

Ain Shams University
Faculty of Engineering

***NEW DYNAMIC TECHNIQUES FOR MAPPING
GENERAL PARALLEL NESTED LOOPS ON
MULTIPROCESSOR SYSTEMS***

By

HANY ISKANDER FAHMY ISKANDER
(B.Sc.)

A Thesis



Submitted in the fulfillment of the Requirements of
the degree of Master of Science in Electrical
Engineering

[Computer and Systems Engineering]

621.391
H. I

Supervised by

Prof. Dr. OSMAN A. BADR

Professor, Faculty of Engineering, Ain Shams University

Dr. SALWA M. NASSAR

Associate Professor, Electronics Research Institute

CAIRO-NOV. 1992



To my father and mother
whose love, encouragement, patience
and faith made this all possible.



ACKNOWLEDGEMENTS

This work was carried out jointly between Ain Shams University and the Electronics Research Institute (National Research Center)

I would like to acknowledge my heartfelt thanks to Professor Osman A. Badr and Dr. Salwa M. Nassar for their invaluable help and sustained interest they have taken in this work.

I am deeply indebted to Dr. Osman and Dr. Salwa for their guidance, assistance, friendship, patience, enthusiasm and encouragement throughout the duration of this work. I am also grateful for their invaluable advice and fruitful discussions.

Finally, I am especially grateful to my parents for their thoughtfulness, understanding and appreciation.

STATEMENT

This disseration is submitted to Ain Shams University for degree of M.Sc. in Electrical Engineering.

The work included in this thesis was carried out by the author in the Departement of Computer and Systems Engineering, Ain Shams University, and in the Departement of Computers and Systems in the Electronics Research Institute (National Research Center).

This thesis had been published in two papers, but no Part of this thesis has been submitted for a degree or qualification at any other University or Institution.

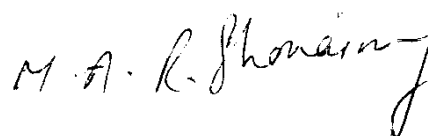
Date : 9/11/92
Signature : Hany Iskander
Name : Hany Iskander Fahmy

EXAMINAR COMMITTEE

NAME, Title & Affiliation

Signature

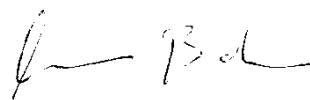
1. Prof. Dr. M.A.R. GHONAIMY
Ain Shams University



2. Prof. Dr. Fathy H. Saleh
Cairo University



3. Prof. Dr. Osman A. Badr
Ain Shams University



4. Dr. SALWA M. NASSAR
Electronics Research Institute



TABLE OF CONTENTS

Table of contents

Abstract

CHAPTER 1: Introduction

1.1 Introduction 1

CHAPTER 2: Survey on Parallel Processing Systems

2.1 Introduction 3

2.2 Historical Achievement 3

2.3 Levels of Parallel Processing 4

2.4 Parallelism in Uniprocessor Systems 4

2.4.1 Multiplicity of Functional Units 5

2.4.2 Parallelism within the CPU 5

2.4.3 Overlapping the CPU and I/O operations 5

2.4.4 Using Hierarchical Memory Systems 5

2.4.5 Balancing of the Subsystem Bandwidth 5

2.4.6 Multiprogramming and Time Sharing 6

2.5 Parallel Computer Structures 6

2.5.1 Pipelined Computers 6

2.5.2 Array Computers 8

2.5.3 Multiprocessor Systems 8

2.5.3.1 Loosely Coupled Multiprocessors 9

2.5.3.2 Tightly Coupled Multiprocessors 9

2.5.3.3. Interconnection Networks	10
2.5.3.3.1 Time Shared or Common Buses	10
2.5.3.3.2 Crossbar Switch and Multiport Memories	10
2.5.3.3.3 Multistage Networks for Multiprocessor	10
2.6 Performance of Parallel Computers	11
2.7 Data Flow Computers	12
2.8 Architecture Classification Schemes	13
2.9 Serial Versus Parallel Processing	14
2.10 Parallel Programming	15
2.11 Parallel Processing Applications	16
2.11.1 Predictive Modeling and Simulation	16
2.11.2 Engineering Design and Automation	18
2.11.3 Energy Resources Exploration	20
2.11.4 Medical, Military, Basic Research	22
CHAPTER 3: Multiprocessing Scheduling Algorithms	
3.1 Introduction	24
3.2 Machine Model	25
3.3 Data Dependences	25
3.4 Scheduling Schemes	27
3.5 Design Rules for Run-Time Scheduling Schemes	28
3.5.1 Complete Program Graphs	29
3.5.2 Parallel Loops	30
3.6 Self-Scheduling Using Implicit Coalescing	32
3.6.1 The Guided Self-Scheduling (GSS()) Algorithm	33

3.6.2 Further Reduction of Synchronization operations	41
3.7 Self-Scheduling Using Two-Level Techniques	42
3.7.1 The Macro-Dataflow Graph and its Presentation	43
3.7.2 Two-Level Self-Scheduling	46
3.7.3 Low-Level Scheduling Considerations	46
CHAPTER 4: New Multiprocessing Scheduling Algorithms Using Guided Self-Scheduling Techniques	
4.1 Disadvantages of Low-level Self-Scheduling	48
4.2 Two-Level Guided Self-Scheduling	48
4.2.1 The Task Pool Data Structure	48
4.2.2 The Scheduling Algorithm	51
4.2.3 Low-Level Guided Self-Scheduling	52
4.2.4 High-Level Self-Scheduling	53
4.2.5 Low-Synchronization Low-Level Guided Self-Scheduling Algorithm (GSS(2))	59
4.3 Advantages of Low-Level Guided Self-Scheduling	59
4.4 High Gain Two-Level Guided Self-Scheduling Algorithm	60
CHAPTER 5: Multiprocessor System Simulation	
5.1 Simulation	63
5.1.1 Machine to be Simulated	63
5.1.2 Simulation Media	64
5.1.3 Simulation Media	65
5.1.4 Simulation Assumptions and Analogies	65

5.1.5 Simulation Correctness Checking	66
5.2 Simulated Algorithms Listing	67
CHAPTER 6: Guided Self-Scheduling Algorithms Performance Measurement	
6.1 Performance Measurement	68
6.1.1 Selecting a working Parallel Program	68
6.1.2 T.L.G.S.S. Versus T.L.S.S. Measurement	69
6.1.3 L.S.T.L.G.S.S. Versus T.L.S.S. Measurement	72
6.1.4 H.G.T.L.G.S.S. Versus T.L.S.S. Measurement	73
CHAPTER 7: Future Trends In Multiprocessor Scheduling Algorithms	
7.1 Advantages of Distributed Memory Multiprocessor systems	75
7.2 Compiling Global Name-Space Parallel Loops for Distributed Executions	75
7.3 Partitioning and mapping nested loops on multiprocessor systems	76
7.4 Compile-Time Techniques for Data Distribution in Distributed Memory machines	77
7.5 A loop Transformation Theory and an Algorithm to Maximize Parallelism	78
CHAPTER 8: Conclusion	
8.1 Conclusion	82
REFERENCES	
APPENDIX A	

APPENDIX B

APPENDIX C

APPENDIX D

APPENDIX E

NEW DYNAMIC TECHNIQUES FOR MAPPING GENERAL PARALLEL NESTED LOOPS ON MULTIPROCESSOR SYSTEMS

Abstract .- As device technology approaches physical limitations , parallel processor systems offers a promising and powerful alternative for high performance computing. In principle parallelism offers performance that can increase without bounds and depends only on the applications in hand. In reality parallelism have its limitations, such limitations is due to two main obstacles.

The first obstacle is that the technology is still unable to support the realization of parallel machine with large number of general purpose processors. The second and the greater obstacle appears which is our inability to efficiently utilize such massively parallel systems. Problems such as specifying parallelism, mapping or scheduling parallel programs on a given architecture , synchronizing the execution of a parallel programs, memory management in parallel processing environment ,and compiling for parallel machines remain areas for new research.

In numerical or scientific programs, nested loops are the most time consuming parts in serial processing but at the same time offer the most amount of parallelism from the sight of parallel distributed processing. Our concern in this thesis will be the schedules which instruct such nested parallel loops for the processors.

One of the most promising schedules which appears in this field is the "Two-Level Self-Scheduling" algorithm which divides the nested parallel loops scheduling process into two levels of scheduling. a) The high level scheduling ; in which the loop instances are placed in a task pool and then chosen for execution, and b) the low level scheduling; in which the processors dispatches a single iterations from any task which requires processors to complete its iterations.

The main disadvantage in this algorithm is that the loop indexes shared variables (which requires semaphore locking and unlocking) had to be updated a number of times equal to the loop bound due to the mandatory private access to such indexes.

In this thesis we present the "Two-Level Guided Self-Scheduling " algorithm and the "High Gain Two-level Guided Self-Scheduling" algorithm , two new approaches for scheduling

arbitrarily nested parallel programs on shared memory multiprocessor systems.

The first proposed algorithm is a modification for the low level part of the "Two-Level Self-Scheduling" algorithm using the "Guided Self-Scheduling" technique, such that each processor instead of dispatching a single iteration each time according to the "Self-Scheduling" technique, it dispatches a number of iterations calculated according to the "Guided Self-Scheduling" technique. This reduces the number of updates of the tasks indexes from a value equal to the loop bound to a value equal to the number of processors. This causes this algorithm to become of more load balancing and of very low synchronization overhead.

The second proposed algorithm contains a farther enhancements to the "Two-Level Guided Self-Scheduling" algorithm to decrease its scheduling time and increase its time gain greatly over the "Two-Level Self-Scheduling" algorithm in the case of systems with large number of processors but with some restrictions in the algorithm applicability.

After presenting the proposed algorithms a real simulation environment based on a Local Area Network system (LAN) to simulate a time shared BUS tightly coupled multiprocessor machine.

In this simulation the LAN workstations is the analogy of the processors of the multiprocessor system (by making such workstation execute the scheduling algorithm) and the LAN server hard disk is the analogy of the shared memory of the system (by creating the task pool on this disk as a number of data files)

Based on this simulation environment a performance comparison is made between the "Two-Level Guided Self-Scheduling" algorithm and the "Two-Level Self-Scheduling" algorithm to show to what extent such algorithm is useful.

From the results obtained in such performance comparison are the Time versus Number of processors, the Multiprocessor system time utilization percentage versus the Loops bound and the Multiprocessor system time utilization percentage versus the Number of processor for both the "Two-Level Self-Scheduling" algorithm (T.L.S.S.) and the "Two-Level Guided Self-Scheduling" algorithm (T.L.G.S.S.) and the time gain percentage of the T.L.G.S.S. algorithm over the T.L.S.S. algorithm.

From these results it was clear that the "Two-Level Guided Self-Scheduling" algorithm increases the Multiprocessor system time utilization and decreases the time required to execute the same tasks over the "Two-Level Self-Scheduling" algorithm.

Also based on this simulation a performance comparison is done between the "High Gain Two-Level Self-Scheduling" algorithm (H.G.T.L.S.S.) and the "Two-Level Self-Scheduling" algorithm (T.L.S.S.) including the time gain percentage of the H.G.T.L.G.S.S. algorithm over the T.L.S.S. algorithm.

From this comparison it was clear that the H.G.T.L.G.S.S. algorithm gives a linear increase in the time gain percentage over the T.L.S.S. algorithm in the case of large number of processors in the contrary to the T.L.G.S.S. which gives a very small time gain percentage over the T.L.S.S. algorithm in this region.

Finally some notes about the future trends in scheduling parallel programs on distributed memory multiprocessing systems is included.

CHAPTER 1

Introduction