000	65,536	9	2.00
10,000	65,536	10	2.04
10,000	131,072	3	2.00
10,000	131,072	4	2.05
10,000	131,072	5	2.20
10,000	131,072	6	2.03
10,000	131,072	7	2.22
10,000	131,072	8	2.22

10,000	131,072	9	2.08
10,000	131,072	10	2.02
10,000	262,144	3	2.02
10,000	262,144	4	2.02
10,000	262,144	5	2.02
10,000	262,144	6	2.00
10,000	262,144	7	2.06
10,000	262,144	8	2.02
10,000	262,144	9	2.04
10,000	262,144	10	2.00
100,000	1,024	3	6,460.60
100,000	1,024	4	4,654.55
100,000	1,024	5	3,675.55
100,000	1,024	6	2,920.35
100,000	1,024	7	2,588.18
100,000	1,024	8	2,233.00
100,000	1,024	9	2,041.65
100,000	1,024	10	1,678.88
100,000	2,048	3	3,194.56
100,000	2,048	4	2,265.90
100,000	2,048	5	1,756.10
100,000	2,048	6	1,522.60
100,000	2,048	7	1,228.67
100,000	2,048	8	1,132.18
100,000	2,048	9	975.85
100,000	2,048	10	1,065.39
100,000	4,096	3	1,664.64
100,000	4,096	4	1,209.18
100,000	4,096	5	940.42
100,000	4,096	6	821.53
100,000	4,096	7	671.73
100,000	4,096	8	754.07

100,000	4,096	9	555.69
100,000	4,096	10	581.57
100,000	8,192	3	898.53
100,000	8,192	4	665.61
100,000	8,192	5	533.39
100,000	8,192	6	611.30
100,000	8,192	7	412.65
100,000	8,192	8	428.67
100,000	8,192	9	448.22
100,000	8,192	10	476.87
100,000	16,384	3	519.48
100,000	16,384	4	404.17
100,000	16,384	5	435.15
100,000	16,384	6	474.69
100,000	16,384	7	277.10
100,000	16,384	8	273.52
100,000	16,384	9	273.66
100,000	16,384	10	272.42
100,000	32,768	3	443.80
100,000	32,768	4	269.15
100,000	32,768	5	270.28
100,000	32,768	6	271.00
100,000	32,768	7	268.51
100,000	32,768	8	269.44
100,000	32,768	9	270.55
100,000	32,768	10	269.08
100,000	65,536	3	258.56
100,000	65,536	4	257.87
100,000	65,536	5	259.91
100,000	65,536	6	256.81
100,000	65,536	7	259.49
100,000	65,536	8	261.27

100,000	65,536	9	260.60
100,000	65,536	10	262.26
100,000	131,072	3	17.77
100,000	131,072	4	17.71
100,000	131,072	5	17.76
100,000	131,072	6	17.72
100,000	131,072	7	17.79
100,000	131,072	8	17.70
100,000	131,072	9	17.75
100,000	131,072	10	17.72
100,000	262,144	3	17.74
100,000	262,144	4	17.71
100,000	262,144	5	17.68
100,000	262,144	6	17.63
100,000	262,144	7	17.69
100,000	262,144	8	17.66
100,000	262,144	9	17.81
100,000	262,144	10	17.71
1,000,000	1,024	3	620,312.00
1,000,000	1,024	4	433,655.00
1,000,000	1,024	5	323,834.00
1,000,000	1,024	6	262,856.00
1,000,000	1,024	7	218,638.00
1,000,000	1,024	8	196,387.00
1,000,000	1,024	9	170,981.00
1,000,000	1,024	10	153,228.00
1,000,000	2,048	3	310,958.00
1,000,000	2,048	4	217,163.00
1,000,000	2,048	5	162,791.00
1,000,000	2,048	6	131,737.00
1,000,000	2,048	7	111,464.00
1,000,000	2,048	8	98,813.60

1,000,000	2,048	9	85,907.70
1,000,000	2,048	10	78,607.00
1,000,000	4,096	3	155,586.00
1,000,000	4,096	4	110,991.00
1,000,000	4,096	5	82,477.60
1,000,000	4,096	6	66,526.30
1,000,000	4,096	7	56,270.20
1,000,000	4,096	8	49,977.40
1,000,000	4,096	9	45,123.70
1,000,000	4,096	10	41,091.80
1,000,000	8,192	3	78,462.50
1,000,000	8,192	4	55,998.10
1,000,000	8,192	5	42,894.90
1,000,000	8,192	6	35,264.60
1,000,000	8,192	7	30,182.90
1,000,000	8,192	8	26,937.10
1,000,000	8,192	9	24,311.50
1,000,000	8,192	10	21,357.40
1,000,000	16,384	3	41,200.90
1,000,000	16,384	4	29,857.90
1,000,000	16,384	5	23,364.20
1,000,000	16,384	6	19,587.30
1,000,000	16,384	7	16,919.70
1,000,000	16,384	8	14,002.20
1,000,000	16,384	9	12,865.40
1,000,000	16,384	10	11,358.10
1,000,000	32,768	3	20,627.50
1,000,000	32,768	4	14,806.90
1,000,000	32,768	5	12,630.00
1,000,000	32,768	6	9,601.08
1,000,000	32,768	7	8,166.80
1,000,000	32,768	8	9,117.31

1,000,000	32,768	9	7,153.74
1,000,000	32,768	10	7,757.29
1,000,000	65,536	3	12,216.50
1,000,000	65,536	4	8,105.64
1,000,000	65,536	5	7,033.23
1,000,000	65,536	6	5,443.19
1,000,000	65,536	7	5,995.07
1,000,000	65,536	8	6,359.41
1,000,000	65,536	9	4,243.17
1,000,000	65,536	10	4,401.09
1,000,000	131,072	3	7,136.54
1,000,000	131,072	4	6,111.23
1,000,000	131,072	5	4,376.05
1,000,000	131,072	6	4,588.90
1,000,000	131,072	7	4,943.23
1,000,000	131,072	8	2,862.75
1,000,000	131,072	9	2,828.80
1,000,000	131,072	10	2,830.56
1,000,000	262,144	3	4,469.31
1,000,000	262,144	4	2,773.73
1,000,000	262,144	5	2,785.67
1,000,000	262,144	6	2,785.72
1,000,000	262,144	7	2,788.57
1,000,000	262,144	8	2,749.21
1,000,000	262,144	9	2,746.67
1,000,000	262,144	10	2,758.05
2,000,000	1,024	3	killed
2,000,000	1,024	4	killed
2,000,000	1,024	5	killed
2,000,000	1,024	6	killed
2,000,000	1,024	7	killed
2,000,000	1,024	8	killed

2,000,000	1,024	9	killed
2,000,000	1,024	10	killed
2,000,000	2,048	3	1,240,780.00
2,000,000	2,048	4	867,997.00
2,000,000	2,048	5	650,327.00
2,000,000	2,048	6	523,204.00
2,000,000	2,048	7	438,612.00
2,000,000	2,048	8	392,072.00
2,000,000	2,048	9	341,123.00
2,000,000	2,048	10	306,656.00
2,000,000	4,096	3	620,102.00
2,000,000	4,096	4	435,111.00
2,000,000	4,096	5	326,220.00
2,000,000	4,096	6	263,003.00
2,000,000	4,096	7	223,403.00
2,000,000	4,096	8	197,828.00
2,000,000	4,096	9	171,983.00
2,000,000	4,096	10	156,950.00
2,000,000	8,192	3	312,740.00
2,000,000	8,192	4	221,225.00
2,000,000	8,192	5	163,656.00
2,000,000	8,192	6	133,162.00
2,000,000	8,192	7	113,041.00
2,000,000	8,192	8	100,068.00
2,000,000	8,192	9	89,788.10
2,000,000	8,192	10	82,097.60
2,000,000	16,384	3	157,050.00
2,000,000	16,384	4	112,366.00
2,000,000	16,384	5	86,250.90
2,000,000	16,384	6	70,222.80
2,000,000	16,384	7	60,026.00
2,000,000	16,384	8	53,870.30

2,000,000	16,384	9	48,677.00
2,000,000	16,384	10	42,549.40
2,000,000	32,768	3	82,232.60
2,000,000	32,768	4	59,421.50
2,000,000	32,768	5	46,551.10
2,000,000	32,768	6	38,914.80
2,000,000	32,768	7	33,934.40
2,000,000	32,768	8	28,203.00
2,000,000	32,768	9	25,648.60
2,000,000	32,768	10	22,803.70
2,000,000	65,536	3	41,327.50
2,000,000	65,536	4	29,613.00
2,000,000	65,536	5	25,257.30
2,000,000	65,536	6	18,991.10
2,000,000	65,536	7	16,376.00
2,000,000	65,536	8	18,037.10
2,000,000	65,536	9	14,364.20
2,000,000	65,536	10	15,461.60
2,000,000	131,072	3	24,350.50
2,000,000	131,072	4	16,377.10
2,000,000	131,072	5	14,142.00
2,000,000	131,072	6	10,892.40
2,000,000	131,072	7	11,977.90
2,000,000	131,072	8	12,791.40
2,000,000	131,072	9	8,560.70
2,000,000	131,072	10	8,850.69
2,000,000	262,144	3	14,417.60
2,000,000	262,144	4	12,381.20
2,000,000	262,144	5	8,697.57
2,000,000	262,144	6	9,217.97
2,000,000	262,144	7	9,905.99
2,000,000	262,144	8	5,637.49

2,000,000	262,144	9	5,648.09
2,000,000	262,144	10	5,637.84
3,000,000	1,024	3	killed
3,000,000	1,024	4	killed
3,000,000	1,024	5	killed
3,000,000	1,024	6	killed
3,000,000	1,024	7	killed
3,000,000	1,024	8	killed
3,000,000	1,024	9	killed
3,000,000	1,024	10	killed
3,000,000	2,048	3	killed
3,000,000	2,048	4	killed
3,000,000	2,048	5	killed
3,000,000	2,048	6	killed
3,000,000	2,048	7	killed
3,000,000	2,048	8	killed
3,000,000	2,048	9	killed
3,000,000	2,048	10	killed
3,000,000	4,096	3	1,397,930.00
3,000,000	4,096	4	975,986.00
3,000,000	4,096	5	733,043.00
3,000,000	4,096	6	589,696.00
3,000,000	4,096	7	491,449.00
3,000,000	4,096	8	442,079.00
3,000,000	4,096	9	388,788.00
3,000,000	4,096	10	347,498.00
3,000,000	8,192	3	699,684.00
3,000,000	8,192	4	488,153.00
3,000,000	8,192	5	371,703.00
3,000,000	8,192	6	301,000.00
3,000,000	8,192	7	247,769.00
3,000,000	8,192	8	226,807.00

3,000,000	8,192	9	196,524.00
3,000,000	8,192	10	175,898.00
3,000,000	16,384	3	356,312.00
3,000,000	16,384	4	247,773.00
3,000,000	16,384	5	187,917.00
3,000,000	16,384	6	152,631.00
3,000,000	16,384	7	129,707.00
3,000,000	16,384	8	119,109.00
3,000,000	16,384	9	100,282.00
3,000,000	16,384	10	94,132.20
3,000,000	32,768	3	181,678.00
3,000,000	32,768	4	129,819.00
3,000,000	32,768	5	96,701.20
3,000,000	32,768	6	82,780.30
3,000,000	32,768	7	70,774.50
3,000,000	32,768	8	58,312.80
3,000,000	32,768	9	56,145.90
3,000,000	32,768	10	52,835.40
3,000,000	65,536	3	94,506.70
3,000,000	65,536	4	64,581.30
3,000,000	65,536	5	54,875.80
3,000,000	65,536	6	40,435.90
3,000,000	65,536	7	38,492.90
3,000,000	65,536	8	35,470.20
3,000,000	65,536	9	30,772.00
3,000,000	65,536	10	25,158.80
3,000,000	131,072	3	47,585.00
3,000,000	131,072	4	39,056.70
3,000,000	131,072	5	30,085.80
3,000,000	131,072	6	26,484.30
3,000,000	131,072	7	21,801.20
3,000,000	131,072	8	23,696.70

3,000,000	131,072	9	17,616.50
3,000,000	131,072	10	18,537.30
3,000,000	262,144	3	29,026.30
3,000,000	262,144	4	21,245.00
3,000,000	262,144	5	17,157.40
3,000,000	262,144	6	18,674.70
3,000,000	262,144	7	12,913.60
3,000,000	262,144	8	13,565.30
3,000,000	262,144	9	14,288.80
3,000,000	262,144	10	14,754.00
4,000,000	1,024	3	killed
4,000,000	1,024	4	killed
4,000,000	1,024	5	killed
4,000,000	1,024	6	killed
4,000,000	1,024	7	killed
4,000,000	1,024	8	killed
4,000,000	1,024	9	killed
4,000,000	1,024	10	killed
4,000,000	2,048	3	killed
4,000,000	2,048	4	killed
4,000,000	2,048	5	killed
4,000,000	2,048	6	killed
4,000,000	2,048	7	killed
4,000,000	2,048	8	killed
4,000,000	2,048	9	killed
4,000,000	2,048	10	killed
4,000,000	4,096	3	2,480,180.00
4,000,000	4,096	4	1,739,100.00
4,000,000	4,096	5	1,299,890.00
4,000,000	4,096	6	1,047,780.00
4,000,000	4,096	7	875,952.00
4,000,000	4,096	8	788,290.00

4,000,000	4,096	9	682,492.00
4,000,000	4,096	10	614,905.00
4,000,000	8,192	3	1,244,530.00
4,000,000	8,192	4	873,563.00
4,000,000	8,192	5	653,801.00
4,000,000	8,192	6	528,884.00
4,000,000	8,192	7	447,180.00
4,000,000	8,192	8	395,866.00
4,000,000	8,192	9	345,131.00
4,000,000	8,192	10	314,637.00
4,000,000	16,384	3	625,357.00
4,000,000	16,384	4	443,421.00
4,000,000	16,384	5	329,431.00
4,000,000	16,384	6	266,568.00
4,000,000	16,384	7	225,575.00
4,000,000	16,384	8	200,591.00
4,000,000	16,384	9	180,166.00
4,000,000	16,384	10	165,068.00
4,000,000	32,768	3	315,106.00
4,000,000	32,768	4	225,052.00
4,000,000	32,768	5	172,340.00
4,000,000	32,768	6	141,224.00
4,000,000	32,768	7	120,498.00
4,000,000	32,768	8	108,067.00
4,000,000	32,768	9	97,731.80
4,000,000	32,768	10	85,259.80
4,000,000	65,536	3	164,619.00
4,000,000	65,536	4	119,917.00
4,000,000	65,536	5	93,645.50
4,000,000	65,536	6	77,931.20
4,000,000	65,536	7	68,212.90
4,000,000	65,536	8	56,749.90

4,000,000	65,536	9	51,625.70
4,000,000	65,536	10	45,694.10
4,000,000	131,072	3	83,256.80
4,000,000	131,072	4	59,298.00
4,000,000	131,072	5	50,916.50
4,000,000	131,072	6	37,921.90
4,000,000	131,072	7	32,654.80
4,000,000	131,072	8	36,561.30
4,000,000	131,072	9	28,814.10
4,000,000	131,072	10	31,013.00
4,000,000	262,144	3	48,857.00
4,000,000	262,144	4	32,662.40
4,000,000	262,144	5	28,318.20
4,000,000	262,144	6	21,870.90
4,000,000	262,144	7	24,157.40
4,000,000	262,144	8	25,757.20
4,000,000	262,144	9	17,195.10
4,000,000	262,144	10	17,814.10
5,000,000	1,024	3	killed
5,000,000	1,024	4	killed
5,000,000	1,024	5	killed
5,000,000	1,024	6	killed
5,000,000	1,024	7	killed
5,000,000	1,024	8	killed
5,000,000	1,024	9	killed
5,000,000	1,024	10	killed
5,000,000	2,048	3	killed
5,000,000	2,048	4	killed
5,000,000	2,048	5	killed
5,000,000	2,048	6	killed
5,000,000	2,048	7	killed
5,000,000	2,048	8	killed

5,000,000	2,048	9	killed
5,000,000	2,048	10	killed
5,000,000	4,096	3	killed
5,000,000	4,096	4	killed
5,000,000	4,096	5	killed
5,000,000	4,096	6	killed
5,000,000	4,096	7	killed
5,000,000	4,096	8	killed
5,000,000	4,096	9	killed
5,000,000	4,096	10	killed
5,000,000	8,192	3	1,948,840.00
5,000,000	8,192	4	1,368,530.00
5,000,000	8,192	5	1,040,350.00
5,000,000	8,192	6	828,354.00
5,000,000	8,192	7	706,264.00
5,000,000	8,192	8	652,001.00
5,000,000	8,192	9	565,482.00
5,000,000	8,192	10	499,140.00
5,000,000	16,384	3	1,017,700.00
5,000,000	16,384	4	701,765.00
5,000,000	16,384	5	548,472.00
5,000,000	16,384	6	421,498.00
5,000,000	16,384	7	361,367.00
5,000,000	16,384	8	334,453.00
5,000,000	16,384	9	288,298.00
5,000,000	16,384	10	258,286.00
5,000,000	32,768	3	500,957.00
5,000,000	32,768	4	357,222.00
5,000,000	32,768	5	261,125.00
5,000,000	32,768	6	217,015.00
5,000,000	32,768	7	185,060.00
5,000,000	32,768	8	161,163.00

5,000,000	32,768	9	139,291.00
5,000,000	32,768	10	126,154.00
5,000,000	65,536	3	249,972.00
5,000,000	65,536	4	183,959.00
5,000,000	65,536	5	134,107.00
5,000,000	65,536	6	118,356.00
5,000,000	65,536	7	96,737.40
5,000,000	65,536	8	86,619.90
5,000,000	65,536	9	82,833.10
5,000,000	65,536	10	76,476.00
5,000,000	131,072	3	133,429.00
5,000,000	131,072	4	98,573.50
5,000,000	131,072	5	78,174.60
5,000,000	131,072	6	65,020.70
5,000,000	131,072	7	59,526.10
5,000,000	131,072	8	53,274.50
5,000,000	131,072	9	44,373.60
5,000,000	131,072	10	49,570.30
5,000,000	262,144	3	76,544.40
5,000,000	262,144	4	57,589.50
5,000,000	262,144	5	42,015.70
5,000,000	262,144	6	36,891.00
5,000,000	262,144	7	38,682.00
5,000,000	262,144	8	30,807.00
5,000,000	262,144	9	34,433.60
5,000,000	262,144	10	33,327.70
6,000,000	1,024	3	killed
6,000,000	1,024	4	killed
6,000,000	1,024	5	killed
6,000,000	1,024	6	killed
6,000,000	1,024	7	killed
6,000,000	1,024	8	killed

6,000,000	1,024	9	killed
6,000,000	1,024	10	killed
6,000,000	2,048	3	killed
6,000,000	2,048	4	killed
6,000,000	2,048	5	killed
6,000,000	2,048	6	killed
6,000,000	2,048	7	killed
6,000,000	2,048	8	killed
6,000,000	2,048	9	killed
6,000,000	2,048	10	killed
6,000,000	4,096	3	killed
6,000,000	4,096	4	killed
6,000,000	4,096	5	killed
6,000,000	4,096	6	killed
6,000,000	4,096	7	killed
6,000,000	4,096	8	killed
6,000,000	4,096	9	killed
6,000,000	4,096	10	killed
6,000,000	8,192	3	2,873,950.00
6,000,000	8,192	4	2,273,970.00
6,000,000	8,192	5	1,722,720.00
6,000,000	8,192	6	1,399,740.00
6,000,000	8,192	7	1,208,650.00
6,000,000	8,192	8	1,055,550.00
6,000,000	8,192	9	927,867.00
6,000,000	8,192	10	868,308.00
6,000,000	16,384	3	1,705,570.00
6,000,000	16,384	4	1,134,960.00
6,000,000	16,384	5	862,781.00
6,000,000	16,384	6	699,246.00
6,000,000	16,384	7	576,382.00
6,000,000	16,384	8	521,518.00

6,000,000	16,384	9	451,011.00
6,000,000	16,384	10	404,705.00
6,000,000	32,768	3	834,911.00
6,000,000	32,768	4	572,060.00
6,000,000	32,768	5	436,397.00
6,000,000	32,768	6	355,096.00
6,000,000	32,768	7	302,749.00
6,000,000	32,768	8	274,651.00
6,000,000	32,768	9	230,498.00
6,000,000	32,768	10	216,122.00
6,000,000	65,536	3	426,396.00
6,000,000	65,536	4	302,177.00
6,000,000	65,536	5	225,281.00
6,000,000	65,536	6	192,339.00
6,000,000	65,536	7	165,743.00
6,000,000	65,536	8	134,882.00
6,000,000	65,536	9	130,319.00
6,000,000	65,536	10	121,835.00
6,000,000	131,072	3	222,276.00
6,000,000	131,072	4	150,448.00
6,000,000	131,072	5	128,576.00
6,000,000	131,072	6	93,952.40
6,000,000	131,072	7	89,316.40
6,000,000	131,072	8	82,054.20
6,000,000	131,072	9	70,307.40
6,000,000	131,072	10	58,328.10
6,000,000	262,144	3	111,818.00
6,000,000	262,144	4	90,838.90
6,000,000	262,144	5	70,405.80
6,000,000	262,144	6	61,569.60
6,000,000	262,144	7	50,580.10
6,000,000	262,144	8	55,253.50

6,000,000	262,144	9	40,445.00
6,000,000	262,144	10	42,884.10
7,000,000	1,024	3	killed
7,000,000	1,024	4	killed
7,000,000	1,024	5	killed
7,000,000	1,024	6	killed
7,000,000	1,024	7	killed
7,000,000	1,024	8	killed
7,000,000	1,024	9	killed
7,000,000	1,024	10	killed
7,000,000	2,048	3	killed
7,000,000	2,048	4	killed
7,000,000	2,048	5	killed
7,000,000	2,048	6	killed
7,000,000	2,048	7	killed
7,000,000	2,048	8	killed
7,000,000	2,048	9	killed
7,000,000	2,048	10	killed
7,000,000	4,096	3	killed
7,000,000	4,096	4	killed
7,000,000	4,096	5	killed
7,000,000	4,096	6	killed
7,000,000	4,096	7	killed
7,000,000	4,096	8	killed
7,000,000	4,096	9	killed
7,000,000	4,096	10	killed
7,000,000	8,192	3	3,947,150.00
7,000,000	8,192	4	2,755,160.00
7,000,000	8,192	5	2,152,610.00
7,000,000	8,192	6	2,064,080.00
7,000,000	8,192	7	1,745,420.00
7,000,000	8,192	8	1,540,570.00

7,000,000	8,192	9	1,370,430.00
7,000,000	8,192	10	1,228,440.00
7,000,000	16,384	3	2,452,700.00
7,000,000	16,384	4	1,623,970.00
7,000,000	16,384	5	1,248,880.00
7,000,000	16,384	6	989,657.00
7,000,000	16,384	7	849,097.00
7,000,000	16,384	8	720,964.00
7,000,000	16,384	9	657,053.00
7,000,000	16,384	10	576,198.00
7,000,000	32,768	3	1,213,180.00
7,000,000	32,768	4	861,459.00
7,000,000	32,768	5	645,948.00
7,000,000	32,768	6	497,909.00
7,000,000	32,768	7	428,817.00
7,000,000	32,768	8	373,971.00
7,000,000	32,768	9	333,379.00
7,000,000	32,768	10	292,970.00
7,000,000	65,536	3	597,395.00
7,000,000	65,536	4	449,275.00
7,000,000	65,536	5	340,118.00
7,000,000	65,536	6	272,397.00
7,000,000	65,536	7	222,387.00
7,000,000	65,536	8	215,502.00
7,000,000	65,536	9	201,099.00
7,000,000	65,536	10	170,729.00
7,000,000	131,072	3	330,902.00
7,000,000	131,072	4	219,910.00
7,000,000	131,072	5	180,633.00
7,000,000	131,072	6	139,926.00
7,000,000	131,072	7	116,491.00
7,000,000	131,072	8	117,554.00

7,000,000	131,072	9	105,921.00
7,000,000	131,072	10	84,147.70
7,000,000	262,144	3	157,676.00
7,000,000	262,144	4	118,032.00
7,000,000	262,144	5	98,767.20
7,000,000	262,144	6	89,180.40
7,000,000	262,144	7	77,128.50
7,000,000	262,144	8	61,903.30
7,000,000	262,144	9	67,167.50
7,000,000	262,144	10	49,483.60
8,000,000	1,024	3	killed
8,000,000	1,024	4	killed
8,000,000	1,024	5	killed
8,000,000	1,024	6	killed
8,000,000	1,024	7	killed
8,000,000	1,024	8	killed
8,000,000	1,024	9	killed
8,000,000	1,024	10	killed
8,000,000	2,048	3	killed
8,000,000	2,048	4	killed
8,000,000	2,048	5	killed
8,000,000	2,048	6	killed
8,000,000	2,048	7	killed
8,000,000	2,048	8	killed
8,000,000	2,048	9	killed
8,000,000	2,048	10	killed
8,000,000	4,096	3	killed
8,000,000	4,096	4	killed
8,000,000	4,096	5	killed
8,000,000	4,096	6	killed
8,000,000	4,096	7	killed
8,000,000	4,096	8	killed

8,000,000	4,096	9	killed
8,000,000	4,096	10	killed
8,000,000	8,192	3	5,947,870.00
8,000,000	8,192	4	3,510,560.00
8,000,000	8,192	5	2,728,550.00
8,000,000	8,192	6	2,106,420.00
8,000,000	8,192	7	1,810,300.00
8,000,000	8,192	8	1,564,200.00
8,000,000	8,192	9	1,394,380.00
8,000,000	8,192	10	1,228,420.00
8,000,000	16,384	3	2,528,700.00
8,000,000	16,384	4	1,746,590.00
8,000,000	16,384	5	1,363,640.00
8,000,000	16,384	6	1,057,070.00
8,000,000	16,384	7	928,244.00
8,000,000	16,384	8	793,940.00
8,000,000	16,384	9	713,244.00
8,000,000	16,384	10	637,070.00
8,000,000	32,768	3	1,268,630.00
8,000,000	32,768	4	889,193.00
8,000,000	32,768	5	686,851.00
8,000,000	32,768	6	537,998.00
8,000,000	32,768	7	471,045.00
8,000,000	32,768	8	407,481.00
8,000,000	32,768	9	376,280.00
8,000,000	32,768	10	332,736.00
8,000,000	65,536	3	640,593.00
8,000,000	65,536	4	456,260.00
8,000,000	65,536	5	362,308.00
8,000,000	65,536	6	286,302.00
8,000,000	65,536	7	252,428.00
8,000,000	65,536	8	220,323.00

8,000,000	65,536	9	203,895.00
8,000,000	65,536	10	174,284.00
8,000,000	131,072	3	338,278.00
8,000,000	131,072	4	243,687.00
8,000,000	131,072	5	197,788.00
8,000,000	131,072	6	157,899.00
8,000,000	131,072	7	143,908.00
8,000,000	131,072	8	115,760.00
8,000,000	131,072	9	107,645.00
8,000,000	131,072	10	97,386.80
8,000,000	262,144	3	171,310.00
8,000,000	262,144	4	120,112.00
8,000,000	262,144	5	107,035.00
8,000,000	262,144	6	80,028.20
8,000,000	262,144	7	69,173.40
8,000,000	262,144	8	74,250.40
8,000,000	262,144	9	64,660.80
8,000,000	262,144	10	65,018.70
9,000,000	1,024	3	killed
9,000,000	1,024	4	killed
9,000,000	1,024	5	killed
9,000,000	1,024	6	killed
9,000,000	1,024	7	killed
9,000,000	1,024	8	killed
9,000,000	1,024	9	killed
9,000,000	1,024	10	killed
9,000,000	2,048	3	killed
9,000,000	2,048	4	killed
9,000,000	2,048	5	killed
9,000,000	2,048	6	killed
9,000,000	2,048	7	killed
9,000,000	2,048	8	killed

9,000,000	2,048	9	killed
9,000,000	2,048	10	killed
9,000,000	4,096	3	killed
9,000,000	4,096	4	killed
9,000,000	4,096	5	killed
9,000,000	4,096	6	killed
9,000,000	4,096	7	killed
9,000,000	4,096	8	killed
9,000,000	4,096	9	killed
9,000,000	4,096	10	killed
9,000,000	8,192	3	killed
9,000,000	8,192	4	killed
9,000,000	8,192	5	killed
9,000,000	8,192	6	killed
9,000,000	8,192	7	killed
9,000,000	8,192	8	killed
9,000,000	8,192	9	killed
9,000,000	8,192	10	0.00
9,000,000	16,384	3	3,239,490.00
9,000,000	16,384	4	2,239,800.00
9,000,000	16,384	5	1,728,940.00
9,000,000	16,384	6	1,383,730.00
9,000,000	16,384	7	1,226,900.00
9,000,000	16,384	8	1,090,650.00
9,000,000	16,384	9	972,824.00
9,000,000	16,384	10	851,800.00
9,000,000	32,768	3	1,616,150.00
9,000,000	32,768	4	1,129,690.00
9,000,000	32,768	5	876,609.00
9,000,000	32,768	6	681,869.00
9,000,000	32,768	7	590,936.00
9,000,000	32,768	8	523,093.00

9,000,000	32,768	9	471,067.00
9,000,000	32,768	10	410,181.00
9,000,000	65,536	3	821,802.00
9,000,000	65,536	4	570,465.00
9,000,000	65,536	5	455,216.00
9,000,000	65,536	6	358,140.00
9,000,000	65,536	7	301,110.00
9,000,000	65,536	8	269,333.00
9,000,000	65,536	9	252,574.00
9,000,000	65,536	10	222,917.00
9,000,000	131,072	3	443,392.00
9,000,000	131,072	4	300,467.00
9,000,000	131,072	5	248,030.00
9,000,000	131,072	6	188,873.00
9,000,000	131,072	7	170,915.00
9,000,000	131,072	8	141,976.00
9,000,000	131,072	9	135,395.00
9,000,000	131,072	10	119,678.00
9,000,000	262,144	3	213,244.00
9,000,000	262,144	4	166,177.00
9,000,000	262,144	5	131,149.00
9,000,000	262,144	6	101,693.00
9,000,000	262,144	7	92,668.20
9,000,000	262,144	8	78,200.80
9,000,000	262,144	9	86,401.30
9,000,000	262,144	10	67,660.30