CHAPTER 7

DISCUSSIONS

7.1 Introduction

This chapter presents discussions of the sequential and parallel experiments in figures and Tables as shown in chapter 6. Comparison of sequential and parallel results of various schemes on 1-D Parabolic and Bio-Heat Equations, 2-D Parabolic and Bio-Heat Equations and 1-D, 2-D,3-D sequential and parallel results are given.

7.2 Comparison of various schemes on 1-D Parabolic and Bio-Heat Equations (Sequential Results)

Table 6.1 shows comparative results for the 1-D Parabolic Equation given by (Saulev (1964)) using the stationary and classes of alternating direction explicit schemes for $\lambda = 0.5,\ t = 0.25,\ eps = 10^{-4}$. The number of iterations, the average absolute error (abs), and the RMS in sequential algorithms are shown. The sequential solutions to the Parabolic Equation for the same problem obtained by means of the various different schemes at the same time levels are presented. The results demonstrate that for the experiment, the fourth-order IADE and fourth-order AGE are the most accurate in comparison to other schemes as well as second-order IADE methods. Hence, the fourth-order AGE scheme is more accurate in comparison in Table 6.1.

The Table also shows that the fourth-order AGE scheme achieves a low iteration count by using a convergence requirement of $\varepsilon = 10^{-4}$. This indicates that the algorithm is a fast numerical solver. The fourth-order IADE scheme is also seen to have low iteration count than the classes of IADE schemes employing the Mitchell & Fairweather variant is fourth-order accurate in space and second-order accurate in time whereas the IADE-MF and IADE-DY variants are second-order accurate in both space and time. As for the IADE variant, its truncation error is $O\left[(\Delta x)^2 + (\Delta t)\right]$. The GS accuracy is the least among all the methods under consideration.

The Crank-Nicolson type approximation gives better accuracy than the implicit method, thus producing a more accurate solution as compared to the IADE-DY (IMP). However, a drawback in using the former is that it incurs more work than the latter in the derivation of its formulae. In conclusion, the fourth-order AGE scheme has merits as an alternative method with respect to stability, accuracy and rate of convergence. Its computational properties are well-suited for implementation on parallel computers.

Table 6.2 shows comparative results for the 1-D Bio-Heat Equation using the stationary and classes of alternating direction explicit schemes for $\lambda = 0.5,\ t = 0.25,\ eps = 10^{-4}$. The average absolute errors of the various stationary and iterative methods are shown. The sequential solutions of the 1-D Bio-Heat Equation for the same problem obtained by means of the various different schemes at the same time levels are presented. The results demonstrate that for the experiment, the fourth-order AGE is the most accurate in comparison to other schemes.

7.3 Parallel Results for 1-D Parabolic and Bio-Heat Equations

In Table 6.3, MPI and PVM parallel results on 1-D Parabolic Equation for various algorithms for 300 mesh sizes are presented. In the Table, we also listed the elapsed time for the master task, T_w, the master CPU time, T_m, and the slave data communication time, T_{sd}, speedup and efficiency all in seconds for the PVM and recorded speedup and efficiency for MPI. In Table 6.3, it is observed that as the processors increases T_w deceases and T_m remains constant. We can also observe that as we scale up the processing units the speedup and efficiency get better. The data listed in the Table 6.3 gives values of the speedup and efficiency both for PVM and MPI for the number of processors used in the computation for 300 mesh size.

The total execution time increases due to the communication cost and additional cost for computing the solution on interface boundaries of each time-step. From the results, it is shown that the processors involved in the computation of solution spend most of their time on computations. The data in Table 6.3 shows that as the number of processors increases the parallel efficiency decreases and the speedup increases for all the schemes. The decrease in efficiency is due to the communication cost and additional computation cost with the increase in the number of processors. The low computation cost and the low computation overhead of the algorithms are reasons for high speedup and efficiency.

The speedup with increasing processors for the GS and SOR schemes on PVM and MPI platform for 100 and 300 mesh sizes is plotted in Fig. 6.1 for 1-D Parabolic case and Fig. 6.2 shows the speedup of GS and SOR for 100 and 300 mesh sizes with MPI and PVM for 1-D Bio-Heat. In Fig. 6.1, we observed that the implementation with MPI gives more linearity conformity than the schemes with the PVM implementation and the SOR performs better in comparative. Fig. 6.2 gives a little improvement on Fig. 6.1 in terms of linearity conformity. We have in Fig. 6.3 the GS and SOR efficiency of 100 and 300 mesh sizes with PVM and MPI for the 1-D Parabolic case and Fig. 6.4 gives the GS and SOR efficiency of 100 and 300 mesh sizes with PVM and MPI for the Bio-Heat case. Fig. 6.4 shows a slight improvement over Fig. 6.3 considering the SOR class, particularly the MPI implementation. Similarly, the speedup of 100 and 300 mesh sizes for 1-D Parabolic case of IADE-DY and IADE-MF PVM and MPI is shown in Fig. 6.5. Fig. 6.6 shows the speedup of IADE-DY and IADE-MF for 100 and 300 mesh sizes for PVM and MPI in the 1-D Bio-Heat case. We observe that as the mesh size increases the linearity of the speedup improves.

Efficiency for 1-D Bio-Heat of 100 and 300 mesh sizes for PVM and MPI is shown in Fig. 6.7. In the figure we observe that as the meshes increases, the efficiency of the schemes get better. However, the IADE-MF class of method is more efficient compared to IADE-DY and SOR. Similarly, the implementation on MPI platform shows better conformity. The speedup for the IADE-4th order scheme and the AGE-4th order scheme for 100 and 300 mesh sizes for 1-D Parabolic and 1-D Bio-Heat are shown in Fig. 6.8 and Fig. 6.9 respectively. In these figures, we observed that the AGE-4th order scheme has better linearity property in comparison to the IADE-4th order scheme especially for the MPI implementation. Moreover, their efficiencies are shown in Fig. 6.10 and Fig. 6.11. In the figures for the efficiencies, we observed that the AGE class of method for Parabolic and Bio-Heat show closeness to unity than every other scheme considered in the experiment. In conclusion, the linearity of the speedups in Fig. 6.9 and Fig. 6.8 are more visible than Fig. 6.6, 6.5, 6.2 and 6.1. In Fig. 6.10 and Fig. 6.11, the efficiencies are closer to unity than Fig. 6.7, 6.4 and 6.3. From the figures and Tables, the parallel algorithm maintains good stability by having a low maximal error confirming the mathematical theory given in (Dou & Phien-Thien (1997)).

7.4 Comparison of the various Schemes on 2-D Parabolic and Bio-Heat Equation (Sequential Experiments)

Comparative results for Eq. (6.7) and (6.8) using various schemes, showing the average of all absolute errors, root mean square errors (RMS), iteration number, $\Delta x, \Delta t, \lambda, t$ and $\varepsilon = 10^{-4}$ are given in Table 6.7 for the Parabolic case. Similarly, comparative sequential results for Eq. (6.9) and (6.10) with the various schemes, showing the average of all absolute errors, RMS, iteration number, Δx, Δt, λ, t and $\varepsilon = 10^{-4}$ are given in Table 6.8 for the Bio-Heat case. The results demonstrate that for the experiment, the AGE class is the most accurate in comparison. The Table also shows that the AGE scheme achieves a low iteration count by using a convergence requirement of $\varepsilon = 10^{-4}$. This indicates that the algorithm is a fast numerical solver for both Parabolic and Bio-Heat Equations. The MF-DS scheme is also seen to have low iteration count than the classes of IADE schemes. As for the IADE variant, its truncation error is $O\left[(\Delta x)^2 + (\Delta t)\right]$. The GS accuracy is the least among all the methods under consideration. In conclusion, the AGE scheme has merits as an alternative method with respect to stability, accuracy and rate of convergence. Its computational properties are well-suited for implementation on parallel computers.

7.5 Parallel Results for 2-D Parabolic and Bio-Heat Equations

In Table 6.9, the MPI and PVM results for 2-D Parabolic Equation for 300x300 mesh sizes for various algorithms are presented. In the Table, we also listed the elapsed time for the master task, T_w, the master CPU time, T_m, and the slave data communication time, T_{sd}, speedup and efficiency all in seconds for the PVM and recorded speedup and efficiency for MPI. Compared to Table 6.3, here, we observe an increase in speedup and efficiency. This means, as the mesh sizes increases the speedup and efficiency get better as observed. Also, the T_w increases as the mesh sizes increases and T_m remains constant for their increased values. The AGE class of scheme show better speedup both for PVM and MPI than other schemes. Fig. 6.12 shows the speedup for GS and SOR PVM implementation for 100x100 to 300x300 mesh sizes for 2-D Parabolic and Fig. 6.13 shows the speedup for IADE-DY, MF-DS and the AGE PVM implementation for 100x100 to 300x300 mesh sizes for 2-D Bio-Heat. We observed that the speedup in Fig.

6.13 shows better linearity property, especially the AGE class of method for higer mesh sizes. This shows that as the mesh increases, the speedup improves. Fig. 6.14 shows the speedup for the MPI implementation using IADE-DY, MF-DS and the AGE for the various meshes. The speedup for the various stipulated meshes for IADE-DY, MF-DS and AGE is shown in Fig. 6.15 for the 2-D Bio-Heat. In this figure, we observed improved linearity conformity for the AGE class of method in comparative.

The results in Fig. 6.16 and Fig. 6.17 show the efficiency of various mesh sizes with PVM and MPI for IADE-DY, MF-DS and AGE method respectively, for 2-D Parabolic. We observed that the MPI implementation is closer to unity. Fig. 6.18 and Fig. 6.19 show the efficiency of various mesh sizes with PVM and MPI for the 2-D Bio-Heat showing the IADE-DY, MF-DS and AGE schemes. We observed from these figures that the MPI efficiency is closer to unity than the PVM efficiency for both differential equations under consideration. That is, the AGE class scheme MPI implementation is more conformed to unity and more accurate than other class of schemes under consideration. However, the parallel efficiency decreases with the increasing block number for given grid size. Given other parameters the speedup increases with the number of processors (Fig. 6.2 – 6.15). At a large number of processors, Amdahl’s law starts to operate, imposing a limiting speed-up due to the constant serial time. Note that the elapsed time is a strong function of the background activities of the cluster. When the number of processors is small, the wall time decreases with the number of processors. As the number of processors become large, however, the wall time increases with the number of processors.

7.6 Comparison of the various Schemes on 1-D Telegraph Equation Sequential Experiments

Table 6.21 provides a comparison of the accuracy of the methods under consideration in terms of absolute error and RMS for the problem given in Eq. (6.11) and (6.12) at the appropriate grid points for the mesh ratio $\lambda = 0.5$ at time levels of $t = 0.25, eps = 10^{-4}, \Delta t = 5\times10^{-3}, \Delta x = 10^{-1}.$ The result of the three level implicit schemes is compared to the results of IADE-MF. The results in Table 6.21 demonstrate that the IADE-MF method has comparable accuracy. In reference to (Evan & Hassan (2003)), more reasons for the performance of the alternating class method has been given. It is evident that the three level implicit scheme is less accurate than the IADE-MF scheme. The iteration number necessary for convergence for IADE-MF is lesser.

7.7 Comparison of the various Schemes on 1-D Telegraph Equation Parallel Experiments

Table 6.22 shows the results for the implicit and IADE schemes with PVM and MPI speedup and efficiency. In the Table, we also listed the elapsed time for the master task, T_w, the master CPU time, T_m, and the slave data communication time, T_{sd}, speedup and efficiency all in seconds for the PVM and recorded speedup and efficiency for MPI. From the Table, we observe that as the number of processors increases, the execution time decreases. This shows that as the number of processors increases, though it might lead to a decrease in execution time but will get to a point that increasing the processors will not have much impact on total execution time. Comparing the performances of the two schemes from the Table, we observe that the IADE-MF scheme gives better parallelism and scalability than the three level implicit scheme. Hence, performance can be improved when the processors are sufficiently large.

Fig. 6.24 shows the PVM speedup for 1-D Telegraph using the IMP and IADE scheme for various mesh sizes and Fig. 6.25 shows the respective MPI implementation for various mesh sizes. We observe that, as the mesh sizes increases the linearity becomes visible. The IADE-MPI (300 mesh) in Fig. 6.25 shows slight linearity over the IADE-PVM (300 mesh) in Fig. 6.24. However, the parallel results in Table 6.22 show that the IADE-MF has a slight increase speedup and efficiency than the IMP scheme. The efficiency is shown in Fig. 6.26 for 1-D Telegraph PVM implementation using IMP and IADE schemes and Fig. 6.27 shows the efficiency for MPI implementation using the IMP and IADE scheme on 1-D Telegraph. Similarly, as the various mesh sizes increase, the efficiency of higher mesh sizes are near unity than the lower mesh sizes. The time spend in data exchange will be significant compared to the time spend in computation and the parallel efficiency goes down. Hence, when the number of processors increases, balancing the number of computational cells per processors will become a difficult task due to significant load imbalance. Hence, increasing mesh sizes improves parallelization in a distributed system due to utilizing processors effectiveness. We observed that the efficiency of the three meshes decreases with increasing processors and the efficiency for the higher mesh sizes are better than the smaller mesh size. Hence, by inference the larger the mesh sizes up to certain number, the speedup improvement is near linearity. The performance begins to degrade with an effect caused by increase in communication overhead as the mesh sizes increases.

The problem size is scaled up following the memory-bounded constraint. This phenomenon is well under expected since the implicit replacement has a very low computation overhead as implemented on problems. The computation time increases very slowly as the number of processors increases. However, these jumps in communication time which are relatively larger than the others are mainly caused by the architecture of the communication between the processors, that is, due to the underlying machine architecture not the algorithm.

7.8 Comparison of various Schemes on 2-D Telegraph Equation Sequential Experiments

Fig. 6.28 and Fig. 6.29 provide a comparison of the accuracy of the methods for Eq. (6.13 and 6.14) at the appropriate grid points for the mesh ratio at time levels of $t = 0.5, t = 1.0, t = 1.5 \, and \, t = 2.0$. The average of the absolute errors can be seen in Fig. 6.28 and the root mean square errors in Fig. 6.29 (from Table 6.25 – 6.26) of the methods under consideration for the sequential algorithms. The results of the ADI and IADE-DY methods are compared to the results of MF-DS. The results in these figures demonstrate that the MF-DS method is more accurate compared to the ADI method of (Peaceman & Rachford (1955)). Similar reasons show that MF-DS is more accurate than the ADI in (Mitchell & Fairweather, (1964) and Sahimi et al., (2006). We used $\lambda = 0.5, \Delta t = 0.05, \Delta x = 0.5, \Delta y = 0.5, r = 0.5$, and $\varepsilon = 10^{-4}$ in Fig. 6.28 and Fig. 6.29.

The results in Fig. 6.28 also show that the IADE-DY scheme is unconditionally stable and seems to be more accurate than the ADI scheme as well. However, the MS-DS and IADE-DY are accurate than the ADI. Based on (Sahimi et. al. (2006)), the DS-MF is convergent, computationally stable and highly accurate. The figures have been able to show this.

7.9 Comparison of various Schemes on 2-D Telegraph Equation Parallel Experiments

In Table 6.27, the MPI and PVM results on 2-D Telegraph Equation for 300 x 300 mesh size for various algorithms are presented. In the Table, we also listed the elapsed time for the master task, T_w, the master CPU time, T_m, and the slave data communication time, T_{sd}, speedup and efficiency all in seconds for the PVM and recorded speedup and efficiency for MPI. Compared to Table 6.22, here, we observe an increase in speedup and efficiency. The MF-DS class of scheme is more linear in speedup than other schemes. Fig. 6.30 shows various speedups for ADI, IADE and MF-DS schemes with PVM implementation for 100x100 to 300x300 mesh sizes for 2-D Telegraph and Fig. 6.31 shows the various speedups for ADI, IADE and MF-DS with MPI implementation for 100x100 to 300x300 mesh sizes for 2-D Telegraph. From the figures we observed that as the mesh sizes increase, the linearity get visible. However, the MF-DS-MPI (300x300 mesh) speedup values in Table 6.9 shows slight improvement over the MF-DS-PVM (300X300 mesh) speedup. Hence, the MF-DS scheme performs better in comparison to the IADE scheme. Fig. 6.32 shows efficiency of various mesh sizes with PVM for 2-D Telegraph and Fig. 6.33 shows the efficiency for various meshes with MPI for 2-D Telegraph. From the two figures, we observed that as the mesh sizes increase, the efficiency gets closer to unity as visibly observed for the MF-DS scheme. The linearity property and closeness to unity in the figures are most observed for the MF-DS MPI.

When a single processor is used to calculate a large problem, the local memory of the processor is insufficient to store all the data and a large part of the data has to be distributed to remove memory by the distributed memory system. Therefore, the serial computation seems to be relatively less efficient or slower than its parallel counterparts.

These results show that the parallel efficiency increases with increasing mesh sizes. Comparing the performances of the different schemes, Table 6.27 and Fig. 6.30 to Fig. 6.33, we observed that the MF-DS has slight improvement over IADE-DY and better improvement over the ADI scheme with MPI and PVM. Hence, the MF-DS is an improvement on the IADE-DY and can improve performance when the CPUs are sufficiently large. From Fig. 6.30 and Fig. 6.31, as the mesh sizes increases, the speedup approaching linearity becomes visible. MPI is focused on message passing and explicitly states that resource management and the concept of a virtual machine are outside the scope of the MPI standard (Foster et al., (1998) & Groop et al., (1999)). Hence, MPI is a much richer communication method than PVM. PVM provides only simple message passing. Similarly, as the mesh sizes increases, the efficiency gets closer to unity. The implementation with MPI in terms of the algorithm performance and convergence shows improvement as compared to the implementation with PVM in relation to the results obtained in the tables and figures. The MPI is expected to run faster within a large multiprocessor. It has many more point-point and collective communication options than PVM. Here, the MPI has the ability to specify a logical communication topology. The improved performance of the MPI is due to the concepts of context and a group of processes into a communicator and is suitable for non-blocking send used in our experiments.

Hence, we see that the parallel efficiency is strongly dependent on the problem size and the number of processors. Generally, it increases with the increasing problem size. The magnitude of the parallel efficiency can be estimated. However, the domain decomposition greatly influences the performance of the parallel computation. For a given mesh size, there is an optimum number of sub-domains which maximizes the domain decomposition efficiency. This optimum number of sub-domains increases with the mesh size. However, the speedup increases with the number of processors, both for

MPI and PVM. Their relative performance show that MPI perform slightly better than PVM. We also observed that as the mesh sizes increases, the execution time increases as well with a proportionate decrease in time giving the efficiency in Fig. 6.32 for PVM and Fig. 6.33 for MPI as the processors increases. The time spend in data exchange will be significant compared to the time spend in computation and the parallel efficiency goes down. We observed that, the efficiency of the three meshes decreases with increasing processors and the efficiency for the higher mesh sizes are better than the smaller mesh sizes.

7.10 Discussions of 3-D ADI Scheme on 3-D Telegraph Equation Sequential Experiment

From Eq. (5.1) using the 3-D ADI scheme from Eq. (5.14) – Eq. (5.16), the values of the root mean square errors (Table 6.38) against different levels of t is plotted in Fig. 6.38 for $t = 1.0, t = 2.0, t = 1.0 \text{ and } t = 2.0 \text{ and } h = 1/16, 1/32 \, and \, 1/64$. Similarly, the values of the RMS errors (Table 6.39) for the 3-D ADI is given in Fig. 6.39 for $\lambda = 3.2$ $t = 1.0, t = 2.0, t = 1.0 \text{ and } t = 2.0$ and $h = 1/16, 1/32 \, and \, 1/64$. The results only confirm the unconditional stability which has been proven (see Mohanty, (2009)). This indicates that the algorithm is a numerical solver for the Telegraph Equation. In conclusion, the 3-D ADI scheme has merits as an alternative method with respect to stability, accuracy and rate of convergence. Its computational properties are well-suited for implementation on parallel computers.

7.11 Discussions of 3-D ADI Scheme on 3-D Telegraph Equation Parallel Experiments

In Table 6.40, the MPI and PVM results on 3-D Telegraph Equation for 300 x 300x300 mesh size using the 3-D ADI scheme are presented. In the Table, we also listed the elapsed time for the master task, T_w, the master CPU time, T_m, and the slave data communication time, T_{sd}, speedup and efficiency all in seconds for the PVM and recorded speedup and efficiency for MPI. Comparing Table 6.27 for the 2-D Telegraph Equation to Table 6.40 of the 3-D Telegraph Equation, here, we observe an increase in speedup and efficiency. Fig. 6.40 shows the speedup for 3-D ADI 300x300x300 mesh sizes with PVM and MPI and Fig. 6.41 shows the efficiency of 300x300x300 mesh sizes with PVM and MPI. The linearity property and closeness to unity in the figures are most observed for the 3-D ADI MPI implementation.

7.12 Performance Improvement Comparison of Schemes

Since iterative methods are usually needed in high resolution simulations, we examine the performance of the schemes used in Chapters 3, 4 and 5 for solving the large sparse linear systems arising from discretized 1-D, 2-D and 3-D Parabolic and Telegraphic equations. In Tables 6.5, 6.6, 6.11 and 6.12, we give the performance improvement for the stationary iterative methods and iterating alternating methods. Table 6.4 shows the effectiveness of various schemes with PVM and MPI for 300 mesh size on 1-D Parabolic case. From the Table, it is observed that as the number of processor increases the efficiency gets better in respect to convergence, and the effectiveness for the MPI implementation is slightly better. In the same table, the AGE-4th scheme class of methods give better effectiveness than other schemes both for MPI and PVM.

The improvement given in Tables 6.5 for 1-D Parabolic 300 mesh size, 6.11 for 2-D Parabolic, 6.24 for 1-D Telegraph and 6.29 2-d Telegraph case for various schemes are implemented by using $1-T_{high\,scheme}/T_{lower\,scheme}$, where $T_{highscheme}$ and $T_{lowscheme}$ are the times required for the various schemes. In the Tables, $T_{highscheme}$ equals varying schemes for the alternating iterating schemes and $T_{lowscheme}$ represents the stationary iterating schemes, respectively. The algorithm in the Tables 6.6 to 6.29 shows significant improvement in many CPUs. However, when implemented in a small number of CPUs, the improvement is not so evident since the communication cost only a small portion of the total wall time. As the number of CPU increases, the bottleneck neck of parallel computers appears and the global reduction consumes a large part of time; we anticipate that the improvement will move to be significant. In Table 6.6, we observed that as the number of processor increases the time for each scheme~~s~~ decreases and the performance comparison shown for various schemes show comparative improvement, respectively. In Table 6.11 and 6.12, we observed that the performance comparison of the AGE class of methods give better performance compared to other class of iterative methods. Table 6.24 shows performance for two schemes on 1-D Telegraph equation, and it is observed that the implementation with MPI shows slight improvement than PVM. In Table 6.29, it is observed also that as the number of processor increases the performance improvements get slightly better due to convergence criterion. Hence, MF-DS class of methods yield better convergence.

Table 6.10 shows effectiveness for various schemes with PVM and MPI for 300 x 300 mesh size parabolic case, and the same improvement was observed for different schemes both for PVM and MPI. Here, the AGE class of method as above is more effective as it combines the element of numerical stability. Table 6.23 gives the effectiveness of various schemes with PVM and MPI for 300 mesh size on 1-D Telegraph equation, and the IADE-MF class is more effective than the IMP method for both MPI and PVM. Effectiveness of various schemes on 2-D Telegraph 300 x 300 mesh size both for PVM and MPI is shown in Table 6.28, with the MF-DS class being more effective. The values in the Tables indicate that the stationary iterative methods are not very scalable with respect to the problem size for solving discretized higher Parabolic and Telegraphic equations. However, the SOR method with optimum relaxation parameter is a significant improvement over the GS method. The average number of the iterative alternating iterations increases faster than that of the stationary iterations, indicating that, for a very large size problem, the iterating iteration performs better than stationary methods.

7.13 Parallel Efficiency of Schemes

To obtain a high efficiency, the slave computational time T_{sc} should be significantly larger than the serial time. When the number of processors is small, the wall time decreases with the number of processors. When the number of processors becomes large, however, the wall time increases with the number of processors. The total CPU time is composed of the waste time, slave data time and slave computational time. Data communication at the end of every iteration is necessary in this strategy. Indeed, the updated values of the solution variables on the full domain are multicast to all slaves after each iteration since a slave can be assigned a different sub-domain under the pool-of-task paradigm. For a given grid size, the CPU time for task id to slaves increases with block numbers (Tables 6.13, 6.14, 6.30 and 6.31).

7.14 Numerical Efficiency of the Schemes

The numerical efficiency E_{num} includes the DD efficiency E_{DD} and convergence rate behavior N_o / N_1, as defined in Eq. 5.68. The DD efficiency $E_{DD} = T_{B=1}^{No}(1) / T_{B=B}^{No}(1)$ includes the increase of floating point operations induced by grid overlap at interfaces and the CPU time variation generated by DD techniques.

Tables 6.13, 6.14, 6.30 and 6.31 shows the CPU time for the slave computation for different grid sizes which is the parallelizable part, running with one processor with varying sub-domain numbers and grid sizes. Due to the repeat computation at the interfaces, the number of operations actually changes with block sizes as seen from the Tables (6.13, 6.14, 6.30 and 6.31). It is interesting to note that there is an optimum block number which minimizes the slave CPU time for computation. This is due to two reasons:

1. The doubling of the calculations at the interfaces of the sub-domains, which increases with block number; and
2. A CPU time reduction owing to a small ratio of memory access to memory occupation for a large block number.

The convergence rate behavior N_o / N_1, the ratio of the iteration number for the best sequential CPU time on one processor and the iteration number for the parallel CPU time on *n* processor, describe the increase in the number of iterations required by the parallel method to achieve a specified accuracy, as compared to the serial method. This increase is caused mainly by the deterioration in the rate of convergence with increasing number of processors and sub-domains. Because the best serial time is not known generally, we take the existing parallel program running on one processor to replace it. Now the problem is that how decomposition strategy affects the convergence rate? The results are summarized in Tables 6.15, 6.16, 6.17, 6.18, 6.19 and 6.20 for parabolic case and Tables 6.32, 6.33, 6.34, 6.35 and 6.36 for both PVM and MPI. Their figures are shown in Fig. 6.21, 6.23, 6.34, 6.35, 6.36 and 6.37 for both PVM and MPI.

It can be seen that N_o / N_1 decreases with increasing block number and increasing number of processors for a given size. The larger the grid size is, the higher is the convergence rate. For a given block number, a higher convergence rate is obtained with less processors. This is because one processor may be responsible for a few sub-domains at each iteration. If some of these sub-domains share some common interfaces, the subsequent blocks to be computed will use the new updated boundary values, and therefore, an improved convergence rate results. The convergence rate is reduced when the block number is large. The reason for this is evident: the boundary conditions propagate to the interior domain in the serial computation after one iteration. But this is delayed in the parallel computation. In addition, the values of variables at the interfaces used in the current iteration are the previous values obtained in the last iteration. Therefore, the parallel algorithm is less “implicit” than the serial one. Despite of these inherent short comings, a high efficiency is obtained for large scale problems. Many improvements have been made and a more detailed analysis has been given. This has led to a more systematic analysis of the results presented in Chapter 6.