Automatic Differentiation on Distributed Memory MIMD Systems

Luigi De Luca and Piero Fiorino

Department of Electronics, Informatics and Systems
University of Calabria - 87036 Rende (Italy)
E-mail: deluca@deis2.deis.unical.it

Abstract
The main target of this paper is to develop an innovative software for the Automatic Differentiation of separable functions, exploiting the parallel features of a distributed memory parallel system (MIMD architecture).

The developed software, written in Fortran, uses the Express tool, thus being easily portable on the several parallel systems supporting Express. It consists of a set of subroutines calculating the function, gradient and hessian values; due to its user friendly interface, it is greatly suitable for using inside Fortran iterative algorithms needing these values; it is not necessary any pre-compiling phase.

Introduction
The term Automatic Differentiation outlines a particular algorithm for differential calculus. Through the computation of the derivatives of elementary functions, obtained by the successive decomposition of the original function, it is possible to reconstruct the derivative values in a given point for very complex structured functions as well as for simple functions.

The Automatic Differentiation techniques are a good alternative to the other methods. In fact, these techniques allow computing of the exact derivative values without symbol handling, but through numerical operations only. There are no numerical approximations but those connected to the arithmetic accuracy of the machine. Moreover, the computational effort to evaluate the derivative values could be greatly lower than using finite differences methods.

The specific contexts which can obtain benefits from the use of the Automatic Differentiation are:
- non-linear optimization problems;
- "real-time" problems in which it is necessary to obtain reliable and fast solutions;
- analysis of dynamic system controlled by differential equations, such as applications of the structural analysis and fluid-dynamics.

AutoDif: an Automatic Differentiation Software Tool
There are several papers concerning the automatic differentiation; in particular, many works are related to the automatic differentiation theory. The contribution of this work consists in the development and the implementation of a parallel approach to Automatic Differentiation technique, for the computation of the gradient and the hessian of a function

\[ f: \mathbb{R}^n \rightarrow \mathbb{R} \]

in a distributed memory multiprocessor system; moreover, the implementation supplies a user friendly interface: in fact, it includes a sort of compiler so that the user only has to write the input function in a file with a shape similar to mathematical
formalism; the program reads this function and codifies it to successive automatic differentiation (see fig. 1). Using the software inside iterative algorithms, this phase will be executed only once, before starting iterations.

![Diagram](image)

The automatic differentiation technique being used is the reverse mode[6] both for gradient and hessian calculation[2].

The functions considered in this paper belong to the class of the "separable" functions with a sparse-structured hessian, in the form of:

\[ f = \sum_{i=1}^{n} f_i \]

where each \( f_i \) depends from \( x_i \) and a small number of other independent variables; all the \( f_i \) have the same structure. For example:

\[ f(x) = \sum_{i=2}^{n} x_i (\sin(x_{i+1}) - \cos(x_{i-1})) \]

This kind of functions, massively used in the optimization problems, has a particular structure giving a good exploitation of the parallel machine capabilities.

Computing System and Software

The computational experiments were carried out on a INMOS T800 transputer network clocked at 25 MHz. The software environment used for developing the parallel code is Express 3.0 and the 3L fortran compiler.

Test Problems

Several test problems, with different characteristics, were considered. They are characterised by the possibility of changing the number of variables and represent meaningful tests due to their analytical structure. For all the test functions, the number of variables selected was: 50, 100, 150, 200. The available memory prevented the possibility to increase the number of variables.

Numerical Results

The numerical results are collected with the aim to point out the numerical behaviour of the Automatic Differentiation techniques in a parallel environment, using an increasing number of processors.

In the tables shown below, referred to some of the considered test problems[7], the CPU time and the speed up factor have been reported for different numbers of processors in the transputer network and for different problem sizes.