Gruppe 15 | Tobias Schoch, Luis Nothvogel

Simulation wurde auf dem HTWG Container ausgeführt

21.1

Beim Container:

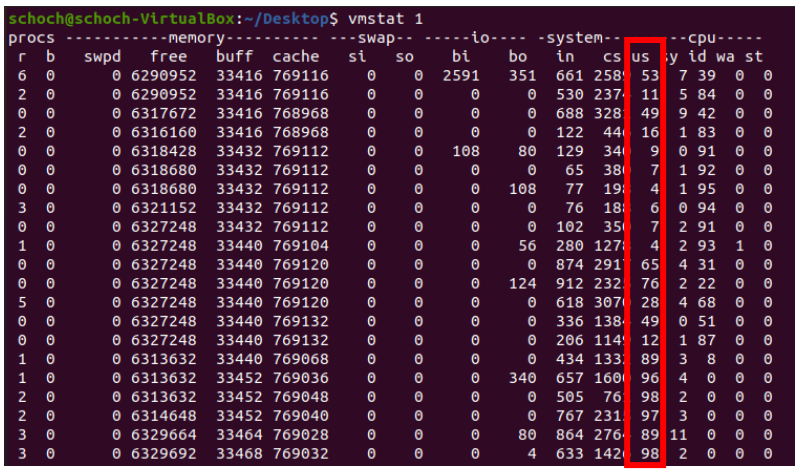
```
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 4  0 507200 1038792      0 221648  865  941   916   965    6    6  7  0 93  0  0
 6  2 507260 1015124      0 252932 77944 84040 109196 84084 11523 8757 14  0 86  0  0
 4  0 502580 993328       0 252932 79760 58400 79776 58400 6996 8329 17  0 83  0  0
 6  1 502200 968288       0 253856 60396 59652 61344 59688 5418 7077 44  0 56  0  0
 4  0 502196 990404       0 253856 75260 84520 75892 84620 13579 12704 33  0 67  0  0
 5  0 502196 990252       0 253856 72608 63584 72608 63596 5760 8904 26  0 74  0  0
 7  0 502152 972764       0 253856 60836 68044 60836 68092 7357 7514 42  0 58  0  0
 3  1 501760 981968       0 258080 91452 83760 96948 83896 12494 9823 30  0 70  0  0
 5  0 501724 983112       0 258080 67156 66852 67156 66928 5683 7478 11  0 89  0  0
 3  0 500584 977420       0 261248 61620 77556 64696 77692 9376 7090 20  0 80  0  0
 4  0 500312 976760       0 261644 90640 72532 91192 72824 8841 8896  1  0 99  0  0
```

Wenn man mem.c mit 1 aufruft, kann man eigentlich kaum eine Veränderung der Werte feststellen. 1 MB ist aber auch nicht sonderlich viel, die hier benötigt werden da der Container bis zu 2 GB physikalischen Speicher hat und darauf nochmal 3 GB Swap Space kommen. Um das ganze laufen zu lassen muss nichts geswapped werden, es kann eigentlich immer komplett in die Speicher geladen werden. Man kann anhand der us-Statistik sehen ab wann das Programm gestartet wurde, da der Wert dort in die Höhe schießt, auch kann man daran dann ablesen wann es noch läuft und ab wann es gestoppt wurde. Die user time statistik gibt gemäß Manpage an wie viel Zeit damit verbraucht non-kernel code auszuführen. Ob die user time colum hier Sinn ergibt, ist schwierig zu sagen, da noch andere Prozesse zusätzlich laufen, die das ganze beeinflussen. Da es aber mehr wird, wenn man das Programm ausführt scheint es erstmal Sinn zu ergeben.

```
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 4  0 1015428 1510392      0 171928  876  957   927   980   10   10  7  0 93  0  0
 4  0 1015428 1509412      0 171796    0    0     0   164 2937 4270 10  0 90  0  0
 8  0 1015420 1506560      0 171796   76    0    76   100 2902 3778 47  0 53  0  0
 5  0 1015420 1506884      0 171796    0    0     0     0 2769 2896 49  0 51  0  0
 5  0 1015416 1500352      0 171796    4    0     4    36 3331 3737 48  0 52  0  0
 8  0 1015204 1488164      0 171928    8    0     8    36 2907 3440 47  0 53  0  0
 9  0 1015144 1486576      0 171928   20    0    20   160 3418 3681 48  0 52  0  0
 6  0 1015136 1506108      0 172060    0    0  1280   140 3719 5315 48  0 52  0  0
 7  0 1015124 1506088      0 172060    0    0     0    32 2697 2451 48  0 52  0  0
 6  0 1015124 1485196      0 172060    4    0     4    36 3285 3190 47  0 53  0  0
 7  0 1015124 1506104      0 172060    0    0     0    36 2595 2819 49  0 51  0  0
 9  0 1015124 1506164      0 172060    4    0     4   108 3492 3765 48  0 52  0  0
 7  0 1015124 1497724      0 172060    0    0     0     0 2449 2379 49  0 51  0  0
 6  0 1015124 1497560      0 172060    0    0     0    96 2834 2653 49  0 51  0  0
 6  0 1015124 1487152      0 172060    4    0     4   160 3129 3462 48  0 52  0  0
 7  0 1015124 1506252      0 172060    0    0     0    68 3357 4214 48  0 52  0  0
 6  0 1015068 1507268      0 172060   28    0    28    96 3525 4712 48  0 52  0  0
 2  0 1015064 1495464      0 171888    8    0  1288     0 3292 4695 45  0 55  0  0
 4  0 1015060 1509660      0 171888   80    0    80     0 2185 1736  4  0 96  0  0
 2  0 1015060 1509496      0 171888    0    0     0    68 2531 3187  1  0 99  0  0
```

Wenn man mehr als eine Instanz laufen lässt, (hier 3) sieht man, dass sich im Vergleich zu vorher nicht wirklich viel ändert. Da wir ja immer noch jeweils 1 MB pro Instanz benötigen ist das immer noch kaum merklich.

I7 4790K 1 CPU-Kern + 980TI + 8GB Ram für die VM:



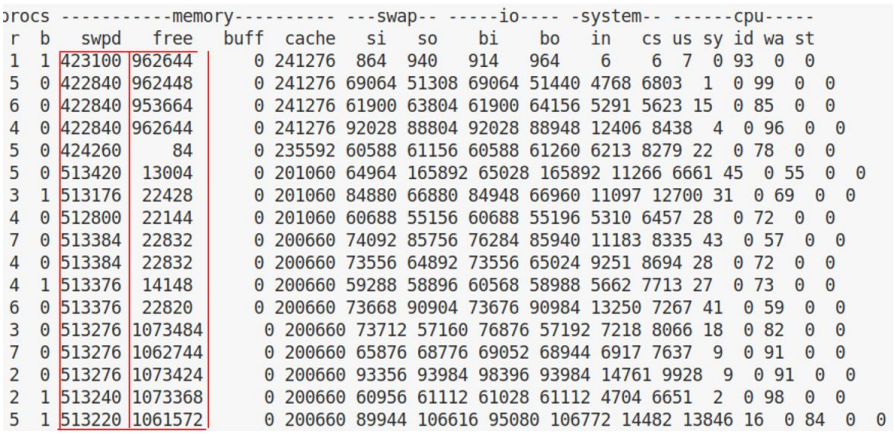
```
schoch@schoch-VirtualBox:~/Desktop$ vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 6  0      0 6290952  33416 769116    0    0  2591   351  661 2589 53  7 39  0  0
 2  0      0 6290952  33416 769116    0    0     0     0  530 2374 11  5 84  0  0
 0  0      0 6317672  33416 768968    0    0     0     0  688 3283 49  9 42  0  0
 2  0      0 6316160  33416 768968    0    0     0     0  122  440 16  1 83  0  0
 0  0      0 6318428  33432 769112    0    0   108    80  129  340  9  0 91  0  0
 0  0      0 6318680  33432 769112    0    0     0     0   65  380  7  1 92  0  0
 0  0      0 6318680  33432 769112    0    0     0   108   77  190  4  1 95  0  0
 3  0      0 6321152  33432 769112    0    0     0     0   76  188  6  0 94  0  0
 0  0      0 6327248  33432 769112    0    0     0     0  102  350  7  2 91  0  0
 1  0      0 6327248  33440 769104    0    0     0    56  280 1270  4  2 93  1  0
 0  0      0 6327248  33440 769120    0    0     0     0  874 2911 65  4 31  0  0
 0  0      0 6327248  33440 769120    0    0     0   124  912 2321 76  2 22  0  0
 5  0      0 6327248  33440 769120    0    0     0     0  618 3070 28  4 68  0  0
 0  0      0 6327248  33440 769132    0    0     0     0  336 1384 49  0 51  0  0
 0  0      0 6327248  33440 769132    0    0     0     0  206 1149 12  1 87  0  0
 1  0      0 6313632  33440 769068    0    0     0     0  434 1332 89  3  8  0  0
 1  0      0 6313632  33452 769036    0    0     0   340  657 1600 96  4  0  0  0
 2  0      0 6313632  33452 769048    0    0     0     0  505  761 98  2  0  0  0
 2  0      0 6314648  33452 769040    0    0     0     0  767 2311 97  3  0  0  0
 3  0      0 6329664  33464 769028    0    0     0    80  864 2764 89 11  0  0  0
 3  0      0 6329692  33468 769032    0    0     0     4  633 1420 98  2  0  0  0
```

Auch hier ist kein sichtlicher Unterschied zu erkennen. 6,3 GB sind frei von 8GB was Sinn ergibt, da OS und andere Programme wie z.B. der offene Firefox auch Ram verbrauchen.

Die user-time Statistik ähnelt der Containerausführung.

21.2

Beim Container:



```
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  1 423100 962644      0 241276  864  940   914   964    6    6  7  0 93  0  0
 5  0 422840 962448      0 241276 69064 51308 69064 51440 4768 6803  1  0 99  0  0
 6  0 422840 953664      0 241276 61900 63804 61900 64156 5291 5623 15  0 85  0  0
 4  0 422840 962644      0 241276 92028 88804 92028 88948 12406 8438  4  0 96  0  0
 5  0 424260     84      0 235592 60588 61156 60588 61260 6213 8279 22  0 78  0  0
 5  0 513420  13004      0 201060 64964 165892 65028 165892 11266 6661 45  0 55  0  0
 3  1 513176  22428      0 201060 84880 66880 84948 66960 11097 12700 31  0 69  0  0
 4  0 512800  22144      0 201060 60688 55156 60688 55196 5310 6457 28  0 72  0  0
 7  0 513384  22832      0 200660 74092 85756 76284 85940 11183 8335 43  0 57  0  0
 4  0 513384  22832      0 200660 73556 64892 73556 65024 9251 8694 28  0 72  0  0
 4  1 513376  14148      0 200660 59288 58896 60568 58988 5662 7713 27  0 73  0  0
 6  0 513376  22820      0 200660 73668 90904 73676 90984 13250 7267 41  0 59  0  0
 3  0 513276 1073484      0 200660 73712 57160 76876 57192 7218 8066 18  0 82  0  0
 7  0 513276 1062744      0 200660 65876 68776 69052 68944 6917 7637  9  0 91  0  0
 2  0 513276 1073424      0 200660 93356 93984 98396 93984 14761 9928  9  0 91  0  0
 2  1 513240 1073368      0 200660 60956 61112 61028 61112 4704 6651  2  0 98  0  0
 5  1 513220 1061572      0 200660 89944 106616 95080 106772 14482 13846 16  0 84  0  0
```

Hier kann man recht leicht erkennen ab wann das Programm gestartet wurde. Nämlich sobald der free wert fällt. Denn dieses Mal führen wir das Programm ja mit 1,024 GB aus. Also die Hälfte unsere verfügbaren Rams, dies bedeutet natürlich, dass mehr Speicher allokiert werden muss. Man kann auch sehr leicht erkennen ab wann das Programm gestoppt wird, nämlich sobald der free wert wieder drastisch in die Höhe schießt. Das System hat danach unerwartet viel mehr Speicher frei als davor. Man müsste eigentlich davon ausgehen, dass das System wieder genau gleich viel wie vorher zu Verfügung hat.

I7 4790K 1 CPU-Kern + 980TI + 8GB Ram für die VM:

```
schoch@schoch-VirtualBox:~/Desktop$ vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 3  0      0 6231200  34004 800716    0    0  1768   349  530 2150 43  5 52  0  0
 0  0      0 6231200  34004 800716    0    0     0     0  271 1220  5  2 93  0  0
 0  0      0 6231200  34012 800716    0    0     0    36  365 1653 10  2 88  0  0
 0  0      0 6231200  34012 800716    0    0     0     0  362 1612 49  2 49  0  0
 0  0      0 6231200  34012 800728    0    0     0     0   76  213  3  1 96  0  0
 0  0      0 6231200  34012 800728    0    0     0     0   88  256 11  1 88  0  0
 0  0      0 6231200  34012 800728    0    0     0     0   72  169  4  1 95  0  0
 1  0      0 5726184  34012 800728    0    0     0     0  321  319 47 27 26  0  0
 1  0      0 5181360  34012 800728    0    0     0     0  431  237 66 34  0  0  0
 1  0      0 5181360  34012 800728    0    0     0     0  449  275 100  0  0  0  0
 1  0      0 5181360  34012 800728    0    0     0     0  426  297 99  1  0  0  0
 1  0      0 5181360  34012 800728    0    0     0     0  425  414 99  1  0  0  0
 1  0      0 5181360  34012 800728    0    0     0  2268  453  407 100  0  0  0  0
 3  0      0 6230948  34012 800728    0    0     0     0  694 2987 71 11 18  0  0
```

Auch hier erkennt man gut, wo der Prozess begonnen hat um Speicher zu reservieren. In der letzten Zeile wird der Prozess dann geschlossen und steigt ungefähr wieder um die zuvor reservierte Menge an, nachdem diese befreit wurde. Da 1024MB leicht zu reservieren sind in unserem 8GB System, muss nicht auf die Disk geswapped werden.

21.3

Beim Container: 2 GB Ram

```
lu851not@ct-bsys-ss20-15:~/z-drive/5.Semester/BSYS/Chap21$ ./mem 1000
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  0 483004 943188      0 274368  866  944   917   968    7    7  7  0 93  0  0
 2  0 483004 943324      0 274368    4    0     4    32 2100 2785  1  0 99  0  0
 8  0 483128 406876      0 274368   24    0    24    80 2528 2689 20  0 80  0  0
 4  0 540300     96      0 226096    0 57144     4 57192 4096 2488 34  0 66  0  0
 5  0 540300     96      0 226096    0    0     0    32 2745 2810 25  0 75  0  0
 6  0 554916   7532      0 218124    0 13980   112 14136 2913 2215 36  0 64  0  0
 4  0 554916  22624      0 218124   32    0    32    64 3397 4123 33  0 67  0  0
 4  0 554884  22432      0 218124  160    0   160    72 2308 3928 28  0 72  0  0
 5  0 555008   9268      0 217988    0  152     0   164 3368 4278 41  0 59  0  0
 2  0 554980  22516      0 217988   96    0    96    96 2838 3071 30  0 70  0  0
 2  0 554980 1048656      0 217988    8    0  1292    80 3552 5642 11  0 89  0  0
 1  0 554980 1048640      0 217988    8    0     8     0 2776 3119 19  0 81  0  0
 7  0 554980 1048636      0 217988    4    0     4   128 1630 3492  0  0 100  0  0
```

Die Si spalte gibt an wie viel von der Disk reingeswapped wurde. Und die so spalte gibt an wie viel zur Disk geswapped wurde. Ja sie geben auch mal nicht 0 Werte zurück. Man kann dies sehr gut beobachten, wenn der free Wert bei 96 ist. Da musste sehr viel Speicher allokiert werden und anscheinend so viel, dass eine bestimmte Menge zur Disk ausgeswapped werden musste. Ähnliches gilt für den Si wert, denn es kann nun mal sein, dass ein Teil des Programms auf der Disk ausgelagert wurde und nur bei Bedarf reinswapped wird, daher die kleineren zahlen.

```
lu851not@ct-bsys-ss20-15:~/z-drive/5.Semester/BSYS/Chap21$ ./mem 2000
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 2  0 1132336 1725828      0  75340  867  945   918   969    7    7  0 93  0  0
 4  0 1131796 1716036      0  75340  388    0   404    36 1816 2754  4  0 96  0  0
 8  0 1131724 1699376      0  75340  172    0  1452    64 3362 4874 17  0 83  0  0
 2  0 1131640 524464       0  75340   32    0    32    96 2707 2314 24  0 76  0  0
 3  0 1181868      4       0  50516 1656 45180  1956 45292 4434 2454 34  0 66  0  0
 2  2 1309520     56       0  41428 2796 128672 13512 128724 14516 3483 44  0 56  0  0
 4  0 1426408    112       0  44672  836 117092  7472 117124 7974 2865 31  0 69  0  0
 5  0 1436528    236       0  51648 68860 70956 77820 70956 8468 7061 20  0 80  0  0
 3  1 1465416  21036       0  59492 66120 91368 77268 91468 13081 6936 35  0 65  0  0
 3  0 1443744    124       0  58824 92816 70372 92816 70496 9428 9639 27  0 73  0  0
 4  0 1446476     80       0  61440 67804 70728 71728 70940 7087 7463 19  0 81  0  0
 9  0 1466888  20168       0  60632 64832 82884 65056 82960 12012 7600 34  0 66  0  0
 4  0 1446660    180       0  61368 90216 66560 94184 66652 12243 12630 27  0 73  0  0
 2  1 1452660    104       0  67416 71400 73248 80408 73280 9276 7977 21  0 79  0  0
 3  0 1471636    108       0  64176 73424 88692 74344 88752 13892 12696 34  0 66  0  0
 2  1 1448616     88       0  63636 86952 59236 87012 59348 8719 8291 24  0 76  0  0
 3  1 1451508    128       0  66112 65732 65196 69972 65336 6268 7553 18  0 82  0  0
 6  0 1467068      0       0  65840 71124 85596 71844 85628 10882 6895 31  0 69  0  0
 4  0 1450484    140       0  65164 84408 67260 84504 67516 9398 7934 28  0 72  0  0
 3  0 1449820    108       0  64636 67688 65580 67944 65676 5694 6479 18  0 82  0  0
 4  1 1463208    240       0  64104 70928 82632 71124 82668 11130 7385 31  0 69  0  0
 4  0 1449636    144       0  64104 87456 72072 88616 72176 10538 9271 27  0 73  0  0
 3  1 1449292    120       0  63704 67792 65880 68432 66056 5897 7960 20  0 80  0  0
 3  0 1462056     40       0  62912 68540 79108 69064 79224 11506 7616 31  0 69  0  0
 2  1 1447856    104       0  61988 86332 69408 86332 69536 10435 8197 28  0 72  0  0
 3  0 1447536      0       0  61592 69332 65520 69572 65636 5542 6627 19  0 81  0  0
 4  1 1459180    448       0  61064 70988 80032 70988 80116 10616 6573 29  0 71  0  0
 3  1 1446532    144       0  60404 89556 73304 89556 73336 10799 7165 30  0 70  0  0
 2  0 1445648    124       0  59080 69316 64952 69316 64992 5478 6341 19  0 81  0  0
 5  0 1452684    996       0  57892 72032 75796 72032 76192 9941 13006 28  0 72  0  0
 3  0 1443024    100       0  56700 89192 74540 89192 74684 12056 9163 30  0 70  0  0
 6  0 1442672    132       0  56304 67848 63876 69128 63960 6264 9286 19  0 81  0  0
 4  1 1464316     96       0  55376 59232 78344 59232 78408 9466 6121 26  0 74  0  0
 4  0 1441408    140       0  54980 97980 71372 97980 71436 12209 9044 32  0 68  0  0
 2  1 1442392     84       0  56168 67560 66448 69432 66564 5841 6966 19  0 81  0  0
 4  0 1458640    152       0  55108 60908 75028 60908 75132 8869 6330 24  0 76  0  0
 2  0 1236568 1846856       0  55768 76008 71004 78108 71080 12702 8880 36  0 64  0  0
```

```
lu851not@ct-bsys-ss20-15:~/z-drive/5.Semester/BSYS/Chap21$ ./mem 3000
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  0 1191732 1778816      0  78604  868  946   919   969    7    7  7  0 93  0  0
 1  0 1191716 1778548      0  78604   12    0    12    68 2869 3938  1  0 99  0  0
 5  0 1191140 1761856      0  78604  816    0   832    32 2575 3518 13  0 87  0  0
 4  0 1191028 960468       0  78604  136    0   136     8 2633 2331 25  0 75  0  0
 5  0 1201216    140       0  69588  484 9304   484  9460 3641 4316 24  0 76  0  0
 3  0 1330532    100       0  63140 7076 130652  7824 130752 14762 6198 43  0 57  0  0
 4  0 1436556     40       0  56316  460 105964  1868 106072 9725 10048 34  0 66  0  0
 4  0 1531124    100       0  52836  416 95448   416 95480 4031 3269 24  0 76  0  0
 4  0 1666832     44       0  53364  404 135772  3760 136296 12377 2190 40  0 60  0  0
 4  0 1776588     32       0  53364  356 110248   356 110360 10454 1748 35  0 65  0  0
 3  0 1872884    184       0  53100  308 96052   396 96108 4839 3538 24  0 76  0  0
 5  0 1999320    152       0  53100  228 126576   408 126688 12726 1488 38  0 62  0  0
 4  0 2116848    124       0  53360   48 117320  1284 117416 10837 1831 37  0 63  0  0
 3  1 2211844     72       0  53756   76 95436   700 95468 4210 1542 25  0 75  0  0
 4  0 2351228     16       0  55340 4776 143616  6728 143668 16427 6052 40  0 60  0  0
 5  1 2467864     16       0  55340 15572 124976 15572 125096 13224 6207 35  0 65  0  0
 3  1 2467848    148       0  55340 55180 41596 55820 41672 6614 9689 13  0 87  0  0
 4  0 2490072    100       0  55208 58928 76200 59568 76300 9531 7881 26  0 74  0  0
 2  0 2467136    156       0  54944 97252 67360 97252 67416 12615 9031 32  0 68  0  0
 4  0 2470040    144       0  57836 61020 67468 64824 67556 6834 9740 20  0 80  0  0
 6  0 2490296    236       0  60344 67752 91516 71708 91584 11060 9112 28  0 72  0  0
 3  1 2473660     76       0  61260 88460 81348 91180 81460 13761 8354 36  0 64  0  0
 4  0 2473620     92       0  61120 66772 56908 67444 56972 6227 7193 17  0 83  0  0
 4  0 2485288     28       0  60184 72744 71840 72744 71876 9230 7576 24  0 76  0  0
 3  0 2475776   4192       0  59240 111716 85188 111844 85236 15949 11696 39  0 61  0  0
 3  0 2471396     24       0  58712 68180 53568 68180 53656 6268 7366 17  0 83  0  0
 6  1 2477384     64       0  58052 61840 63316 63120 63380 8453 8799 21  0 79  0  0
 3  1 2491196  21224       0  57512 33728 99880 33832 99956 14752 5711 40  0 60  0  0
 2  1 2469724     24       0  57248 75764 25400 75764 25468 6892 9912 10  0 90  0  0
 3  0 2470912      0       0  56984 58556 39272 58588 39364 6870 8228 13  0 87  0  0
 4  0 2486148     56       0  54192 74288 65528 74288 65608 13196 15422 36  0 64  0  0
 5  0 2464224     48       0  51624 74392 27124 74392 27156 6902 8951 12  0 88  0  0
 3  1 2463008     12       0  50292 57636 37588 57636 37684 5715 7676 13  0 87  0  0
 4  1 2476176    264       0  49892 75484 64564 75484 64632 11863 9414 33  0 67  0  0
 2  0 2462376    128       0  49756 79572 43028 79628 43068 8376 10066 17  0 83  0  0
 5  0 2462416    148       0  49756 67792 49060 67792 49180 6017 8385 16  0 84  0  0
 4  1 2473292   2384       0  49888 81252 74528 82920 74560 13670 11431 35  0 65  0  0
 6  0 2464156    184       0  51336 90012 63440 91708 63472 10632 10050 25  0 75  0  0
 2  0 2464668     84       0  51864 65580 53836 66144 53912 6166 7214 18  0 82  0  0
 2  1 2484452  17312       0  51864 75952 83668 75952 83768 12663 7668 35  0 65  0  0
```

```
lu851not@ct-bsys-ss20-15:~/z-drive/5.Semester/BSYS/Chap21$ ./mem 4000
allocating 4194304000 bytes (4000.00 MB)
  number of integers in array: 1048576000
Killed
```

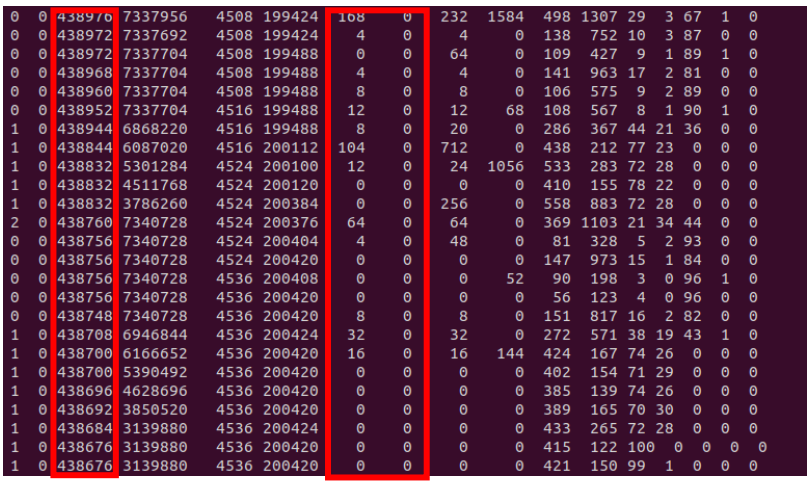
```
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 3  0 1128356 1692184      0 108296  869  948   920   971    8    7  7  0 93  0  0
 3  0 1128068 1691716      0 108296  460    0   460    32 2562 4248  2  0 98  0  0
 5  0 1127812 1684668      0 108296   20    0   660   148 2802 3771  2  0 98  0  0
 2  0 1127808 1691680      0 108296    0    0   640    64 2800 3719 17  0 83  0  0
 4  0 1126772 1515068      0 109220  740    0  1660    80 2663 7652  4  0 96  0  0
 3  0 1126744 322576       0 109220   16    0    16   144 2591 2333 33  0 67  0  0
 4  0 1188020    116       0  88904  100 52200   188 52232 4132 3039 36  0 64  0  0
 2  0 1287908    188       0  74352 1632 95024  1880 95560 4781 3017 25  0 75  0  0
 3  0 1414148     52       0  66508   68 126148  1572 126256 11164 2546 38  0 62  0  0
 6  0 1535684    152       0  65652  304 121408  3052 121440 11598 1880 37  0 63  0  0
 3  0 1632188     92       0  64028  132 96188   132 96216 4597 3376 24  0 76  0  0
 5  0 1765516      4       0  63340   84 133608  1676 133756 12180 2111 38  0 62  0  0
 4  0 1913124      8       0  61876   56 146268  1392 146336 14163 4124 42  0 58  0  0
 3  0 2007920    228       0  61348  312 95004   312 95036 3999 2789 24  0 76  0  0
 6  0 2123580      0       0  60556  188 115092   296 115164 9853 7879 33  0 67  0  0
 2  0 2262956    112       0  59500  328 138836   328 138920 14651 2493 44  0 56  0  0
 2  0 2357804    196       0  59368  228 94744   228 94788 4411 2917 24  0 76  0  0
 4  0 2485124     96       0  59368  232 124388   232 124488 8449 2041 31  0 69  0  0
 5  0 2649336     68       0  60160  204 163092  1320 163124 17826 2616 49  0 51  0  0
 5  1 2741052      8       0  58432 2240 94040  2384 94072 5198 3513 25  0 75  0  0
 5  0 2835660     48       0  57900 8356 102916  8400 103000 9614 7701 26  0 74  0  0
 3  0 2995524     84       0  57768 1528 161264  1596 161348 21052 9250 50  0 50  0  0
 3  0 3086676     20       0  57768  356 91124  1636 91252 5367 4846 26  0 74  0  0
 7  3 2253036      0       0  15260  260 65236 11172 65632 4174 3554 27  0 73  0  0
 3  0 1190940 1788280       0  63680 1052 9384 50412  9424 4171 4779 19  0 81  0  0
```

Bei größeren MB werten verhält es sich logischerweise etwas anders. Denn es muss im Fall mehr geswapped werden. Das spiegelt sich auch in den Si & So Werten wider. Die Werte ergeben auch Sinn. Denn es wird in diesem Fall nur die Arraystelle reingeswapped die gerade beschrieben wird. Daher ist der Si Wert so gering.

Da der freie Speicher sehr leer ist und zum Teil sogar auf 0 sinkt, muss auch dementsprechend viel Speicher auf die Disk geswapped werden. Daher ist auch dementsprechend der „so“ Wert hoch.

I7 4790K 1 CPU-Kern + 980TI + 8GB Ram für die VM:

4GB reserviert

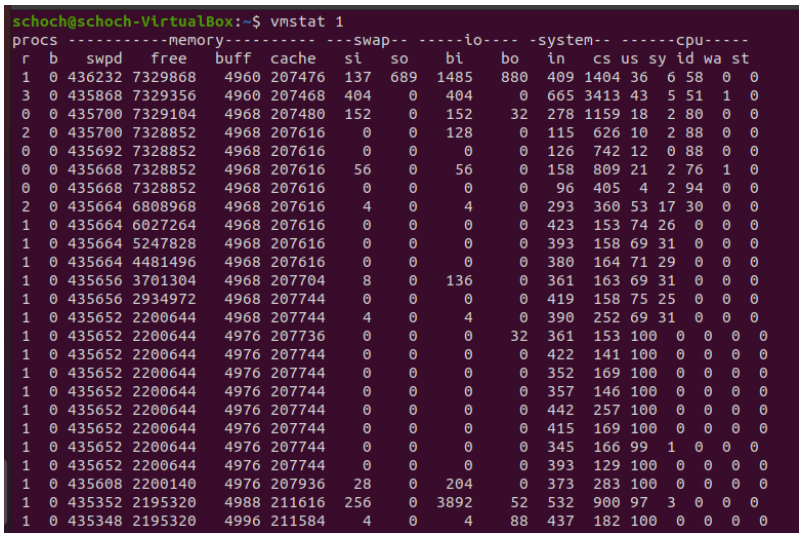


r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa	st
0	0	438976	7337956	4508	199424	168	0	232	1584	498	1307	29	3	67	1	0
0	0	438972	7337692	4508	199424	4	0	4	0	138	752	10	3	87	0	0
0	0	438972	7337704	4508	199488	0	0	64	0	109	427	9	1	89	1	0
0	0	438968	7337704	4508	199488	4	0	4	0	141	963	17	2	81	0	0
0	0	438960	7337704	4508	199488	8	0	8	0	106	575	9	2	89	0	0
0	0	438952	7337704	4516	199488	12	0	12	68	108	567	8	1	90	1	0
1	0	438944	6868220	4516	199488	8	0	20	0	286	367	44	21	36	0	0
1	0	438844	6087020	4516	200112	104	0	712	0	438	212	77	23	0	0	0
1	0	438832	5301284	4524	200100	12	0	24	1056	533	283	72	28	0	0	0
1	0	438832	4511768	4524	200120	0	0	0	0	410	155	78	22	0	0	0
1	0	438832	3786260	4524	200384	0	0	256	0	558	883	72	28	0	0	0
2	0	438760	7340728	4524	200376	64	0	64	0	369	1103	21	34	44	0	0
0	0	438756	7340728	4524	200404	4	0	48	0	81	328	5	2	93	0	0
0	0	438756	7340728	4524	200420	0	0	0	0	147	973	15	1	84	0	0
0	0	438756	7340728	4536	200408	0	0	0	52	90	198	3	0	96	1	0
0	0	438756	7340728	4536	200420	0	0	0	0	56	123	4	0	96	0	0
0	0	438748	7340728	4536	200420	8	0	8	0	151	817	16	2	82	0	0
1	0	438708	6946844	4536	200424	32	0	32	0	272	571	38	19	43	1	0
1	0	438700	6166652	4536	200420	16	0	16	144	424	167	74	26	0	0	0
1	0	438700	5390492	4536	200420	0	0	0	0	402	154	71	29	0	0	0
1	0	438696	4628696	4536	200420	0	0	0	0	385	139	74	26	0	0	0
1	0	438692	3850520	4536	200420	0	0	0	0	389	165	70	30	0	0	0
1	0	438684	3139880	4536	200424	0	0	0	0	433	265	72	28	0	0	0
1	0	438676	3139880	4536	200420	0	0	0	0	415	122	100	0	0	0	0
1	0	438676	3139880	4536	200420	0	0	0	0	421	150	99	1	0	0	0

Nachdem wir nun mehrmals Speicher reserviert hatten, ist nun der erste geswappte Speicher ersichtlich in der „swpd“ Spalte.

Interessant ist hier, dass auf diesem System „so“ komplett 0 hat, was auch Sinn macht, da genügend Restspeicher vorhanden ist auf der Memory (3,1GB) und nichts geswapped werden muss. Si hingegen lädt ein paar Pages von der Disk zurück auf die Memory.

5GB reserviert



```
schoch@schoch-VirtualBox:~$ vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  0 436232 7329868   4960 207476  137  689  1485   880  409 1404 36  6 58  0  0
 3  0 435868 7329356   4960 207468  404    0   404     0  665 3413 43  5 51  1  0
 0  0 435700 7329104   4968 207480  152    0   152    32  278 1159 18  2 80  0  0
 2  0 435700 7328852   4968 207616    0    0   128     0  115  626 10  2 88  0  0
 0  0 435692 7328852   4968 207616    0    0     0     0  126  742 12  0 88  0  0
 0  0 435668 7328852   4968 207616   56    0    56     0  158  809 21  2 76  1  0
 0  0 435668 7328852   4968 207616    0    0     0     0   96  405  4  2 94  0  0
 2  0 435664 6808968   4968 207616    4    0     4     0  293  360 53 17 30  0  0
 1  0 435664 6027264   4968 207616    0    0     0     0  423  153 74 26  0  0  0
 1  0 435664 5247828   4968 207616    0    0     0     0  393  158 69 31  0  0  0
 1  0 435664 4481496   4968 207616    0    0     0     0  380  164 71 29  0  0  0
 1  0 435656 3701304   4968 207704    8    0   136     0  361  163 69 31  0  0  0
 1  0 435656 2934972   4968 207744    0    0     0     0  419  158 75 25  0  0  0
 1  0 435652 2200644   4968 207744    4    0     4     0  390  252 69 31  0  0  0
 1  0 435652 2200644   4976 207736    0    0     0    32  361  153 100  0  0  0  0
 1  0 435652 2200644   4976 207744    0    0     0     0  422  141 100  0  0  0  0
 1  0 435652 2200644   4976 207744    0    0     0     0  352  169 100  0  0  0  0
 1  0 435652 2200644   4976 207744    0    0     0     0  357  146 100  0  0  0  0
 1  0 435652 2200644   4976 207744    0    0     0     0  442  257 100  0  0  0  0
 1  0 435652 2200644   4976 207744    0    0     0     0  415  169 100  0  0  0  0
 1  0 435652 2200644   4976 207744    0    0     0     0  345  166 99  1  0  0  0
 1  0 435652 2200644   4976 207744    0    0     0     0  393  129 100  0  0  0  0
 1  0 435608 2200140   4976 207936   28    0   204     0  373  283 100  0  0  0  0
 1  0 435352 2195320   4988 211616  256    0  3892    52  532  900 97  3  0  0  0
 1  0 435348 2195320   4996 211584    4    0     4    88  437  182 100  0  0  0  0
```

Während der Swap Speicher sich weiterhin langsam entleert, werden auch 5GB leicht reserviert. So bleiben danach noch 2,2GB Ram übrig.

6GB reserviert

```
schoch@schoch-VirtualBox:~$ vmstat 1
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 2  0 434192 7317740   5196 215844  128  628  1367   804  408 1322 40  6 54  0  0
 0  0 434160 7317732   5196 215852   44    0    44     0  674 3519 59  1 40  0  0
 0  0 434160 7317732   5196 215852    0    0     0     0  131  839 10  2 88  0  0
 0  0 434160 7317732   5196 215852    0    0     0     0  124  637  8  2 90  0  0
 1  0 434160 7277624   5196 215852    0    0     0     0  290 1542 16  7 76  0  0
 1  0 433168 6576056   5196 215980 1220    0  1372     0  626  942 75 25  0  0  0
 1  0 433168 5822324   5196 216004    0    0     0     0  378  179 76 24  0  0  0
 1  0 433164 5047928   5196 216004    4    0     4     4  407  174 71 29  0  0  0
 1  0 433108 4309316   5196 216004   60    0    60     0  430  229 74 26  0  0  0
 1  0 433108 3522320   5196 216008    0    0     0    44  412  150 69 31  0  0  0
 1  0 433108 2765312   5196 216008    0    0     0     0  428  155 70 30  0  0  0
 1  0 433108 2002508   5196 216008    0    0     0     0  397  124 77 23  0  0  0
 1  0 433108 1271708   5196 216008    0    0     0     0  355  151 71 29  0  0  0
 1  0 433108 1161836   5204 216000    0    0     0    32  396  268 96  4  0  0  0
 1  0 433108 1161836   5204 216008    0    0     0     0  386  143 100  0  0  0  0
 2  0 433108 1162340   5204 217504    0    0  1544     0  430  355 100  0  0  0  0
 1  0 433100 1174380   5204 218428   12    0   912     0  429  799 99  1  0  0  0
 1  0 432900 1174436   5204 218460  252    0   264     0  474  659 99  1  0  0  0
 1  0 432900 1174436   5204 218464    0    0     0     0  412  183 100  0  0  0  0
```

Der Swap Speicher wird weiterhin entleert, während das System auch 6 GB reserviert und danach noch 1,1GB übrighat.

7 GB reserviert

```
0  0 416980 7235480   8540 273212  148    0   704     0  561 2064 48  6 46  0  0
0  0 416980 7235480   8540 273216    0    0     0     0  112  496 11  1 88  0  0
3  0 416740 7244540   8540 275720  288    0  2816     0  226 1228 14  2 83  1  0
1  0 416736 6533408   8540 275740    4    0     4     0  424  571 74 25  1  0  0
1  0 416736 5790008   8540 275740    0    0     0     0  390  180 73 27  0  0  0
1  0 416736 5020400   8540 275740    0    0     0     0  390  130 73 27  0  0  0
1  0 416736 4226096   8540 275740    0    0     0     0  402  165 76 24  0  0  0
1  0 416732 3471608   8540 275740    4    0     4     0  346  176 76 24  0  0  0
1  0 416732 2688392   8548 275740    0    0     0    20  391  138 75 25  0  0  0
1  0 416732 1939952   8548 275740    0    0     0     0  407  218 70 30  0  0  0
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 2  0 416732 1173872   8548 275740    0    0     0     0  415  168 72 28  0  0  0
 2  0 416732 419888   8548 275740     0    0     0     0  383  122 70 30  0  0  0
 1  0 436064 146112   7092 211552   56 19300    60 19300 1562 1022 66 34  0  0  0
 1  0 436064 146112   7092 211552    0    0     0     0  426  151 100  0  0  0  0
```

Nachdem das System beinahe die gesamten 7 GB reserviert und nur noch 150 MB übrighat, wurden vom System 19300 kb (19,3 MB) geswapped auf die Disk, was man schön in der vorletzten Zeile sehen kann.

8 GB reserviert

```
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0 434016 7335240   7348 214220  124  524  1184   669  407 1166 47  6 47  0  0
 1  0 433760 7335240   7348 214256   28    0    28     0  531 2932 28  4 68  0  0
 0  0 433760 7335240   7348 214268   32    0    32     4  368 1517 34  3 63  0  0
 0  0 433760 7335240   7348 214268    0    0     0     0  133  776 13  3 84  0  0
 0  0 433760 7334988   7356 214268   20    0    20    12  173  959 16  3 80  1  0
 1  0 433760 6582256   7356 214272    8    0    16     0  425  472 64 35  1  0  0
 1  0 433760 5793748   7356 214276    0    0     0     0  444  227 70 30  0  0  0
 1  0 433760 4997932   7356 214276    0    0     0     0  414  156 69 31  0  0  0
 1  0 433760 4222024   7356 214276    0    0     0     0  408  161 74 26  0  0  0
 1  0 433760 3444100   7356 214276    0    0     0     0  404  160 70 30  0  0  0
 1  0 433760 2691376   7356 214276    0    0     0     0  414  192 73 27  0  0  0
 1  0 433760 1911184   7356 214276    0    0     0     0  392  154 74 26  0  0  0
 1  0 433760 1136788   7356 214276    0    0     0     0  373  146 71 29  0  0  0
 1  0 433760 353068   7356 214276    0    0     0     0  384  120 71 29  0  0  0
 2  0 459260 101364    176 102036   40 25640   852 25640 1846 1204 40 60  0  0  0
 1  0 459252 7443904    368 133144    8    0 62768    44 1132 1617  4 91  1  4  0
```

Das System schafft es nicht 8 GB zu reservieren. Es versucht es jedoch, bis es merkt, dass weder die 459 MB Swapspeicher reichen, noch die restlichen 7,3 GB freier Speicher.

Daher gibt es im Anschluss Killed zurück.

```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 8000
allocating 8388608000 bytes (8000.00 MB)
  number of integers in array: 2097152000
Killed
```

Virtual Box erlaubt es nicht mehr RAM zu reservieren, als das System zur Verfügung hat.

Daher funktioniert es auch nicht mehr als 8GB RAM zu reservieren.

21.4

Beim Container: mit gleichem Screenshot wie bei 21.3

Man kann sehen das bei steigender übergebenen MB zahl, steigt auch die Bi & Bo Zeit. Was logisch ist, da wir immer mehr swappen müssen bei größer werdendem Array. Die CPU utilization time verändert sich nicht großartig.

Bi- und Bo beschreibt Blocks, die zwischen Disk und Memory gesendet werden, in Blocks pro Sekunde.

I7 4790K 1 CPU-Kern + 980TI + 8GB Ram für die VM:

Die Bi- und Bo-Blocks verhalten sich sehr ähnlich zu si und so. Dabei werden logischerweise die Blocks zwischen Disk und Memory geswapped werden.

21.5

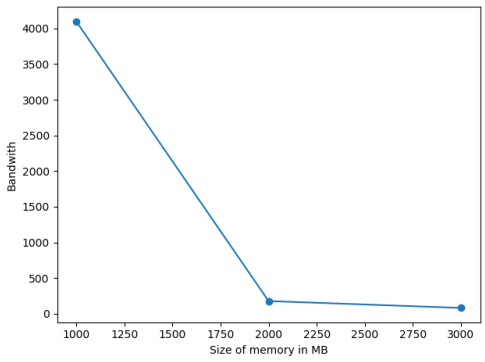
Beim Container:

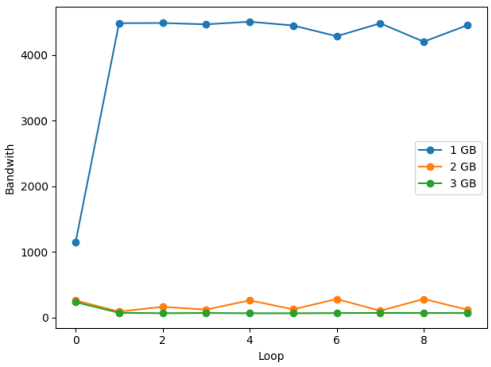
```
lu851not@ct-bsys-ss20-15:~/z-drive/5.Semester/BSYS/Chap21$ ./mem 1000
allocating 1048576000 bytes (1000.00 MB)
  number of integers in array: 262144000
loop 0 in 879.65 ms (bandwidth: 1136.82 MB/s)
loop 1 in 224.01 ms (bandwidth: 4464.03 MB/s)
loop 2 in 224.08 ms (bandwidth: 4462.73 MB/s)
loop 3 in 224.42 ms (bandwidth: 4455.95 MB/s)
loop 4 in 224.22 ms (bandwidth: 4459.99 MB/s)
loop 5 in 225.70 ms (bandwidth: 4430.76 MB/s)
loop 6 in 293.03 ms (bandwidth: 3412.64 MB/s)
loop 7 in 256.81 ms (bandwidth: 3893.92 MB/s)
loop 8 in 225.09 ms (bandwidth: 4442.67 MB/s)
loop 9 in 232.98 ms (bandwidth: 4292.21 MB/s)
loop 10 in 225.59 ms (bandwidth: 4432.84 MB/s)
```

```
lu851not@ct-bsys-ss20-15:~/z-drive/5.Semester/BSYS/Chap21$ ./mem 3000
allocating 3145728000 bytes (3000.00 MB)
  number of integers in array: 786432000
loop 0 in 12489.73 ms (bandwidth: 240.20 MB/s)
loop 1 in 51920.06 ms (bandwidth: 57.78 MB/s)
loop 2 in 46107.18 ms (bandwidth: 65.07 MB/s)
loop 3 in 42890.80 ms (bandwidth: 69.95 MB/s)
loop 4 in 42936.60 ms (bandwidth: 69.87 MB/s)
loop 5 in 44321.07 ms (bandwidth: 67.69 MB/s)
```

Beim ersten Fall, also wo wir eine Größe von einem 1 GB nehmen, kann man sehen, dass das ganze recht schnell abläuft. Da nicht sonderlich viel geswapped werden muss, daher ist die Bandbreite und Geschwindigkeit auch recht hoch. Man sieht auch das Loop 0 im Vergleich zu Loop 1 eine deutlich langsamere Bandbreite hat, dies liegt daran, dass dies der erste speicherzugriff ist und das ganze erst in den Cache muss, danach ist es deutlich schneller da es ja schon im Cache ist.

Beim zweiten Fall sieht es allerdings etwas anders aus, dort nehmen wir 3 GB als Größe. D.H es passt auf jeden Fall nicht in dem Ram rein und es muss geswapped werden und dies spiegelt sich auch in der Zeit und in der Bandbreite wider. Denn dadurch, dass immer auf die Disk gewartet werden muss, und diese nicht grade schnell ist im Vergleich zum Memory, dauert das ganze länger. Man sieht auch das Loop 0 im Vergleich zu Loop 1 deutlich schneller ist. Dies liegt daran, dass für die späteren Loops immer geswapped werden muss, daher dauert dies bei diesen länger.





I7 4790K 1 CPU-Kern + 980TI + 8GB Ram für die VM:

```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 4096
allocating 4294967296 bytes (4096.00 MB)
  number of integers in array: 1073741824
loop 0 in 5419.26 ms (bandwidth: 755.82 MB/s)
loop 1 in 3705.59 ms (bandwidth: 1105.36 MB/s)
loop 2 in 3808.93 ms (bandwidth: 1075.37 MB/s)
loop 3 in 3851.40 ms (bandwidth: 1063.51 MB/s)
loop 4 in 3786.91 ms (bandwidth: 1081.62 MB/s)
loop 5 in 3792.79 ms (bandwidth: 1079.94 MB/s)
loop 6 in 3841.90 ms (bandwidth: 1066.14 MB/s)
loop 7 in 3787.10 ms (bandwidth: 1081.57 MB/s)
loop 8 in 4156.59 ms (bandwidth: 985.42 MB/s)
loop 9 in 4032.62 ms (bandwidth: 1015.72 MB/s)
```

```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 5000
allocating 5242880000 bytes (5000.00 MB)
  number of integers in array: 1310720000
loop 0 in 6637.53 ms (bandwidth: 753.29 MB/s)
loop 1 in 4620.57 ms (bandwidth: 1082.12 MB/s)
loop 2 in 4596.49 ms (bandwidth: 1087.79 MB/s)
loop 3 in 4593.59 ms (bandwidth: 1088.47 MB/s)
loop 4 in 4505.42 ms (bandwidth: 1109.78 MB/s)
loop 5 in 4561.96 ms (bandwidth: 1096.02 MB/s)
loop 6 in 4998.29 ms (bandwidth: 1000.34 MB/s)
loop 7 in 4514.45 ms (bandwidth: 1107.56 MB/s)
loop 8 in 4541.88 ms (bandwidth: 1100.87 MB/s)
loop 9 in 4659.73 ms (bandwidth: 1073.02 MB/s)
```

```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 6000
allocating 6291456000 bytes (6000.00 MB)
  number of integers in array: 1572864000
loop 0 in 8239.71 ms (bandwidth: 728.18 MB/s)
loop 1 in 5830.05 ms (bandwidth: 1029.15 MB/s)
loop 2 in 5626.87 ms (bandwidth: 1066.31 MB/s)
loop 3 in 5856.70 ms (bandwidth: 1024.47 MB/s)
loop 4 in 5998.81 ms (bandwidth: 1000.20 MB/s)
loop 5 in 5680.76 ms (bandwidth: 1056.20 MB/s)
loop 6 in 5751.09 ms (bandwidth: 1043.28 MB/s)
loop 7 in 5696.59 ms (bandwidth: 1053.26 MB/s)
loop 8 in 6130.30 ms (bandwidth: 978.74 MB/s)
loop 9 in 5841.34 ms (bandwidth: 1027.16 MB/s)
```

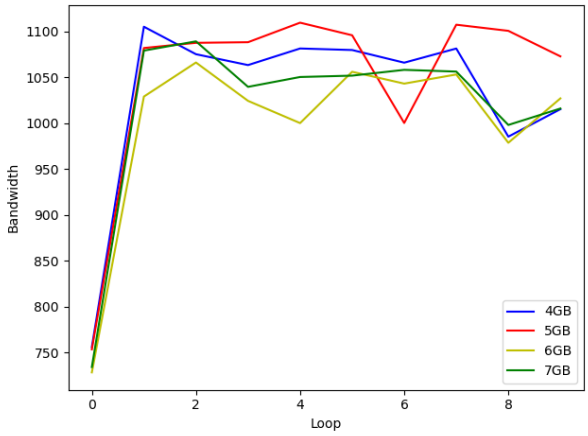
```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 7000
allocating 7340032000 bytes (7000.00 MB)
  number of integers in array: 1835008000
loop 0 in 9536.53 ms (bandwidth: 734.02 MB/s)
loop 1 in 6485.63 ms (bandwidth: 1079.31 MB/s)
loop 2 in 6425.54 ms (bandwidth: 1089.40 MB/s)
loop 3 in 6732.39 ms (bandwidth: 1039.75 MB/s)
loop 4 in 6663.53 ms (bandwidth: 1050.50 MB/s)
loop 5 in 6654.03 ms (bandwidth: 1051.99 MB/s)
loop 6 in 6614.30 ms (bandwidth: 1058.31 MB/s)
loop 7 in 6626.12 ms (bandwidth: 1056.42 MB/s)
loop 8 in 7012.83 ms (bandwidth: 998.17 MB/s)
loop 9 in 6889.02 ms (bandwidth: 1016.11 MB/s)
```

Die Reservierung von 4GB, 5GB, 6GB, und 7GB läuft sehr schnell ab und ungefähr der selben Bandbreite, da hierfür unsere Memory große genug ist. Loop 0 ist stets ca. 25% langsamer als die anderen Loops, da beim ersten Durchlauf noch keine Cacheeinträge vorhanden sind.

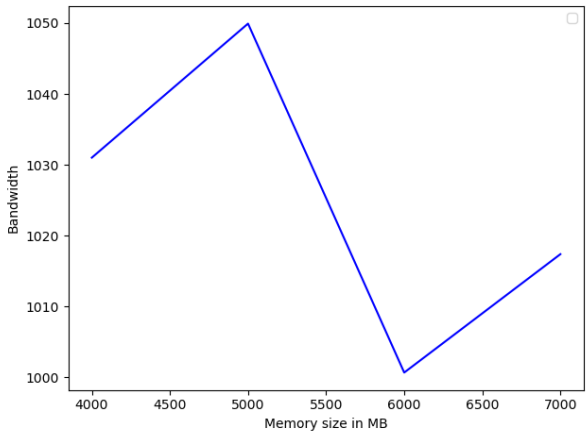
```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 8000
allocating 8388608000 bytes (8000.00 MB)
  number of integers in array: 2097152000
Killed
```

```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 12000
allocating 12582912000 bytes (12000.00 MB)
memory allocation failed
```

Da unser Swap Speicher sehr klein ist, kann aufwärts von 8GB kein Speicher reserviert werden. Bei einer Speicherreservierung von 8GB versucht das System noch den Speicher zu reservieren, bis es merkt, dass der Swap Speicher auch nicht ausreicht und beendet den Prozess. Bei 12GB hingegen, versucht das System es erst gar nicht und gibt die Fehlermeldung zurück: „Memory allocation failed“. Dies passiert, wenn das System NULL zurück gibt bei der Reservierung.



Hier kann man nochmal schön sehen, wie die Loops verlaufen. Wie bereits erwähnt, ist die erste Loop etwas langsamer und hat danach einen gleichmäßigen Verlauf.



Hier sehen wir zwar einen immensen Ausschlag, wenn man sich allerdings die Skalierung der y-Achse anschaut wirkt der Ausschlag nur minimal. Der Speicher wurde in allen 4 Versuchen sehr gleichmäßig reserviert.

21.6

Beim Container:

```
Filename                                Type            Size    Used    Priority
none                                    virtual         3145728 1195400 0
```

Wenn man mem.c mit Werten ab 4 GB aufruft wird es wie im Screenshot oben einfach gekillt. Die Grenze wird hier vermutlich die Größe des Speichers + Swap Space Größe sein also in unserem Fall: 2 GB + 3 GB = 5 GB. Allerdings nur wenn sonst nichts anderes laufen würde, was nun mal nicht passiert. Hier im Bild oben sieht man, dass vom Swap Space schon ca. 1 GB belegt sind, d.h. es bleiben 2 GB übrig. Wenn jetzt der RAM auch nicht komplett frei ist kommen wir ab 4 nicht mehr hin. 2 GB könnten geswapped werden aber der Rest müsste in den Hauptspeicher geladen werden und das geht nur wenn dieser komplett leer ist.

I7 4790K 1 CPU-Kern + 980TI + 8GB Ram für die VM:

```
schoch@schoch-VirtualBox:~$ swapon -s
Filename                                Type            Size    Used    Priority
/swapfile                               file            459260  453096  -2
```

Die Größe des Swapbereiches in der Disk, ist 459 MB groß. Interessanterweise ist der Swapbereich im Container 3 GB groß, also fast 6-mal so groß. Davon ist aufgrund der 7 GB Speicherreservierung nahezu alles belegt.

21.7

Es ist schwer die in-memory Performance vom RAM zu erreichen mit einer SSD oder einer HDD. Ein kleiner Statistikvergleich wie schnell meine unterschiedlichen Hardware Komponenten sind:

HDD: 130 MB/s

SSD: 530 MB/s

RAM: 16 GB/s

Wir haben die Ergebnisse in zwei identischen VM's laufen lassen, die lediglich unterschiedliche Festplatten haben. Die eine mit SSD die andere mit HDD. Daher ist der Hauptfaktor beider Systeme nur die Geschwindigkeit der Festplatte.

Wir konnten zwar Swapspeicher aktivieren und deaktivieren auf den jeweiligen Systemen, aber leider keine andere Partition zuweisen, da wir einer VM leider keine mehrfachen Festplatten geben konnten.

```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 7000
allocating 7340032000 bytes (7000.00 MB)
  number of integers in array: 1835008000
loop 0 in 9081.27 ms (bandwidth: 770.82 MB/s)
loop 1 in 6184.91 ms (bandwidth: 1131.79 MB/s)
loop 2 in 6200.06 ms (bandwidth: 1129.02 MB/s)
loop 3 in 6180.36 ms (bandwidth: 1132.62 MB/s)
loop 4 in 6145.05 ms (bandwidth: 1139.13 MB/s)
loop 5 in 6146.70 ms (bandwidth: 1138.82 MB/s)
```

```
schoch@schoch-VirtualBox:~/Desktop$ ./mem 7000
allocating 7340032000 bytes (7000.00 MB)
  number of integers in array: 1835008000
loop 0 in 10190.41 ms (bandwidth: 706.55 MB/s)
loop 1 in 6522.13 ms (bandwidth: 1103.93 MB/s)
loop 2 in 6509.66 ms (bandwidth: 1106.05 MB/s)
loop 3 in 6517.51 ms (bandwidth: 1104.72 MB/s)
loop 4 in 6529.61 ms (bandwidth: 1102.67 MB/s)
loop 5 in 6475.24 ms (bandwidth: 1111.93 MB/s)
```

Wie zu erwarten ist die SSD etwas schneller. Um genau zu sein, 30MB/s Warum ist die SSD lediglich nur ein paar MB/s schneller? Dies könnte an unserem sehr kleinen Swapspeicher liegen. Wir haben nur einen sehr kleinen Swapspeicher mit einer Größe von 459MB. Da die SSD 5-mal so schnell ist wie die HDD und nur so einen kleinen Unterschied macht, ist es bei unserem kleinen Swapspeicher nahezu unmöglich an die Geschwindigkeit des RAMS von 16GB/s ranzukommen. Wäre der Swapspeicher größer, dann wäre vermutlich auch die Bandweite des Systems mit SSD schneller.