# A Compiler-automated Array Compression Scheme for Optimizing Memory Intensive Programs

Lixia Liu
Department of Computer Science
Purdue University, West Lafayette, IN 7907
liulixia@cs.purdue.edu

Zhiyuan Li
Department of Computer Science
Purdue University, West Lafayette, IN 7907
li@cs.purdue.edu

## ABSTRACT

This paper proposes a compiler-automated array compression scheme to reduce the memory bandwidth consumption of programs and thereby to improve their execution speed. Three encoding methods are developed for such compression. Formulas are derived to analyze the cost and benefit of such methods. To ease the programmer's effort for writing and maintaining complex source code that utilizes compression, we implement our technique in a compiler which automatically transforms the program into different versions corresponding to different encoding methods. The compiler also inserts operations to adaptively invoke the preferred version at run time, including the original version which performs no compression. Results show that our compiler-automated adaptive scheme improves the execution speed over the original version by an average of 9% for a set of benchmark programs which perform memory-intensive sparse matrix-vector multiplications (SpMV). These results take into account of overhead to make the adaptive decision. When tested separately, the individual encoding methods speed up program execution by as high as 41%, which compares favorably against previous compression methods manually applied to SpMV.

## Categories and Subject Descriptors

D.3.4 [**Programming Languages**]: Processors-Optimization

## General Terms

Algorithms, Performance, Design

## Keywords

Compression, memory intensive programs, adaptive code selection, bandwidth consumption reduction, compiler implementation

## 1. INTRODUCTION

Research has shown that the memory bus on multicore chips is a major performance bottleneck for many numerical applications [7][10][11]. In this paper, we propose a compiler-automated compression scheme to reduce memory bandwidth consumed by integer or pointer arrays. Several previous studies manually applied compression to index arrays used in sparse matrix-vector multiplication (SpMV) [1][2][6]. Experiments showed that, while the bandwidth reduction can be quite high in many cases, the overhead introduced by the compression operations can also be significant. Branch mispredictions and loop overhead may also be increased. Performance of numerical kernels can be sensitive to these factors. In certain cases, compression may actually degrade the performance [6]. Thus, it is important to understand the benefit and the cost of compression quantitatively when one designs an efficient compression method. Ideally, the compression decision must be made adaptively, based on the cost-benefit analysis with parameters obtained at run time.

To pursue such an adaptive scheme, we develop three encoding methods for array compression. We show by experiments that, when applied separately, these methods compare favorably with previous methods that were manually applied. More importantly, we are able to develop formulas to quantify the benefit and cost of each of the encoding methods based on parameters available at run time. The decision on whether to compress and which encoding method to use can then be made at run time. Since the adaptive scheme requires the program to incorporate all three encoding methods in addition to the code without compression, the program structure can be quite complex. To ease the programmer's effort for writing and maintaining the adaptive program, we implement our technique in a compiler which automatically transforms the original program (without compression) into different versions corresponding to different encoding methods. The compiler also inserts operations to adaptively invoke the preferred version at run time. This is in contrast to previous compression methods which require the special handling of compressed arrays to be programmed manually, in some cases even in an assembly language [6].

The rest of the paper is organized as follows. In Section 2, three encoding methods used in our adaptive compression scheme are introduced. Section 3 describes the framework and benefit model of our scheme. Experimental results are presented in Section 4 to demonstrate the effectiveness of the scheme. We discuss related work in Section 5 and make concluding remarks in Section 6.

## 2. ENCODING METHODS

In this section, we present three encoding methods used in our adaptive compression scheme, namely Double Array Compression (DAC), Delta Double Array Compression (DDAC), and Special Delta Double Array Compression (SDDAC). These methods are simple enough to be automated by compiler transformation and, at the same time, they are highly competitive so far as the decompression overhead is concerned.


*ICS'10*, June 2-4, 2010, Tsukuba, Ibaraki, Japan.