Exercise: Dynamic Memory Allocation

1. Introduction

For a real-time system it is important that worst case execution times (WCET) are known for each task that has a hard deadline.

Imagine your computation task also needs to allocate some memory during the runtime, but the amount of new memory needed can only be determined at runtime: e.g., the amount of new memory needed could depend on the sensor data you get in each step (e.g., depending on the number of 3D points from a Lidar scanner).

So, dynamic memory allocation and deallocation times also play an important role if we want to determine the WCET of a task.

2. Experiment

In this exercise your task is to write a simulation for dynamic memory allocation / deallocation and measure the time needed for a C `malloc()` and `free()` call as a function of the size of the memory block to be allocated / deallocated.

For this: Write a C program that simulates 2000 dynamic memory allocation/freeing procedures by determining a random amount of bytes to be allocated, allocates the blocks using `malloc()` and de-allocates it using `free()`. Measure the corresponding times needed using the `clock()` command.

Write the data collected to a file `alloc_and_free_times.txt` that can later be used for plotting the data as a graph. For this, write the memory block size in a first column, the time needed to allocate it using `malloc()` in a second column, and the time needed to `free()` it in a third column:

A row in this file could look like this:

```
42 28 5
```

E.g., this line would mean, for allocating a memory block of 42 MB in the experiment we needed 28ms and for deallocating it 5ms in the experiment.

Then plot the results. Use the data file to plot a graph that shows how much time t(S) we need in order to allocate / free a memory block of size S.

If you use gnuplot for plotting the graph, you can use a command similar to this one:

plot "alloc_and_free_times.txt" using 1:2, "alloc_and_free_times.txt" using 1:3

3. Questions

Look at the generated graphs and answer the following questions:

- Is the time needed to allocate / free a memory block independent of its size S?
- How good can we predict the time needed to allocate / free a block?
- Can we predict allocation / free times better for small or large block sizes?