Automated parallelization of sequential C-programs on the example of two applications from the field of laser material processing

M.S. Baranov[1,2], D.I. Ivanov[1,2], N.A. Kataev[1], A.A. Smirnov

Keldysh Institute of Applied Mathematics Russian Academy of Sciences[1], Lomonosov Moscow State University[2]

Optimization and parallelization of programs suppose an execution of a sequence analysis and transform passes. The choice of the optimal sequence depends on the application, on the purpose of optimization, on the architecture of the target computer system and technologies used for parallel programming. A huge size of space of possible optimization sequences complicates the automatic search for the best sequence for a certain program. Although manual parallelization takes into consideration these peculiarities, it is still a complicated and time-consuming process.

The approach proposed in this article involves an automatic execution of individual passes in the order specified by the user. Semi-automatic tools for transformation and analysis were developed.

Several types of static analysis can be performed: data dependence analysis with the detection of the dependence vector, privatizable variables analysis, induction and reduction variables recognition.

Each transform pass consists of a set of basic transformations selected by the user: variable propagation, loop-invariant code motion, loop unrolling, loop distribution, loop swapping, loop merging, loop permutation, iteration space shifting. The basic transformations are specified in the source code of the program as directives which are specified by using #pragma mechanism provided by the C standard. Transformations are performed over nests of perfectly nested loops. In some cases a transformation of arbitrary code blocks is allowed. There are two types of transformations: safe and unsafe ones. Before executing safe transformations their permissibility is checked by the mentioned transformation tool, in the case of unsafe transformations the responsibility for them lies on the user.

The proposed approach was applied for semi-automatic parallelization of two applications for laser material processing. After semi-automatic analysis and transformation these two programs were successfully manually parallelized using parallel programming technologies OpenMP and OpenACC. Table 1 shows the time of execution of three-dimensional problem in seconds on 4-cores processor Intel Core i7-3770 CPU 3.40GHz with active Hyper Threading and graphics accelerator NVIDIA GTX Titan. All versions of the program were compiled with option -O3. The original and transformed programs were also parallelized in automatic way using Intel compiler with option -parallel.

Table 1: Execution time (in seconds) of the programs Powder 3D (100x100x100, 100 iterations)

	Original	**Transformed**	**Auto (original)**	**Auto (transformed)**	**OpenMP**	**OpenACC**
1 thread	50.59	19.77			22.81	92.53
8 threads			62.38	11.61	11.30	
1 GPU						19.46

We plan to introduce the developed tools into the system for automated parallelization SAPFOR [1] in order to achieve a serial implementation of the parallelized program which may be efficiently mapped on modern clusters by the automatic parallelization compiler included in the system.

References

1. Bahtin V.A., Borodich I.G., Kataev N.A., Klinov M.S., Kovaleva N.V., Krukov V.A., Podderugina N.V. Dialog s programmistom v sisteme avtomatizacii rasparallelivanija SAPFOR [Dialogue with a programmer in the automatic parallelization environment SAPFOR]. Vestnik Nizhegorodskogo universiteta im. N.I. Lobachevskogo [Vestnik of Lobachevsky State University of Nizhni Novgorod]. 2012 No. 5 (2). P. 242–245.