# Absoft AWE Auto-Parallel Example

This document describes the example program supplied with Pro Fortran that demonstrates the auto-parallelism capabilities of the Absoft compiler. The program also demonstrates features of the **A**bsoft **W**indow **E**nvironment (AWE) including menus, menu commands, alert boxes, spreadsheets, and plots.

The program can be compiled and run unchanged on any operating system platform supported by Absoft, including Windows, Linux, and OS X. Additional documentation on AWE can be found in the *Absoft User Guide* for your compiler.

## Example Description

The example sets up 3 menu entries and callback functions for the entries. The first menu command is **Solve** and calls routines to accumulate timings for program runs with increasing number of threads dedicated to the task. The second menu command is **Chart** which creates a spread sheet in AWE and displays the results. The third menu command is **Plot** which displays the results as a plot. The **Chart** and **Plot** menu entries are disabled until after the **Solve** command is issued to accumulate the data.

The example consists of 3 source files and an AbsoftTools project file. The project file is described in the next section, *Building the Example*. The source files are:

### `AWE_Preferences.f95`

This file contains the call-back functions for customizing the behavior of AWE. It is automatically added to your project when an AWE application target type is selected in AbsoftTools. The supplied file is unchanged from the default.

### `solver.f90`

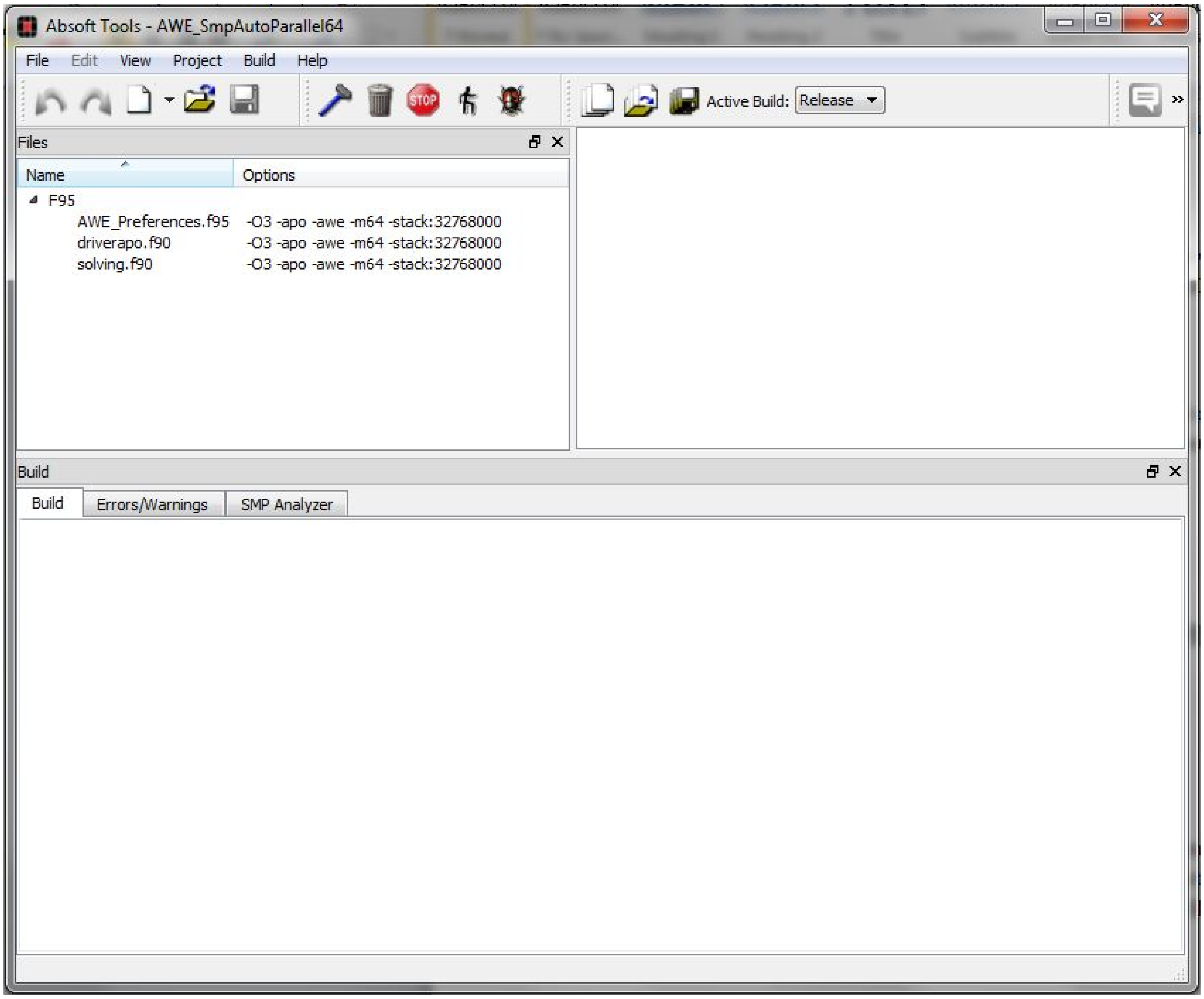
This is the problem solving subroutine for the example. Its only purpose is to load the computer CPU to accumulate execution timings. The code is automatically parallelized with the `–apo` compiler option.

### `driverapo.f90`

This is the driver program for the example. It consists of a module to maintain global program data entities, a main program for initialization, and the 3 menu command callback functions.

# Building the Example

To build the example, first start AbsoftTools. Select **Open Project** from the AbsoftTools **Project** menu and browse to the "examples" directory in the main Absoft directory on your computer. Open the directory named "AWE_APO_Example" and then open the project file named "AWE_SmpAutoParallel64.atools". Your AbsoftTools window will now look similar to this:



Absoft Tools - AWE_SmpAutoParallel64

Active Build: Release

| Name | Options |
|---|---|
| F95 | |
| AWE_Preferences.f95 | -O3 -apo -awe -m64 -stack:32768000 |
| driverapo.f90 | -O3 -apo -awe -m64 -stack:32768000 |
| solving.f90 | -O3 -apo -awe -m64 -stack:32768000 |

From the **Build** menu, select **Build** or click on the hammer icon. The application will be ready to run when the Build completed message is displayed in the **Build** pane in the bottom half of the AbsoftTools window.

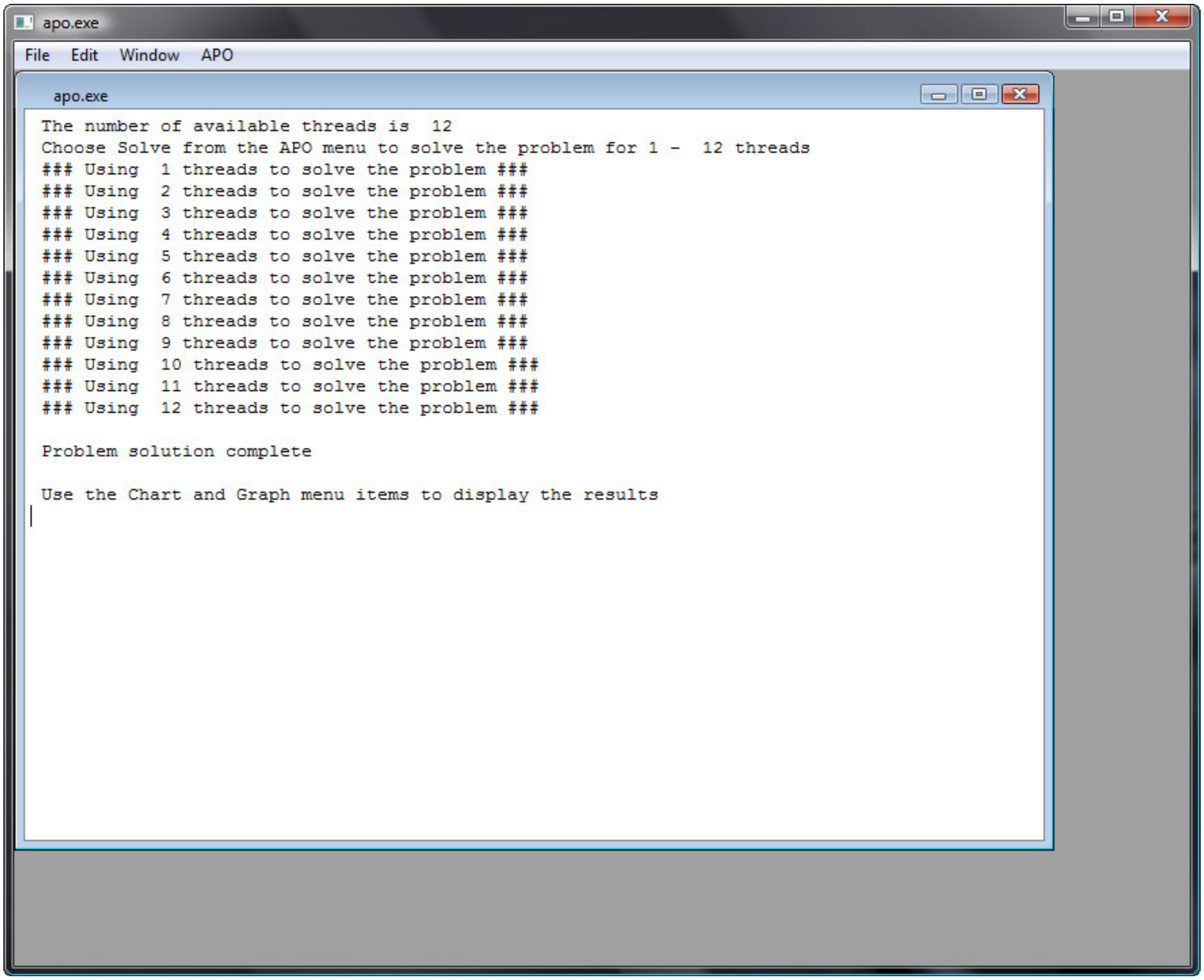
## Running the Example

To run the example, select **Execute** from the AbsoftTools **Build** menu or click on the running man icon. An AWE window will open and the following text will be displayed in the text window:

```
The number of available threads is 12
Choose Solve from the APO menu to solve the problem for 1 - 12 threads
```

**Note:** the number of threads will vary depending on your specific hardware configuration.

From the **APO** menu, select **Solve** to calculate the timings for increasing numbers of threads dedicated to the calculation. When the calculations are complete, the AWE window will look similar to this:

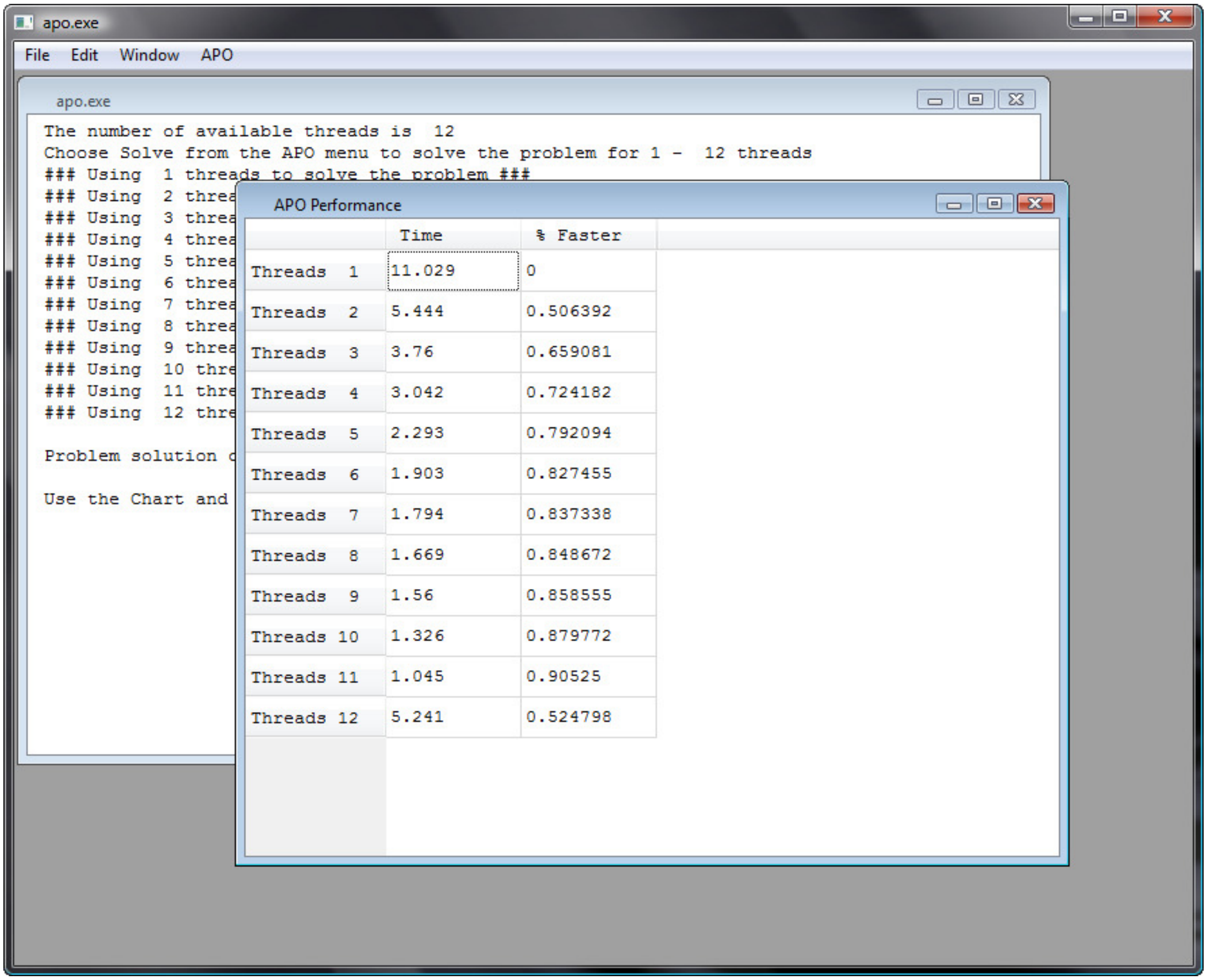


```
The number of available threads is  12
Choose Solve from the APO menu to solve the problem for 1 -  12 threads
### Using  1 threads to solve the problem ###
### Using  2 threads to solve the problem ###
### Using  3 threads to solve the problem ###
### Using  4 threads to solve the problem ###
### Using  5 threads to solve the problem ###
### Using  6 threads to solve the problem ###
### Using  7 threads to solve the problem ###
### Using  8 threads to solve the problem ###
### Using  9 threads to solve the problem ###
### Using  10 threads to solve the problem ###
### Using  11 threads to solve the problem ###
### Using  12 threads to solve the problem ###

Problem solution complete

Use the Chart and Graph menu items to display the results
```

Now the **Chart** and **Plot** menu commands can used to display the results. The spreadsheet will look similar to this:



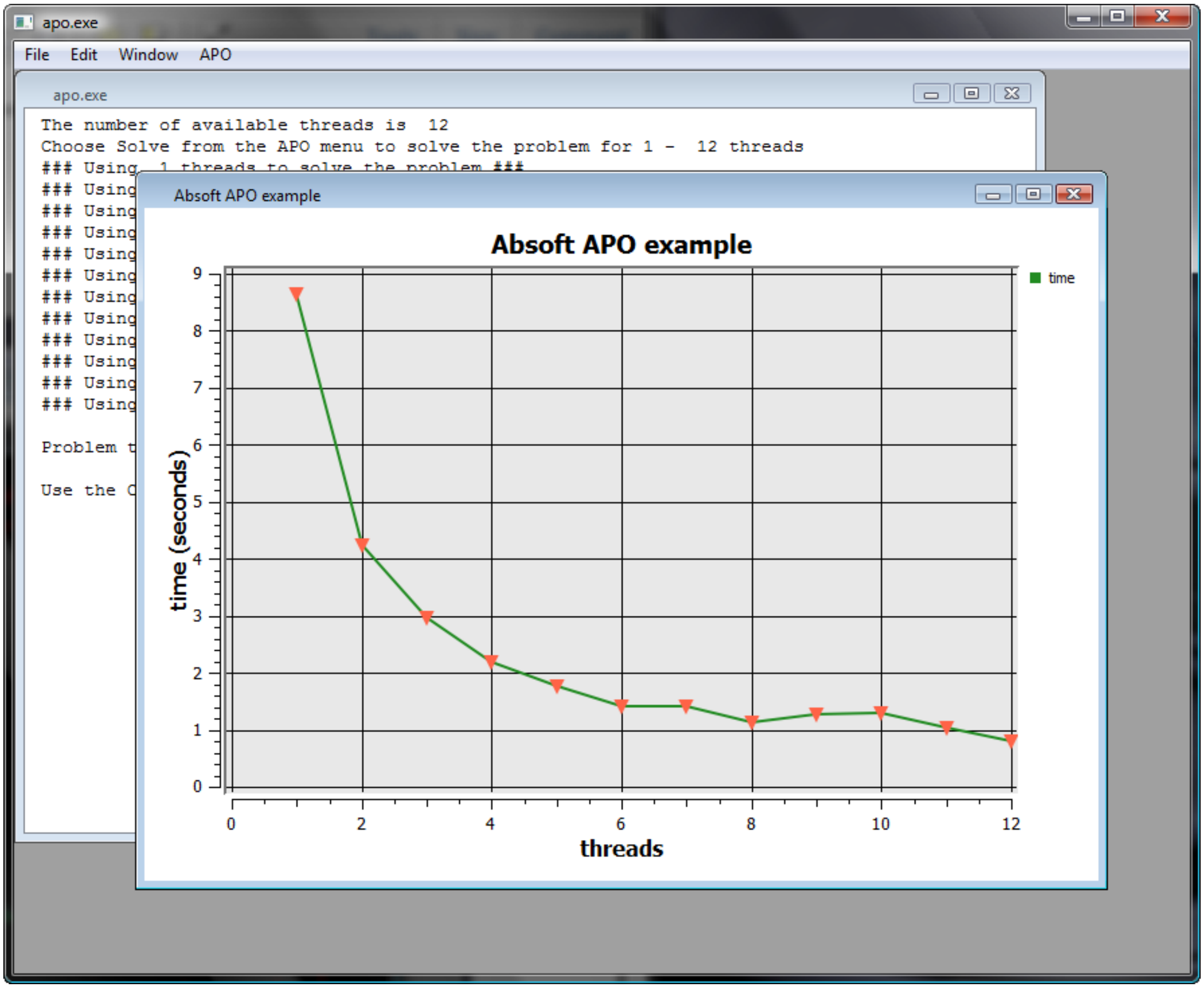
```
The number of available threads is  12
Choose Solve from the APO menu to solve the problem for 1 -  12 threads
### Using  1 threads to solve the problem ###
### Using  2 threa
### Using  3 threa
### Using  4 threa
### Using  5 threa
### Using  6 threa
### Using  7 threa
### Using  8 threa
### Using  9 threa
### Using  10 thre
### Using  11 thre
### Using  12 thre

Problem solution c

Use the Chart and
```

| | Time | % Faster |
|---|---|---|
| Threads 1 | 11.029 | 0 |
| Threads 2 | 5.444 | 0.506392 |
| Threads 3 | 3.76 | 0.659081 |
| Threads 4 | 3.042 | 0.724182 |
| Threads 5 | 2.293 | 0.792094 |
| Threads 6 | 1.903 | 0.827455 |
| Threads 7 | 1.794 | 0.837338 |
| Threads 8 | 1.669 | 0.848672 |
| Threads 9 | 1.56 | 0.858555 |
| Threads 10 | 1.326 | 0.879772 |
| Threads 11 | 1.045 | 0.90525 |
| Threads 12 | 5.241 | 0.524798 |

The plot will look similar to this:



```
The number of available threads is  12
Choose Solve from the APO menu to solve the problem for 1 -  12 threads
### Using  1 threads to solve the problem ###
### Using
### Using
### Using
### Using
### Using
### Using
### Using
### Using
### Using
### Using
### Using

Problem t

Use the C
```