OpenSees and Allinea

I had access to the Allinea profiler tools for a week. I used:

- `perf-tool` this tool gives an overall result of the whole run as a webpage.
- `map` this is a very fancy profiler that runs the code for a given time defined by the user and then shows what parts of the code were the ones where we spent most of the time

In this page I will summarize some of the findings.

Target Problem

I used the Lamb's problem developed by Seokho. This is a fairly small model with some 50K+ DOF.

Perf-tool

OpenSeesSP_2p_1n_2016-10-18_11-49.html

OpenSeesSP_4p_1n_2016-10-18_11-44.html

OpenSeesSP_8p_1n_2016-10-18_11-55.html

Some comments on the results:

- When using two processors, the recommendation is to use more processes as the communication is quite low and we spend most of the time doing computations.
- If we increase to 4 processors, then almost 40% of the time is spent on MPI point-to-point communications, which is quite a high number. Also, and most importantly, there is almost no gain with respect to 2 core execution time: it is a couple of seconds slower on 4 processors ⚠️.
- Trying 8 processors for this problem seems to be a huge overkill. We see an even bigger execution time and a lot more communications.
- Something very interesting that was pointed out by this tool is the fact that during calculations, most of the time data needs to be retrieved from memory (80-90% of the whole time). This means that the code is highly ineffective, but optimization of this part often is very complicated and requires domain specific knowledge.

MAP

This tool is very interesting, I will try to detail some interesting results:

- Most of the computational time is spent on the dgemm function from BLAS (matrix multiplication).
- The point-to-point communications are mostly related to MUMPS inner communications, so there is not much to be done here.