Chapter 1.
Getting Started With the Parallel Analyzer View

This chapter helps you get the ProDev ProMP parallel analyzer view running on your system. It contains the following sections:

- Setting up your system, see Section 1.1.
- Running the parallel analyzer view: general features, see Section 1.2.
- Tutorials, see Section 1.3.

  Note: This product was formerly called WorkShop Pro MPF.

1.1 Setting Up Your System

To install the ProDev ProMP software, you should have at least 16 MB of memory; 32 MB improves overall performance.

ProDev ProMP requires the following software versions (or later versions):

- IRIX system software version 6.2
- MIPSpro Auto−Parallelizing Fortran 77, release 7.2.1
- MIPSpro Auto−Parallelizing Fortran 90, release 7.3
- MIPSpro Auto−Parallelizing C, release 7.3
- ToolTalk 1.1
- WorkShop 2.0

To determine what software is installed on your system, enter the following at the shell prompt:

```
% versions
```

If the items mentioned in this section are not installed, consult your sales representative, or in the United States call the Silicon Graphics Technical Assistance Center at 1−(800)−800−4SGI. To order additional memory, consult your sales representative or call 1−(800)−800−SGI1.

If you have all the software and memory you need, you can install the ProDev ProMP software.

- For general instructions about software installation, consult the man pages `inst(1M)` and `swmgr(1M)`, and the manual, *IRIX Admin: Software Installation and Licensing*.
- See also *Developer Magic: ProDev Pro MP Release Notes* for specific installation instructions.

The executable is `cvpav`(1), which is installed in `/usr/sbin`.

1.2 Running the Parallel Analyzer View: General Features

The process of using the parallel analyzer view involves two steps:

1. Compiling a program with appropriate options.
2. Reading the compiled files with the parallel analyzer view.

1.2.1 Compiling a Program for the Parallel Analyzer View

Before starting the parallel analyzer view to analyze your source (in this case, Fortran source), run one of the auto−parallelizing compilers with the appropriate options. For the tutorials presented in subsequent chapters, makefiles are provided. You can adapt these to your specific source or enter one of the following commands:

```
% f90 -apo keep -O3 sourcefile.f  
% f77 -apo keep -O3 sourcefile.f
```

The compiler generates its usual output files and an analysis file (`sourcefile.anl`), which the parallel analyzer reads.

The command−line options have the following effects:

`-apo keep`
: Saves a `.anl` file, which has necessary information for the parallel analyzer view.

`-O3`
: Sets the compiler for aggressive optimization. The optimization focuses on maximizing code performance, even if that requires extending the compile time or relaxing language rules.

See the *MIPSpro Fortran 77 Programmer's Guide*, *MIPSpro Compiling and Performance Tuning Guide*, and the `f90`(1) or `f77`(1) man page for more information.

> **Note:** The `cvpav` command assumes that the **`-apo keep`** option was used on each of the Fortran source files named in a single executable or file specifying several executables. If this is not the case, a warning message is posted, and the unprocessed files are marked by an error icon within the parallel analyzer's subroutines and files view, see Section 6.6.4.

1.2.1.1 Generating Other Reports

While they are not part of the parallel analyzer view, other parallelization reports can be generated using the following command−line options:

`-apo list`
: Produces a `.l` file, a listing of those parts of the program that can run in parallel and those that cannot.

`-mplist`
: Generates the equivalent parallelized program in a `.w2f.f` file.

These reports are text files that can be used for analysis. For more detailed information, see *MIPSpro Auto−Parallelizing Option Programmer's Guide.*

1.2.1.2 OpenMP and PCF Directive Support

The MIPSpro auto−parallelizing Fortran compilers support OpenMP directives, unless you are compiling with the `-o32` option. If you put OpenMP directives in your `o32` code, they are treated as comments rather than being interpreted. For more information on OpenMP directives, see the following:

- Section 2.11
- The OpenMP Architecture Review Board web site at the following URL: `http://www.openmp.org/`

Although using OpenMP directives is recommended, the auto−parallelizing compilers still support PCF directives. For information on analyzing loops containing PCF directives, see Appendix A.

1.2.2 Reading Files With the Parallel Analyzer View

You can run the parallel analyzer view on any of the following objects:

- A source file
- An executable
- A list of files

To run the parallel analyzer view for one of these cases, enter one of the following commands:

```
% cvpav -fsourcefile.f  
  
% cvpav -eexecutable  
% cvpav -Ffileset-file
```

The `cvpav` command reads information from all source files compiled into the application.

The parallel analyzer view has several other command line options, as well as several X Window System resources that you can set. See the man page `cvpav(1)`(1) for more information.

> **Note:** If you receive a message related to licensing when you start `cvpav`, refer to Chapter 7 in the *ProDev Pro MP Release Notes*. To access the notes, enter the `grelnotes`(1) command and choose `Products > ProMP`.

1.3 Tutorials

For a more detailed introduction to the parallel analyzer view, follow one of tutorials provided with the product in the following chapters:

- Examining Loops for Fortran 77 Code, seeChapter 2.
- Examining Loops for Fortran 90 Code, see Chapter 3.
- Examining Loops for C Code, see Chapter 4.
- Using WorkShop With Parallel Analyzer View, see Chapter 5.
- Examining Loops Containing PCF Directives, see Appendix A.