Задание: Протестировать функцию malloc/free и построить график зависимости времени выделения от размера запрашиваемой памяти.

Усложнение: сравнить с другим малоками.

Task: Test the malloc/free function and plot the allocation time versus the requested memory size.

Complication: compare with other maloks.

Ход работы

Для сравнения были выбраны следующие реализации функции malloc: tcmalloc, jemalloc, memalloc, hoard malloc.

Приведем краткую информацию о каждой из них:

tcmalloc – в его основе лежит идея разделения памяти на несколько уровней ради уменьшения фрагментации памяти. Внутри TCMalloc управление памятью делится на две части: работа с памятью потоков и работа с кучей.

jemalloc – это реализация malloc общего назначения, в которой особое внимание уделяется предотвращению фрагментации и масштабируемой поддержке параллелизма.

memalloc – простой распределитель памяти (это все, что о нем известно).

hoard malloc – это замена malloc, которая может значительно повысить производительность приложений, особенно многопоточных программ, работающих на многопроцессорных и многоядерных процессорах.

Для тестирования использовался следующий код на языке C++:

```
#include <iostream>
#include <chrono>
#include <fstream>
//#include <jemalloc/jemalloc.h>
using namespace std;
#define MAX_MEM_COUNT 4294967297
int main() {
	ofstream res("data.csv");
	for(unsigned long long i=1; i < MAX_MEM_COUNT; i*=2) {
		auto begin = chrono::steady_clock::now();
		auto ptr = malloc(i);
		auto end = chrono::steady_clock::now();
		auto elapsed_ns = chrono::duration_cast<chrono::nanoseconds>(end - begin);
		res << i << "," << elapsed_ns.count() << endl;
		if (ptr != NULL){
```

```
                        free(ptr);
                }
        }
        res.close();
}
```

Установка малоков:

```
tcmalloc:
sudo apt-get install google-perftools
export LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libtcmalloc_minimal.so.4
sudo ln -s /usr/lib/x86_64-linux-gnu/libtcmalloc_minimal.so.4
sudo apt -y install libgoogle-perftools-dev
g++ -ltcmalloc_minimal malloc_test.cpp -o malloc_test

jemalloc: раскомментировать строку //#include <jemalloc/jemalloc.h>
sudo apt-get install libjemalloc-dev
export LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libjemalloc.so

hoardmalloc:
git clone https://github.com/emeryberger/Hoard
cd Hoard/src
make
export LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libhoard.so

memalloc:
git clone https://github.com/arjun024/memalloc
cd memalloc/
gcc -o memalloc.so -fPIC -shared memalloc.c
export LD_PRELOAD=$PWD/memalloc.so
```

Для каждого малока проводилось по 5 тестов и для каждого размера выделяемой памяти находилось среднее время. На графике максимальным размером является 134217728 байт, поскольку для tcmalloc время выделения в дальнейшем достигало недопустимых значений. Возможно, это было вызвано тем, что для tcmalloc существует максимальный размер выделяемой памяти, записанный в переменную TCMALLOC_LARGE_ALLOC_REPORT_THRESHOLD.

Вывод: Из графика видно, что лучше всего с большими объемами памяти справляется memalloc, а для меньших объемов лучше использовать hoard malloc. Стоит отметить, что выделение 8388608 байт с помощью jemalloc требует намного больше времени, чем в случае с остальными аллокаторами, а классифицировать этот показатель как выброс едва ли возможно, поскольку эксперимент проводился несколько раз. Также заметим, что, начиная с какого-то размера выделяемой памяти, для большинства малоков время выделения памяти почти не меняется.