Assignment 2 Report

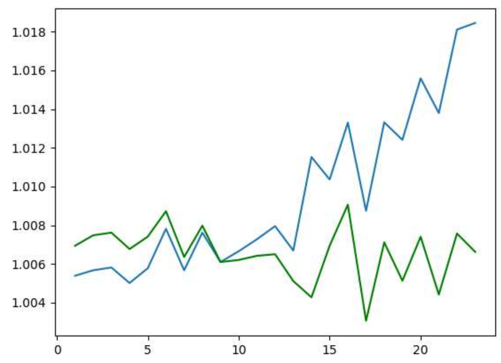
Ippokratis Koukoulis, Christos Nanis

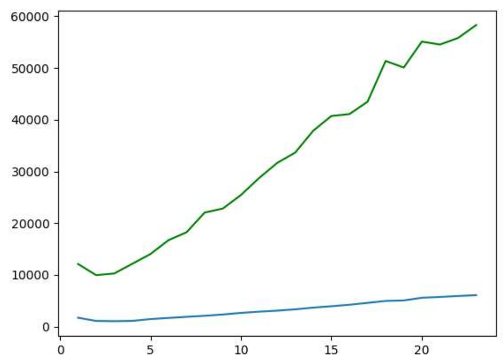
Outline

The header of the page block structure is of 96-B and contains a 16-B header with a pointer that points to the first page of the page block, with additional info on the kind of objects stored in that memory (we differentiate between small and big (i.e. below or over 2048B), based on a setup-time magic number provided). All headers are 16B multiples so that all objects are properly aligned to a 16B boundary. Our size taxonomy follows that our classes range from 16B – 2kB and properly aligned as prior described. Our implementation does not use a superpage-backend manager but nevertheless incorporates both levels of caching also performed on the paper.

Experimentation

To compare and evaluate our implemented memory allocator, we used an *alloc-test* repo fork and showed the impact of our implementation vs the *jemalloc* one on memory blowup and the total duration of the executable program related to an increasing number of deployed threads. For an experiment of 22 deployed threads, an example run is as follows:





We can observe in the first plot, that the memory blowup expressed as (RSSmax << 12)/maxalloc behaves as: For the first few number of threads the behaviour is similar to the jemalloc one, whereas from the 9-point, the behaviour starts to be dramatically more stable and well-fluctuating than the jemalloc case, showing that our direction manages efficiently memory for more threads. On the second plot, we can observe a mostly linear increase in runtime for an increasing number of threads which indicates that optimization around execution time must be performed to achieve better in that direction, combined with the fact that the blowout is handled well.