Sisal 3.2 Language Features Overview*

Alexander Stasenko

A.P. Ershov Institute of Informatics Systems
astasenko@gmail.com

Abstract. This paper contains a short introduction of Sisal language and an overview of features introduced by Sisal 3.2 version compared to Sisal 3.1 version. Sisal 3.2 features a multidimensional array support, new abstractions like parametric types and generalized procedures, more flexible user-defined reductions, an improved interoperability with other programming languages and a specification of several optimizing source text annotations. Sisal 3.x version is used as an input language of a system of functional programming (SFP).

Keywords: functional programming, dataflow languages, scientific computations.

1 Introduction

Imperative programming languages and their traditional extensions like OpenMP [1] are not very convenient for a parallel program development because they require a low level specification of parallelism that can lead to subtle program errors that are hard to detect and fix. In addition, existing popular technologies for a parallel program specification such as OpenMP and nVidia CUDA [2] often rely on a specific machine architecture (e.g. OpenMP is designed to produce SMP-friendly code and CUDA was designed for nVidia GPUs) thus these technologies are not suitable for a portable specification of parallelism.

To overcome the above mentioned limitations in imperative languages other extensions such as OpenCL [3] and Intel Parallel Building Block are introduced. For example Intel Parallel Building Blocks augment C++ language with technologies like Threading Building Blocks [4] library for task-centric parallelism, Array Building Blocks for data-centric parallelism and Intel Ct [5] that combines task and data flow parallelism. These technologies allow developer to be more independent of hardware architecture as for example some work can be seamlessly offloaded to GPU.

There are several areas where parallelism is very important. One is game programming (visualization, physics) and other is scientific computations. C/C++ language that is targeted by most parallel extensions is quite suitable for programming games however scientific world is much more conservative and Fortran

* Work is partly supported by the Russian Foundation for Basic Research (RFBR grant no. 07-07-12050).

Acronym SMP stands for Symmetric Multiprocessing.
programming language [4] is still quite popular there. Since Fortran is also an imperative language that is designed to specify computations for a sequential Von Neumann architecture it is not easy to write parallel programs in Fortran even despite some its inherently parallel features such as built-in arrays.

Sisal programming language [7] was created to be a successor of Fortran language and to be more suitable for a parallel programming [8]. To achieve this Sisal was designed to be functional programming language with types and constructions that allow easy parallelization (that is why it is often called a dataflow language) on variety of machine architectures including quite exotic dataflow supercomputers. To ease transformation of imperative style Fortran programs Sisal contains different kinds of loop expressions that in fact are quite unusual in a world of functional languages. Sisal computations are always deterministic and can be described in a form of acyclic dataflow graph where nodes represent operations and edges represent data. Sisal supports exception handling in a form of special error value that every type contains.

It was demonstrated that Sisal performance need not be worse than programs written in imperative languages [8, 9] as it was for example demonstrated with controlled comparison on real-world image processing benchmark code [10] and other applications such as a Gauss-Jordon linear equation solver, a particle in cell simulation, a protein simulation program [11], the Lawrence Livermore Loops [12], a SIMPLE hydrodynamics code [13] and a one level barotropic weather simulation [14]. The acceptable Sisal performance was reached after some experiments with different forms of algorithms in Sisal language that take into account Sisal implemented compiler optimizations, not with different parallelization techniques. In contrast the imperative program exposes a variety of different parallelization techniques which are independent of algorithm. In addition it was showed that parallel code in imperative language was considerably large then the Sisal code since a large component of parallel imperative code is related to control overheads.

Sisal 3.2 programming language [15] introduced by this paper is a successor of Sisal 3.1 [16] language that was developed in IIS SB RAS. Sisal 3.1 integrated the most important features of Sisal 90 [17] and Sisal 3.2 integrated features of Sisal 2.0 [18] version. This paper contains an overview of features introduced by Sisal 3.2 version compared to Sisal 3.1 version. Sisal 3.x version is used as an input language of a system of functional programming (SFP) [19]. SFP aims to provide programmer with a convenient parallel program development environment on his personal computer and seamless transfer of his program to supercomputer environment without need for its adaptation.

This paper is organized in a following way. The section 2 describes the general features of Sisal language. The section 3 describes multidimensional arrays that came from Sisal 2.0 language. The section 4 describes new language abstractions such as parametric types and generalized functions and operations. The section 5 describes new way to specify user-defined reductions that were introduced in Sisal 90. The section 6 describes the way Sisal 3.2 programs can interoperable with other programming languages. The section 7 describes existing optimizing annotations or pragma statements.