UNIVERSITI PUTRA MALAYSIA

PARALLEL DIAGONALLY IMPLICIT RUNGE-KUTTA METHODS FOR SOLVING ORDINARY DIFFERENTIAL EQUATIONS

UMMUL KHAIR SALMA BINTI DIN

FS 2009 46

PARALLEL DIAGONALLY IMPLICIT RUNGE-KUTTA METHODS FOR SOLVING ORDINARY DIFFERENTIAL EQUATIONS

By

UMMUL KHAIR SALMA BINTI DIN

Thesis Submitted to the School of Graduate Studies, Universiti Putra Malaysia, in Fulfilment of the Requirements for the Degree of Doctor of Philosophy

December 2009

To my late father,

Haji Din bin Haji Ahmad

…who had always believed in the importance of knowledge.

Abstract of thesis presented to the Senate of Universiti Putra Malaysia in fulfilment of the requirement for the degree of Doctor of Philosophy

PARALLEL DIAGONALLY IMPLICIT RUNGE-KUTTA METHODS FOR SOLVING ORDINARY DIFFERENTIAL EQUATIONS

By

UMMUL KHAIR SALMA BINTI DIN

December 2009

Chairman: Fudziah binti Ismail, PhD

Faculty: Science

This thesis focuses on the derivations of diagonally implicit Runge-Kutta (DIRK) methods with the capability to be implemented by parallel executions. A few new methods are proposed by having sparsity patterns which enable the parallelization of methods. In the first part of the thesis, a fifth order DIRK suitable for two processors parallel executions and DIRK methods of fourth and fifth orders suitable for three processors are proposed. The executions of these methods are done by using fixed stepsizes on a set of nonstiff problems. The regions of stability are presented and numerical results of the methods are compared to the existing methods. Parallel computations show significant time reduction when solving large systems of nonstiff ordinary differential equations (ODEs).

The subsequent part of the thesis discusses on embedded DIRK methods suitable for two processors implementations. Two 4(3) and also two 5(4) embedded DIRK methods with adequate stability regions to solve stiff ODEs are proposed. Numerical experiments on stiff test problems are done based on variable stepsize strategy. An existing code for solving stiff ODEs suitable for embedded DIRK with equal diagonal elements is modified to accommodate the new methods with alternate diagonal elements. Comparisons on numerical results to existing methods show a competitive efficiency when solving small systems of stiff ODEs.

A parallel code is developed with the same capability of the modified sequential code to handle stiff ODEs, linear and nonlinear problems. All algorithms are written in C language and the parallel code is implemented on Sun Fire V1280 distributed memory system. Three large scales of stiff ODEs are used to measure the parallel performances of the new embedded methods. Results show that speedups increased as the dimensions of the problems gets larger which is a significant contribution in reducing the cost of computations.

Abstrak tesis yang dikemukakan kepada Senat Universiti Putra Malaysia
sebagai memenuhi keperluan untuk ijazah Doktor Falsafah

KAEDAH RUNGE-KUTTA PEPENJURU TERSIRAT SELARI BAGI MENYELESAI PERSAMAAN PEMBEZAAN BIASA

Oleh

UMMUL KHAIR SALMA BINTI DIN

Disember 2009

Pengerusi: Fudziah binti Ismail, PhD

Fakulti: Sains

Tesis ini tertumpu kepada penerbitan kaedah Runge-Kutta pepenjuru tersirat (RKPT) yang berupaya untuk dilaksanakan secara selari. Beberapa kaedah dicadangkan yang mempunyai bentuk yang bertaburan jarang bagi membolehkan kaedah itu diselarikan. Dalam bahagian pertama tesis ini, satu kaedah RKPT berperingkat lima sesuai dilaksanakan secara selari menggunakan dua pemproses dan kaedah berperingkat empat dan lima yang sesuai untuk tiga pemproses dicadangkan. Pelaksanaan kaedah ini dilakukan dengan menggunakan saiz langkah tetap ke atas satu set masalah tak kaku. Rantau kestabilan bagi kesemua kaedah dikemukakan dan keputusan berangka dibandingkan dengan beberapa kaedah sedia ada. Pengiraan secara selari menunjukkan pengurangan masa yang signifikan ketika menyelesaikan masalah persamaan pembezaan biasa (PPB) tak kaku yang bersaiz besar.

Bahagian selanjutnya dalam tesis ini membincangkan kaedah terbenam RKPT yang sesuai untuk pelaksanaan menggunakan dua pemproses yang bertujuan untuk menyelesaikan PPB kaku. Dua kaedah terbenam RKPT 4(3) dan juga dua 5(4) dengan rantau kestabilan yang mencukupi untuk menyelesaikan PPB kaku dicadangkan. Ujikaji berangka ke atas masalah kaku dijalankan berasaskan strategi saiz langkah boleh ubah. Satu kod sedia ada untuk menyelesaikan PPB kaku sesuai untuk kaedah terbenam RKPT dengan unsur pepenjuru yang sama diubahsuai agar bersesuaian dengan kaedah baru yang mempunyai unsur pepenjuru yang berselang-seli. Perbandingan ke atas keputusan berangka terhadap kaedah sedia ada menunjukkan kecekapan yang kompetitif semasa menyelesaikan sistem PPB bersaiz kecil.

Satu kod selari dibina dengan keupayaan yang sama dengan kod jujukan yang telah diubahsuai bagi menangani masalah PPB kaku, linear dan tak linear. Semua algoritma ditulis dalam bahasa C dan kod selari dilaksanakan di sistem memori bertaburan Sun Fire V1280. Tiga PPB kaku berskala besar digunakan untuk mengukur prestasi selari kaedah terbenam yang baru tersebut. Hasil menunjukkan kecepatan meningkat apabila dimensi masalah bertambah besar yang memberikan sumbangan yang signifikan dalam mengurangkan kos pengiraan.

ACKNOWLEDGEMENTS

In the name of Allah, the Most Gracious, the Most Merciful.
Thank you Ya Allah for the good health, patience and inspiration during the completion of this work.

First and foremost I would like to express my heartfelt gratitude to the chairman of the supervisory committee, Associate Professor Dr. Fudziah binti Ismail for her wise guidance, invaluable advice and patience. This work would not have been completed without her constant encouragement. I would also like to extend my appreciation to the supervisory committee members, Y.Bhg. Professor Dato' Dr. Mohamed bin Suleiman, Dr. Zanariah binti Abdul Majid and Y.Bhg. Professor Dr. Mohamed bin Othman for their kind support and suggestions. A special thanks to Professor J.C. Butcher from The University of Auckland and Dr. Rokiah @ Rozita Ahmad from UKM for sharing their invaluable knowledge in this field of study.

The financial support and study leave from the Ministry of Higher Education and Universiti Kebangsaan Malaysia is gratefully acknowledged. Many thanks also go to all my research colleagues and friends for their friendship and generous help throughout my few years in UPM.

I would like to thank both mother and mother-in-law, Hajjah Che Wa and Hajjah Zahrah, and also my brothers and sisters for their support and *do'a*. Finally, my deepest appreciation goes to my beloved husband Zafarin Abdul Ghaffar who has been very supportive and has inspires me to excel in life. I would also like to extend this gratitude to all my everloving strengths of my life, Iqbal, Nu'man, Tasnim and Dina who never fail to support me as their mother in any way they can. May Allah bless all of you.

I certify that a Thesis Examination Committee has met on 7 December 2009 to conduct the final examination of Ummul Khair Salma binti Din on her thesis entitled "Parallel Diagonally Implicit Runge-Kutta Methods for Solving Ordinary Differential Equations" in accordance with the Universities and University Colleges Act 1971 and the Constitution of the Universiti Putra Malaysia [P.U.(A) 106] 15 March 1998. The Committee recommends that the student be awarded the Doctor of Philosophy.

Members of the Thesis Examination Committee were as follows:

Malik bin Hj. Abu Hassan, PhD
Professor
Faculty of Science
Universiti Putra Malaysia
(Chairman)

Azmi bin Jaafar, PhD
Associate Professor
Faculty of Science Computer and Information Technology
Universiti Putra Malaysia
(Internal Examiner)

Zarina Bibi binti Ibrahim, PhD
Senior Lecturer
Faculty of Science
Universiti Putra Malaysia
(Internal Examiner)

Abdul Razak bin Yaakub, PhD
Professor
College of Art and Sciences
Universiti Utara Malaysia
(External Examiner)

BUJANG KIM HUAT, PhD
Professor and Deputy Dean
School of Graduate Studies
Universiti Putra Malaysia

Date:

This thesis submitted to the Senate of Universiti Putra Malaysia and has been accepted as fulfilment of the requirement for the degree of Doctor of Philosophy. The members of the Supervisory Committee were as follows:

Fudziah binti Ismail, PhD
Associate Professor
Faculty of Science
Universiti Putra Malaysia
(Chairperson)

Zanariah binti Abdul Majid, PhD
Lecturer
Faculty of Science
Universiti Putra Malaysia
(Member)

Mohamed bin Othman, PhD
Professor
Faculty of Science Computer and Information Technology
Universiti Putra Malaysia
(Member)

Dato' Mohamed bin Suleiman, PhD
Professor
Faculty of Science
Universiti Putra Malaysia
(Member)

HASANAH MOHD GHAZALI, PhD
Professor and Dean
School of Graduate Studies
Universiti Putra Malaysia

Date: 11 February 2010

TABLE OF CONTENTS

		Page
DEDICATION		ii
ABSTRACT		iii
ABSTRAK		v
ACKNOWLEDGEMENTS		vii
APPROVAL		viii
DECLARATION		x
LIST OF TABLES		xiv
LIST OF FIGURES		xvii
LIST OF ABREVIATIONS		xxi

CHAPTER

1	**INTRODUCTION**	
	1.1 General Introduction	1
	1.1.1 Objectives of the Thesis	3
	1.1.2 Outline of the Thesis	3
	1.1.3 Scope and Limitation	6
	1.2 The Initial Value Problem	7
	1.3 The Runge-Kutta Method	8
	1.3.1 The Diagonally Implicit Runge-Kutta Method	10
	1.3.2 Stability of Runge-Kutta Method	11
	1.3.3 A-stablity, A(α)-stability and L-stability	14
	1.4 Order Conditions	15
	1.5 Error Considerations	16
	1.6 Stiff Systems of ODEs	18
2	**DERIVATION OF FIFTH ORDER DIRK METHOD SUITABLE FOR PARALLEL USING TWO PROCESSORS**	
	2.1 Introduction	22
	2.2 Parallel Runge-Kutta Methods	22
	2.3 The Type II Methods	24
	2.4 Derivation of Fifth Order Parallel Runge-Kutta Method	28
	2.4.1 Stability of the Method	35
	2.4.2 Numerical Experiments	36
	2.4.3 Tested Problems	37
	2.4.4 Numerical Results	40
	2.5 Discussion	43
3	**PARALLELIZATION OF FIFTH ORDER DIRK METHOD ON TWO PROCESSORS**	
	3.1 Introduction	44
	3.2 An Overview on Parallel Computing	45
	3.2.1 High Performance Computing	48

	3.2.2 Message Passing Interface	48
	3.3 The Workload and the Needs for Parallel Computing in Runge-Kutta Methods	49
	3.4 The Performance of Parallel Processing	50
	3.5 The Architecture of the Parallel Algorithm for P2DIRK5	51
	3.6 Numerical Experiments	52
	3.6.1 Tested Problems	53
	3.6.2 Numerical Results	54
	3.7 Discussion	61
4	**PARALLEL FOURTH AND FIFTH ORDER DIRK METHODS USING THREE PROCESSORS**	
	4.1 Introduction	62
	4.2 The Type IV Methods	62
	4.3 The Fourth Order Methods	64
	4.4 The Fifth Order Methods	68
	4.5 Derivation of Fifth Order Methods	69
	4.6 The Parallel Architecture of the Type IV Method	76
	4.7 Numerical Experiments	77
	4.7.1 The Accuracy	77
	4.7.2 The Execution Time	77
	4.8 Discussion	89
	4.8.1 The Accuracy	89
	4.8.2 The Execution Time	89
5	**EMBEDDED DIRK 4(3) METHODS SUITABLE FOR PARALLEL USING TWO PROCESSORS**	
	5.1 Introduction	91
	5.2 Embedded Runge-Kutta Methods	91
	5.3 The 4(3) Pairs	93
	5.4 Solving Stiff Problems	98
	5.5 The Program - Billington's Code	99
	5.6 The Architecture for Parallel Implementation of P2DIRK4(3)a	104
	5.7 From Single to Dual	106
	5.8 Numerical Experiments	110
	5.8.1 Stiff Test Problems	110
	5.8.2 The Comparison Method	114
	5.8.3 Results	114
	5.9 Discussion	131
6	**EMBEDDED DIRK 5(4) METHODS SUITABLE FOR PARALLEL USING TWO PROCESSORS**	
	6.1 Introduction	132
	6.2 The Derivation of 5(4) Pairs	132
	6.3 Numerical Experiments	142
	6.4 Results	142
	6.5 Discussion	158

7	**THE IMPLEMENTATION OF PARALLEL EMBEDDED DIRK METHODS**	
	7.1 Introduction	159
	7.2. The Parallel Algorithm	160
	7.3 The Large Scale Systems of Stiff ODEs	166
	7.4 Segmentation of the Computing Time	169
	7.5 The Numerical Experiments	171
	7.5.1 P2DIRK4(3)a	171
	7.5.2 P2DIRK5(4)b	176
	7.6 Discussion	181
	7.6.1 P2DIRK4(3)a	181
	7.6.2 P2DIRK5(4)b	182
8	**CONCLUSION**	
	8.1 Summary	184
	8.2 Future Work	187
REFERENCES		189
APPENDICES		194
BIODATA OF STUDENT		212
LIST OF PUBLICATIONS		212

LIST OF TABLES

Table		Page
1.1	Number of order conditions	15
2.1	Runge-Kutta matrices and digraph	23
2.2	Equations of order conditions for Runge-Kutta methods of order 5	28
2.3	Numerical results for test problems 2.1-2.4 using IN4a, R5, CS5 and P2DIRK5	41
2.4	Numerical results for test problems 2.5-2.8 using IN4a, R5, CS5 and P2DIRK5	42
3.1	Stepsize and its total steps	52
3.2	The speedup and efficiency for solving Problem 3.1	55
3.3	The speedup and efficiency for solving Problem 3.2	56
3.4	The speedup and efficiency for solving Problem 3.3	57
4.1	Numerical results for Problems 2.1-2.4 using JN4, IN4b, P3DIRK4a and P3DIRK4b	79
4.2	Numerical results for Problems 2.5-2.4 using JN4, IN4b, P3DIRK4a and P3DIRK4b	80
4.3	Numerical results for Problems 2.1-2.4 using R5, P3DIRK5a and P3DIRK5b	81
4.4	Numerical results for Problems 2.5-2.8 using R5, P3DIRK5a and P3DIRK5b	82
4.5	The speedup and efficiency of P3DIRK4a in solving Problem 3.1	83
4.6	The speedup and efficiency of P3DIRK4a in solving Problem 3.2	84
4.7	The speedup and efficiency of P3DIRK5a in solving Problem 3.1	85

5.1	The 4(3) pairs	114
5.2	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.1	116
5.3	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.2	117
5.4	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.3	118
5.5	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.4	119
5.6	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.5	120
5.7	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.6	121
5.8	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.7	122
5.9	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.8	123
5.10	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.9	124
5.11	Performance comparison between HW4(3), P2DIRK4(3)a and P2DIRK4(3)b for solving Problem 5.10	125
6.1	The 5(4) pairs	142
6.2	Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.1	143
6.3	Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.2	144
6.4	Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.3	145
6.5	Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.4	146
6.6	Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.5	147

6.7 Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.6 148

6.8 Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.7 149

6.9 Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.8 150

6.10 Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.9 151

6.11 Performance comparison between K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b for solving Problem 5.10 152

7.1 Case 1: diff1>TOL/10 and diff2>TOL/10 163

7.2 Case 2: diff1<TOL/10 and diff2>TOL/10 163

7.3 Case 3: diff2<TOL/10 and diff1>TOL/10 164

7.4 The speedup and efficiency of P2DIRK4(3)a in solving Problem 7.1 172

7.5 The speedup and efficiency P2DIRK4(3)a in solving Problem 7.2 173

7.6 The speedup and efficiency P2DIRK4(3)a in solving Problem 7.3 174

7.7 The speedup and efficiency of P2DIRK5(4)b in solving Problem 7.1 177

7.8 The speedup and efficiency P2DIRK5(4)b in solving Problem 7.2 178

7.9 The speedup and efficiency P2DIRK5(4)b in solving Problem 7.3 179

LIST OF FIGURES

Figure		Page
1.1	Butcher's Array	9
1.2	A Simplified Butcher's Array	9
1.3	The Chronology of Work on DIRK	11
2.1	Fourth Order - Type II Method	25
2.2	Method by Jackson & Norsett (JN4)	26
2.3	Method by Iserles and Norsett (IN4a)	26
2.4	Method by Iserles and Norsett (IN4b)	27
2.5	Sparsity Structure and Digraph for a Fifth Order Runge-Kutta Method with Six Stages	27
2.6	The stability region of P2DIRK5	36
3.1	A Dependence Graph Exhibiting Data Parallelism (Quinn, 2004)	47
3.2	A Dependence Graph Exhibiting Functional Parallelism (Quinn, 2004)	47
3.3	The Parallel Processes of P2DIRK5	51
3.4	Results for Problem 3.1	58
3.5	Results for Problem 3.2	59
3.6	Results for Problem 3.3	60
4.1	Sparsity Structure and Digraph for Runge-Kutta Methods with Six Stages for Three Processors	63
4.2	Butcher's Array for *2-parallel 3-processors* Runge-Kutta Method	63
4.3	The P3DIRK4a	65

4.4	The Stability Region for P3DIRK4a	66
4.5	The P3DIRK4b	67
4.6	The Stability Region for P3DIRK4b	67
4.7	The Modified Sparsity Structure and Digraph for Runge-Kutta Methods for Three Processors	68
4.8	The P3DIRK5a	72
4.9	The Stability Region of P3DIRK5a	73
4.10	The P3DIRK5b	74
4.11	The Stability Region of P3DIRK5b	75
4.12	The Parallel Process for Type IV Methods	76
4.13	Results of P3DIRK4a in Solving Problem 3.1	86
4.14	Results of P3DIRK4a in Solving Problem 3.2	87
4.15	Results of P3DIRK5a in Solving Problem 3.1	88
5.1	The P2DIRK4(3)a	95
5.2	The P2DIRK4(3)b	96
5.3	The Stability Region of P2DIRK4(3)a	97
5.4	The Stability Region of P2DIRK4(3)b	98
5.5	Flowchart for Phases in BiCODE	103
5.6	The Basis for Parallelization of BiCODE	105
5.7	The Source Code in C Language for Phase II of BiCODE	107
5.8	The Pseudocode for the "Detour" Process	109
5.9	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.1	126
5.10	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.2	126
5.11	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.3	127

5.12	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.4	127
5.13	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.5	128
5.14	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.6	128
5.15	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.7	129
5.16	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.8	129
5.17	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.9	130
5.18	Log MAXE versus TIME for HW4(3), P2DIRK4(3)a and P2DIRK4(3)b in Solving Problem 5.10	130
6.1	The Modified Sparsity Structure for 5(4) Pair	133
6.2	The Stability Region of P2DIRK5(4)	136
6.3	The Stability Region of P2DIRK5(4)a	140
6.4	The Stability Region of P2DIRK5(4)b	141
6.5	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.1	153
6.6	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.2	153
6.7	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.3	154
6.8	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.4	154
6.9	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.5	155
6.10	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.6	155
6.11	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.7	156

6.12	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.8	156
6.13	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.9	157
6.14	Log MAXE versus TIME for K5(4), F5(4), P2DIRK5(4)a and P2DIRK5(4)b in Solving Problem 5.10	157
7.1	The Convergence Test	162
7.2	The Program Fragment of the Convergence Test for Processor 1	164
7.3	The Program Fragment of the Convergence Test for Processor 2	165
7.4	Segmentation of Computing Time in Solving Problem 7.1	170
7.5	Segmentation of Computing Time in Solving Problem 7.2	170
7.6	Segmentation of Computing Time in Solving Problem 7.3	171
7.7	The Speedup of P2DIRK4(3)a when Solving Problem 7.1	175
7.8	The Speedup of P2DIRK4(3)a when Solving Problem 7.2	175
7.9	The Speedup of P2DIRK4(3)a when Solving Problem 7.3	176
7.10	The Speedup of P2DIRK5(4)b when Solving Problem 7.1	180
7.11	The Speedup of P2DIRK5(4)b when Solving Problem 7.2	180
7.12	The Speedup of P2DIRK5(4)b when Solving Problem 7.3	181
8.1	Proposed Hierarchical Tree for Future Work	187

LIST OF ABBREVIATIONS

ODEs	Ordinary differential equations
IVPs	Initial value problems
DIRK	Diagonally implicit Runge-Kutta
MPI	Message Passing Interface
SDIRK	Singly diagonally implicit Runge-Kutta
LTE	Local truncation error
FSAL	First Same As Last
JN4	Method by Jackson and Nørsett
IN4a	Method by Iserles and Nørsett with L-stability
IN4b	Method by Iserles and Nørsett with A-stability
P2DIRK5	Fifth order DIRK suitable for two processors
R5	Fifth order DIRK by Al-Rabeh
CS5	Fifth order DIRK by Cooper and Sayfy
HPC	High Performance Computing
SISD	Single Instruction Single Data
MISD	Multiple Instruction Single Data
SIMD	Single Instruction Multiple Data
MIMD	Multiple Instruction Multiple Data
P3DIRK4a	Fourth order DIRK suitable for three processors with A-stability
P3DIRK4b	Fourth order DIRK suitable for three processors with stability region [-51.44,0]

P3DIRK5a	Fifth order DIRK suitable for three processors with stability region [-16.18,0]
P3DIRK5b	Fifth order DIRK suitable for three processors with stability region [-9.86,0]
P2DIRK4(3)a	Embedded 4(3) DIRK suitable for two processors with four stages error estimator
P2DIRK4(3)b	Embedded 4(3) DIRK suitable for two processors with five stages error estimator
BiCODE	Billington's code for DIRK with equal diagonal elements
BiCODE-2	Modified Billington's code for DIRK with alternate diagonal elements
HW4(3)	Embedded 4(3) DIRK by Hairer and Wanner
P2DIRK5(4)	Embedded 5(4) DIRK suitable for two processors with stability region [-13.33,0]
P2DIRK5(4)a	Embedded 5(4) DIRK suitable for two processors with stability region [-159.2,0]
P2DIRK5(4)b	Embedded 5(4) DIRK suitable for two processors with $A^*(\alpha)$-stability
K5(4)	Embedded 5(4) DIRK by Kværnø
F5(4)	Embedded 5(4) DIRK by Ismail

Equation Chapter 1 Section 1

CHAPTER 1

INTRODUCTION

1.1 General Introduction

Numerical analysis is the area of mathematics and computer science that has a great importance in solving many physical problems represented by mathematical models. It creates, analyzes, and implements algorithms to give the best numerical approximation to the problems of continuous mathematics which originate generally from real-world applications of algebra, geometry, and calculus. These problems occur throughout the natural sciences, social sciences, medicine, engineering, and business which then are classified as linear or nonlinear and stiff or non-stiff problems. When simulating the behaviour of those systems, mathematical models often include one or more ordinary differential equations (ODEs). Almost always numerical techniques must be used to obtain approximate solutions to the ODEs since analytical techniques available are not powerful enough to solve any ODEs except the simplest (Gupta et al., 1985).

The early work on numerical ordinary differential equations has been built since the 19^{th} century where the 1883 paper of Bashforth and Adams and the 1895 paper of Runge have presented the initial ideas in developing modern softwares of numerical methods (Butcher, 2000). Since then, further ideas were suggested with few being the main choice of techniques when solving ODEs. There are two main approaches for numerical methods which are linear multistep methods and the one-step methods. As how they are named, the approximation of the solution value for a given x is based on a number of previously computed points for linear multistep methods while the approach for the one-step methods is restricted to only on the most recent point already computed in a previous step. Both classes of methods have their own strengths and it is up to users to consider which is more convenience and suitable to use. Adams methods are widely known for the linear multistep users while Runge-Kutta methods have been used extensively in a one-step algorithm. Even though the classical methods for Adams and Runge-Kutta methods have proved to be useful for many problems, research in these methods are still actively conducted where many arising issues are tackled and more new methods are proposed.

The growth in power and availability of digital computers has led to an increasing use of realistic mathematical models. Numerical analysis of increasing sophistication has been needed to solve these more complex models of many physical problems. The wide variety of new computer architectures has created more option to improve the implementation of numerical algorithms. One of the intensive researches that have been conducted is the parallel implementation of numerical methods. According to Jackson (1991), the desire for parallel solvers, in particular for solving initial value problems (IVPs) for ODEs, arises from the need to solve many important problems more rapidly than is currently possible. The reduction in cost particularly time, is undeniably give great motivation in developing this idea. A few ideas of parallelism have been suggested with all having the same purpose as to have methods with better performance.