

HASIL EVALUASI KINERJA METODE ITERATIF PARALEL IMPLISIT MULTISTEP RUNGE-KUTTA PADA SISTEM PARALEL MPI-LINUX

Alhadi Bustamam¹, Heru Suhartanto² dan T. Basaruddin² LL

¹ Jurusan Matematika FMIPA-UI, Depok, Jawa Barat, 16424. Email: alhadi@makara.cso.ui.ac.id

² Fakultas Ilmu Komputer UI, Depok, Jawa Barat, 16424. Email: { [heru_chan](mailto:heru_chan}@cs.ui.ac.id) }@cs.ui.ac.id

ABSTRAK

Pengembangan metode iPIMRK untuk menyelesaikan persoalan yang stiff dan implementasi perangkat lunaknya telah dilakukan oleh Suhartanto dan Burrage. Implementasi ini menggunakan FORTRAN90 dan dijalankan pada mesin *shared memory*-MPMD SGI-ORIGIN2000. Paralelisasi dilakukan secara iteratif untuk tiga proses utama yang disebut dengan *parallel_stages*, *parallel_factors* dan *parallel_solves* yang dapat dijalankan pada sejumlah *s*-stages prosesor. Proses integrasi menggunakan ukuran langkah berubah dan pada setiap langkah iterasi digunakan dua teknik perhitungan untuk koefisien parameter metode yaitu: koefisien tetap (*fixed coefficients*: FC-iPIMRK) atau koefisien berubah (*variable coefficients*: VC-iPIMRK). Bustamam dan Suhartanto et. al. berhasil mengimplementasikan kembali metode iPIMRK tersebut secara SPMD pada mesin paralel virtual menggunakan teknologi pertukaran pesan MPI dan GNU-FORTRAN77. Pada makalah ini akan ditampilkan hasil evaluasi kinerja metode iPIMRK untuk menyelesaikan sistem persamaan diferensial biasa berupa persoalan-persoalan yang *stiff* yang diimplementasikan oleh Bustamam dan Suhartanto et. al. pada sistem paralel MPI-LINUX di Laboratorium HPCCSUI Fakultas Ilmu Komputer UI Depok. Dari hasil eksperimen terlihat bahwa metode VC-iPIMRK dibandingkan dengan metode FC-iPIMRK lebih baik dari sisi *speed-up*, efisiensi dan akurasi tetapi lebih buruk dari sisi waktu komputasi. Kontribusi positif terhadap kinerja proses paralel hanya diberikan oleh proses *parallel_stages* dan *parallel_factors* sedangkan proses *parallel_solves* ternyata memberikan kontribusi negatif. Untuk meningkatkan kinerja maka sebaiknya proses *parallel_solves* tidak diaktifkan.

Kata kunci: *solver*, persoalan nilai awal yang *stiff*, metode iPIMRK, SPMD, sistem paralel MPI-LINUX.

Makalah diterima [27 Agustus 2002]. Revisi akhir [15 Oktober 2002].

1. IMPLEMENTASI SPMD METODE IPIMRK PADA SISTEM PARALEL MPI-LINUX

Pengembangan teknik paralelisasi metode iteratif paralel implisit *multistep* Runge-Kutta (iPIMRK), implementasi secara *single program multiple data* (SPMD) pada sistem paralel MPI-LINUX dan pembuatan *solver* SVMRK untuk menyelesaikan sistem persamaan diferensial biasa yang kaku (*stiff ODEs*) telah dilakukan oleh Bustamam dan Suhartanto et. al. [1]-[4]. Tujuan utamanya adalah untuk mendapatkan *solver* dengan kinerja yang lebih baik (lebih cepat dan lebih akurat dari *solver* sekuensial tertentu) dan murah dari sisi biaya komputasi dan biaya produksi (dibandingkan mesin paralel multiprosesor *shared memory*). Metode *r-steps* dan *s-stages* iPIMRK tipe RADAU yang memiliki orde $(2s+r-2)$ dirancang untuk peningkatan kecepatan dan akurasi dengan biaya komputasi yang lebih murah. Sistem paralel MPI-LINUX menggunakan perangkat lunak LAM-MPI versi 6.5.4 [9]-[11], sistem operasi Linux Mandrake 8.0 dan bahasa pemrograman GNU-FORTRAN77 [7] digunakan untuk mendapatkan biaya produksi dan *setup* sistem yang lebih murah, lihat Bustamam dan Suhartanto et. al. [1]-[4].

Pada metode iPIMRK terdapat tiga aspek yang menonjol yang menjadi dasar paralelisasi pada metode ini yaitu proses evaluasi fungsi (*PARALLEL_STAGES*), faktorisasi matrik iterasi (*PARALLEL_FACTORS*), dan perhitungan solusi sistem persamaan linier pada iterasi Newton (*PARALLEL_SOLVES*). Ketiga proses tersebut dikerjakan secara paralel pada masing-masing *stages* dari setiap *step* integrasi secara paralel di *s-stages* prosesor pada sistem paralel MPI-LINUX dengan model program SPMD. Satu prosesor (prosesor-0) berfungsi sebagai prosesor utama yang bertugas membagi pekerjaan dan data serta mengumpulkan hasil komputasi lokal pada semua prosesor (termasuk prosesor-0 itu sendiri), dan *s-1* prosesor lainnya bertugas melakukan komputasi paralel secara serentak berdasarkan *taskinfo* dan *compinfo* yang dia terima dari prosesor-0.

Untuk implementasi SPMD ini dibutuhkan satu prosedur global yang berfungsi sebagai pengontrol utama untuk semua *task* dari MPI dan tiga buah prosedur

khusus sebagai pengontrol sekunder untuk ketiga proses paralel tersebut, lihat Bustamam dan Suhartanto et. al. [2]. Berdasarkan hasil-hasil eksperimen pada makalah ini akan dilihat pengaruh paralelisasi oleh ketiga aspek ini pada peningkatan *speed-up* komputasi kemudian akan dilihat juga aspek mana yang paling dominan dalam memberikan kontribusi pada peningkatan *speed-up* dari metode iPIMRK tersebut.

Untuk perhitungan tebakan solusi awal dari metode iPIMRK ini digunakan beberapa macam metode prediktor yaitu: prediktor P1 berdasarkan turunan pada r -titik langkah sebelumnya, prediktor P2 berdasarkan turunan pada r -titik langkah sebelumnya dan s -titik tahapan pada langkah sebelumnya, prediktor P3 berdasarkan turunan pada s -titik tahapan pada langkah sebelumnya dan prediktor P4 berdasarkan turunan pada s -titik tahapan pada langkah sebelumnya dan t_{n-1} . Selain itu dalam implementasi metode iPIMRK terdapat dua teknik yang berbeda dalam pemilihan koefisien-koefisien parameter metode implisit MRK yang digunakan yaitu: metode iPIMRK dengan koefisien-koefisien parameter konstan pada setiap langkah (*fixed coefficients-FC-iPIMRK*) dan koefisien-koefisien berubah-ubah pada setiap langkah tergantung pada rasio besarnya suatu langkah dengan besar langkah sebelumnya (*variable coefficients-VC-iPIMRK*), lihat Bustamam [1], Burrage dan Suhartanto [5],[6].

Pada eksperimen ini akan dilihat pengaruh perbedaan pemilihan koefisien parameter ini pada lamanya waktu integrasi, peningkatan *speed-up* yang diperoleh dan keakuratan solusinya. Selanjutnya pada eksperimen ini juga akan dilihat pengaruh dari perbedaan jumlah prosesor, jumlah *steps* dan jumlah *stages* yang digunakan pada metode iPIMRK (khususnya antara metode iPIMRK₃₃ yang menggunakan tiga prosesor dengan metode iPIMRK₂₂ yang menggunakan dua prosesor), baik dari segi waktu komputasi, *speed-up*, efisiensi dan akurasi yang diperoleh. Kriteria evaluasi kinerja ini menggunakan definisi kinerja dari Freeman [8].

2. TES PROBLEM

Tes problem yang digunakan pada penelitian ini adalah beberapa tes problem standar untuk persoalan sistem ODE yang kaku yang diambil dari tes set standar CWI (<http://www.cwi.nl/static/projects/IVPtestset/>), tes problem *cusp* dari Jeff Cash (<http://www.ma.ic.ac.uk/~jcash/IVPsoftware/>), dan tes problem *dense* dan *brusselator* dari Suhartanto [12],[13] (<http://www.math.uq.edu.au/~hs/>). Tes problem tersebut pada dasarnya dibedakan menjadi dua karakteristik sistem matrik yaitu berbentuk *full matrix* atau *banded matrix*. Sedangkan tes problem *dense* dan *brusselator* merupakan

tes problem yang dimensi persoalannya dapat diubah-ubah, di mana tes problem *dense* berbentuk *full* matrik sedangkan tes problem *brusselator* berbentuk *banded* matrik. Pada eksperimen ini akan dilihat juga pengaruh dari perbedaan bentuk sistem matrik dan juga pengaruh perbedaan dimensi N pada kinerja yang diperoleh.

3. HASIL-HASIL EKSPERIMEN DAN EVALUASI KINERJA

Berikut ini ditampilkan beberapa hasil eksperimen dan evaluasi kinerja metode iPIMRK untuk beberapa tes problem di atas.

3.1. Tes Problem Dengan Dimensi N Kecil (*Full/Banded Matrix*) dan Tes Problem *Brusselator*.

Tes ini menggunakan tes standar CWI dengan program *driver* DR_SVMRK, tes problem *cusp* dengan program *driver* DR_CUSP dan tes problem *brusselator* dengan program DR_SBRUSS. Hasil eksperimen tidak menunjukkan *speed-up* yang memadai (*speed-up* < 1)

Tabel A.1: Hasil eksperimen DR_CUSP untuk tes problem CUSP (*banded matrix*, $N=96$, $\text{Atol}=1.d-07$)

Metode	Tpred	NFCN	NDEC	NSOL	Seq Time	Par Time	Speed-Up
VC-iPIMRK ₃₃	3	960	917	120	0,440	1,390	0,32
FC-iPIMRK ₃₃	3	1012	966	128	0,470	1,590	0,30
VC-iPIMRK ₃₃	1	1819	1750	224	0,850	2,530	0,34
VC-iPIMRK ₃₃	2	1113	1043	142	0,510	1,770	0,29
VC-iPIMRK ₃₃	4	1052	1001	132	0,480	1,620	0,30
VC-iPIMRK ₂₂	3	1438	1221	304	0,410	1,160	0,35
FC-iPIMRK ₂₂	3	1423	1213	303	0,490	1,230	0,40
VC-iPIMRK ₂₂	1	4342	3509	877	1,300	3,000	0,43
FC-iPIMRK ₂₂	1	4366	3533	881	1,340	3,210	0,42
VC-iPIMRK ₂₂	2	1757	1485	370	0,510	1,280	0,40
FC-iPIMRK ₂₂	2	1666	1405	350	0,560	1,250	0,45
FC-iPIMRK ₂₂	4	1822	1565	388	0,610	1,460	0,42
FC-iPIMRK ₂₂	4	1213	1041	255	0,310	0,830	0,37

Keterangan: NFC=Jumlah evaluasi fungsi, DEC=Jumlah dekomposisi sistem matrik, NSOL=Jumlah pemecahan solusi. VC= Variable Coefficients, FC=Fixed Coefficients.

Tabel A.2: Hasil eksperimen DR SVMRK untuk beberapa tes problem (full matrix, ukuran N kecil, atol bervariasi untuk setiap tes problem)

Metode	Tes Prob	N	Tpred	Atol	Seq-time	Par-time	Speed-up
FC-IPIMRK ₃₃	Orego	3	1	1,d-12	0,010	0,550	0,02
VC-IPIMRK ₃₃	Orego	3	1	1,d-12	0,020	0,720	0,03
FC-IPIMRK ₃₃	Rob	3	2	1,d-12	0,010	0,100	0,10
VC-IPIMRK ₃₃	Rob	3	2	1,d-12	0,020	0,120	0,17
FC-IPIMRK ₃₃	B51000	6	3	1,d-04	0,040	1,870	0,02
VC-IPIMRK ₃₃	B51000	6	3	1,d-04	0,080	1,320	0,06
FC-IPIMRK ₃₃	Hires	8	4	1,d-06	0,030	0,870	0,03
VC-IPIMRK ₃₃	Hires	8	4	1,d-06	0,040	0,700	0,06

3.2. Tes Problem Dense Dengan Variasi N Kecil Ke Besar (Full Matrix)

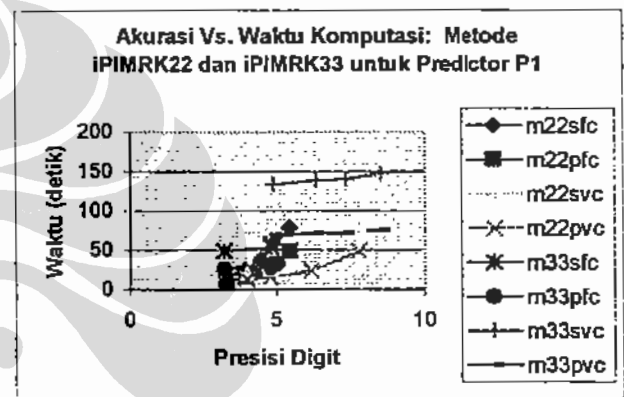
Tabel B.1: Hasil speed-up untuk tes problem dense dengan metode FC-IPIMRK33, prediktor P3, variasi N, dan variasi toleransi absolut (ATOL)

Metode Fixed Coefficients(FC) iPIMRK33				
ATOL				
N	1,d-7	1,d-8	1,d-9	1,d-10
20	0,42	0,54	0,35	0,80
30	1,22	0,79	0,93	0,85
40	1,19	1,00	0,81	1,06
50	1,23	1,11	1,18	0,89
100	1,38	1,37	1,47	1,42
150	1,55	1,53	1,54	1,53
200	1,60	1,64	1,64	1,68
250	1,88	1,89	1,87	1,88

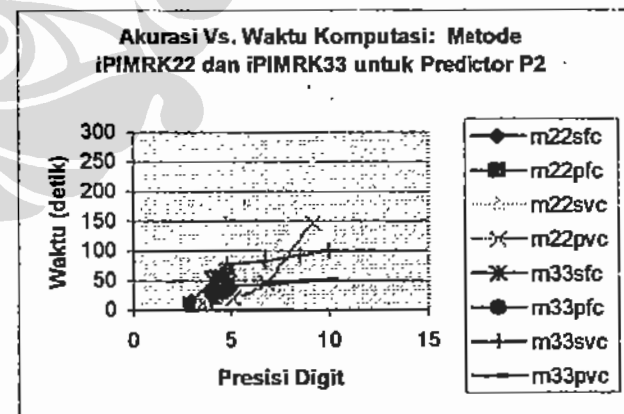
Perbandingan antara akurasi dan waktu komputasi antara metode iPIMRK22 dan metode iPIMRK33 untuk prediktor P1,P2,P3 dan P4 dapat dilihat pada gambar B.1, B.2, B.3 dan B.4.

Tabel B.2: Hasil speed-up untuk tes problem dense dengan metode VC-IPIMRK33, prediktor P3, variasi N, dan variasi toleransi absolut (ATOL)

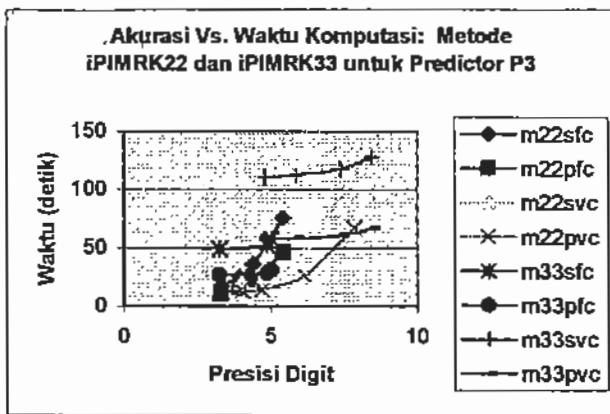
Metode Variable Coefficients(VC) iPIMRK33				
ATOL				
N	1,d-7	1,d-8	1,d-9	1,d-10
20	0,24	1,00	0,64	0,82
30	0,75	1,00	1,14	0,72
40	0,71	0,75	0,89	1,39
50	0,95	1,41	1,26	1,10
100	1,42	1,45	1,49	1,46
150	1,54	1,59	1,50	1,54
200	1,70	1,71	1,74	1,73
250	1,92	1,96	1,95	1,95



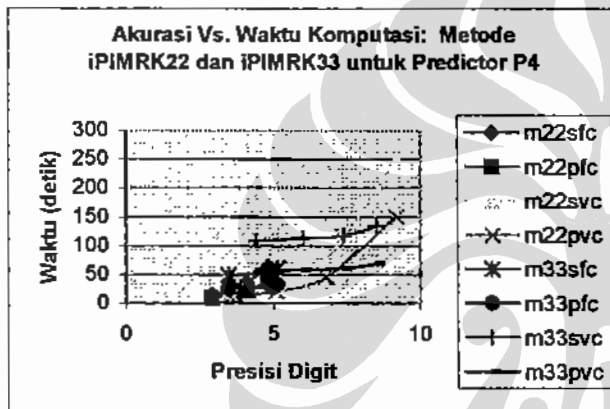
Gambar B.1: Perbandingan Akurasi dan Waktu Eksekusi metode iPIMRK₂₂ dan iPIMRK₃₃ dengan prediktor P1.



Gambar B.2: Perbandingan Akurasi dan Waktu Eksekusi metode iPIMRK₂₂ dan iPIMRK₃₃ menggunakan prediktor P2



Gambar B.3: Perbandingan Akurasi dan Waktu Eksekusi metode iPIMRK₂₂ dan iPIMRK₃₃ menggunakan prediktor P3



Gambar B.4: Perbandingan Akurasi dan Waktu Eksekusi metode iPIMRK₂₂ dan iPIMRK₃₃ menggunakan prediktor P4

{Keterangan notasi pada gambar adalah sebagai berikut: *sfc*=*sequential fixed coefficients*, *svc*=*sequential variable coefficients*, *pfc*=*parallel fixed coefficients*, dan *pvc*=*parallel variable coefficients*}

Berdasarkan Gambar B.1, B.2, B.3 dan B.4 di atas, hasil-hasil eksperimen dan evaluasi beberapa aspek kinerja antara metode iPIMRK₂₂ (menggunakan dua prosesor) dan iPIMRK₃₃ (menggunakan tiga prosesor) serta variasi metode dengan koefisien parameter konstan dan berubah dapat dijelaskan sebagai berikut.

3.2.1. Perbandingan Akurasi dengan Waktu Eksekusi pada N=250 dan Berbagai Prediktor P1,P2,P3,P4

Berdasarkan tabel B.1.a, B.1.b, B.1.c, dan B.1.d, metode iPIMRK₃₃ memberikan akurasi yang lebih baik dari metode iPIMRK₂₂ tetapi akurasi yang lebih tinggi ini membutuhkan waktu eksekusi lebih lama. Selain itu untuk kedua metode terlihat bahwa akurasi yang

diperoleh dari metode dengan koefisien parameter berubah selalu lebih baik dari metode dengan koefisien konstan, tetapi waktu eksekusi yang dibutuhkan metode koefisien berubah selalu lebih lama dari metode dengan koefisien konstan.

Tabel B.1.a. Hasil *Speed-up* dan Efisiensi metode koefisien konstan iPIMRK₂₂

Tpred	Atol	Seq-time	Par-time	Speed-up	Efficiency (%)	Pd
1	0,0001	17,310	11,010	1,57	78,61	3,29
1	1,00E-06	25,160	15,999	1,57	78,63	3,95
1	1,00E-08	38,130	23,400	1,63	81,47	4,45
1	1,00E-10	78,720	48,120	1,64	81,80	5,46
2	0,0001	14,310	9,360	1,53	76,44	2,92
2	1,00E-06	25,230	15,75	1,60	80,10	3,67
2	1,00E-08	36,130	22,090	1,64	81,78	4,08
2	1,00E-10	67,780	41,630	1,63	81,43	4,79
3	0,0001	17,260	11,350	1,52	76,04	3,28
3	1,00E-06	25,230	15,640	1,61	80,66	3,92
3	1,00E-08	36,110	22,520	1,60	80,17	4,38
3	1,00E-10	75,580	46,220	1,64	81,76	5,41
4	0,0001	14,300	9,130	1,57	78,31	2,92
4	1,00E-06	25,180	16,010	1,57	78,64	3,67
4	1,00E-08	36,0704	22,820	1,58	79,03	4,09
4	1,00E-10	63,790	39,150	1,63	81,47	4,82
Rata-rata				1,60	79,77	

Tabel B.1.b : Hasil *Speed-up* dan Efisiensi metode koefisien berubah iPIMRK₂₂

Tpred	Atol	Seq-time	Par-time	Speed-up	Efficiency (%)	Pd
1	0,0001	21,290	13,470	1,58	79,03	4,01
1	1,00E-06	26,190	16,270	1,61	80,48	4,79
1	1,00E-08	40,130	24,920	1,61	80,52	6,19
1	1,00E-10	80,650	48,750	1,65	82,72	7,80
2	0,0001	18,290	11,700	1,56	78,16	3,51
2	1,00E-06	34,110	20,930	1,63	81,49	5,09
2	1,00E-08	77,659	47,080	1,65	82,48	6,78
2	1,00E-10	246,960	148,380	1,66	83,22	9,18
3	0,0001	21,230	13,440	1,58	78,98	4,00
3	1,00E-06	23,220	14,510	1,60	80,01	4,68
3	1,00E-08	43,910	26,620	1,65	82,48	6,15
3	1,00E-10	113,320	67,850	1,67	83,51	7,85
4	0,0001	18,250	11,620	1,57	78,53	3,51
4	1,00E-06	34,140	21,200	1,61	80,52	5,09
4	1,00E-08	76,610	46,320	1,65	82,70	6,78
4	1,00E-10	246,950	149,270	1,65	82,72	9,18
Rata-rata				1,62	81,10	

Khusus untuk metode dengan koefisien berubah terlihat bahwa pada metode iPIMRK₃₃ dengan prediktor P1 dan P3 untuk memperoleh akurasi yang tinggi dibutuhkan waktu eksekusi yang lebih lama dari iPIMRK₂₂, tetapi untuk prediktor P2 dan P4 berlaku sebaliknya. Hal ini dipengaruhi oleh perbedaan jumlah

evaluasi fungsi dan jumlah faktorisasi yang sangat besar pada prediktor P1 dan P3 dibandingkan dengan prediktor P2 dan P4 pada kedua metode, lihat Gambar B.1.a.

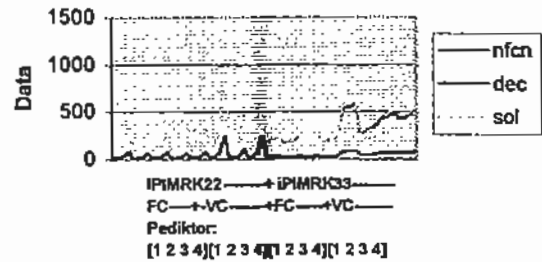
Tabel B.1.c: Hasil Speed-up dan Efisiensi total metode koefisien konstan iPIMRK₃₃

Tpred	Atol	Seq-time	Par-time	Speed-up	Efficiency (%)	pd
1	0,0001	48,570	26,460	1,84	61,19	3,23
1	1,00E-06	53,970	28,820	1,87	62,42	4,82
1	1,00E-08	59,060	31,710	1,86	62,08	4,82
1	1,00E-10	62,720	33,020	1,90	63,32	5,03
2	0,0001	51,520	27,850	1,85	61,66	4,07
2	1,00E-06	55,270	29,900	1,85	61,62	4,27
2	1,00E-08	55,25	29,800	1,85	61,80	4,52
2	1,00E-10	68,320	36,460	1,87	62,46	4,89
3	0,0001	48,740	26,580	1,83	61,12	3,25
3	1,00E-06	52,150	28,260	1,85	61,51	4,87
3	1,00E-08	55,770	29,670	1,88	62,66	4,89
3	1,00E-10	57,560	30,640	1,88	62,62	5,01
4	0,0001	48,750	26,180	1,86	62,07	3,46
4	1,00E-06	54,070	28,980	1,87	62,19	4,93
4	1,00E-08	52,240	27,940	1,87	62,32	5,12
4	1,00E-10	61,020	32,590	1,87	62,41	4,98
Rata-rata				1,86	62,09	

Tabel B.1.d: Hasil Speed-up dan Efisiensi metode koefisien berubah iPIMRK₃₃

Tpred	Atol	Seq-time	Par-time	Speed-up	Efficiency (%)	pd
1	0,0001	133,110	68,410	1,95	64,86	4,01
1	1,00E-06	138,350	71,300	1,94	64,68	4,79
1	1,00E-08	140,510	71,910	1,95	65,13	6,19
1	1,00E-10	147,660	75,740	1,95	64,99	7,80
2	0,0001	77,410	40,930	1,89	63,04	3,51
2	1,00E-06	83,020	43,260	1,92	63,97	5,09
2	1,00E-08	92,410	48,250	1,92	63,84	6,78
2	1,00E-10	99,920	51,699	1,93	64,42	9,18
3	0,0001	110,740	57,950	1,91	63,70	4,00
3	1,00E-06	112,530	58,480	1,92	64,14	4,68
3	1,00E-08	117,840	60,790	1,94	64,62	6,15
3	1,00E-10	128,310	66,930	1,92	63,90	7,85
4	0,0001	108,970	56,340	1,93	64,47	3,51
4	1,00E-06	112,420	57,960	1,94	64,65	5,09
4	1,00E-08	117,750	60,780	1,94	64,58	6,78
4	1,00E-10	133,470	68,610	1,95	64,84	9,18
Rata-Rata				1,93	64,36	

Trend Jumlah Evaluasi Fungsi, Faktorisasi dan Solusi Newton Terhadap Variasi Metode, Predictors dan Coefficiens

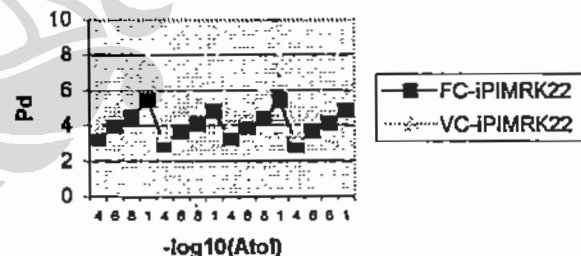


Gambar B.1.a: Trend jumlah evaluasi fungsi, jumlah faktorisasi dan jumlah solusi Newton pada kedua metode iPIMRK₂₂ dan iPIMRK₃₃

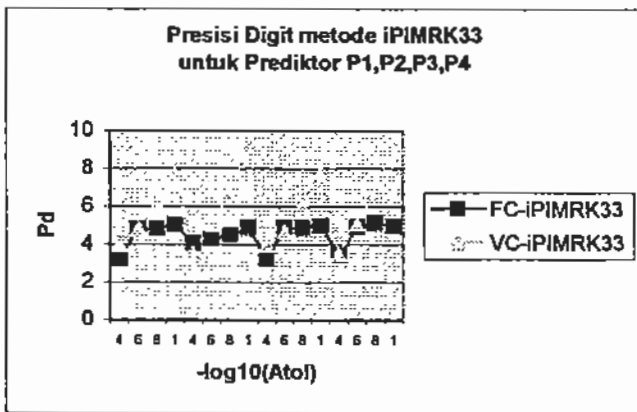
3.2.2. Perbandingan Akurasi pada Berbagai Prediktor P1,P2,P3,P4

Berdasarkan hasil eksperimen pada Gambar B.2.a dan B.2.b terlihat bahwa pada metode iPIMRK₂₂ dan iPIMRK₃₃ dengan koefisien parameter berubah prediktor P1 dan P3 menghasilkan tingkat akurasi yang lebih rendah dari prediktor P2 dan P4. Sebaliknya pada metode iPIMRK₂₂ dan iPIMRK₃₃ dengan koefisien parameter konstan prediktor P1 dan P3 menghasilkan tingkat akurasi yang lebih baik dari prediktor P2 dan P4

Presisi Digit metode iPIMRK₂₂ untuk Prediktor P1,P2,P3,P4



Gambar B.2.a: Perbandingan presisi digit metode koefisien konstan iPIMRK₂₂ dan metode koefisien berubah iPIMRK₂₂



Gambar B.2.b: Perbandingan presisi digit metode koefisien konstan $iPIMRK_{33}$ dan metode koefisien berubah $iPIMRK_{33}$

dilihat bahwa metode $iPIMRK_{33}$ memberikan rata-rata *speed-up* total yang lebih baik dari metode $iPIMRK_{22}$, tetapi ditinjau dari sisi efisiensi berlaku sebaliknya. Selain itu untuk kedua metode terlihat bahwa *speed-up* dan efisiensi yang diperoleh dari metode dengan koefisien parameter berubah selalu lebih baik dari metode dengan koefisien konstan.

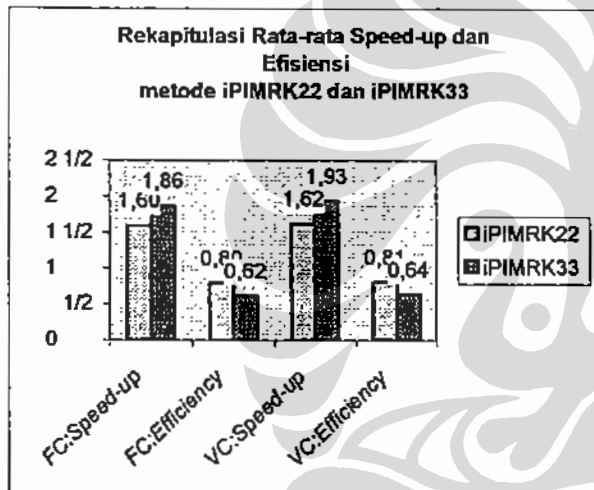
b. Rekapitulasi *Speed-up* dan Efisiensi Parsial (Per Komponen Paralelisasi)

Tabel B.3.b: Hasil Rekapitulasi *Speed-up* & Efisiensi parsial metode $iPIMRK_{22}$ dan $iPIMRK_{33}$

Koefisien Parameter Metode	Rata-rata <i>Speed-up</i> fcn	Rata-rata <i>speed-up</i> dec	Rata-rata <i>speed-up</i> sol
Metode $iPIMRK_{33}$ (3 prosesor)			
Konstan	1,79	1,73	0,67
Berubah	1,87	1,74	0,67
Metode $iPIMRK_{33}$ (3 prosesor)			
Konstan	2,54	2,18	0,48
Berubah	2,73	2,31	0,39
Koefisien Parameter Metode	Rata-rata Efficiency fcn (%)	Rata-rata Efficiency dec (%)	Rata-rata Efficiency sol (%)
Metode $iPIMRK_{22}$ (2 prosesor)			
Konstan	89,31	86,29	33,30
Berubah	93,56	87,09	33,63
Metode $iPIMRK_{33}$ (3 prosesor)			
Konstan	84,82	72,60	16,16
Berubah	91,16	76,98	12,93

3.2.3. Perbandingan *Speed-up* dan Efisiensi

a. Rekapitulasi *Speed-up* dan Efisiensi Total

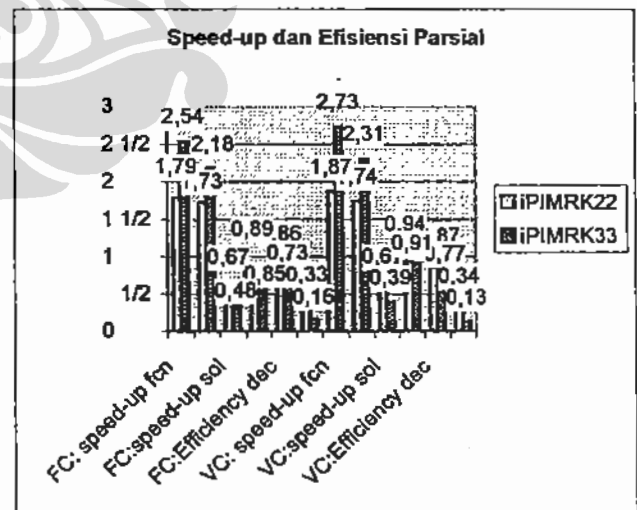


Gambar B.3.a: Rekapitulasi Rata-rata *Speed-up* & Efisiensi metode $iPIMRK_{22}$ dan $iPIMRK_{33}$

Tabel B.3.a: Rekapitulasi Rata-rata *Speed-up* dan Efisiensi Total.

Metode	Koefisien Konstan	
	Rata-rata <i>Speed-up</i>	Rata-rata Efficiency(%)
$iPIMRK_{22}$	1,60	79,77
$iPIMRK_{33}$	1,86	62,09
Metode	Koefisien Berubah	
	Rata-rata <i>Speed-up</i>	Rata-rata Efficiency(%)
$iPIMRK_{22}$	1,62	81,10
$iPIMRK_{33}$	1,93	64,36

Berdasarkan hasil eksperimen untuk *speed-up* dan efisiensi total pada Tabel B.3.a dan Gambar B.3.a, dapat



Gambar B.3.b: Diagram batang terpisah, hasil Rekapitulasi *Speed-up* & Efisiensi parsial metode $iPIMRK_{22}$ dan $iPIMRK_{33}$

Berdasarkan data hasil eksperimen pada Tabel B.3.b yang diplot menjadi grafik pada Gambar B.3.b dapat dilihat bahwa metode iPIMRK₃₃ memberikan rata-rata *speed-up* parsial yang lebih baik dari metode iPIMRK₂₂, tetapi ditinjau dari sisi efisiensi masih berlaku keadaan sebaliknya namun perbedaan efisiensinya sudah cukup kecil. Selain itu pada kedua metode terlihat bahwa kontribusi positif untuk *speed-up* dan efisiensi yang dominan dihasilkan oleh proses paralel evaluasi fungsi dan paralel faktorisasi sedangkan proses paralel solusi sistem persamaan linier selalu memberikan kontribusi negatif pada *speed-up* dan efisiensi. Perlu dipertimbangkan untuk membuang aspek *PARALLEL_SOLVES* dari implementasi metode iPIMRK pada sistem paralel MPI-LINUX di Lab HPCSSUI.

4. KESIMPULAN

Berdasarkan hasil-hasil eksperimen evaluasi kinerja di atas dapat disimpulkan bahwa *speed-up* tidak berhasil diperoleh untuk tes problem yang dimensinya kecil atau sistem matriknya berbentuk *banded matrix* khususnya untuk tes problem yang menggunakan *driver* DR_SVMRK, DR_CUSP, dan DR_SBRUS, sedangkan tes problem yang dimensinya besar dengan sistem matrik berbentuk *full matrix*, khususnya tes problem *dense* dengan *driver* DR_SDENSE untuk dimensi $N=250$ diperoleh *speed-up* dan efisiensi yang cukup signifikan.

Metode dengan koefisien berubah iPIMRK₂₂ dan iPIMRK₃₃ ditinjau dari segi *speed-up*, efisiensi dan presisi digit memiliki kinerja yang lebih baik dari metode dengan koefisien konstan iPIMRK₂₂ dan iPIMRK₃₃, tetapi ditinjau dari sisi waktu eksekusinya berlaku keadaan sebaliknya. Selanjutnya jika dibandingkan antara metode iPIMRK₂₂ yang menggunakan dua prosesor dengan metode iPIMRK₃₃ yang menggunakan tiga prosesor terlihat bahwa metode iPIMRK₃₃ lebih unggul dari sisi *speed-up* dan presisi digitnya tetapi lebih buruk dari sisi efisiensinya, selain itu dari sisi waktu komputasi metode iPIMRK₃₃ cenderung lebih lama dari iPIMRK₂₂ walaupun pada kasus tertentu waktu komputasi pada metode koefisien berubah iPIMRK₂₂ dengan prediktor P2 dan P4 terlihat lebih lama dari metode koefisien berubah iPIMRK₃₃ dengan prediktor P2 dan P4.

Kontribusi positif untuk *speed-up* dan efisiensi didominasi oleh proses *PARALLEL_STAGES* diikuti oleh proses *PARALLEL_FACTORS*, sedangkan proses *PARALLEL_SOLVES* justru memberikan kontribusi negatif. Untuk lebih meningkatkan hasil *speed-up* dan efisiensi metode iPIMRK₂₂ dan iPIMRK₃₃ pada eksperimen sebaiknya proses *PARALLEL_SOLVES*-nya tidak digunakan.

REFERENSI

- [1] Bustamam, A., *Implementasi Metode IPIMRK Untuk Menyelesaikan Persoalan Nilai Awal yang Stiff pada Sistem Paralel MPI-Linux*, Tesis S2 Magister Ilmu Komputer-UI, 2002.
- [2] Bustamam, A., Suhartanto, H. dan Basaruddin, T., "Implementasi Metode Iteratif Paralel Implisit Multistep Runge-Kutta pada Sistem Paralel MPI-Linux", *Prosiding Lokakarya Komputasi dalam Sains dan Teknologi Nuklir-XIII*, BATAN, 2002.
- [3] Bustamam, A., Suhartanto, H. dan Basaruddin, T., "Implementasi Perangkat Lunak SVMRK pada Sistem Paralel MPI-Linux", *Prosiding Seminar Ilmiah Nasional Komputer dan Sistem Intelijen KOMMIT2002*, Gunadarma, 2002.
- [4] Bustamam, A., Suhartanto, H. dan Basaruddin, T., "Beberapa Hasil Eksperimen dan Evaluasi Kinerja Metode iPIMRK pada Sistem Paralel MPI-Linux", *Prosiding Seminar Nasional Ilmu Komputer dan Teknologi Informasi SNIKT1-III*, Tarumanagara, 2002.
- [5] Burrage, K., dan Suhartanto, H., "Parallel Iterated Method Based on Multistep Runge-Kutta of Radau Type for Stiff Problems", *Adv. Comput. Math.*, 1997, vol.7, pp.59-77.
- [6] Burrage, K., dan Suhartanto, H., "Parallel Iterated Method Based on Variable-Step Multistep Runge-Kutta", Submitted to *Adv. Comput. Math.*, 1997.
- [7] Burley, J. C., *Using and Porting GNU Fortran*, Free Software Foundation, Boston, U. S. A., 2001.
- [8] Freeman, L. dan Philips, C., *Parallel Numerical Algorithms*, Prentice Hall International, UK, pp. 22-44, 1992.
- [9] Gropp, W., Lusk, E. dan Skjellum, *Using MPI: Portable Programming with Message Passing Interface*, MIT Press, 1995.
- [10] Message Passing Interface Forum, "MPI: A Message Passing Interface Standar", *Int. Journal of Supercomputer*, 1994, vol 8(3/4), pp. 159-416, Special Issue on MPI.
- [11] Ohio Supercomputer Center, *MPI Primer/Developing with LAM*, the Ohio State University, 1996.
- [12] Suhartanto, H. dan Burrage, K., "S-PMRK: Parallel Stiff ODE-Solver Based on Iterated Variable-Step Multistep Runge-Kutta", Submitted to *Adv. Comput. Math.*, 1997.
- [13] Suhartanto, H., *Parallel Iterated Techniques Based on Multistep Runge-Kutta Method of Radau Type*, Ph.D Thesis, University of Queensland, Brisbane, Australia, 1997.