Report Assignment 2

Counting Sort Algorithm

Lecturer: Francesco Moscato fmoscato@unisa.it

Group:

- Carratù	Arianna	0622701696	a.carratu18@studenti.unisa.it
- Di Rienzo	Matteo	0622701818	m.dirienzo1@studenti.unisa.it
- Gambardella	Giuseppe	0622701666	g.gambardella23@studenti.unisa.it

Problem description

Parallelize and Evaluate Performances of "COUNTING SORT" Algorithm, by using MPI.

Counting Sort is an algorithm for sorting a collection of objects according to keys that are small positive integers. It is an integer sorting algorithm.
Its worst-case performance is O(n+k), where k is the range of the non-negative key values.

For further information, see https://en.wikipedia.org/wiki/Counting_sort.

Solution

We implemented an algorithm that evaluates four different MPI I/O approaches, referred as Versions.

We used MPI primitives to parallelize the counting sort algorithm and see how many cores are needed to increase the speedup.

Steps followed:

- Divide the array in *p* parts of equal length called chunk.
- Create a counting array *'c'* for each process.
- Let each process count the number of elements and store them in *c.* This will take O(k+n/p), where k is the number of possible values.
- Add all counting arrays c together in the first rank.
  This will take O(p*k).
- The first rank will calculate the sorted array in O(k).

The parallelized counting sort is faster than the serial counting sort if and only if p*k<=n. Otherwise, it is slower, since each process has *c* more elements to sum than the serial counting sort.

Version 1

The first version reads each chunk from file in an ordered mode (i.e., using *MPI_File_read_ordered*).

Version 2

Each chunk is read from file using *MPI_File_seek* and *MPI_File_read_all* with a displacement computed through processes’ ranks.

Version 3

Each chunk is read from file using a file view and *MPI_File_read_all* with a displacement computed through processes’ ranks.

Version 4

Each chunk is read from file using a file view and *MPI_File_read* with a displacement computed through processes' ranks.

Experimental Setup 1

Hardware

CPU

```
processor   : 0
vendor_id   : GenuineIntel
cpu family  : 6
model       : 78
model name  : Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz
stepping    : 3
microcode   : 0xea
cpu MHz           : 800.109
cache size  : 4096 KB
physical id : 0
siblings    : 4
core id           : 0
cpu cores   : 2
apicid            : 0
initial apicid    : 0
fpu         : yes
fpu_exception     : yes
cpuid level : 22
wp          : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb
rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology
nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est
tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt
tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch
cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi
flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms
invpcid rtm mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1
xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear
flush_l1d
bugs        : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs taa itlb_multihit srbds
bogomips    : 5599.85
clflush size      : 64
cache_alignment   : 64
address sizes     : 39 bits physical, 48 bits virtual
power management:
```

```
processor   : 1
vendor_id   : GenuineIntel
cpu family  : 6
model       : 78
model name  : Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz
stepping    : 3
microcode   : 0xea
cpu MHz           : 800.044
cache size  : 4096 KB
physical id : 0
siblings    : 4
core id           : 1
cpu cores   : 2
apicid            : 2
initial apicid    : 2
fpu         : yes
fpu_exception     : yes
cpuid level : 22
wp          : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb
rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology
nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est
tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt
tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch
cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi
flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms
invpcid rtm mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1
xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear
flush_l1d
bugs        : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs taa itlb_multihit srbds
bogomips    : 5599.85
clflush size      : 64
cache_alignment   : 64
address sizes     : 39 bits physical, 48 bits virtual
power management:

processor   : 2
vendor_id   : GenuineIntel
cpu family  : 6
model       : 78
model name  : Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz
stepping    : 3
microcode   : 0xea
cpu MHz           : 800.030
cache size  : 4096 KB
physical id : 0
siblings    : 4
core id           : 0
```

```
cpu cores   : 2
apicid            : 1
initial apicid    : 1
fpu         : yes
fpu_exception     : yes
cpuid level : 22
wp          : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb
rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology
nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est
tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt
tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch
cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi
flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms
invpcid rtm mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1
xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear
flush_l1d
bugs        : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs taa itlb_multihit srbds
bogomips    : 5599.85
clflush size      : 64
cache_alignment   : 64
address sizes     : 39 bits physical, 48 bits virtual
power management:

processor   : 3
vendor_id   : GenuineIntel
cpu family  : 6
model       : 78
model name  : Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz
stepping    : 3
microcode   : 0xea
cpu MHz           : 800.071
cache size  : 4096 KB
physical id : 0
siblings    : 4
core id           : 1
cpu cores   : 2
apicid            : 3
initial apicid    : 3
fpu         : yes
fpu_exception     : yes
cpuid level : 22
wp          : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb
rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology
nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est
tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt
```

```
tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch
cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi
flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms
invpcid rtm mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1
xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear
flush_l1d
bugs        : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs taa itlb_multihit srbds
bogomips    : 5599.85
clflush size      : 64
cache_alignment   : 64
address sizes     : 39 bits physical, 48 bits virtual
power management:
```

RAM

```
MemTotal:        7544828 kB
MemFree:         5568100 kB
MemAvailable:    6427248 kB
Buffers:          113108 kB
Cached:          1035960 kB
SwapCached:            0 kB
Active:           904208 kB
Inactive:         725648 kB
Active(anon):     566884 kB
Inactive(anon):    19088 kB
Active(file):     337324 kB
Inactive(file):   706560 kB
Unevictable:       84996 kB
Mlocked:               0 kB
SwapTotal:       2097148 kB
SwapFree:        2097148 kB
Dirty:             10864 kB
Writeback:             0 kB
AnonPages:        565832 kB
Mapped:           355560 kB
Shmem:            108440 kB
KReclaimable:      69280 kB
Slab:             183332 kB
SReclaimable:      69280 kB
SUnreclaim:       114052 kB
KernelStack:        8480 kB
PageTables:        13308 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:     5869560 kB
Committed_AS:    4209324 kB
VmallocTotal:   34359738367 kB
VmallocUsed:       30328 kB
```

```
VmallocChunk:           0 kB
Percpu:              3312 kB
HardwareCorrupted:      0 kB
AnonHugePages:          0 kB
ShmemHugePages:         0 kB
ShmemPmdMapped:         0 kB
FileHugePages:          0 kB
FilePmdMapped:          0 kB
CmaTotal:               0 kB
CmaFree:                0 kB
HugePages_Total:        0
HugePages_Free:         0
HugePages_Rsvd:         0
HugePages_Surp:         0
Hugepagesize:        2048 kB
Hugetlb:                0 kB
DirectMap4k:       227552 kB
DirectMap2M:      4401152 kB
DirectMap1G:      3145728 kB
```

Software

On this Setup Linux is not Virtualized, and it runs with the following software:

```
Distributor ID: Linuxmint
Description:    Linux Mint 20.2
Release:        20.2
Codename:       uma
GCC:            9.3.0
```

Experimental Setup 2

Hardware

CPU

```
processor   : 0
vendor_id   : GenuineIntel
cpu family  : 6
model       : 142
model name  : Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz
stepping    : 10
microcode   : 0xffffffff
cpu MHz           : 1991.999
cache size  : 8192 KB
physical id : 0
siblings    : 1
core id           : 0
cpu cores   : 1
apicid            : 0
```

```
initial apicid    : 0
fpu         : yes
fpu_exception     : yes
cpuid level : 22
wp          : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush mmx fxsr sse sse2 ss syscall nx pdpe1gb rdtscp lm
constant_tsc arch_perfmon nopl xtopology tsc_reliable nonstop_tsc cpuid pni
pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt
tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm
3dnowprefetch invpcid_single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust
bmi1 avx2 smep bmi2 invpcid rdseed adx smap clflushopt xsaveopt xsavec
xgetbv1 xsaves arat flush_l1d arch_capabilities
bugs        : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs srbds
bogomips    : 3983.99
clflush size      : 64
cache_alignment   : 64
address sizes     : 45 bits physical, 48 bits virtual
power management:

processor   : 1
vendor_id   : GenuineIntel
cpu family  : 6
model       : 142
model name  : Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz
stepping    : 10
microcode   : 0xffffffff
cpu MHz           : 1991.999
cache size  : 8192 KB
physical id : 2
siblings    : 1
core id           : 0
cpu cores   : 1
apicid            : 2
initial apicid    : 2
fpu         : yes
fpu_exception     : yes
cpuid level : 22
wp          : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush mmx fxsr sse sse2 ss syscall nx pdpe1gb rdtscp lm
constant_tsc arch_perfmon nopl xtopology tsc_reliable nonstop_tsc cpuid pni
pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt
tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm
3dnowprefetch invpcid_single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust
bmi1 avx2 smep bmi2 invpcid rdseed adx smap clflushopt xsaveopt xsavec
xgetbv1 xsaves arat flush_l1d arch_capabilities
bugs        : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs srbds
```

```
bogomips    : 3983.99
clflush size      : 64
cache_alignment   : 64
address sizes     : 45 bits physical, 48 bits virtual
power management:

processor   : 2
vendor_id   : GenuineIntel
cpu family  : 6
model       : 142
model name  : Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz
stepping    : 10
microcode   : 0xffffffff
cpu MHz           : 1991.999
cache size  : 8192 KB
physical id : 4
siblings    : 1
core id           : 0
cpu cores   : 1
apicid            : 4
initial apicid    : 4
fpu         : yes
fpu_exception     : yes
cpuid level : 22
wp          : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush mmx fxsr sse sse2 ss syscall nx pdpe1gb rdtscp lm
constant_tsc arch_perfmon nopl xtopology tsc_reliable nonstop_tsc cpuid pni
pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt
tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm
3dnowprefetch invpcid_single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust
bmi1 avx2 smep bmi2 invpcid rdseed adx smap clflushopt xsaveopt xsavec
xgetbv1 xsaves arat flush_l1d arch_capabilities
bugs        : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs srbds
bogomips    : 3983.99
clflush size      : 64
cache_alignment   : 64
address sizes     : 45 bits physical, 48 bits virtual
power management:

processor   : 3
vendor_id   : GenuineIntel
cpu family  : 6
model       : 142
model name  : Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz
stepping    : 10
microcode   : 0xffffffff
cpu MHz           : 1991.999
cache size  : 8192 KB
```

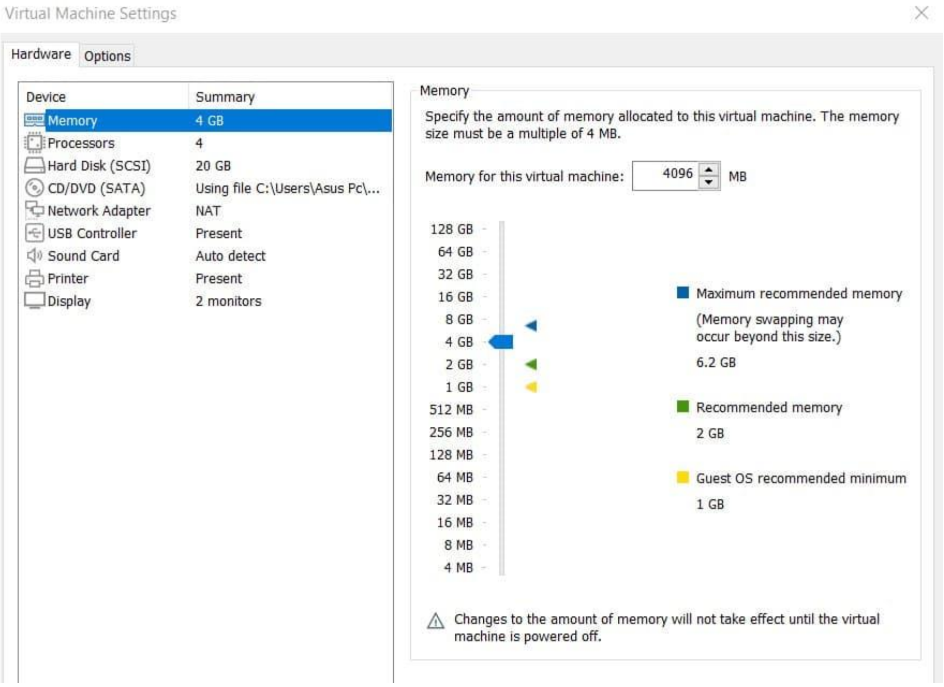
```
physical id : 6
siblings    : 1
core id           : 0
cpu cores   : 1
apicid            : 6
initial apicid    : 6
fpu         : yes
fpu_exception     : yes
cpuid level : 22
wp          : yes
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush mmx fxsr sse sse2 ss syscall nx pdpe1gb rdtscp lm
constant_tsc arch_perfmon nopl xtopology tsc_reliable nonstop_tsc cpuid pni
pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt
tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm
3dnowprefetch invpcid_single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust
bmi1 avx2 smep bmi2 invpcid rdseed adx smap clflushopt xsaveopt xsavec
xgetbv1 xsaves arat flush_l1d arch_capabilities
bugs        : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs srbds
bogomips    : 3983.99
clflush size      : 64
cache_alignment   : 64
address sizes     : 45 bits physical, 48 bits virtual
power management:
```

RAM

```
MemTotal:       3988028 kB
MemFree:        1615208 kB
MemAvailable:   2524864 kB
Buffers:          50904 kB
Cached:         1049492 kB
SwapCached:           0 kB
Active:          451812 kB
Inactive:       1375208 kB
Active(anon):      2280 kB
Inactive(anon):  747840 kB
Active(file):    449532 kB
Inactive(file):  627368 kB
Unevictable:         16 kB
Mlocked:             16 kB
SwapTotal:       945368 kB
SwapFree:        945368 kB
Dirty:               32 kB
Writeback:            0 kB
AnonPages:       726672 kB
```

```
Mapped:          333760 kB
Shmem:            29268 kB
KReclaimable:     74656 kB
Slab:            160188 kB
SReclaimable:     74656 kB
SUnreclaim:       85532 kB
KernelStack:      11248 kB
PageTables:       15028 kB
NFS_Unstable:         0 kB
Bounce:               0 kB
WritebackTmp:         0 kB
CommitLimit:    2939380 kB
Committed_AS:   3812688 kB
VmallocTotal:   34359738367 kB
VmallocUsed:      60324 kB
VmallocChunk:         0 kB
Percpu:           92160 kB
HardwareCorrupted:    0 kB
AnonHugePages:        0 kB
ShmemHugePages:       0 kB
ShmemPmdMapped:       0 kB
FileHugePages:        0 kB
FilePmdMapped:        0 kB
HugePages_Total:      0
HugePages_Free:       0
HugePages_Rsvd:       0
HugePages_Surp:       0
Hugepagesize:      2048 kB
Hugetlb:              0 kB
DirectMap4k:     237376 kB
DirectMap2M:    3956736 kB
DirectMap1G:    2097152 kB
```

VIRTUAL MACHINE SETTINGS



Device	Summary
Memory	4 GB
Processors	4
Hard Disk (SCSI)	20 GB
CD/DVD (SATA)	Using file C:\Users\Asus Pc\...
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	2 monitors

Memory

Specify the amount of memory allocated to this virtual machine. The memory size must be a multiple of 4 MB.

Memory for this virtual machine: 4096 MB

🟦 Maximum recommended memory
(Memory swapping may occur beyond this size.)
6.2 GB

🟩 Recommended memory
2 GB

🟨 Guest OS recommended minimum
1 GB

⚠ Changes to the amount of memory will not take effect until the virtual machine is powered off.

Software

On this Setup Linux is Virtualized, and it runs with the following software:

- Ubuntu 21.04
- GCC 10.3.0

Performance, Speedup & Efficiency

Case study

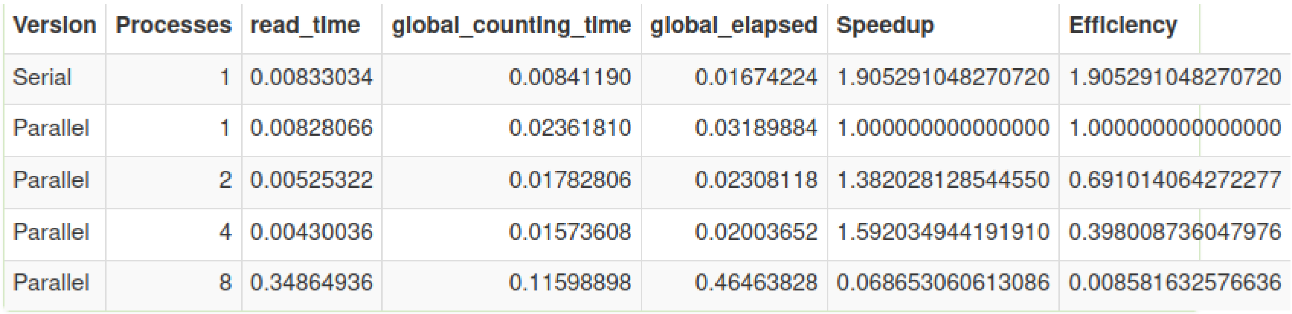
In this case study, the main purpose was to analyze the performance of our program in the following build setup:

● The sequential program is compiled with the gcc optimization O3.

● The parallel programs are all compiled with the gcc optimization O3. So here we want to highlight the difference between a simple sequential program compared to a parallel one, furthermore the case study is done on a non-random array of multiple size that are 5000000, 10000000, 2000000 and with different number of threads (0, 1, 2, 4, 8). In addition, there are four versions of reading the array from files.

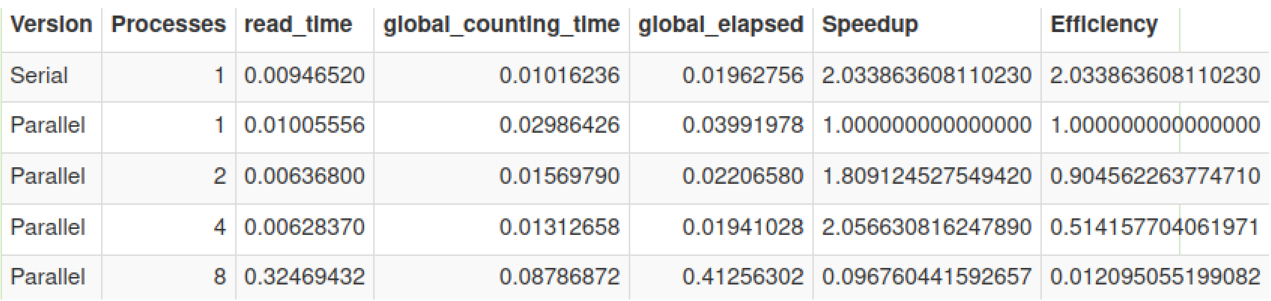
SIZE-5000000-V1

Setup 1



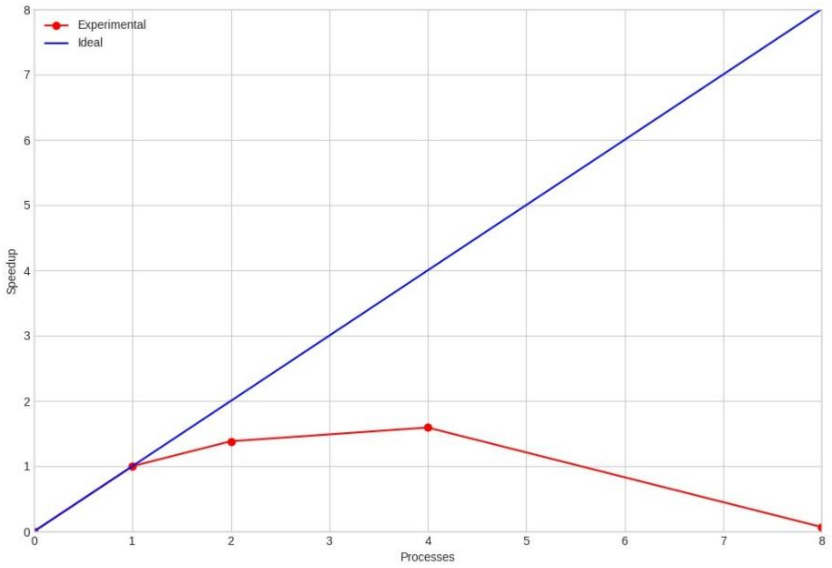
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.00833034	0.00841190	0.01674224	1.905291048270720	1.905291048270720
Parallel	1	0.00828066	0.02361810	0.03189884	1.000000000000000	1.000000000000000
Parallel	2	0.00525322	0.01782806	0.02308118	1.382028128544550	0.691014064272277
Parallel	4	0.00430036	0.01573608	0.02003652	1.592034944191910	0.398008736047976
Parallel	8	0.34864936	0.11598898	0.46463828	0.068653060613086	0.008581632576636

Setup 2

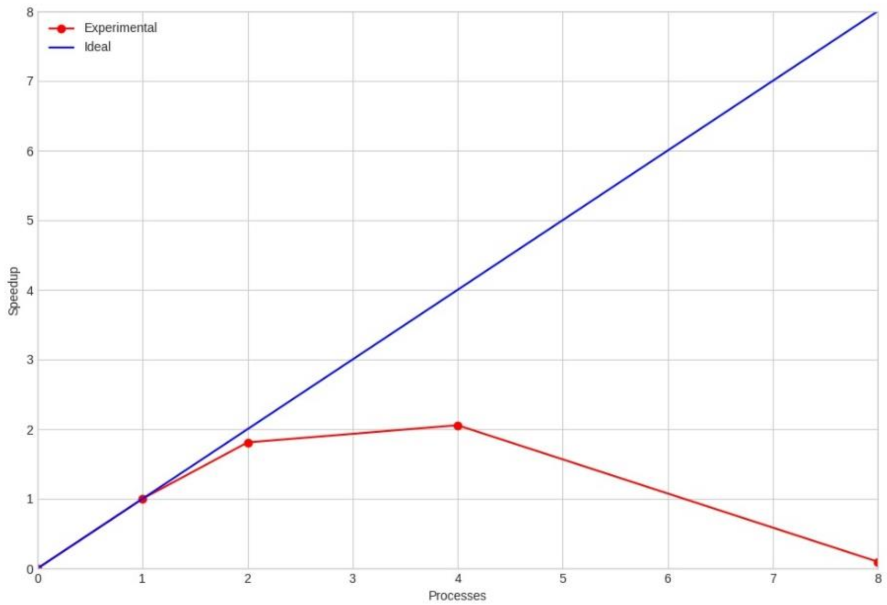


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.00946520	0.01016236	0.01962756	2.033863608110230	2.033863608110230
Parallel	1	0.01005556	0.02986426	0.03991978	1.000000000000000	1.000000000000000
Parallel	2	0.00636800	0.01569790	0.02206580	1.809124527549420	0.904562263774710
Parallel	4	0.00628370	0.01312658	0.01941028	2.056630816247890	0.514157704061971
Parallel	8	0.32469432	0.08786872	0.41256302	0.096760441592657	0.012095055199082

Setup 1

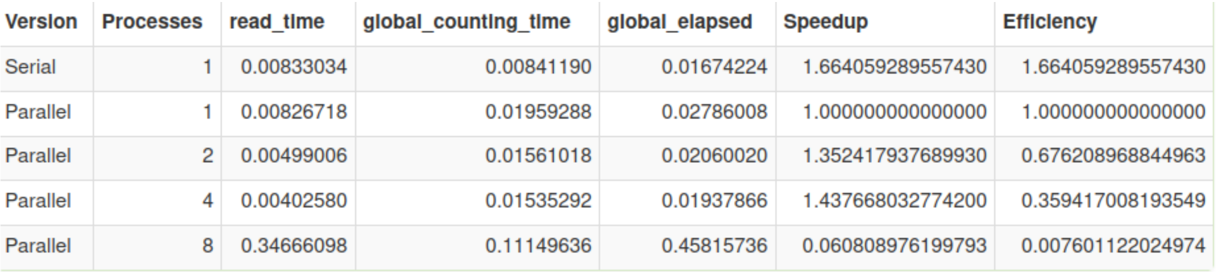


Setup 2



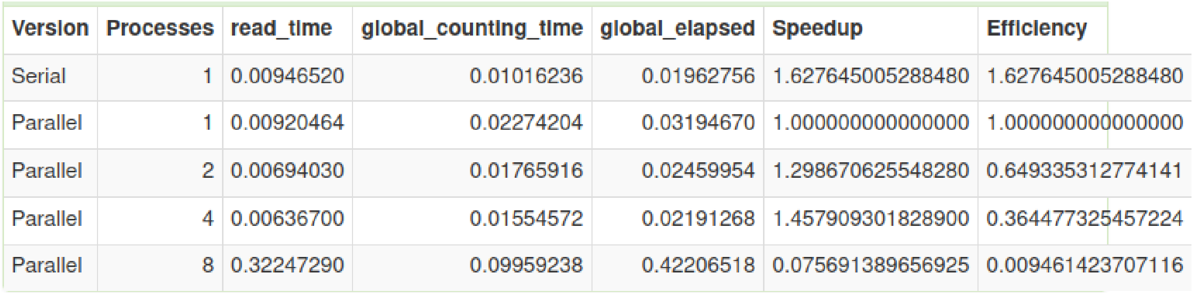
SIZE-5000000-V2

Setup 1



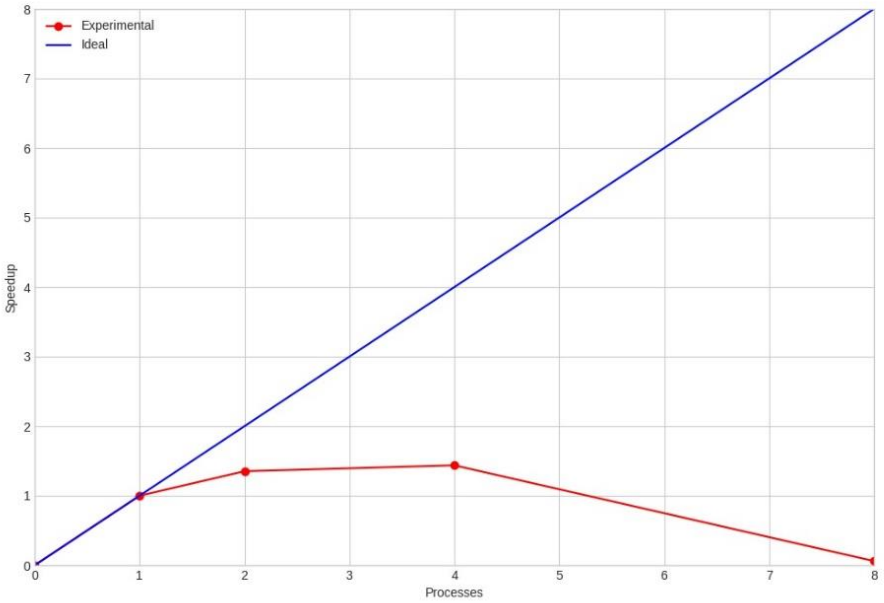
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.00833034	0.00841190	0.01674224	1.664059289557430	1.664059289557430
Parallel	1	0.00826718	0.01959288	0.02786008	1.000000000000000	1.000000000000000
Parallel	2	0.00499006	0.01561018	0.02060020	1.352417937689930	0.676208968844963
Parallel	4	0.00402580	0.01535292	0.01937866	1.437668032774200	0.359417008193549
Parallel	8	0.34666098	0.11149636	0.45815736	0.060808976199793	0.007601122024974

Setup 2

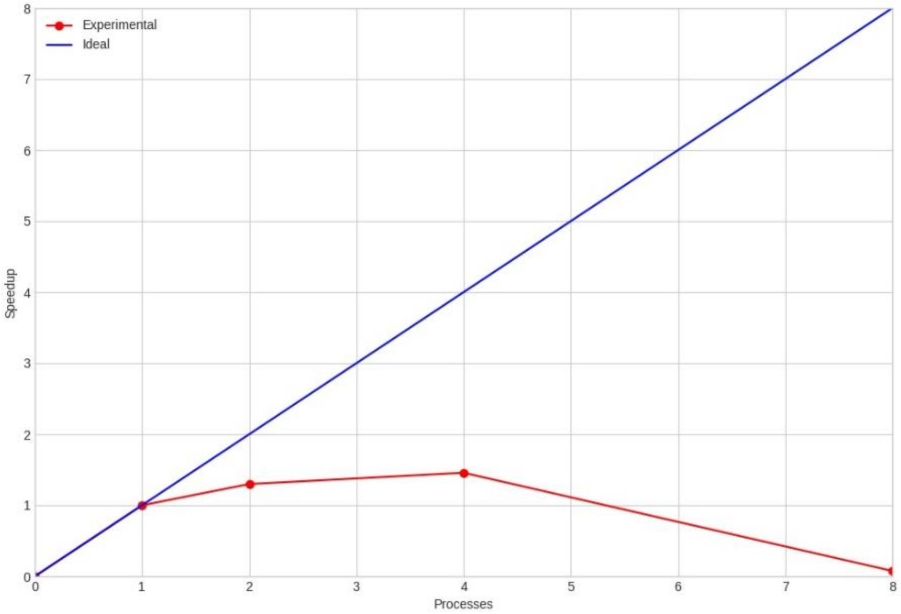


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.00946520	0.01016236	0.01962756	1.627645005288480	1.627645005288480
Parallel	1	0.00920464	0.02274204	0.03194670	1.000000000000000	1.000000000000000
Parallel	2	0.00694030	0.01765916	0.02459954	1.298670625548280	0.649335312774141
Parallel	4	0.00636700	0.01554572	0.02191268	1.457909301828900	0.364477325457224
Parallel	8	0.32247290	0.09959238	0.42206518	0.075691389656925	0.009461423707116

Setup 1

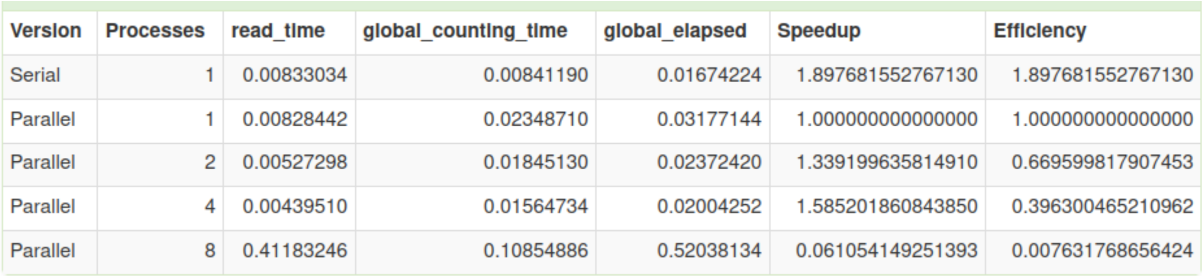


Setup 2



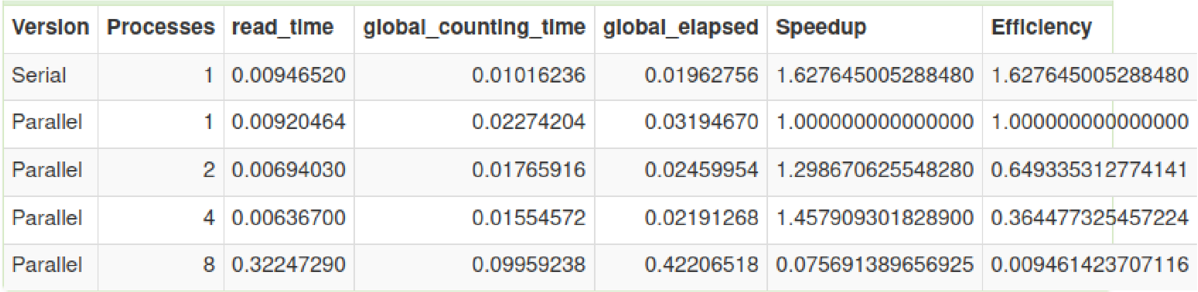
SIZE-5000000-V3

Setup 1



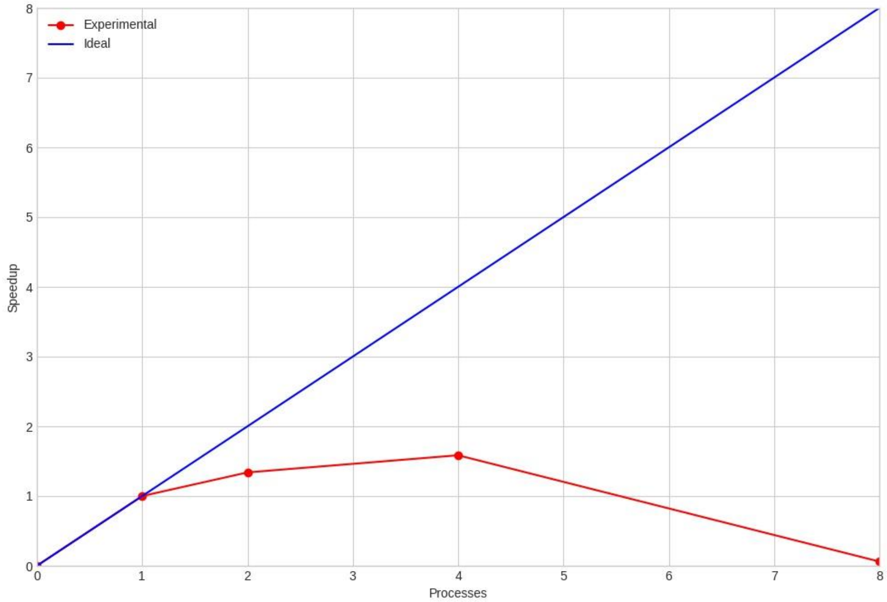
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.00833034	0.00841190	0.01674224	1.897681552767130	1.897681552767130
Parallel	1	0.00828442	0.02348710	0.03177144	1.000000000000000	1.000000000000000
Parallel	2	0.00527298	0.01845130	0.02372420	1.339199635814910	0.669599817907453
Parallel	4	0.00439510	0.01564734	0.02004252	1.585201860843850	0.396300465210962
Parallel	8	0.41183246	0.10854886	0.52038134	0.061054149251393	0.007631768656424

Setup 2

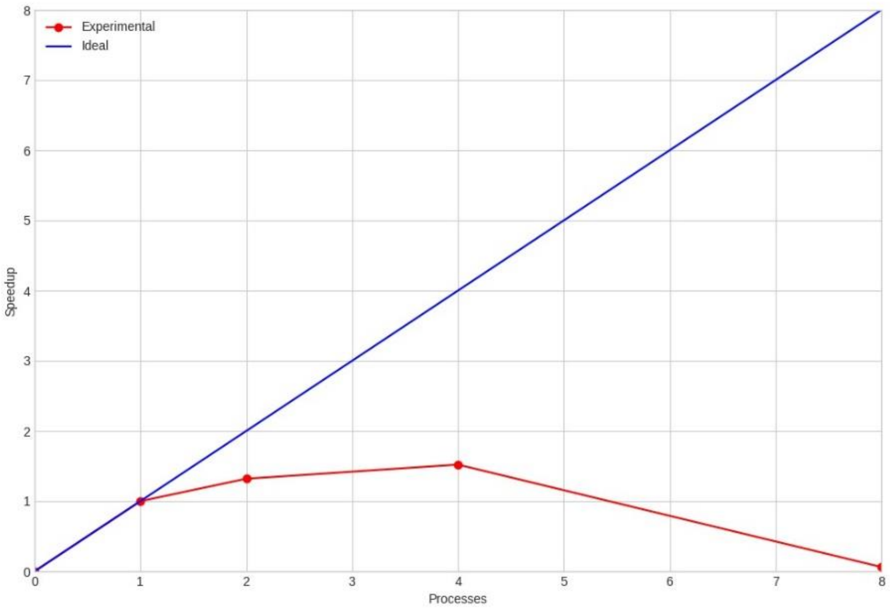


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.00946520	0.01016236	0.01962756	1.627645005288480	1.627645005288480
Parallel	1	0.00920464	0.02274204	0.03194670	1.000000000000000	1.000000000000000
Parallel	2	0.00694030	0.01765916	0.02459954	1.298670625548280	0.649335312774141
Parallel	4	0.00636700	0.01554572	0.02191268	1.457909301828900	0.364477325457224
Parallel	8	0.32247290	0.09959238	0.42206518	0.075691389656925	0.009461423707116

Setup 1

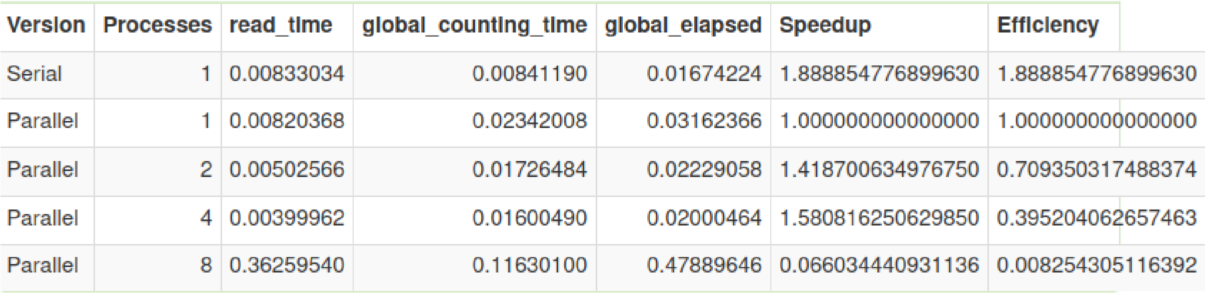


Setup 2



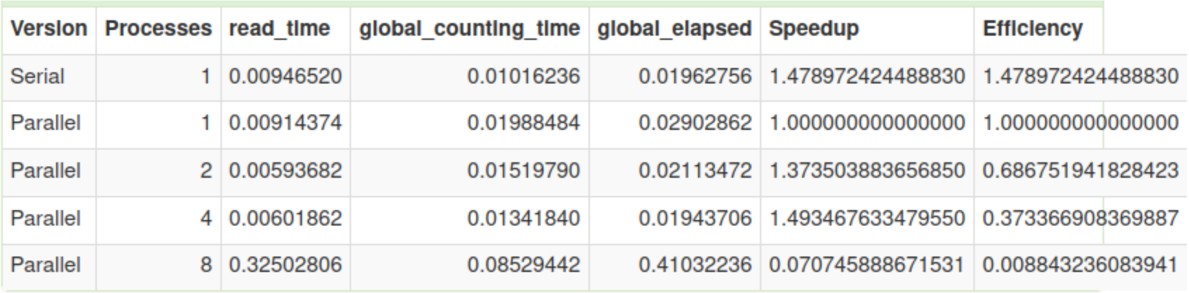
SIZE-5000000-V4

Setup 1



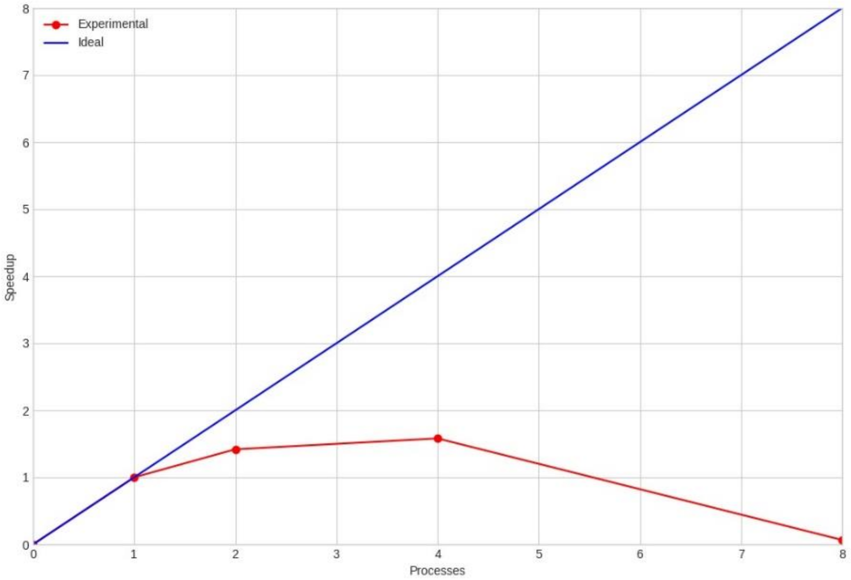
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.00833034	0.00841190	0.01674224	1.888854776899630	1.888854776899630
Parallel	1	0.00820368	0.02342008	0.03162366	1.000000000000000	1.000000000000000
Parallel	2	0.00502566	0.01726484	0.02229058	1.418700634976750	0.709350317488374
Parallel	4	0.00399962	0.01600490	0.02000464	1.580816250629850	0.395204062657463
Parallel	8	0.36259540	0.11630100	0.47889646	0.066034440931136	0.008254305116392

Setup 2

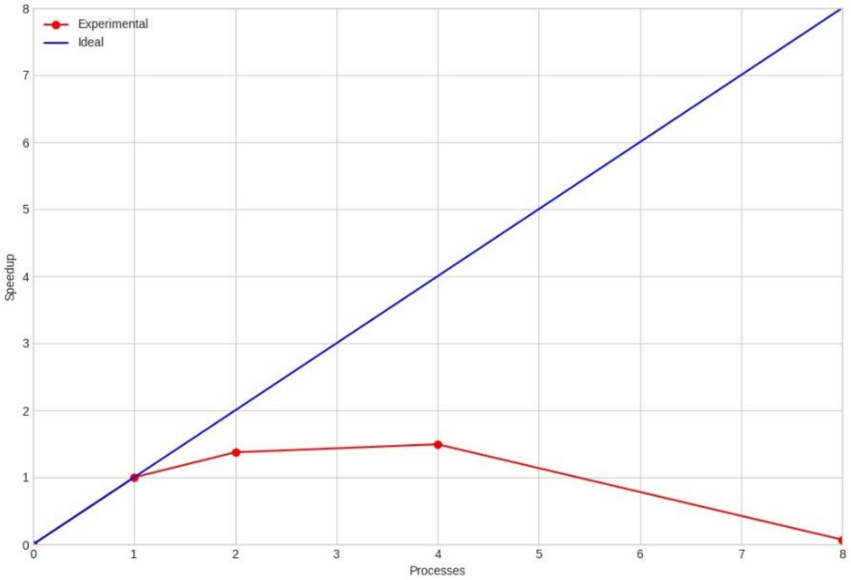


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.00946520	0.01016236	0.01962756	1.478972424488830	1.478972424488830
Parallel	1	0.00914374	0.01988484	0.02902862	1.000000000000000	1.000000000000000
Parallel	2	0.00593682	0.01519790	0.02113472	1.373503883656850	0.686751941828423
Parallel	4	0.00601862	0.01341840	0.01943706	1.493467633479550	0.373366908369887
Parallel	8	0.32502806	0.08529442	0.41032236	0.070745888671531	0.008843236083941

Setup 1

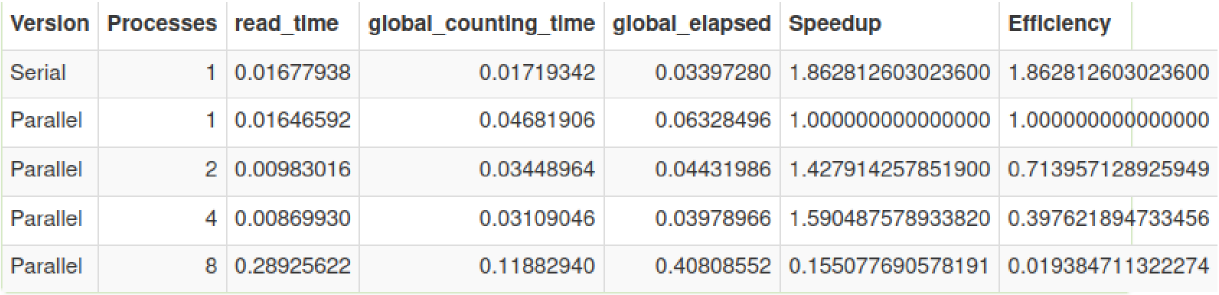


Setup 2



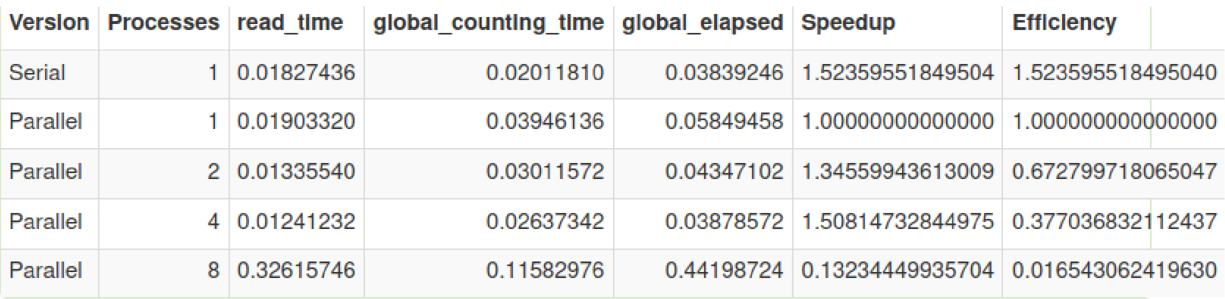
SIZE-10000000-V1

Setup 1



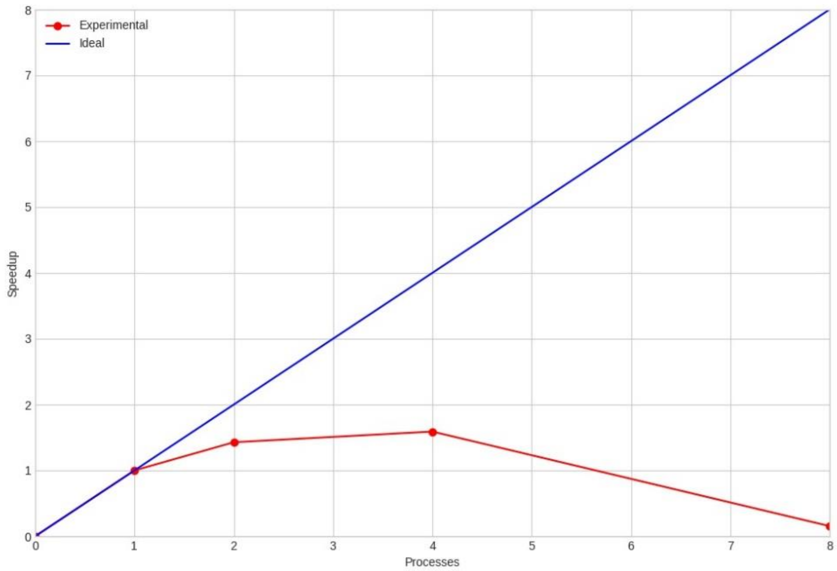
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.01677938	0.01719342	0.03397280	1.862812603023600	1.862812603023600
Parallel	1	0.01646592	0.04681906	0.06328496	1.000000000000000	1.000000000000000
Parallel	2	0.00983016	0.03448964	0.04431986	1.427914257851900	0.713957128925949
Parallel	4	0.00869930	0.03109046	0.03978966	1.590487578933820	0.397621894733456
Parallel	8	0.28925622	0.11882940	0.40808552	0.155077690578191	0.019384711322274

Setup 2

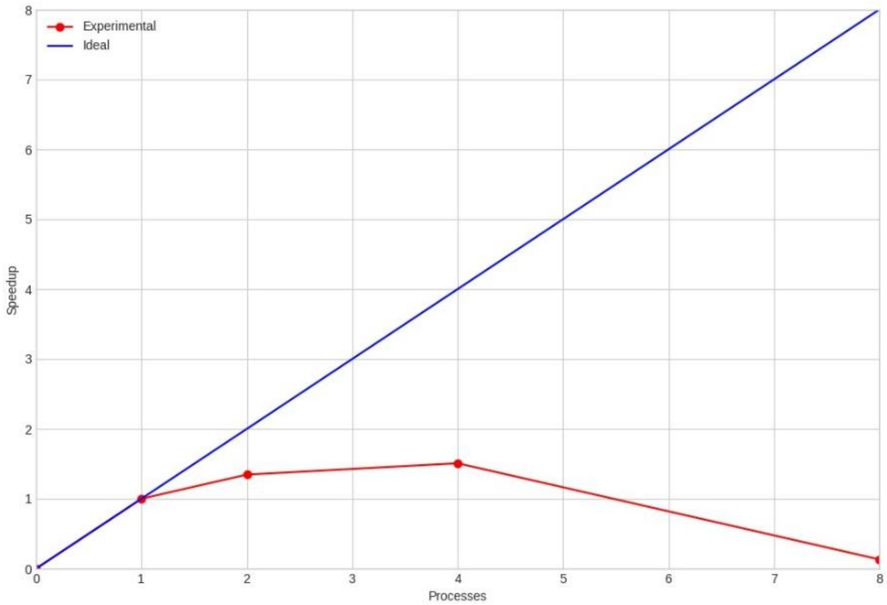


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.01827436	0.02011810	0.03839246	1.52359551849504	1.523595518495040
Parallel	1	0.01903320	0.03946136	0.05849458	1.00000000000000	1.000000000000000
Parallel	2	0.01335540	0.03011572	0.04347102	1.34559943613009	0.672799718065047
Parallel	4	0.01241232	0.02637342	0.03878572	1.50814732844975	0.377036832112437
Parallel	8	0.32615746	0.11582976	0.44198724	0.13234449935704	0.016543062419630

Setup 1

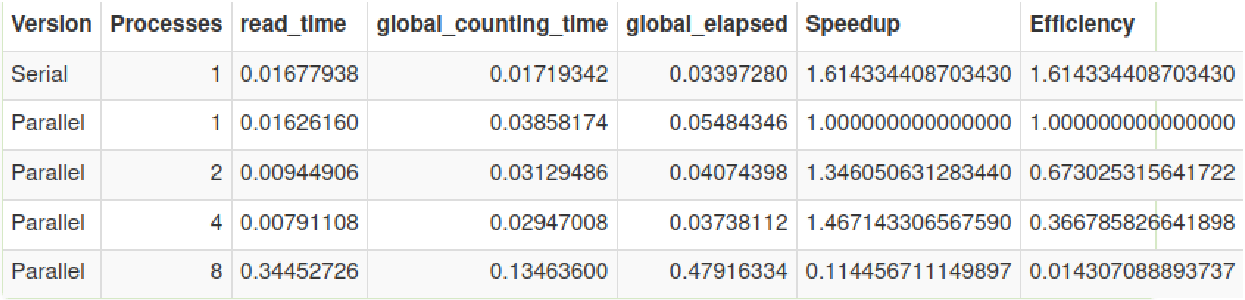


Setup 2



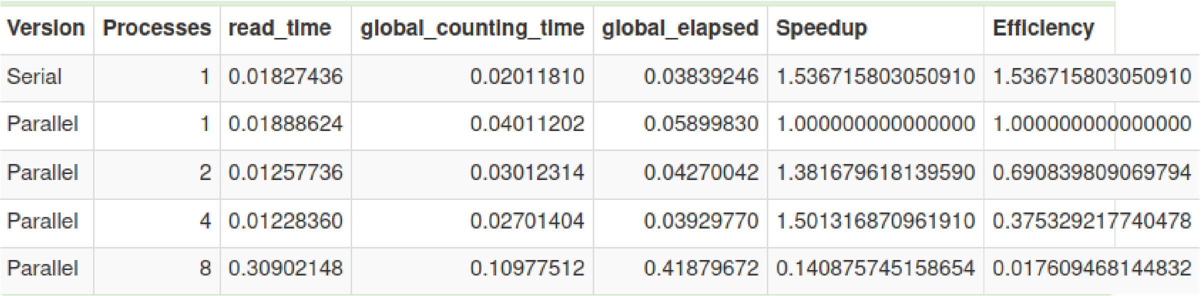
SIZE-10000000-V2

Setup 1



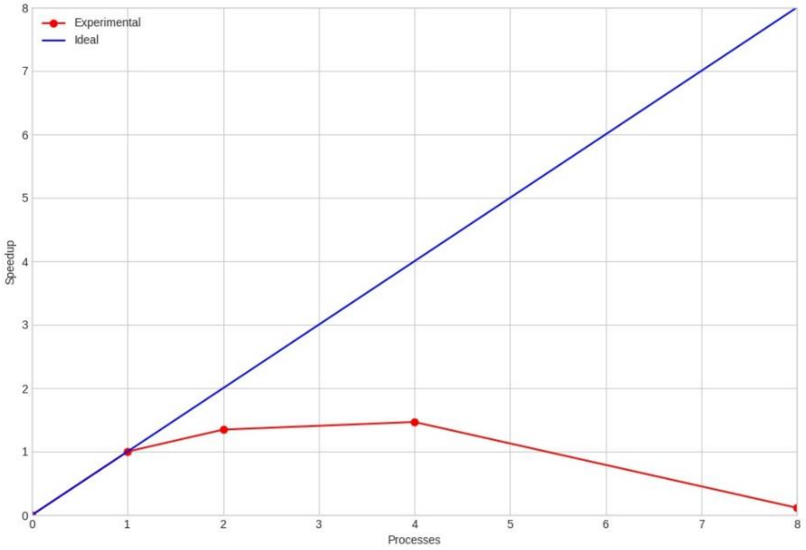
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.01677938	0.01719342	0.03397280	1.614334408703430	1.614334408703430
Parallel	1	0.01626160	0.03858174	0.05484346	1.000000000000000	1.000000000000000
Parallel	2	0.00944906	0.03129486	0.04074398	1.346050631283440	0.673025315641722
Parallel	4	0.00791108	0.02947008	0.03738112	1.467143306567590	0.366785826641898
Parallel	8	0.34452726	0.13463600	0.47916334	0.114456711149897	0.014307088893737

Setup 2

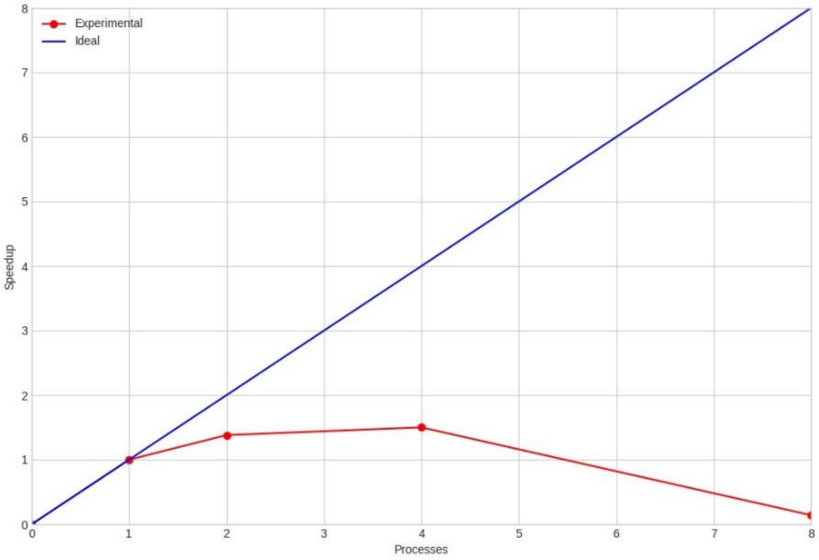


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.01827436	0.02011810	0.03839246	1.536715803050910	1.536715803050910
Parallel	1	0.01888624	0.04011202	0.05899830	1.000000000000000	1.000000000000000
Parallel	2	0.01257736	0.03012314	0.04270042	1.381679618139590	0.690839809069794
Parallel	4	0.01228360	0.02701404	0.03929770	1.501316870961910	0.375329217740478
Parallel	8	0.30902148	0.10977512	0.41879672	0.140875745158654	0.017609468144832

Setup 1

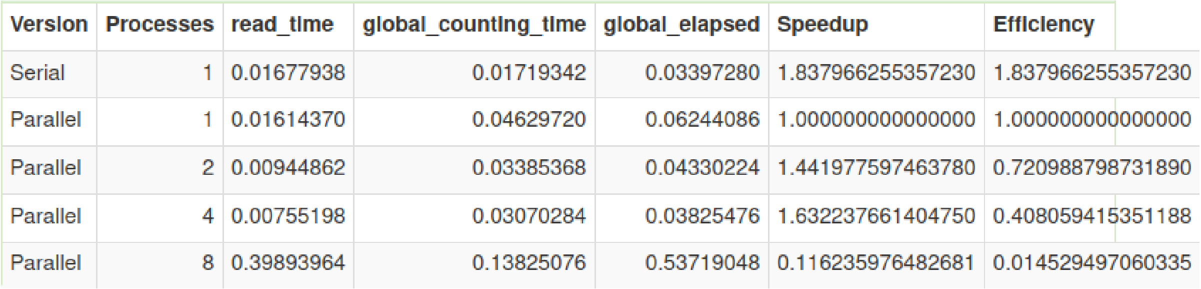


Setup 2



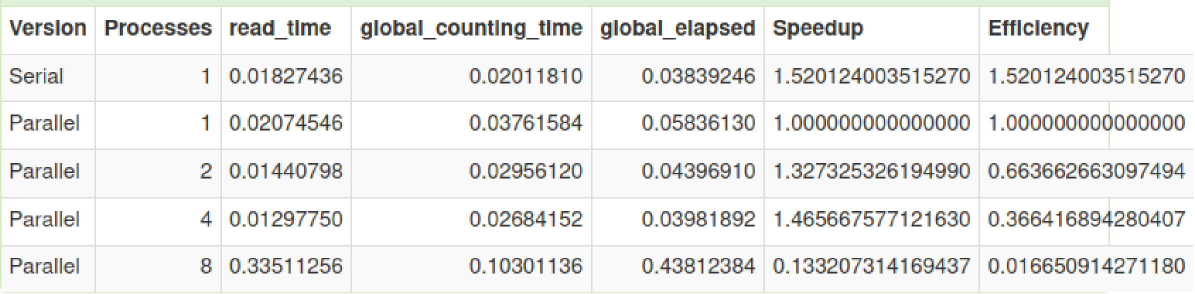
SIZE-10000000-V3

Setup 1



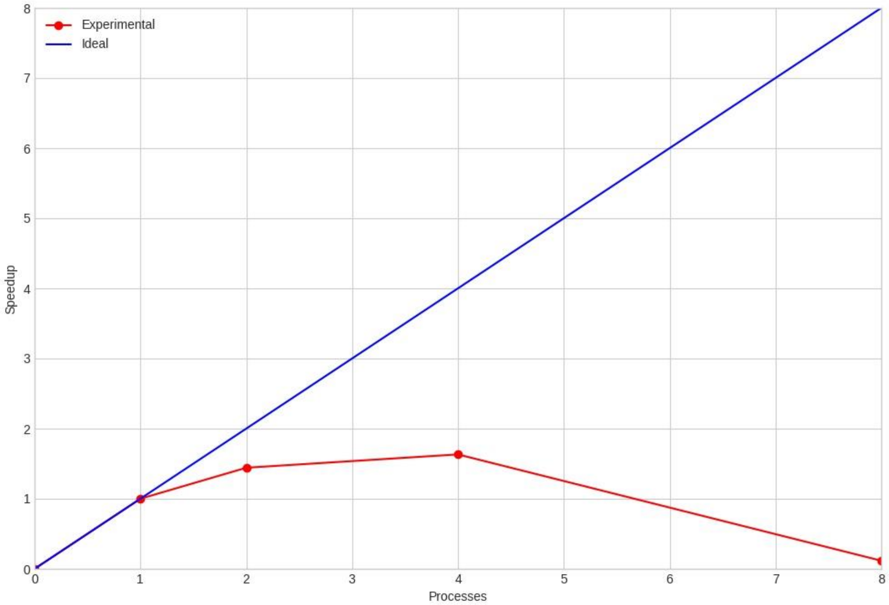
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.01677938	0.01719342	0.03397280	1.837966255357230	1.837966255357230
Parallel	1	0.01614370	0.04629720	0.06244086	1.000000000000000	1.000000000000000
Parallel	2	0.00944862	0.03385368	0.04330224	1.441977597463780	0.720988798731890
Parallel	4	0.00755198	0.03070284	0.03825476	1.632237661404750	0.408059415351188
Parallel	8	0.39893964	0.13825076	0.53719048	0.116235976482681	0.014529497060335

Setup 2

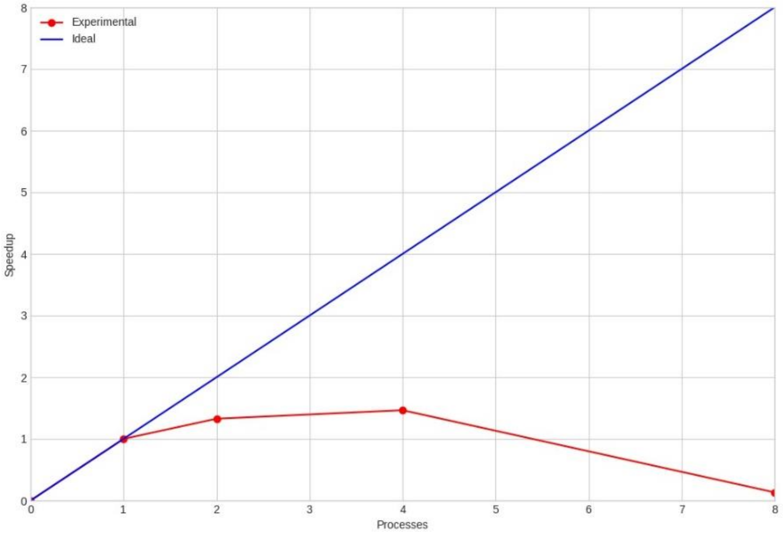


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.01827436	0.02011810	0.03839246	1.520124003515270	1.520124003515270
Parallel	1	0.02074546	0.03761584	0.05836130	1.000000000000000	1.000000000000000
Parallel	2	0.01440798	0.02956120	0.04396910	1.327325326194990	0.663662663097494
Parallel	4	0.01297750	0.02684152	0.03981892	1.465667577121630	0.366416894280407
Parallel	8	0.33511256	0.10301136	0.43812384	0.133207314169437	0.016650914271180

Setup 1

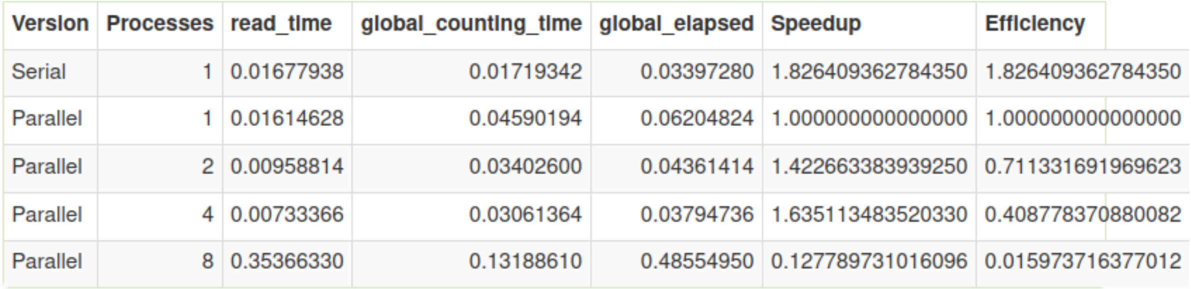


Setup 2



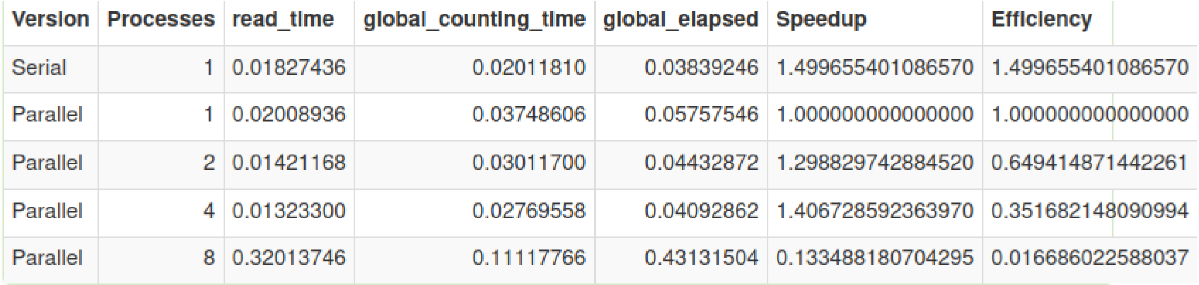
SIZE-10000000-V4

Setup 1



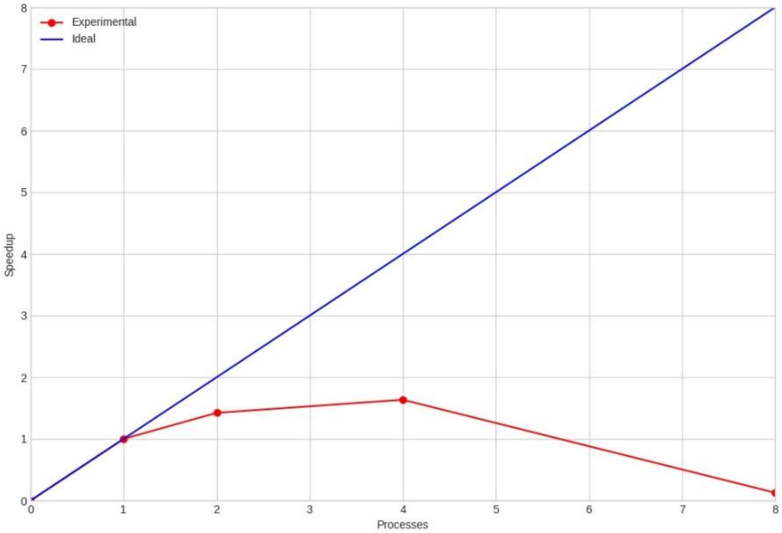
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.01677938	0.01719342	0.03397280	1.826409362784350	1.826409362784350
Parallel	1	0.01614628	0.04590194	0.06204824	1.000000000000000	1.000000000000000
Parallel	2	0.00958814	0.03402600	0.04361414	1.422663383939250	0.711331691969623
Parallel	4	0.00733366	0.03061364	0.03794736	1.635113483520330	0.408778370880082
Parallel	8	0.35366330	0.13188610	0.48554950	0.127789731016096	0.015973716377012

Setup 2

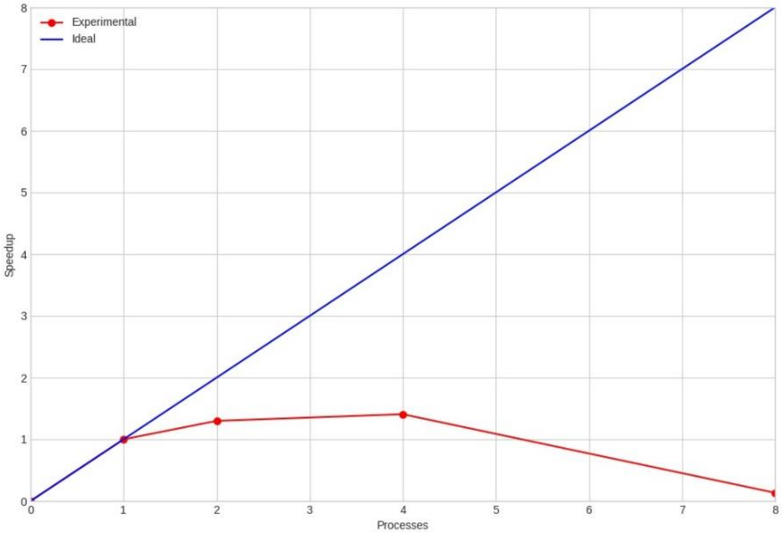


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.01827436	0.02011810	0.03839246	1.499655401086570	1.499655401086570
Parallel	1	0.02008936	0.03748606	0.05757546	1.000000000000000	1.000000000000000
Parallel	2	0.01421168	0.03011700	0.04432872	1.298829742884520	0.649414871442261
Parallel	4	0.01323300	0.02769558	0.04092862	1.406728592363970	0.351682148090994
Parallel	8	0.32013746	0.11117766	0.43131504	0.133488180704295	0.016686022588037

Setup 1

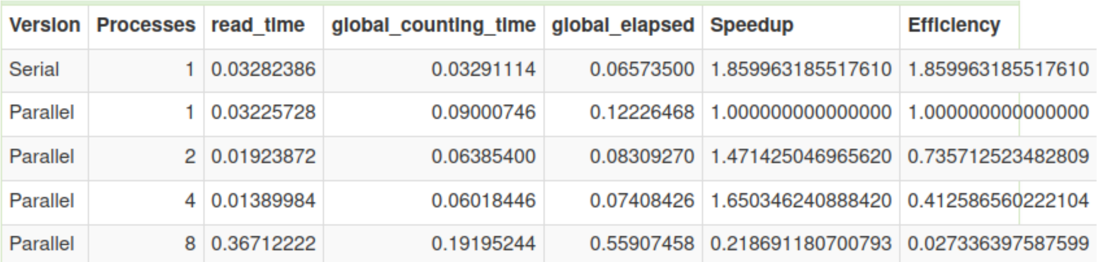


Setup 2



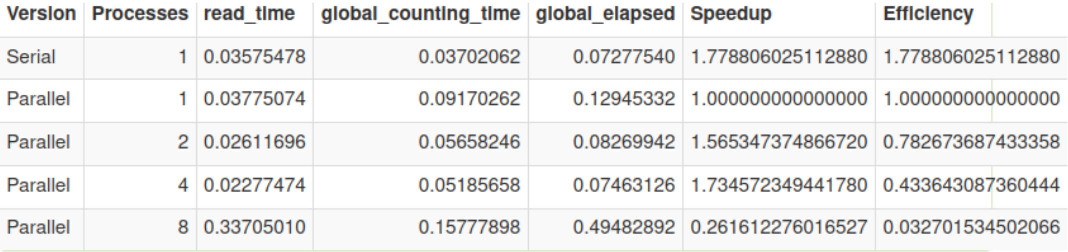
SIZE-20000000-V1

Setup 1



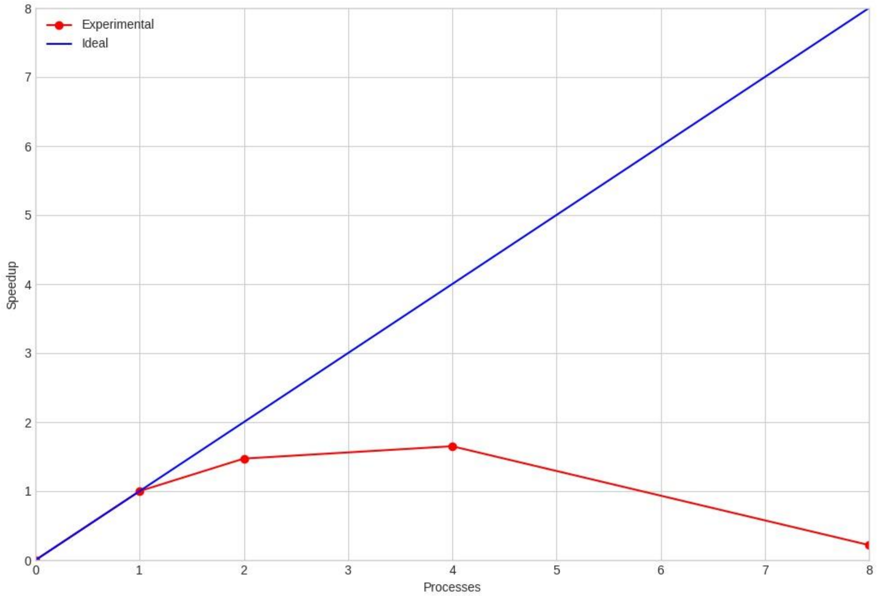
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.03282386	0.03291114	0.06573500	1.859963185517610	1.859963185517610
Parallel	1	0.03225728	0.09000746	0.12226468	1.000000000000000	1.000000000000000
Parallel	2	0.01923872	0.06385400	0.08309270	1.471425046965620	0.735712523482809
Parallel	4	0.01389984	0.06018446	0.07408426	1.650346240888420	0.412586560222104
Parallel	8	0.36712222	0.19195244	0.55907458	0.218691180700793	0.027336397587599

Setup 2

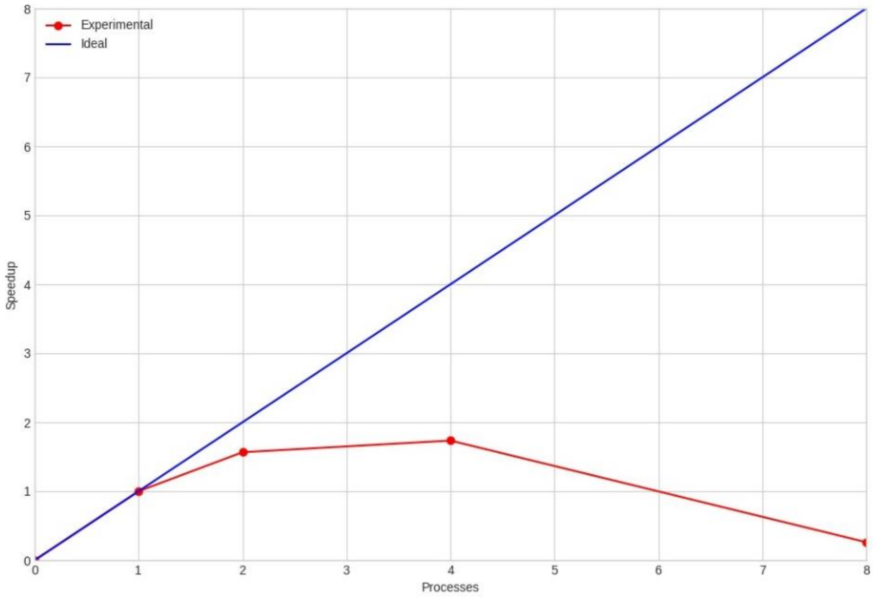


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.03575478	0.03702062	0.07277540	1.778806025112880	1.778806025112880
Parallel	1	0.03775074	0.09170262	0.12945332	1.000000000000000	1.000000000000000
Parallel	2	0.02611696	0.05658246	0.08269942	1.565347374866720	0.782673687433358
Parallel	4	0.02277474	0.05185658	0.07463126	1.734572349441780	0.433643087360444
Parallel	8	0.33705010	0.15777898	0.49482892	0.261612276016527	0.032701534502066

Setup 1

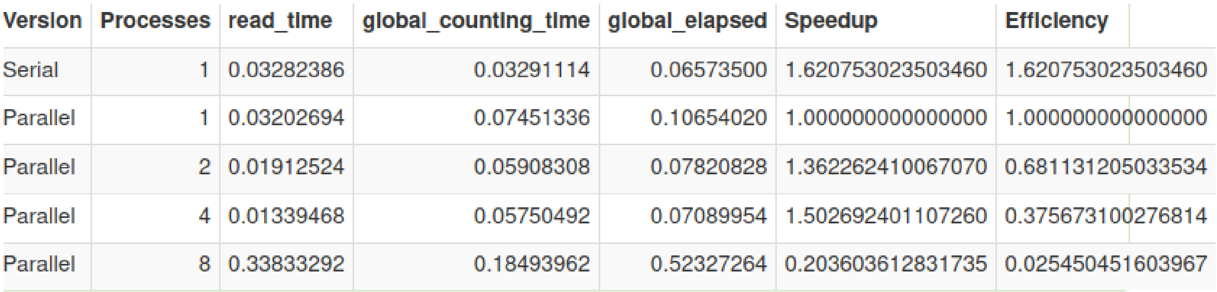


Setup 2



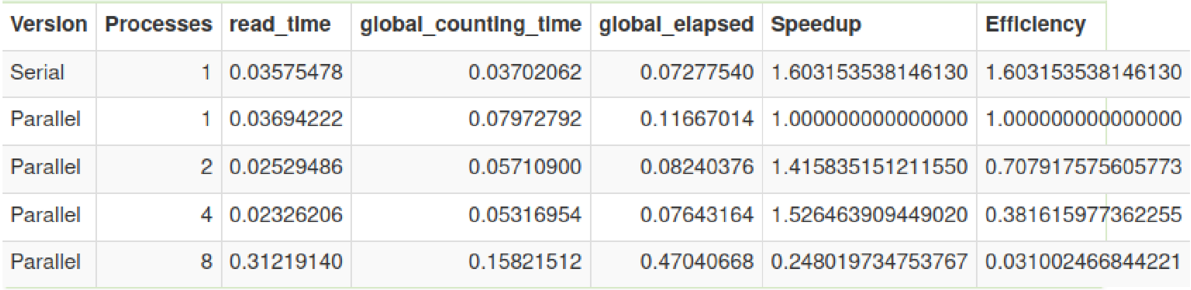
SIZE-20000000-V2

Setup 1



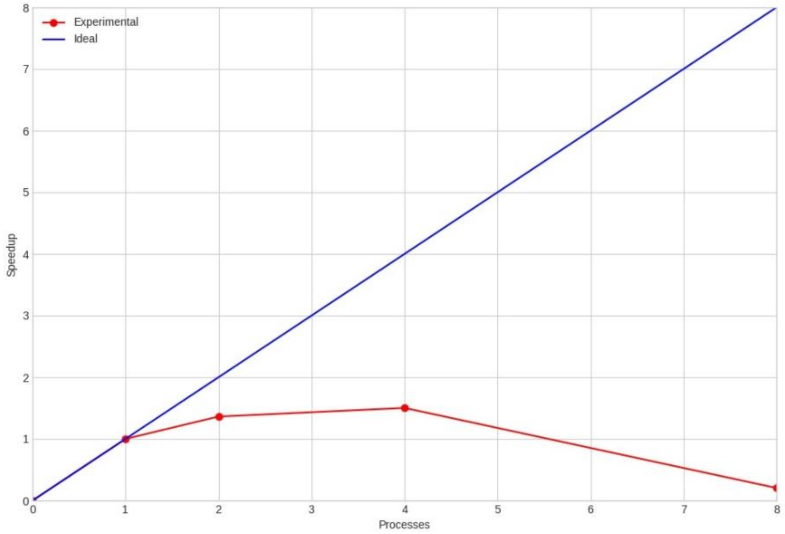
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.03282386	0.03291114	0.06573500	1.620753023503460	1.620753023503460
Parallel	1	0.03202694	0.07451336	0.10654020	1.000000000000000	1.000000000000000
Parallel	2	0.01912524	0.05908308	0.07820828	1.362262410067070	0.681131205033534
Parallel	4	0.01339468	0.05750492	0.07089954	1.502692401107260	0.375673100276814
Parallel	8	0.33833292	0.18493962	0.52327264	0.203603612831735	0.025450451603967

Setup 2

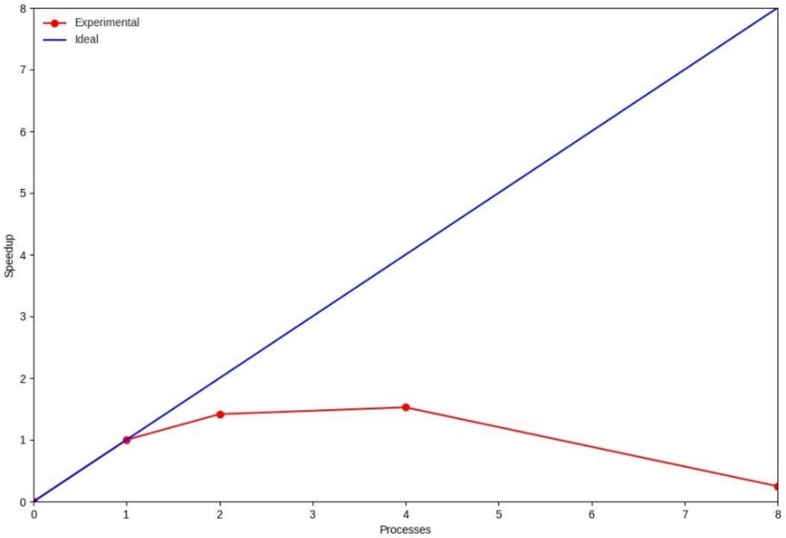


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.03575478	0.03702062	0.07277540	1.603153538146130	1.603153538146130
Parallel	1	0.03694222	0.07972792	0.11667014	1.000000000000000	1.000000000000000
Parallel	2	0.02529486	0.05710900	0.08240376	1.415835151211550	0.707917575605773
Parallel	4	0.02326206	0.05316954	0.07643164	1.526463909449020	0.381615977362255
Parallel	8	0.31219140	0.15821512	0.47040668	0.248019734753767	0.031002466844221

Setup 1

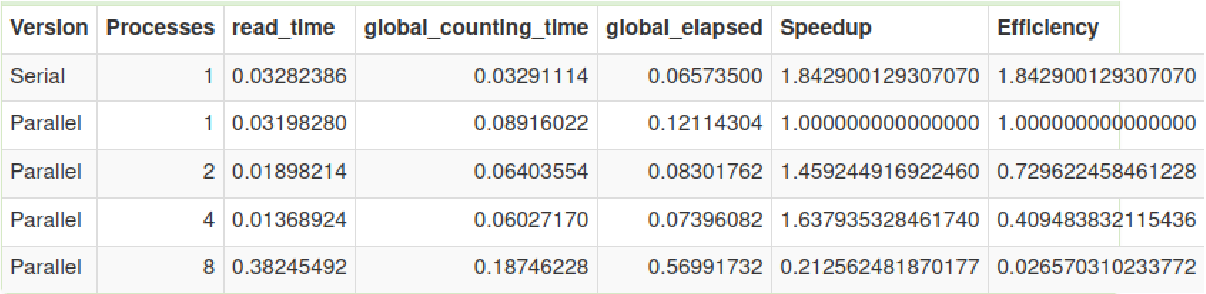


Setup 2



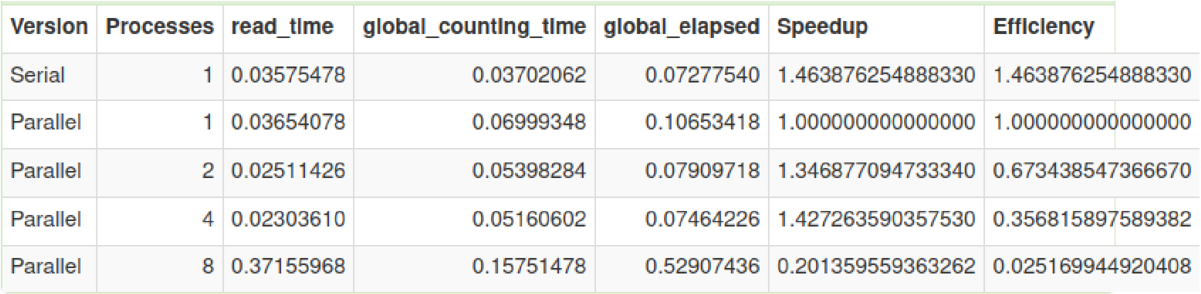
SIZE-20000000-V3

Setup 1



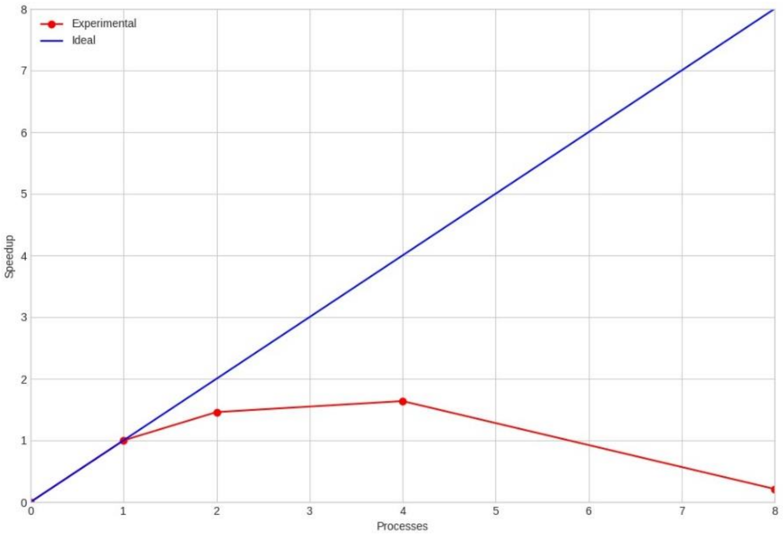
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.03282386	0.03291114	0.06573500	1.842900129307070	1.842900129307070
Parallel	1	0.03198280	0.08916022	0.12114304	1.000000000000000	1.000000000000000
Parallel	2	0.01898214	0.06403554	0.08301762	1.459244916922460	0.729622458461228
Parallel	4	0.01368924	0.06027170	0.07396082	1.637935328461740	0.409483832115436
Parallel	8	0.38245492	0.18746228	0.56991732	0.212562481870177	0.026570310233772

Setup 2

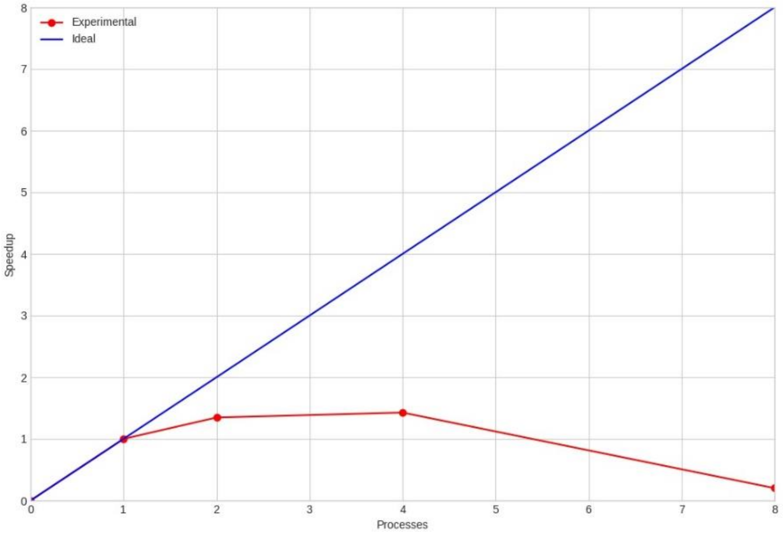


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.03575478	0.03702062	0.07277540	1.463876254888330	1.463876254888330
Parallel	1	0.03654078	0.06999348	0.10653418	1.000000000000000	1.000000000000000
Parallel	2	0.02511426	0.05398284	0.07909718	1.346877094733340	0.673438547366670
Parallel	4	0.02303610	0.05160602	0.07464226	1.427263590357530	0.356815897589382
Parallel	8	0.37155968	0.15751478	0.52907436	0.201359559363262	0.025169944920408

Setup 1

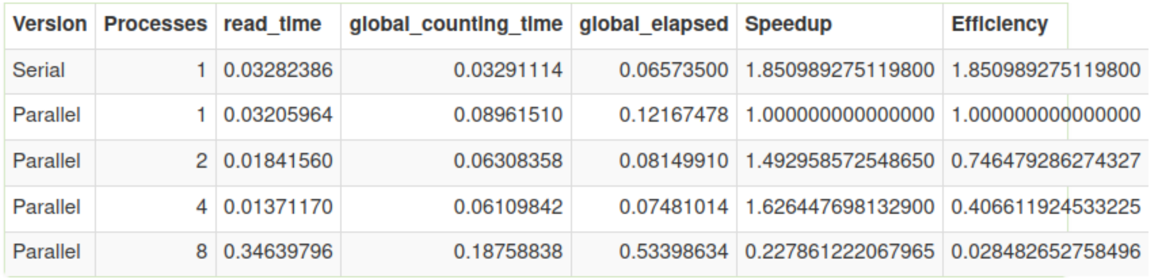


Setup 2



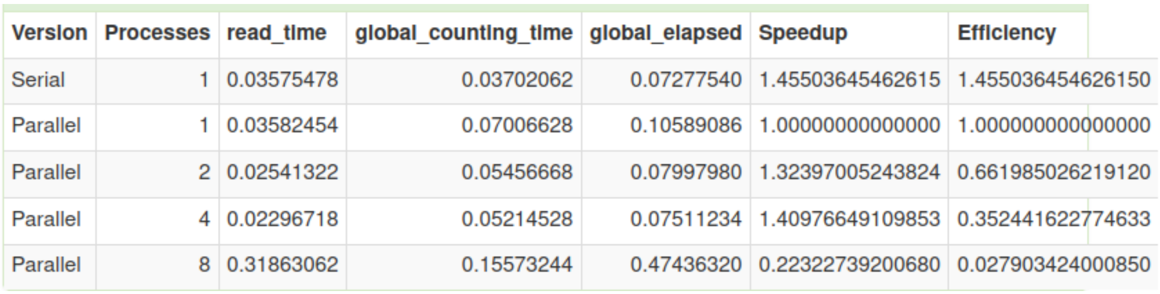
SIZE-20000000-V4

Setup 1



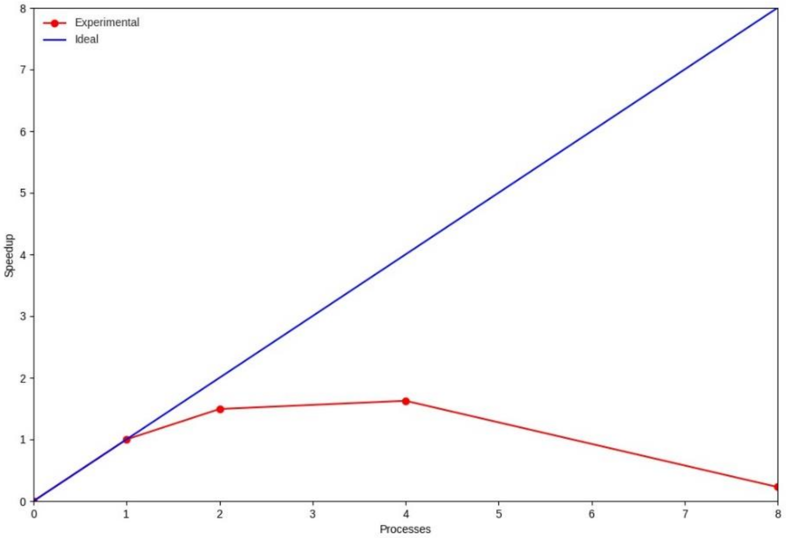
Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.03282386	0.03291114	0.06573500	1.850989275119800	1.850989275119800
Parallel	1	0.03205964	0.08961510	0.12167478	1.000000000000000	1.000000000000000
Parallel	2	0.01841560	0.06308358	0.08149910	1.492958572548650	0.746479286274327
Parallel	4	0.01371170	0.06109842	0.07481014	1.626447698132900	0.406611924533225
Parallel	8	0.34639796	0.18758838	0.53398634	0.227861222067965	0.028482652758496

Setup 2

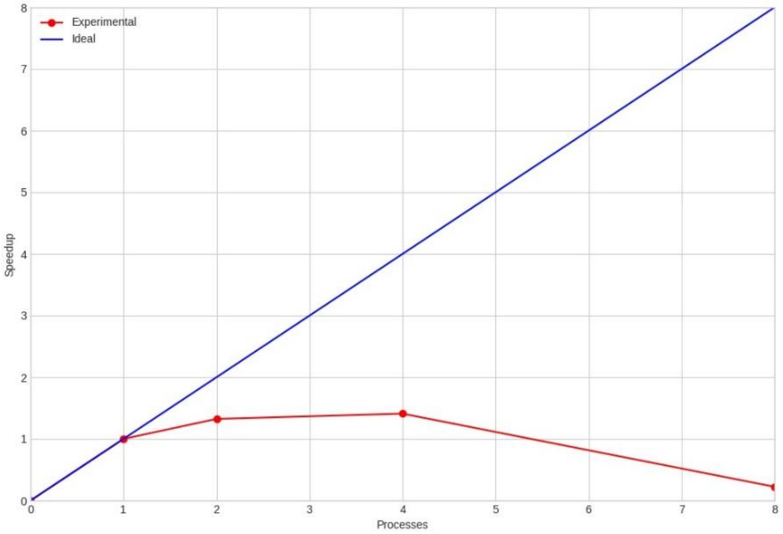


Version	Processes	read_time	global_counting_time	global_elapsed	Speedup	Efficiency
Serial	1	0.03575478	0.03702062	0.07277540	1.45503645462615	1.455036454626150
Parallel	1	0.03582454	0.07006628	0.10589086	1.00000000000000	1.000000000000000
Parallel	2	0.02541322	0.05456668	0.07997980	1.32397005243824	0.661985026219120
Parallel	4	0.02296718	0.05214528	0.07511234	1.40976649109853	0.352441622774633
Parallel	8	0.31863062	0.15573244	0.47436320	0.22322739200680	0.027903424000850

Setup 1



Setup 2



Considerations

Speedup

The speedup is computed in comparison to the one thread parallel program.

The maximum computed speedup is 2.056 in the 5000000-V1 test case with 4 threads and the Setup 2. This means that the parallel version is 2 times faster than the serial one. The reason it does not go over this value is probably the lack of parallelization in the sorting nested loop, which is the last for in the counting_sort function and the most expensive one, with a complexity of O(k) with k equals to (max(array) - min(array) +1).

The lowest speedup is 0.060 in the 5000000-V2 test case with 8 threads and the Setup 1.

We can observe that in each examinate case the speedup with 8 threads is the lowest on both the setups, the reason being the saturation of hardware resources, since the code is being executed on a single machine, and not a cluster.

Efficiency

In most cases the relative efficiency of each experimental setup is comparable. The difference between the various comparisons is around 5% to 30% with the efficiency of the Setup 2 being slightly lower than the Setup 1. We can observe a significative difference between the efficiency of the two setups especially in serial program. This is due to the overhead introduced by the virtualization of the OS in the Setup 2, and the different processor.

In general, the efficiency is constant and not depending on size of the array, so we can assume that this parallel solution is partially scalable.

Read Time

We note that the read time decreases especially in cases with 2 and 4 threads compared to the sequential one. In general, with 8 threads, it is remarkably like the read time of the serial and of the 1 thread parallel program.

We can see that there is an evident improvement in versions 3 and 4 with respect to the others due to the use of file view.

Elapsed Time

We notice that the best of the parallel performances is given when the algorithm runs on 4 processes. The cause of this could be that both systems are quad core.

Furthermore, by increasing the number of processes beyond 4, the algorithm begins to have very poor performance which makes it useless.

The best elapsed time detected is in V3 with size 500000 in Setup 2.
As expected, when the array size gets bigger, performance decrease.

Although the V1 is expected to be less performing because it is based on a shared file pointer, all measurements were made on a single system, therefore in this case it is faster than the other versions.

We observe that most MPI implementations are slower than serial implementation.

This could be caused by the overhead of the MPI environment, which is process based and not thread based.

Code Considerations

The only one *MPI_Barrier* present in the code is used to make the single rank check the correctness of the inputs.

To implement the sum between vectors operation, the *sumVectors* function has been implemented and then passed to *MPI_Op_create*.

The versions are selectable by defining the VERSION x directive with x = 1,2,3,4.

The *MPI_Allreduce* function was used to calculate the global maxima and minima.

The ability not to save the array has also been added by defining the NO_OUTPUT directive.

We considered only case study with O3 optimization because the other optimizations were not so relevant in terms of speedup.

In O0 optimization, the MPI implementation in single process was faster than the serial.
This could be due to MPI optimizations.

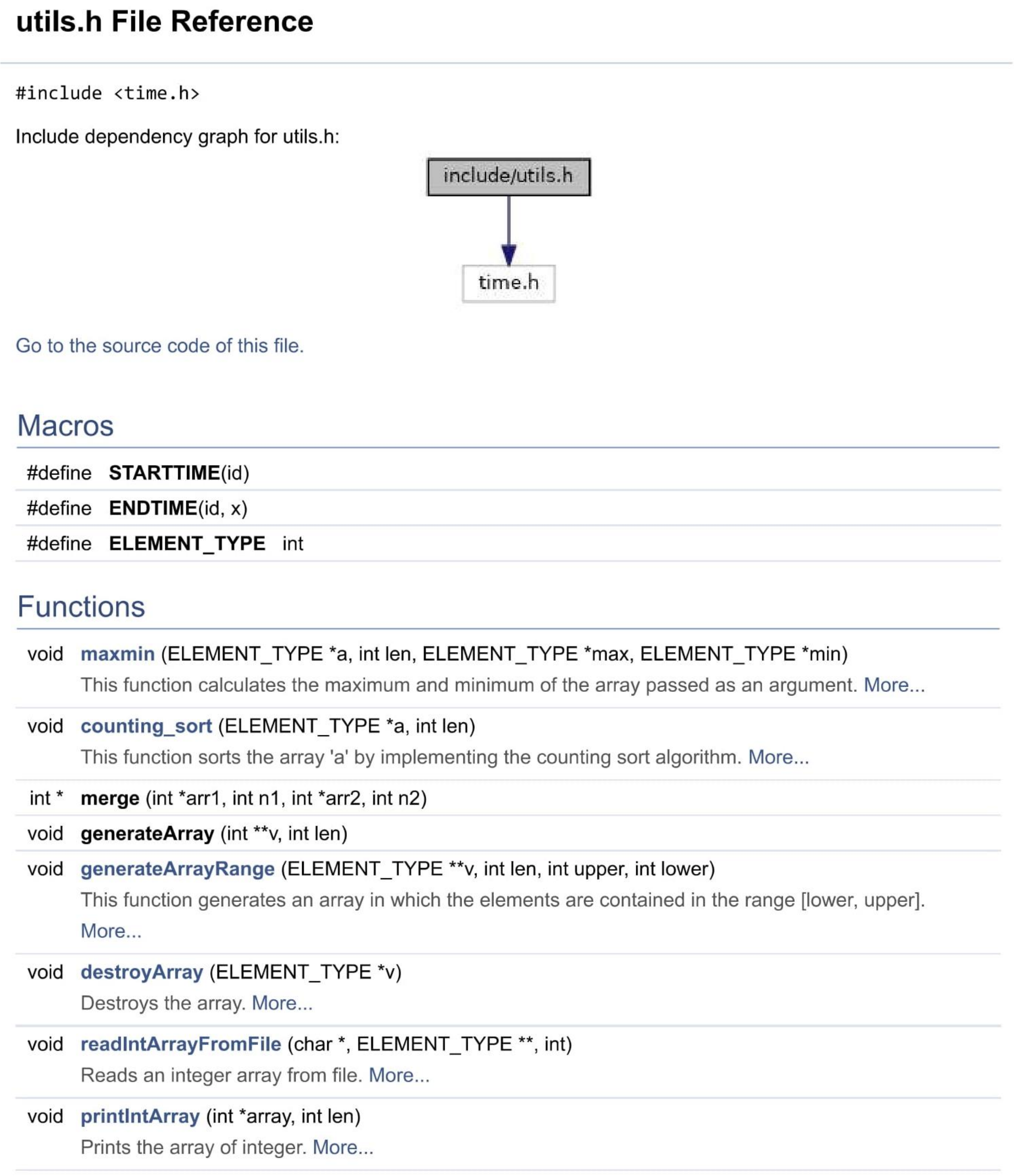
Test case

In the test folder there is a *test.c* file which contains the test cases.

The tests are performed to verify that the sorting is executed correctly.

Since there are 2 programs to test, we have performed our test in the sequential MPI algorithm, and we tested the correctness of the algorithm by using files.

API



utils.h File Reference

```
#include <time.h>
```

Include dependency graph for utils.h:

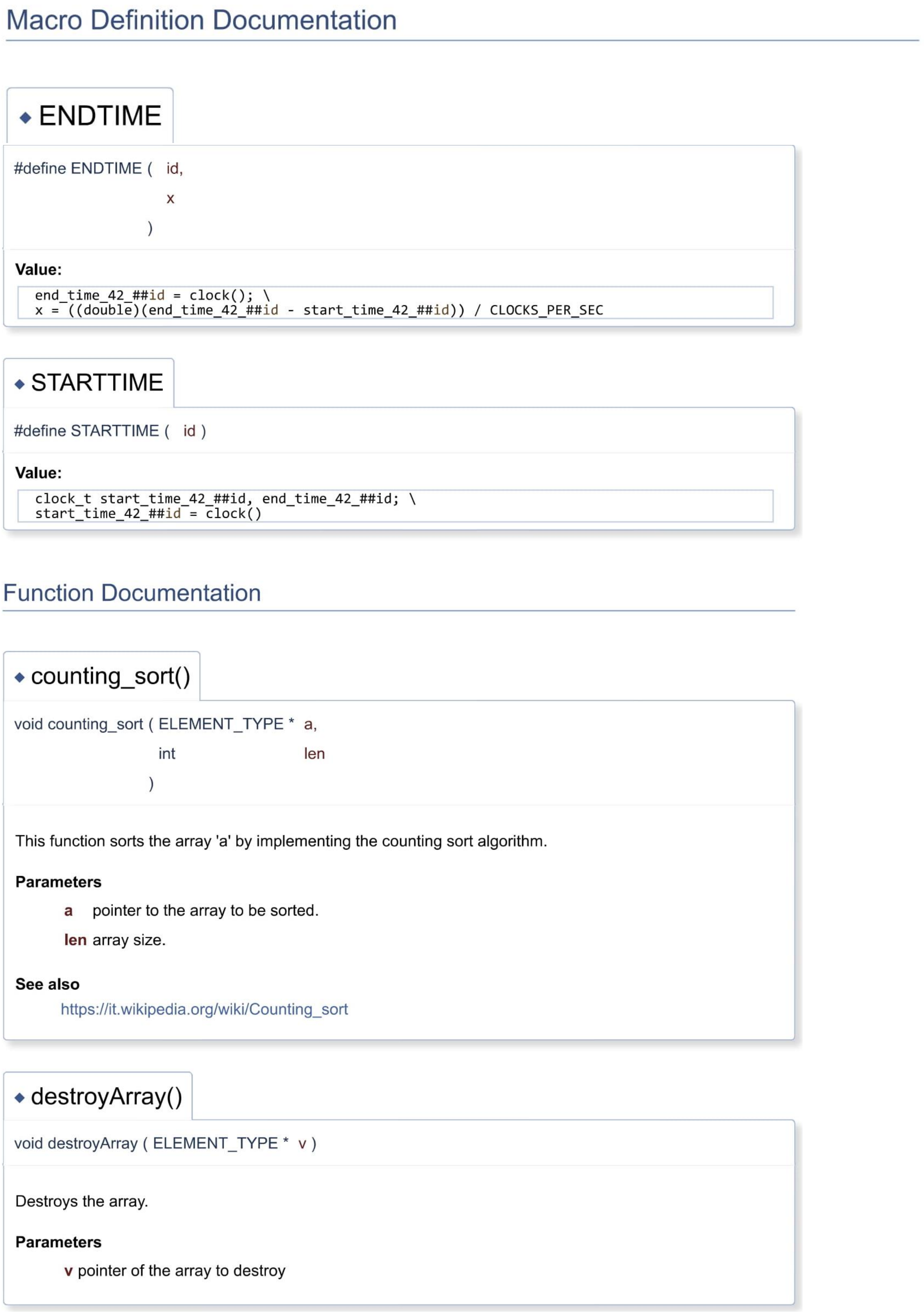
Go to the source code of this file.

Macros

#define	**STARTTIME**(id)
#define	**ENDTIME**(id, x)
#define	**ELEMENT_TYPE** int

Functions

void	**maxmin** (ELEMENT_TYPE *a, int len, ELEMENT_TYPE *max, ELEMENT_TYPE *min) This function calculates the maximum and minimum of the array passed as an argument. More...
void	**counting_sort** (ELEMENT_TYPE *a, int len) This function sorts the array 'a' by implementing the counting sort algorithm. More...
int *	**merge** (int *arr1, int n1, int *arr2, int n2)
void	**generateArray** (int **v, int len)
void	**generateArrayRange** (ELEMENT_TYPE **v, int len, int upper, int lower) This function generates an array in which the elements are contained in the range [lower, upper]. More...
void	**destroyArray** (ELEMENT_TYPE *v) Destroys the array. More...
void	**readIntArrayFromFile** (char *, ELEMENT_TYPE **, int) Reads an integer array from file. More...
void	**printIntArray** (int *array, int len) Prints the array of integer. More...



Macro Definition Documentation

◆ ENDTIME

```
#define ENDTIME (  id,
                   x
                )
```

Value:

```
end_time_42_##id = clock(); \
x = ((double)(end_time_42_##id - start_time_42_##id)) / CLOCKS_PER_SEC
```

◆ STARTTIME

```
#define STARTTIME (  id )
```

Value:

```
clock_t start_time_42_##id, end_time_42_##id; \
start_time_42_##id = clock()
```

Function Documentation

◆ counting_sort()

```
void counting_sort ( ELEMENT_TYPE *  a,
                     int             len
                   )
```

This function sorts the array 'a' by implementing the counting sort algorithm.

Parameters

a	pointer to the array to be sorted.
len	array size.

See also

https://it.wikipedia.org/wiki/Counting_sort

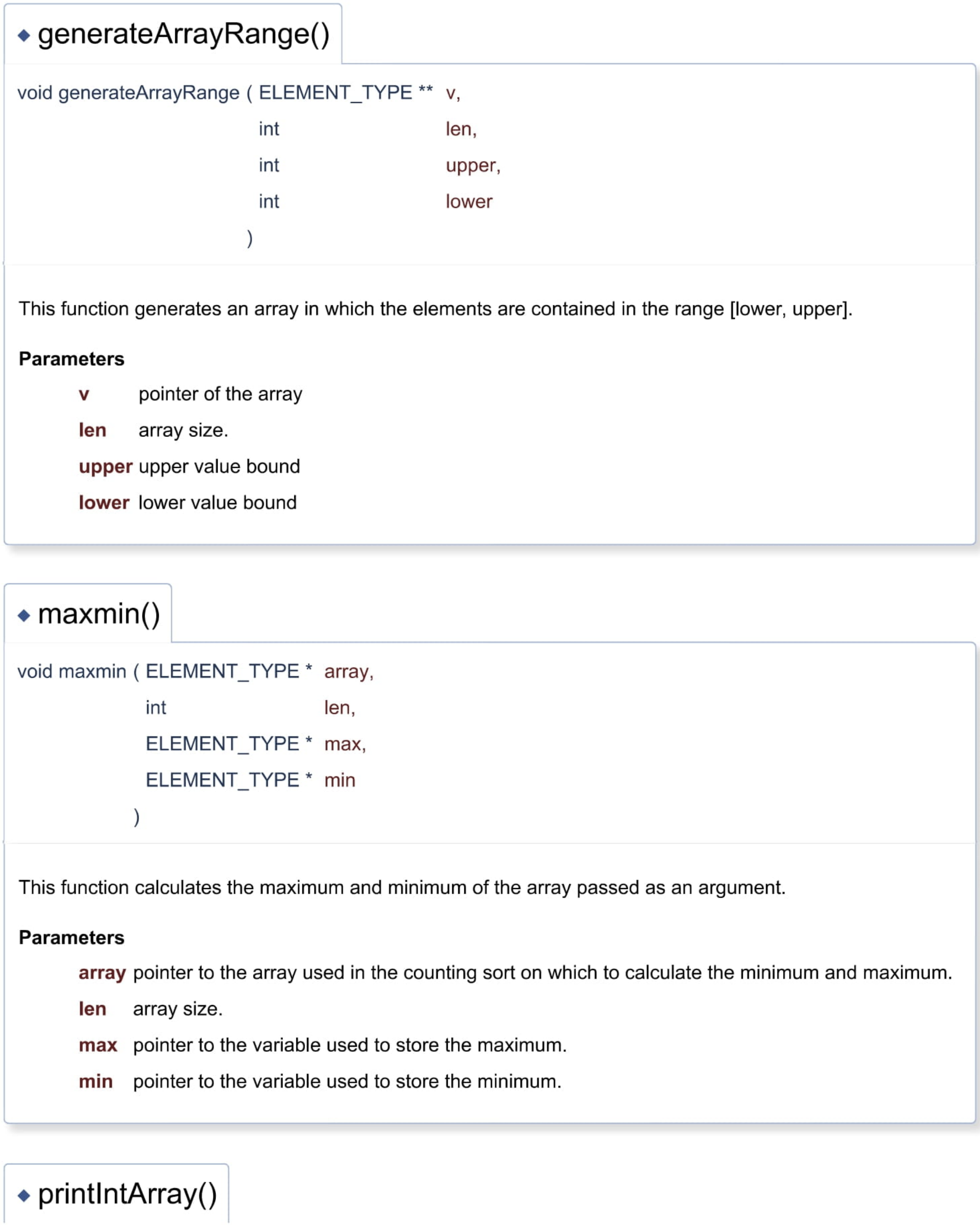
◆ destroyArray()

```
void destroyArray ( ELEMENT_TYPE *  v )
```

Destroys the array.

Parameters

v pointer of the array to destroy



◆ generateArrayRange()

```
void generateArrayRange ( ELEMENT_TYPE **  v,
                          int              len,
                          int              upper,
                          int              lower
                        )
```

This function generates an array in which the elements are contained in the range [lower, upper].

Parameters

v	pointer of the array
len	array size.
upper	upper value bound
lower	lower value bound

◆ maxmin()

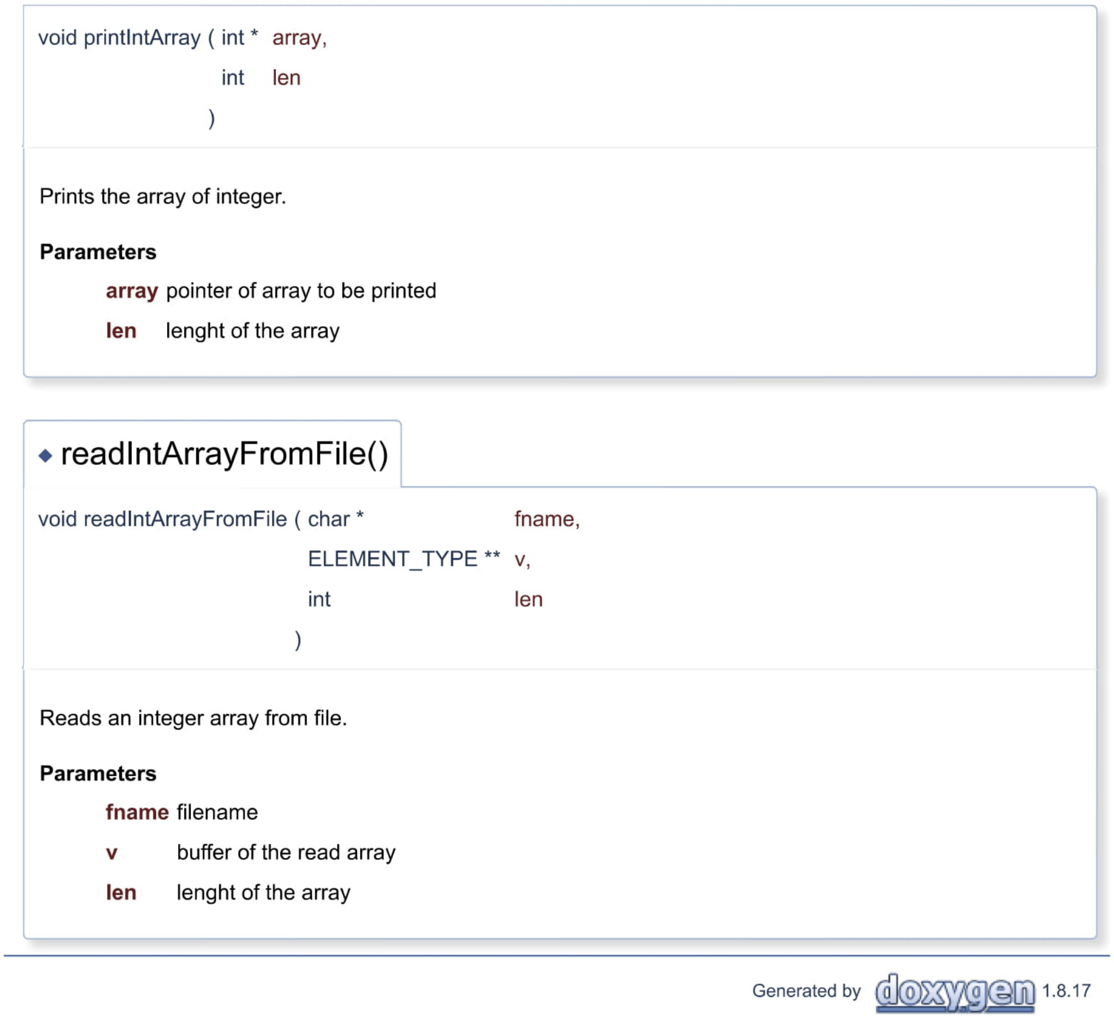
```
void maxmin ( ELEMENT_TYPE *  array,
              int             len,
              ELEMENT_TYPE *  max,
              ELEMENT_TYPE *  min
            )
```

This function calculates the maximum and minimum of the array passed as an argument.

Parameters

array	pointer to the array used in the counting sort on which to calculate the minimum and maximum.
len	array size.
max	pointer to the variable used to store the maximum.
min	pointer to the variable used to store the minimum.

◆ printIntArray()



```
void printIntArray ( int *  array,
                     int    len
                   )
```

Prints the array of integer.

Parameters

array	pointer of array to be printed
len	lenght of the array

◆ readIntArrayFromFile()

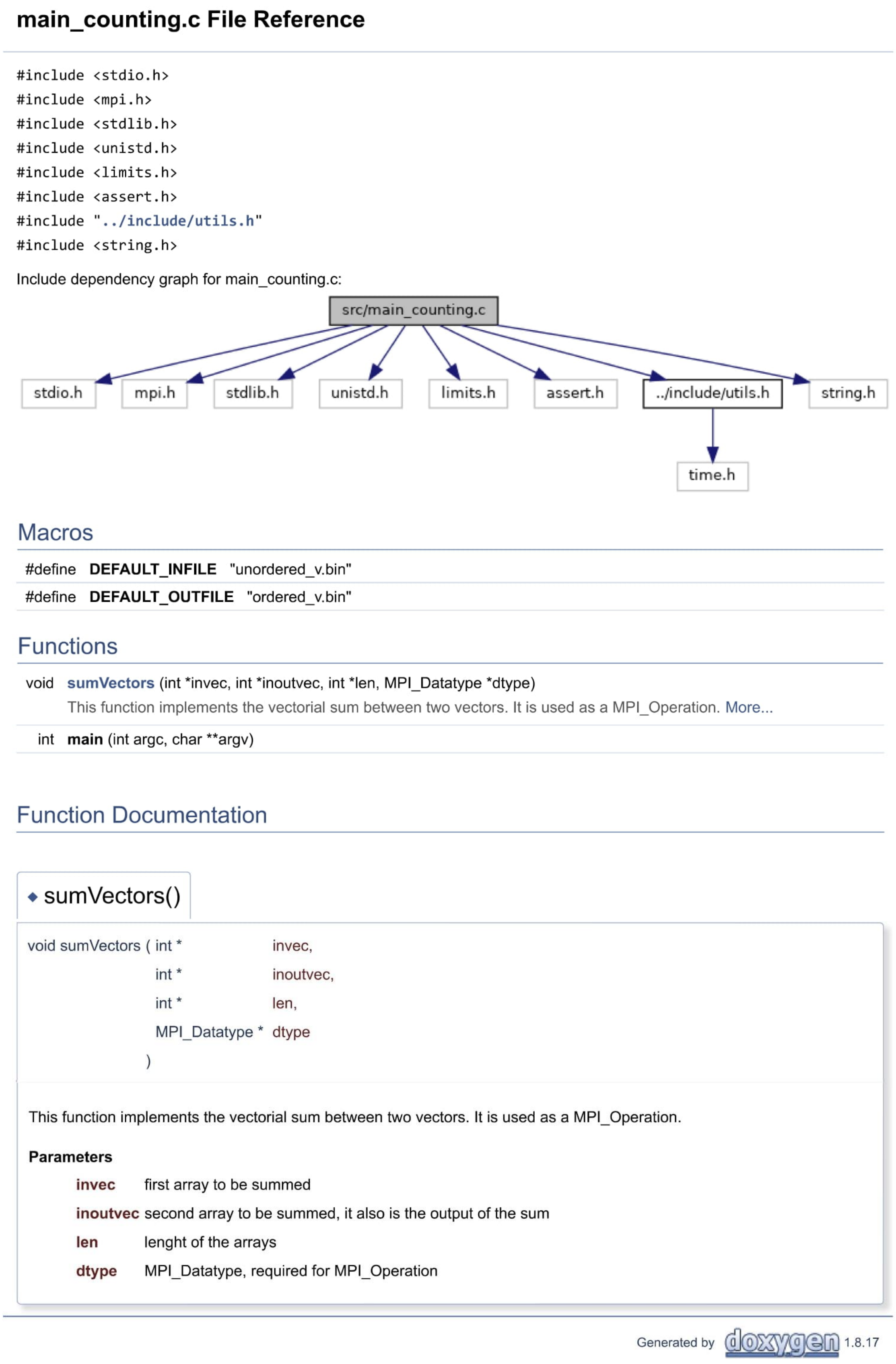
```
void readIntArrayFromFile ( char *            fname,
                            ELEMENT_TYPE **   v,
                            int               len
                          )
```

Reads an integer array from file.

Parameters

fname	filename
v	buffer of the read array
len	lenght of the array

Generated by doxygen 1.8.17



main_counting.c File Reference

```
#include <stdio.h>
#include <mpi.h>
#include <stdlib.h>
#include <unistd.h>
#include <limits.h>
#include <assert.h>
#include "../include/utils.h"
#include <string.h>
```

Include dependency graph for main_counting.c:

Macros

#define **DEFAULT_INFILE** "unordered_v.bin"

#define **DEFAULT_OUTFILE** "ordered_v.bin"

Functions

void **sumVectors** (int *invec, int *inoutvec, int *len, MPI_Datatype *dtype)
This function implements the vectorial sum between two vectors. It is used as a MPI_Operation. More...

int **main** (int argc, char **argv)

Function Documentation

◆ sumVectors()

```
void sumVectors ( int *            invec,
                  int *            inoutvec,
                  int *            len,
                  MPI_Datatype *   dtype
                )
```

This function implements the vectorial sum between two vectors. It is used as a MPI_Operation.

Parameters

invec	first array to be summed
inoutvec	second array to be summed, it also is the output of the sum
len	lenght of the arrays
dtype	MPI_Datatype, required for MPI_Operation

Generated by doxygen 1.8.17

How To Run

1. Create a build directory and launch cmake:

```
mkdir build
cd build
cmake ..
```

2. Generate executables with **make**

3. To generate measures run **make generate_measures**

4. To extract mean times and speedup curves from them run **make extract_measures**

Results can be found in the measures/measure directory, divided by problem size and version.