A multithreaded malloc

Johan Montelius

HT2016

1 Introduction

Your task in this assignment is to implement a *multithreaded malloc* i.e. we should be able to use it in a multithreaded application. The basic requirement is of course that it is *thread safe*, that it still works even if two threads us it at the same time. This is of course very easy to solve by having one central lock that protects any data structures that are used by the module. It is a bit harder to allow multiple threads to actually use the malloc module at the same time to improve efficiency. This is your task; implement the `malloc()` and `free()` procedures in way that allows concurrent access. You should also provide benchmarks of you system that shows that it is faster (or shows that it is not worth the trouble).

2 The implementation

Before you start you should have done the experiments that teach you how to implement `malloc()` and `free()`. Since you will implement a multithreaded system and provide concurrent updates to shared structures it is also advisable to have done the experiments that shows how pthread mutex locks are used. If you can combine this knowledge you should be fine.

Start by implementing a malloc procedure that is protected by one global lock. Run some benchmarks and describe the performance, this is the system that you will try to beat.

Think about what strategy that could be best to explore, depending on which strategy that you choose there will be more or less access to shared data structures and if you want things to run fast you will have to minimize this.

As an extra challenge you can start to think about a thread local cache. If a thread does a free on a data structure it might be a good idea to give this structure back to the same thread instead of to some other thread. This is tricky to solve since all threads execute the same code, yet they should keep track of *thread local data structures*. If you do it right you should see some improvement in execution speed.

3 The benchmarks

Start doing benchmarks from day one. It will be much easier to change your strategy early on if you benchmarks tells you that you're on the wrong track.

In your presentation you should present benchmarks that shows if your approach was successful or not. If you have a brilliant idea that you think would work and the benchmarks proves you wrong, this is also very interesting and might be the conclusion of your report.