A SYSTOLIC ARRAY PARALLELIZING COMPILER

THE KLUWER INTERNATIONAL SERIES IN
ENGINEERING AND COMPUTER SCIENCE

HIGH PERFORMANCE COMPUTING
Systems, Networks and Alrogrithms

Consulting Editor

H.T. Kung

A SYSTOLIC ARRAY PARALLELIZING COMPILER

by

Ping-Sheng Tseng
Bell Communications Research, Inc.

with a foreword by
H.T. Kung

KLUWER ACADEMIC PUBLISHERS
Boston/Dordrecht/London

Distributors for North America:
Kluwer Academic Publishers
101 Philip Drive
Assinippi Park
Norwell, Massachusetts 02061 USA

Distributors for all other countries:
Kluwer Academic Publishers Group
Distribution Centre
Post Office Box 322
3300 AH Dordrecht, THE NETHERLANDS

Library of Congress Cataloging-in-Publication Data

Tseng, Ping-Sheng, 1959-
A systolic array parallelizing compiler / by Ping-Sheng Tseng.
p. cm. — (The Kluwer international series in engineering and computer science ; #106. High performance computing)
Revision of the author's thesis (Ph.D.)—Carnegie Mellon University.
Includes bibliographical references and index.
ISBN-13:978-1-4612-8835-0 e-ISBN-13:978-1-4613-1559-9
DOI:10.1007/978-1-4613-1559-9

1. Parallel processing (Electronic computers) 2. Compilers (Computer programs) I. Title. II. Series: Kluwer international series in engineering and computer science ; SECS 106. III. Series: Kluwer international series in engineering and computer science. High performance computing.
QA76.58.T74 1990
005.4'53—dc20 90-4751
CIP

Contents

List of Figures

Foreword

Widespread use of parallel processing will become a reality only if the process of porting applications to parallel computers can be largely automated. Usually it is straightforward for a user to determine how an application can be mapped onto a parallel machine; however, the actual development of parallel code, if done by hand, is typically difficult and time consuming. Parallelizing compilers, which can generate parallel code automatically, are therefore a key technology for parallel processing.

In this book, Ping-Sheng Tseng describes a parallelizing compiler for systolic arrays, called AL. Although parallelizing compilers are quite common for shared-memory parallel machines, the AL compiler is one of the first working parallelizing compilers for distributed-memory machines, of which systolic arrays are a special case. The AL compiler takes advantage of the fine grain and high bandwidth interprocessor communication capabilities in a systolic architecture to generate efficient parallel code.

While capable of handling an important class of applications, AL is not intended to be a general-purpose parallelizing compiler. Instead, AL is designed to be effective for a special class of computations that use arrays and loops. AL relies on the fact that for these computations, the user can easily provide "hints" or mapping strategies to guide the compiler to distribute data structures and loop iterations. Using these hints, the AL compiler generates the local program for each processor, manages the interprocessor communication, and parallelizes the loop execution. A fundamental contribution of AL, which goes beyond its current implementation, is the identification of what to capture in these hints that the user can easily provide and the compiler can effectively use.

AL has proven to be extremely effective in programming the Warp systolic array developed by Carnegie Mellon. AL was used to port the nontrivial LINPACK QR (SQRDC), SVD (SSVDC), LU (SGEFA), and back substitution (SGESL) routines to Warp all in one person-week. AL has been used in several applications, including the porting of a large Navy signal processing application to Warp. The AL compiler has also been used to generate parallel code for automatic schedulers that map a large set of high-level signal processing tasks onto Warp.

Carnegie Mellon's experience in programming Warp clearly indi-

cates the effectiveness of special-purpose parallelizing compilers such as AL. These tools, which we also call parallel program generators, can improve a programmer's productivity by several orders of magnitude. In fact, some applications would never have been brought up on Warp if the user did not have access to AL; for these applications explicit programming of interprocessor communication is simply too difficult to be done by a human being. Besides AL, the Warp project has developed several other parallel program generators. One of them, called Apply, has been extensively used to generate parallel code for image processing applications. Both AL and Apply generally produce code better than or as good as hand-written code.

The success of parallel program generators such as AL is encouraging. With them we are assured that programming parallel machines can be as easy as programming sequential machines for some important application areas. In this sense, these programming tools have helped legitimatize our effort in building parallel computers. Besides providing useful tools for programming parallel machines for special applications, these parallel program generators are also significant in giving insights into information that more general-purpose parallelizing compilers of the future need to capture.

The book is an outgrowth of Ping-Sheng Tseng's Ph.D thesis from Carnegie Mellon. The book gives a complete treatment of the

design, implementation and evaluation of AL. I am pleased to write the Foreword for this outstanding piece of work and hope that the book will inspire researchers to further this very important area of parallel processing.

H. T. Kung
April, 1990
Pittsburgh, PA

Acknowledgements

This work evolved from my thesis research at Carnegie Mellon University. I want to thank my thesis advisor H. T. Kung. When I started this research project, Kung was the only one who believed that I could make this thesis happen. Without his vision, support and encouragement, I might have changed my thesis topic before I started. Thanks to Monica Lam and Peter Lieu. Monica implemented the W2 compiler directives for me. Peter has been very responsive to my W2 bug reports. Without their help, all the experimental work might have never been completed. The members of my thesis committee are H. T. Kung, Gary Koob, Thomas Gross, and George Cox. Their invaluable comments helped me to improve the quality of this presentation. I want to thank my former officemate Robert Cohn. Robert read my manuscript several times and helped me to make this presentation much more readable. I would also like to thank the Defense Advanced Research Projects Agency for supporting this work. Finally, I would like to thank my wife Ly-June

and my father Po and my mother Sou-Sou for their support in my graduate study.

A SYSTOLIC ARRAY PARALLELIZING COMPILER