In this paper we describe the design of an interactive, knowledge-based system for the semi-automatic transformation of Fortran 77 programs into parallel programs for a new supercomputer. The system is characterized by a powerful analysis component, a catalog of MIMD and SIMD transformations, and a flexible dialog facility. It contains specific knowledge about the parallelization of an important class of numerical algorithms.

Keywords: Multiprocessors, analysis of algorithms, program transformations

1 INTRODUCTION

The objective of the German supercomputer project SUPRENUM is the development of a machine capable of solving very large numerical simulation problems which cannot be handled by current computer systems. This objective is being pursued in two ways: firstly, by designing a highly parallel architecture, and secondly, by providing a powerful and efficient software environment, in particular with regard to application languages, tools, and parallel algorithms for the primary application areas.

The architecture of the SUPRENUM machine is characterized by a hierarchical multiprocessing structure whose essential components are nodes and clusters. The basic processing node is a single board computer, consisting of a 32-bit microprocessor connected to an 8 Mbyte dynamic memory, a 4 MFLOPS vector unit, and a dedicated communication processor. Each node has a local operating system which schedules the processes of the node, supports the communication with other nodes, and manages the local resources. A cluster consists of up to 16 nodes which communicate via an ultra-fast parallel bus with a bandwidth of 256 Mbits per second. The structure of an overall system is characterized by a matrix of clusters whose rows and columns are connected by one slotted ring bus (Fig. 1). A separate operating system machine manages the global resources, the distribution of the workload, and system recovery. Program execution is handled by the set of local node operating systems.

* The work described in this paper is being performed within the German Supercomputer project SUPRENUM and is supported by the Federal Ministry for Research and Technology (BMFT), F.R. Germany
The main application envisaged for the SUPRENUM machine is the numerical simulation of very large problems. Highly specialized algorithms are being developed for these problems, with a particular emphasis on the multigrid method [Stöb 82] which provides one of the fastest known disciplines for solving partial differential equations on general bounded domains. The differential equations are discretized on a grid with a fine mesh size; the solution process is a combination of standard relaxation methods and the computation of corrections on coarser grids. The computations at each grid point are usually local, a typical program will have from $10^3$ to $10^4$ grid points.

The SUPRENUM software environment includes optimizing compilers for extensions of the languages Fortran 77 (this extension will be called C-Fortran in the following) and Modula-2. These language versions specify a process concept, message-based synchronization and communication facilities, and array handling in the style of Fortran 8x [ANSI 86]. In addition, the software environment includes a very high-level specification language, an interactive program development facility and a simulation system. An important criterion for the acceptance of the new machine by the user community is a software tool that supports the transformation of existing Fortran programs into parallel programs that can be efficiently executed on the SUPRENUM computer. This paper describes the design of a semi-automatic parallelization system that solves this problem.

The paper is organized as follows: Section 2 gives an overview of the parallelization system, describes the main properties of its principal components, and relates the design to previous work. The set of all program transformations used in the system is called the catalog; it is treated in Section 3. Section 4 discusses the interactive component, outlining the main mechanisms used for the dialog-controlled analysis and transformation of programs. Our concluding remarks are to be found in Section 5.

2 SYSTEM STRUCTURE AND COMPONENTS

The structure of the parallelization system is shown in Fig. 2. Let $P$ denote a Fortran source program. The overall task of the system, applied to $P$, is the transformation of $P$ into a semantically equivalent parallelized program $P^*$ in C-Fortran. This process can be outlined as follows: First, the Front End is applied to $P$, creating an intermediate program representation. Then, by a coordinated repetitive application of analysis processes and catalog transformations a representation is obtained which can be used by the Back End to generate $P^*$. The whole process can be interactively controlled by the user. In order to provide maximum flexibility, the system may be applied to arbitrary C-Fortran programs including programs that have already been (partially) parallelized.

We now discuss the individual components in more detail: The Front End, applied to a C-Fortran program $P$, creates an intermediate representation consisting of two parts: structural information and analysis information. The structural information specifies the abstract syntax of $P$ together with the results of conventional semantic analysis, auxiliary control information, and a decomposition of the program into segments which are to be executed as processes at run time*. It includes [Aho 86, Hecht 77, *

This is the decomposition initially present in $P$ immediately after application of the Front End; for a sequential program $P$, it will be trivial.

---

* This is the decomposition initially present in $P$ immediately after application of the Front End; for a sequential program $P$, it will be trivial.