Parallel implementations of image reconstruction algorithm for X-ray microtomography

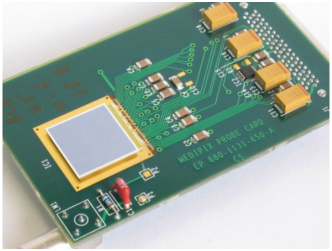
V. Tokareva[1], A. Gridin[1,2], A. Khakimov[3], D. Kozhevnikov[1], O.Strelstova[4], M.Zuev[4]

[1]Dzhelepov Laboratory of Nuclear Problems, JINR, Dubna, Russia
[2]Moscow Institute of Physics and Technology, Dolgoprudny, Russia
[3]Saint Petersburg State University, Saint Petersburg, Russia
[4]Laboratory of Information Technologies, JINR, Dubna, Russia

July 5, 2016

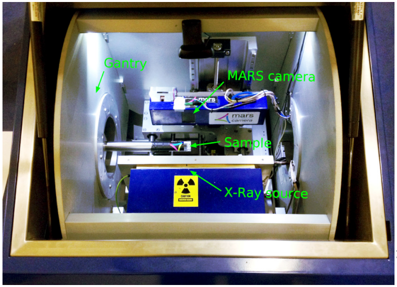
Motivation

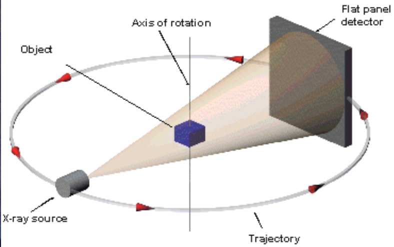
DLNP at JINR works with detectors based on Medipix chip with silicon and GaAs sensors. For study of their properties a CT scanner MARS was obtained. MARS allows to obtain the information about the structure of the scanned object. The reconstruction program written at DLNP works much slower than commercial analogs. The aim of this work is to realize the reconstruction algorithm using different parallel technologies such as OpenMP and Xeon Phi.



MARS microtomograph

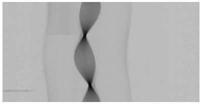
The gantry (x-ray source and camera) with the scanning equipment attached to the MARS tomograph and rotated around the scanned sample. The gantry rotation axis is horizontal. A test sample (up to 100 mm in diameter and 300 mm length) placed in the center and can be moved along the rotation axis.

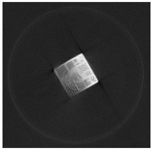


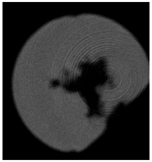


Input and output data

The result of scanning is a set of shadow projections obtained for different angles. The input data after preprocessing is given to reconstruction program as a set of filtered synograms. The program reconstruts the slices of the scanned object.


An example of sinogram


An example of reconstruted pantom slice


An example of reconstructed stone

FDK algorithm description

The FDK method is an approximate reconstruction algorithm for circular cone-beam tomography. The simplicity of the FDK method has made it the most used algorithm for cone-beam reconstruction. The algorithm consists of 3 steps:

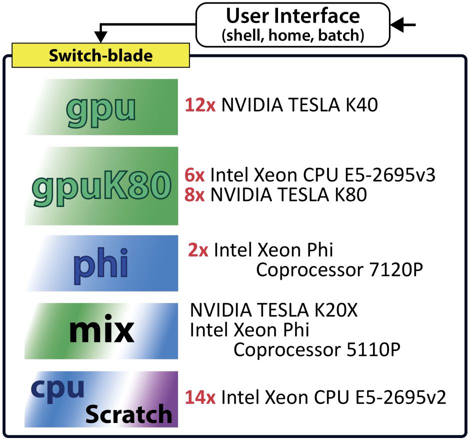
- pre-weighting;
- backprojection;
- summation.

The FDK image reconstruction formula

$$f_{FDK}(x,y,z) = \int_0^{2\pi} \frac{R^2}{(R+x\cos\beta+y\sin\beta)^2}\tilde{p}^F(\beta, R\frac{-x\sin\beta+y\cos\beta}{R+x\cos\beta+y\sin\beta}, z\frac{R}{R+x\cos\beta+y\sin\beta})d\beta,$$

where
$\tilde{p}^F(\beta,a,b) = \left(\frac{R}{\sqrt{R^2+a^2+b^2}}p^F(\beta,a,b)\right) * g^P(a)$ - a convolution of the input data $p^F(\beta,a,b)$ with ramp-filter;
a, b, R, β - geometrical parameters;
z - number of the reconstructed slice;
x, y - coordinates of the reconstructed point.

Hybrilit cluster scheme



Parallel implementations

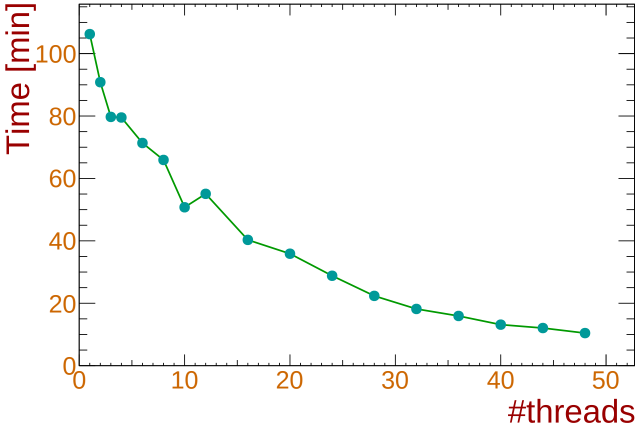
Next algorithm realizations were implemented:

- OpenMP CPU version;
- OpenMP version with extensions for native mode of Intel Xeon Phi coprocessors;
- OpenMP version with extensions for offload mode of Intel Xeon Phi coprocessors;

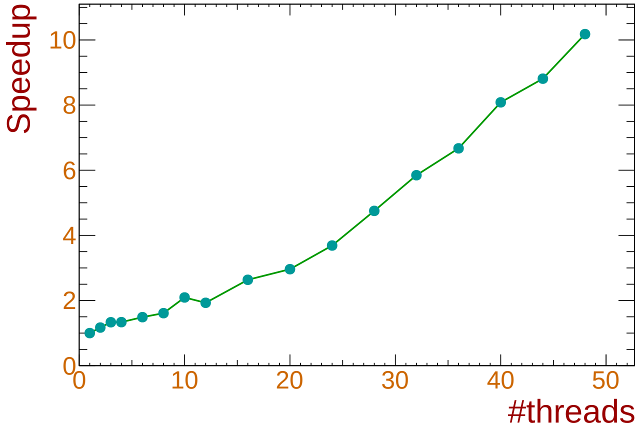
The next figures were received for reconstruction volume 1400*1400*256 pixels.

OpenMP realization

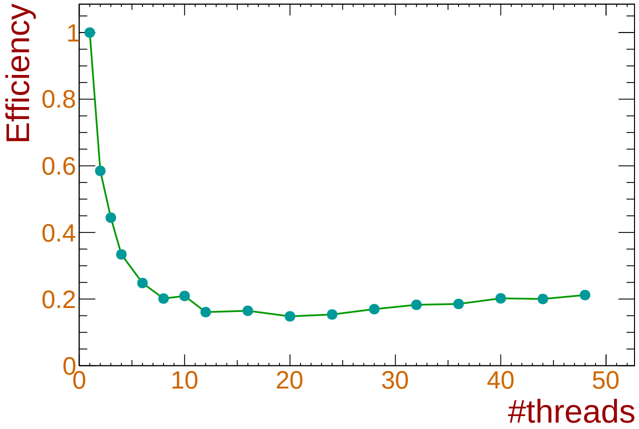
Calculation time for full reconstruction vs. the number of threads, OpenMP CPU realization



Speedup for full reconstruction vs. the number of threads, OpenMP CPU realization

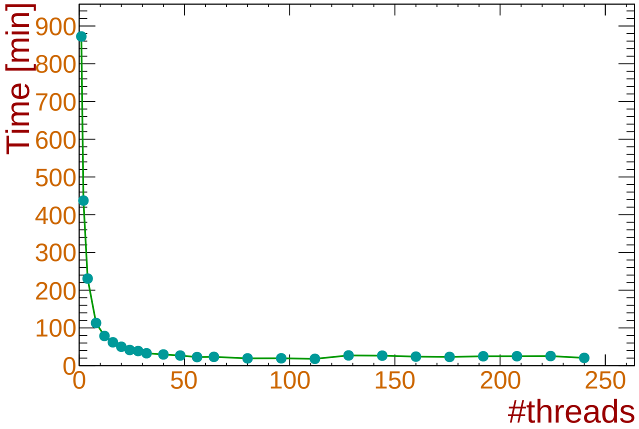


Efficiency for full reconstruction vs. the number of threads, OpenMP CPU realization

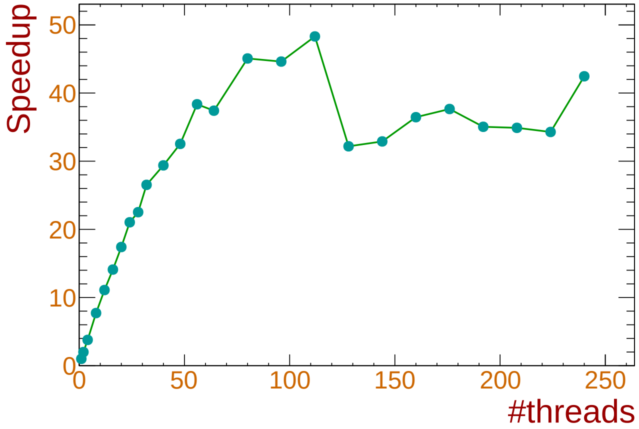


OpenMP realization for Xeon Phi native mode

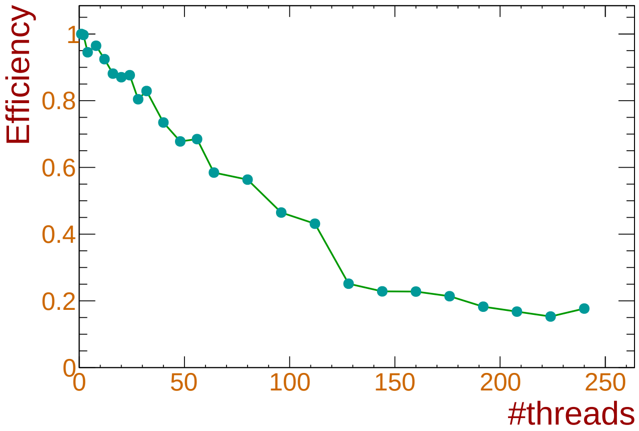
Computation time for full reconstruction vs. the number of threads, OpenMP for Xeon Phi native mode



Speedup for full reconstruction vs. the number of threads, OpenMP for Xeon Phi native mode

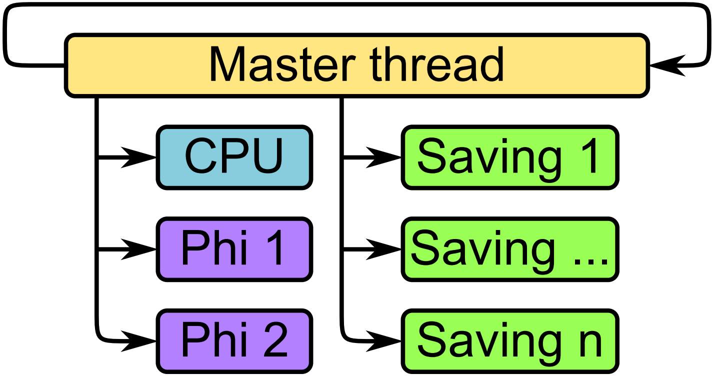


Efficiency for full reconstruction vs. the number of threads, OpenMP for Xeon Phi native mode

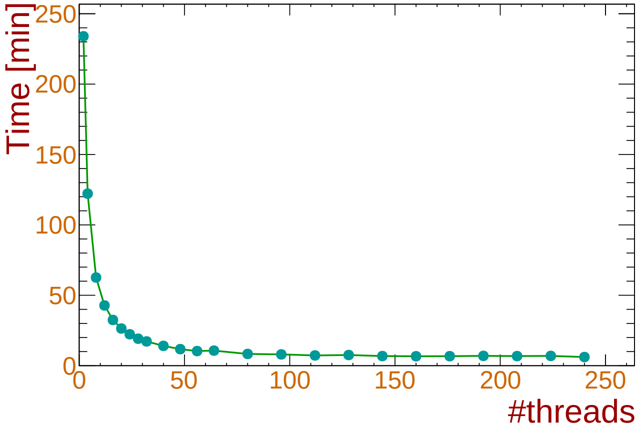


OpenMP realization for Xeon Phi offload mode

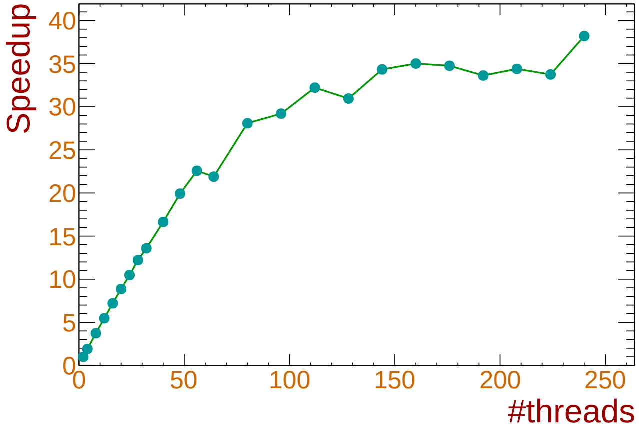
The offload-version architecture



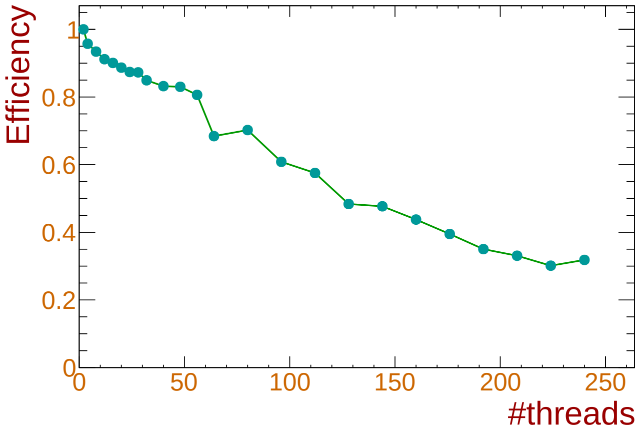
Computation time for full reconstruction vs. the number of threads for Xeon Phi offload mode, OpenMP



Speedup for full reconstruction vs. the number of threads for Xeon Phi offload mode, OpenMP

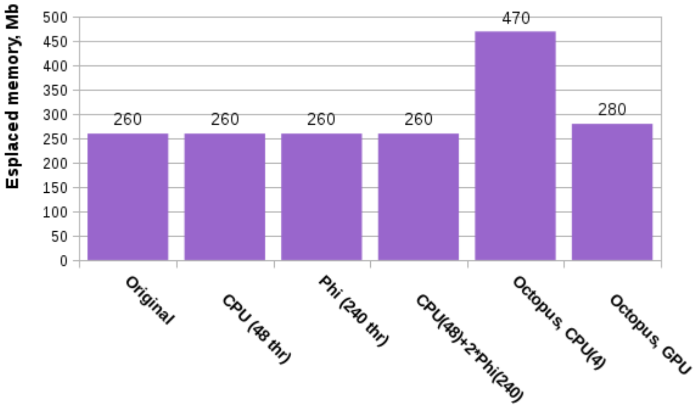


Efficiency for full reconstruction vs. the number of threads for Xeon Phi offload mode, OpenMP

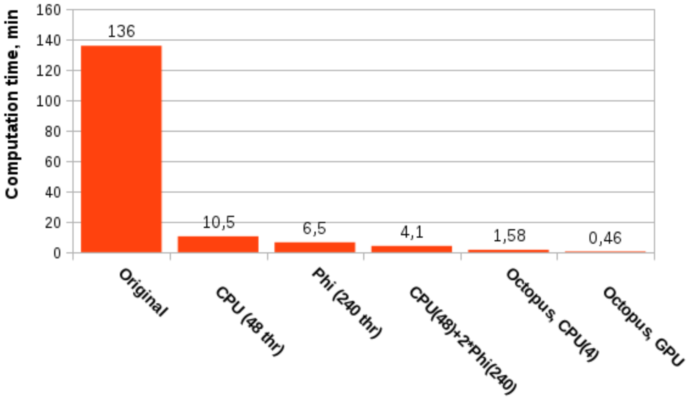


Resource usage

Memory usage



Processing time



Results

- Realisations for CPU and Xeon Phi using OpenMP parallel technology were implemented;
- The architecture was developed allowing simultanious usage of CPU and any number of Xeon Phi coprocessors;
- Parallel algorithms allow to reduce the time of calculations by the factor of 13 for CPU realization, 11.3 for Xeon Phi native mode, 21 for Xeon Phi offload mode and 34 for CPU+Xeon Phi offload case using OpenMP technology.

Outlook

- Study oportunities of algorithm optimization for further speedup;
- CUDA version development using present achivments.

References

A. Gongadze et al. Alignment and resolution studies of a MARS CT scanner, Physics of Particles and Nuclei Letters, September 2015, Volume 12, Issue 5, pp 725-735

L. A. Feldkamp, L. C. Davis, and J. W. Kress. Practical cone-beam algorithm. Journal of the Optical Society of America A., 1984, Vol. 1, Is. 6, p. 612–619.

Octopus Imaging Software, https://octopusimaging.eu/

Thank you for your attention!