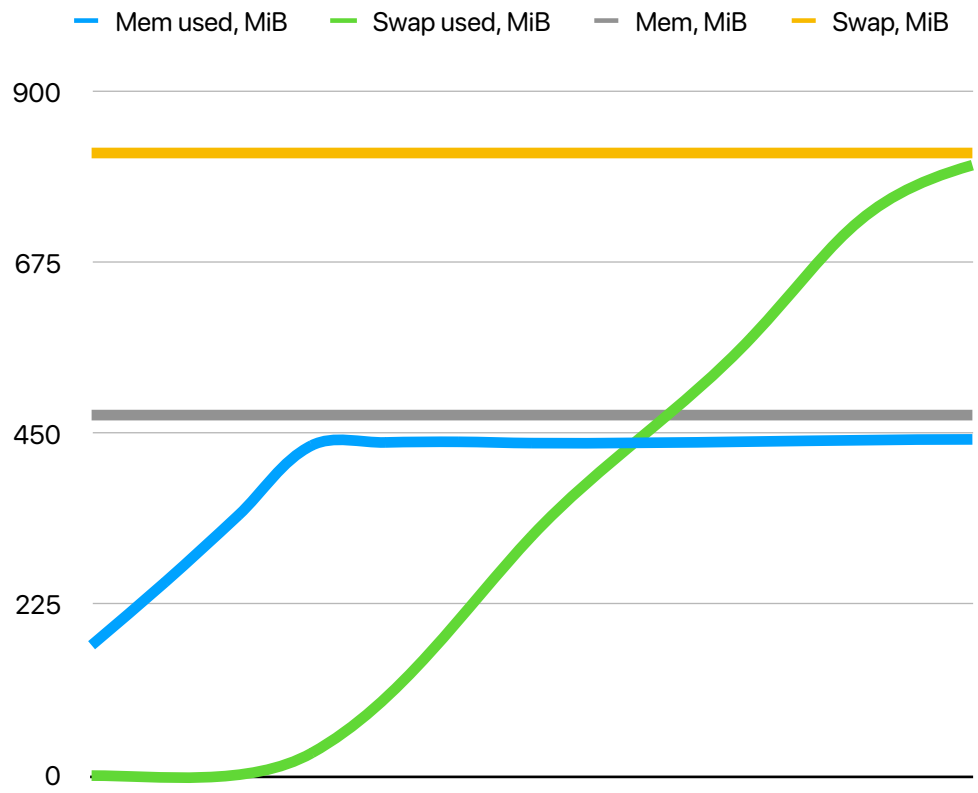
Отчет по лабораторной работе № 5

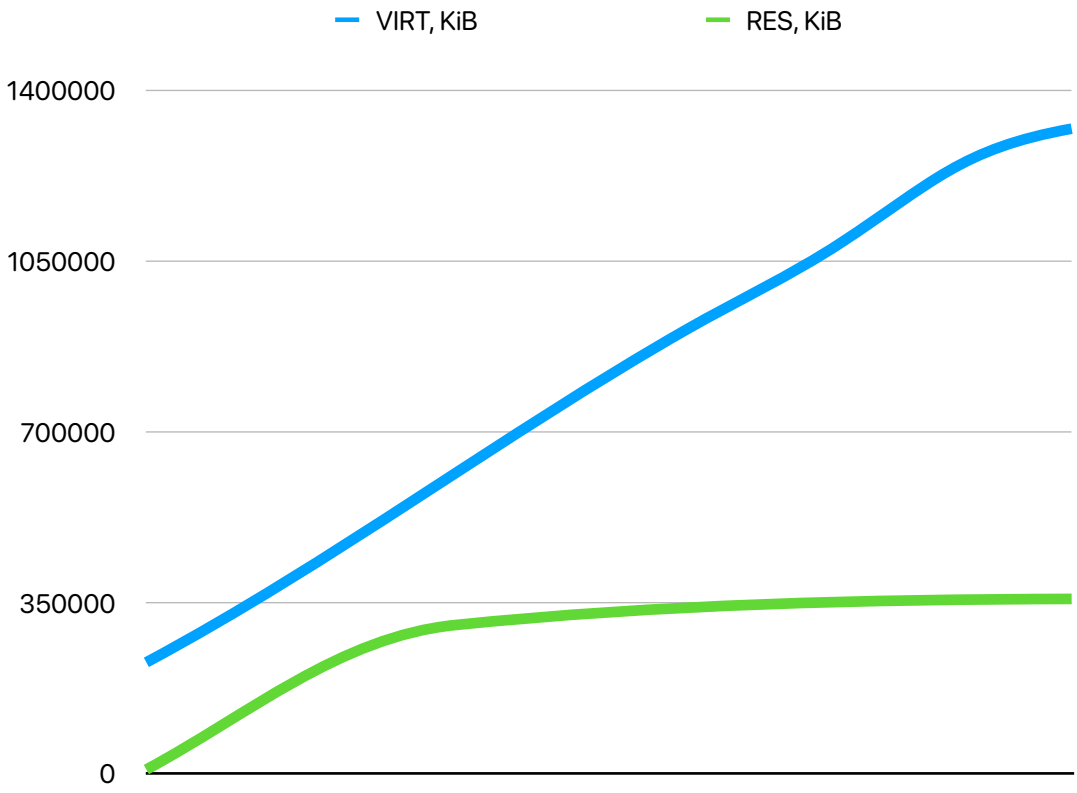
Эксперимент 1

Этап 1

820 MiB swap 474,8 MiB ram 1 скрипт

Mem used, MiB	Swap used, MiB	VIRT, KiB	RES, KiB	%MEM	Mem, MiB
171,7	0	228428	8840	1,8	474,8
254,7	0	313040	93584	19,2	474,8
342,9	3,5	403856	184372	37,9	474,8
434,9	29,8	523580	287336	58,4	474,8
438,8	111,3	612152	304768	62,7	474,8
439,6	199	700988	319524	65,7	474,8
437,9	315,8	818600	332168	68,3	474,8
438,6	405	909812	346468	71,3	474,8
445,6	485	999044	360088	74,1	474,8
440,2	578	1087220	354688	73	474,8
441,8	691,3	1205756	358004	73,6	474,8
442,7	776,5	1293272	357924	73,6	474,8
442,8	804	1321784	358408	73,7	474,8





На полученных графиках мы четко видим работу свапа. Как только у нас начинает заканчиваться физическая память, так сразу наиболее старые страницы начинают сбрасываться на диск чтобы освободить место под выделение новых. Падение происходит после того как кончается место под свап.

Последнее выведение число в лог 14000000

Последние 2 записи в журнале:

- [288,283463] Out of memory: Killed process 1220 (bash) total-vm:1342376kB, anon-rss:361692kB, file-rss:0kB, shmem-rss:0kB, UID:0
- [288,320080] oom_reaper: reaped process 1220 (bash), now anon-rss:0kB, file-rss:0kB, shmem-rss:0kB

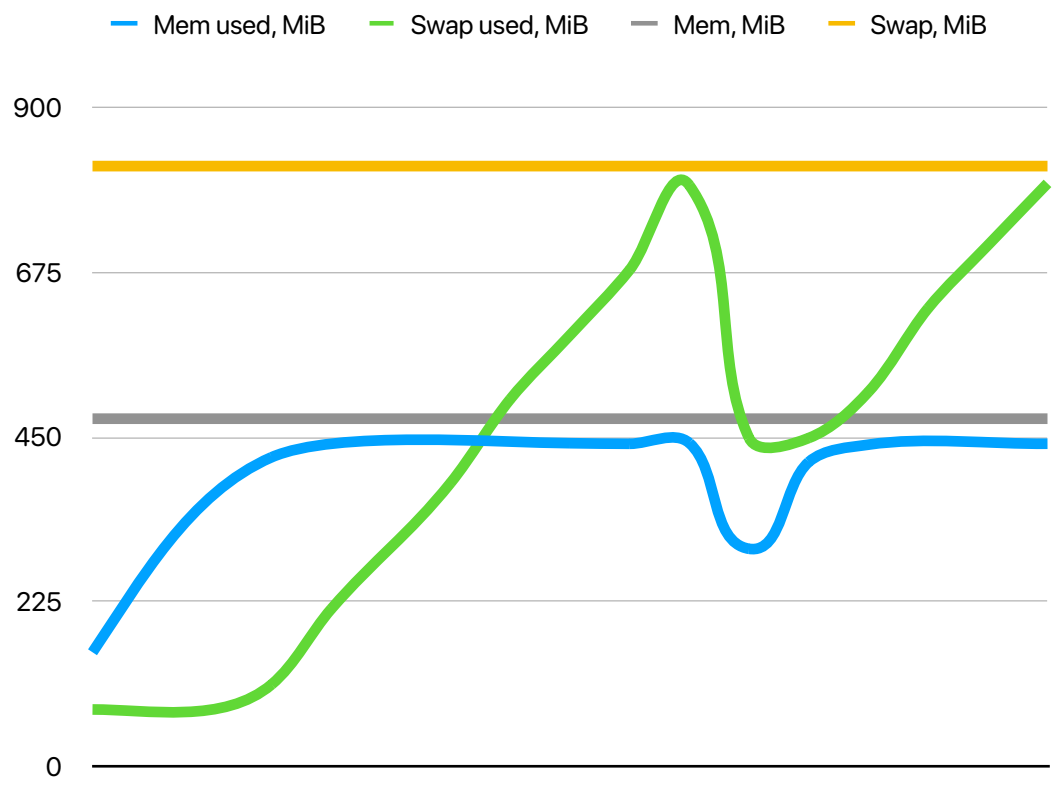
Где первая - убийство процесса из-за нехватки памяти на выделение под новую страницу, а вторая - очистка памяти, занятой убитым процессом.

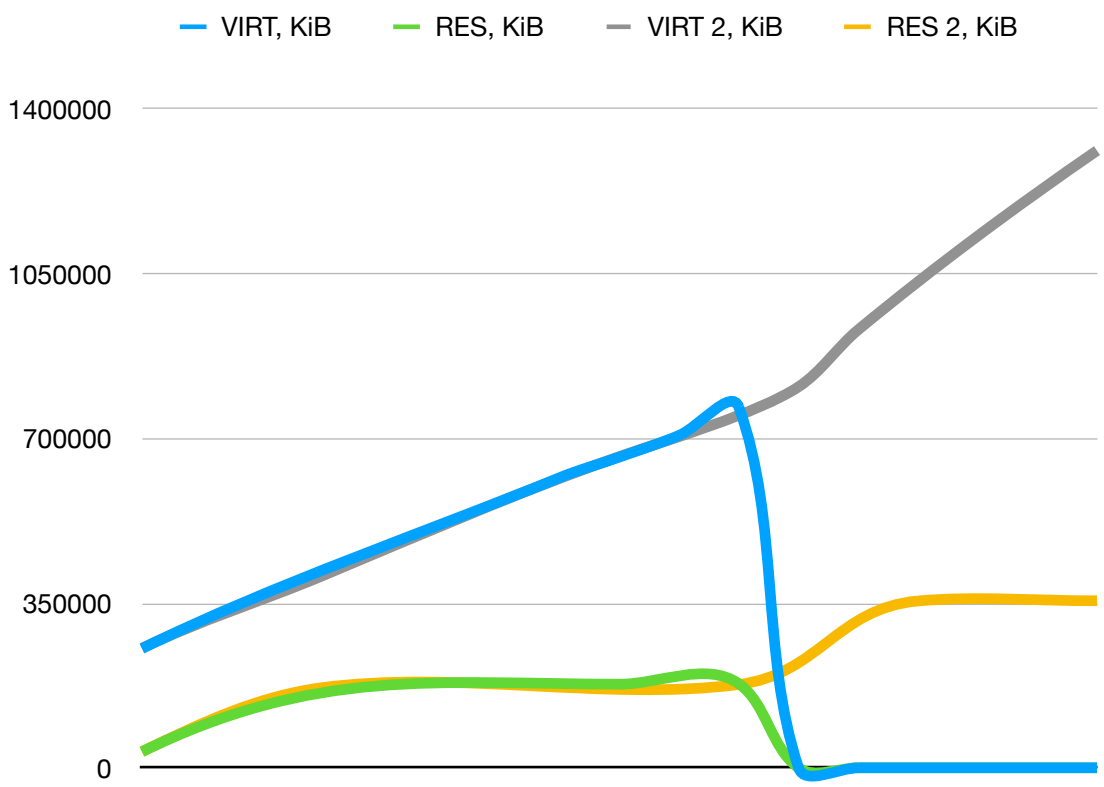
В целом, в данной ситуации можно увидеть то, что при переполнении физической памяти, система сбрасывает самые старые страницы на диск, пока на нем есть место. Как только место кончается и выгружать больше некуда, процесс убивается.

Этап 2

2 скрипта

Mem used, MiB	Swap used, MiB	VIRT, KiB	RES, KiB	%MEM	VIRT 2, KiB	RES 2, KiB	%MEM 2
155,6	77,4	253640	34172	7	253772	34184	7
276,5	77,4	315548	95944	19,7	315416	95960	19,7
366,9	77,4	361880	142408	29,3	361484	141896	29,2
422,1	111,6	407288	170612	35,1	406892	169984	35
437,7	214,8	467480	175928	36,2	467480	176068	36,2
440,4	299,1	512492	177896	36,6	512492	177656	36,5
440,6	387,1	557240	177364	36,5	556712	177464	36,5
442	501,6	616904	178832	36,8	616244	178196	36,7
441,3	589	660596	177700	36,5	661652	178684	36,8
440,5	678,3	706664	177972	36,6	705080	176572	36,3
441,5	793,7	766724	178696	36,8	764612	177024	35,4
297	448,4	0	0	0	809888	209264	43
415,6	448,4	0	0	0	931196	330680	68
439,3	511,5	0	0	0	1019504	354620	72,9
439,6	625,8	0	0	0	1135928	353700	72,7
439,8	711,7	0	0	0	1224632	354480	72,9
440,6	796,4	0	0	0	1312016	355140	73





Эта ситуация уже выглядит интереснее. В начале оба процесса быстро съедают всю память и система начинает выгружаться страницы в свап. Но поскольку первый процесс был запущен несколько раньше, то в критический момент полной нехватки памяти именно он окажется первым. В результате первый процесс убивается, а его память освобождается. Это дает возможность второму процессу продолжить поедать память. Что в итоге переходить к этапу 1. Процесс требует место -> странные страницы выгружаются -> выделяется место под новые. Конец такой же, убийство второго процесса из-за нехватки места.

Лог первого процесса остановился на отметке в 700000, в то время как второй процесс дошел до тех же самых 14000000.

Системный журнал ожидаемо показывает убийство первого процесса и освобождение, занятой им, памяти. А под конец то же самое происходит и со вторым:

- [12075.025022] Out of memory: Killed process 1409 (bash) total-vm:781640kB, anon-rss:179276kB, file-rss:0kB, shmem-rss:0kB, UID:0
- [12075.057345] oom_reaper: reaped process 1409 (bash), now anon-rss:0kB, file-rss:0kB, shmem-rss:0kB

- [12132.333008] Out of memory: Killed process 1410 (bash) total-vm:1340660kB, anon-rss:358500kB, file-rss:0kB, shmem-rss:0kB, UID:0
- [12132.372469] oom_reaper: reaped process 1410 (bash), now anon-rss:0kB, file-rss:0kB, shmem-rss:0kB

Выводы таковы. Система убивает первый процесс подошедший к выделению памяти при ее нехватке. Память освобождается сразу же, что дает возможность другим процессам продолжить работу. Так же мы видим что длинна массива на момент последней одновременной работы 2х процессов ровно в 2 раза меньше придельной длины при работе одного. Это логично, ведь оба процесса работают почти синхронно и поэтому пополам делать потребляемую память.

Эксперимент 2

Исходно за N принимаем 1400000. При K = 10 все проходит в нормальном режиме, процессы корректно завершаются. При смене K на 30 происходит лавинообразное завершение процессов. Что очевидно т.к места на всех не хватает.

Определение идеального N

Начал свою проверку я с самого мне логичного: если процессов стало в 3 раза больше, то на каждого будет в 3 раза меньше памяти. И в целом это так, при N = 14000000/30 = 466666 все прекрасно работает. Но я заметил, что из-за запуска программ с интервалом в 1 секунду, они достаточно растянуты друг от друга во времени чтобы можно было увеличить лимит. И после пары десятков тестов я выяснил что для моей конфигурации он находится в районе 793000.