How to speed up Excel in 6 steps

After the last post on Excel ("Accelerating an Excel Sheet with OpenCL"), there have been various request and discussions how we do "the miracle". Short story: we only apply proper engineering tactics. Below I'll explain how you can also speed up Excel and when you actually have to call us.

Excel is a special piece of software from a developer's perspective. An important rule of software engineering is to keep functionality (code) and data separate. Excel mixes these two as no other, which actually goes pretty well in many cases unless the data gets too big or the computations too heavy. In that case you've reached Excel's limits and need to properly solve it.

Below are the steps to go through, of which most you can do yourself!

Step 1: Try LibreOffice

LibreOffice has a very fast calculation-engine and has ported large parts to OpenCL. It just works with your Excel sheet or gives errors - very simple to find out.

Download here and then just try. Much faster? Problem solved. Crashing or slower? Go to step 2.

Step 2: Restructuring the Excel-sheet

Microsoft offers two first solutions to slow Excel: putting all calculations in VBA scripts, or move to MS Access. Both options need the data and code to be separated. Once you can answer "Where does the compute happen?" with an answer that is far from "here, and there and there and also a bit here and there" then you're finished.

To further optimise the Excel sheet, I found the list on https://chandoo.org/wp/optimize-speedup-excel-formulas/ to be quite complete.

Step 3: Optimising the algorithms

A classic is finding an optimal range of values and a computation is looped until the values don't change anymore. There are two problems with that: temporary values and a slow algorithm. An approximation could help with initialising the data, resulting in much faster final results. Not storing temporary variable and intermediate results (run-in-one-go) could get to the final answer quicker.

There are many other examples that I could go into. Here at Stream HPC we do a lot of algorithm optimisations - knowing your mathematics gives faster software.

Step 4: Add more hardware

Got 4GB? Upgrade immediately. Got 8GB? Computers with 16GB or 32GB of memory will speed up the computer as a whole, but also Excel could get a lot faster. The actual speedup could also be none, ofcourse. Easiest is to simply find a machine with lots of ram and test your Excel sheet.

Microsoft offers Windows HPC and a special office that goes with it. Currently they focus on running the sheet on Azure. And Azure is a lot bigger than what you have on your desk, for sure.

Step 5: Putting the compute-engine outside Excel

When Microsoft Access or Azure is not for you, you could focus on solving the computations in a smarter way. This is where we spoke about in the previous article on Excel and this is also the part where you should contact us, if you want performant version of your Excel sheet.

The creation of such DLL is not a product, but it would contain the algorithms . You can interact with the DLL via Excel, which takes care of the control and holds the data. Going a step further would put the data in a database and build a modern GUI that runs in the browser. And that has been the message: don't use Excel when you need to test its limits.

Step 6: Porting the compute-engine to the GPU.

This is where we're best known for: making GPU-software. We'd take the optimised algorithm from step 3 and the data from step 2, then use the same approach as in step 5 to get software that runs on the GPU. The result is very a fast Excel sheet like the one in the previous Excel article.