PERFORMANCE ASSESSMENT OF THE SIMFAP PARALLEL CLUSTER AT IFIN-HH BUCHAREST

GH. ADAM1,2, S. ADAM1,2, A. AYRIYAN1, E. DUSHANOV1, E. HAYRYAN1, V. KORENKOVI1, A. LUTSENKO1, V. MITSYN1, T. SAPOZHIKOV1, A. SAPOZHIKOV1, O. STRELTSOVA1, F. BUZATU2, M. DULEA2, I. VASILE2, A. SIMA2, C. VIŞAN2, J. BUŞA3, I. POKORNY3

1 Joint Institute for Nuclear Research, 141980 Dubna, Moscow, Russia
2 “Horia Hulubei” National Institute for Physics and Nuclear Engineering (IFIN-HH) 407 Atomistilor, Magurele–Bucharest, 077126, Romania
3 Technical University, Košice, Slovakia

Received December 20, 2007

Performance assessment and case study outputs of the parallel SIMFAP cluster at IFIN-HH Bucharest point to its effective and reliable operation. A comparison with results on the supercomputing system in LIT-JINR Dubna adds insight on resource allocation for problem solving by parallel computing.

1. INTRODUCTION

The prototype parallel SIMFAP cluster at IFIN-HH Bucharest involves an 8-dual-processor array connected by Myrinet 2000 network. It was equipped with open software, both as it concerns the operating system (CentOS 5, which is a freely available Linux distribution that is based on Red Hat’s commercial product Red Hat Enterprise Linux), and the high performance parallel computing message passing interface (MPI) standard.

The present paper reports results concerning performance assessment of the cluster. These point to the effective and reliable operation of the implemented open software infrastructure. A comparison with the peak performance of the recently installed 240-core supercomputing complex at the Central Information and Computer Complex (CICC) of the Laboratory of Information Technologies (LIT) of the Joint Institute for Nuclear Research (JINR) in Dubna provides insight into the size tuning of the processor subset needed for the most effective solution of the existing problems.

The organization of the paper is as follows. In Section 2, results of peak performance measurements of the SIMFAP and CICC systems using the HPL Benchmark [1] are reported. The discussion of the insight obtained from these date is done in Section 3. In Section 4, outputs of two case studies are reported.
the scrutiny of which evidences further features of the parallel computing. The paper ends with conclusions in Section 5.

2. PEAK PERFORMANCE ASSESSMENT

To assess the peak performance, we used the High Performance LINPACK (HPL) Benchmark, developed at the Innovative Computing Laboratory, University of Tennessee [1], and currently used to provide a reliable basis for tracking and detecting trends in high-performance computing. Twice a year, a list of the sites operating the 500 most powerful computer systems is assembled and released [2]. The same benchmark is used for the assessment of the 50 most powerful computers in the Commonwealth of the Independent States (CIS), where the CICC system of JINR occupies the 12-th rank, as of the end of September 2007 [3].

The HPL benchmark measures the computing time $T$ needed for solving systems of linear algebraic equations, based on the LU-decomposition. Given a system of order $N$, the number of elementary floating point operations required for solving it by LU-decomposition is

$$N_{\text{op}} = (2/3) \cdot N^3 + 2 \cdot N^2. \quad (1)$$

Then the cluster performance is given by

$$P = \frac{N_{\text{op}}}{T}. \quad (2)$$

The main characteristics of the SIMFAP and CICC systems can be summarized as follows:

<table>
<thead>
<tr>
<th>Features</th>
<th>SIMFAP</th>
<th>CICC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Processors</td>
<td>Xeon Irwindale</td>
<td>2×Xeon 5150</td>
</tr>
<tr>
<td>Clock frequency</td>
<td>3 GHz</td>
<td>2.66 GHz</td>
</tr>
<tr>
<td>Cores per CPU</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>CPUs per node</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2-level cache memory/CPU</td>
<td>2 MB</td>
<td>4 MB</td>
</tr>
<tr>
<td>RAM on node</td>
<td>4 GB</td>
<td>8 GB</td>
</tr>
<tr>
<td>Nodes within cluster</td>
<td>8</td>
<td>60</td>
</tr>
<tr>
<td>Overall number of CPUs</td>
<td>16</td>
<td>120</td>
</tr>
<tr>
<td>Overall number of cores</td>
<td>16</td>
<td>240</td>
</tr>
<tr>
<td>Overall RAM, $N_{\text{RAM}}$</td>
<td>32 GB</td>
<td>480 GB</td>
</tr>
<tr>
<td>Operating System</td>
<td>CentOS 5</td>
<td>Scientific Linux 4.5</td>
</tr>
<tr>
<td>Network</td>
<td>Myrinet 2000</td>
<td>Gigabit Ethernet</td>
</tr>
<tr>
<td>MPI</td>
<td>Version 1.2.</td>
<td>Version 1.2.7</td>
</tr>
<tr>
<td>Peak theoretical performance</td>
<td>96 GFlops</td>
<td>2553.6 GFlops</td>
</tr>
<tr>
<td>$N_{\text{max}}$</td>
<td>63.2×10^3</td>
<td>244.9×10^3</td>
</tr>
</tbody>
</table>
The last two entries have been estimated as follows:

(i) **The peak theoretical performance**

This parameter is obtained from the following formula which is derived under the assumption that each CPU performs all the elementary operations while spending vanishing time for the information exchange with the cache, the RAM, or the swap disk memory:

\[ P_{\text{theor}} = nkv, \]  

- \( n \) denotes the number of cores within the cluster,
- \( k \) denotes the number of operations done by each core during one CPU period,
- \( v \) denotes the CPU frequency (in GHz).

For the SIMFAP cluster, \( n = 16, k = 2, v = 3 \), whereas for the CICC, \( n = 240, k = 4, v = 2.66 \), wherefrom the above values follow.

(ii) **The maximum system order allowing peak performance**

For a system of \( N \) equations with \( N \) unknowns, the coefficient matrix occupies \( N^2 b \) bytes in the RAM/swap, where \( b \) is the floating point word length in bytes.

Having at our disposal an overall RAM \( N_{\text{RAM}} \), the condition

\[ N_{\text{max}}^2 b < N_{\text{RAM}} \]  

will provide the system order \( N_{\text{RAM}} \) up to which the peak theoretical performance is expected to occur.

We have \( b = 8 \) for the REAL*8 (double precision) data considered in both tests. With the given \( N_{\text{RAM}} \) values for the two systems, the values quoted in the last line follow.

![Fig. 1 – Computing time and performance of SIMFAP cluster.](image-url)
Fig. 2 – Computing time and performance of JINR CICC cluster.

The benchmark outputs are presented in Figs. 1 and 2 respectively. The corresponding numerical data are given in Table 1.

Table 1
Computing time $T$ and performance $P$ vs. the order $N$ of the system

<table>
<thead>
<tr>
<th>SIMFAP IFIN−HH</th>
<th>CICC LIT−JINR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$ (seconds)</td>
<td>$P$ (GFlops)</td>
</tr>
<tr>
<td>10 000</td>
<td>15.31</td>
</tr>
<tr>
<td>20 000</td>
<td>99.29</td>
</tr>
<tr>
<td>30 000</td>
<td>308.29</td>
</tr>
<tr>
<td>40 000</td>
<td>702.15</td>
</tr>
<tr>
<td>50 000</td>
<td>1331.92</td>
</tr>
<tr>
<td>60 000</td>
<td>2241.66</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. DISCUSSION OF PEAK PERFORMANCE RESULTS

The experimental data reported in the previous section unveil several interesting aspects which deserve consideration.

− While the data have not been collected with the aim at doing a systematic comparison of the performances of the two clusters, SIMFAP and CICC respectively, the scrutiny of the times reported in Table 1 allow us to formulate a first important hint on the request of resource allocation for parallel solutions.

**Fact 1.** The optimal resource allocation (i.e., number of CPUs) needed to achieve minimum computing time of the parallel solution depends on the problem size.
Thus, for the $10^4 \times 10^4$ problem of the HPL test, a 16-core solution proves to be more effective than a 240-core solution. For the $2 \times 10^4 \times 2 \times 10^4$ problem the 16-core and 240-core solutions need comparable computing times. However, for the $5 \times 10^4 \times 5 \times 10^4$ problem, the advantage of the 240-core solution becomes overwhelming.

This is nothing but a straightforward consequence of the standard flow of the solution inside a parallel system; \textit{problem splitting} $\rightarrow$ \textit{local derivation of equations}, for the unknown interface parameters $\rightarrow$ \textit{interface data transfer to a single processor to get the interface parameters} $\rightarrow$ \textit{turn back of the interface parameter solutions} $\rightarrow$ \textit{completion of the local solutions}.

Thus, the interprocessor communication is an important factor which is slowing down the parallel solution. Of course, its effects can be alleviated if the inter-CPU communication takes advantage of the multicore architecture and/or it is hierarchically organized to get acceleration to logarithmic times.

For each of the two different systems under consideration, SIMFAP and CICC, it is worthwhile to define the \textit{minimum theoretical computing time of the HPL benchmark at given order $N$}. This is obtained provided the peak theoretical value (3) is taken for the value of $P$ in equation (2):

$$T_{\text{min}}(N) = N_{\text{op}}/P_{\text{theor}}.$$  \hfill (5)

The plot, in log-log scale, on the same figure, of both the $T_{\text{min}}$ and the actually measured computing times for the two parallel clusters (in Fig. 3) provides direct intuitive insight into the above considerations concerning the optimal resource allocation.

![Fig. 3 – Log-log plot of HPL Benchmark times for SIMFAP and CICC clusters. The $T_{\text{min}}(N)$ values following from Eq. (5) are also given.](image-url)
• The ratio of the measured peak performances to the peak theoretical ones shows significantly different values for the SIMFAP and the CICC clusters respectively. Thus, for the SIMFAP cluster, the peak experimental value of 64.24 GFlops compares very favourably with the peak theoretical one of 96 GFlops. For the CICC cluster, however, the corresponding figures of 1124 and 2553.6 respectively, result in a much lower ratio.

There are several reasons for this discrepancy. First, the Myrinet network securing the intercommunication among the processors inside the SIMFAP cluster is about four times more effective as compared to the Gigabit Ethernet network in the CICC cluster. This difference stems from the initial destination of the two computer complexes. Thus, while SIMFAP was specially conceived to be a parallel cluster, CICC was assumed to be exclusively devoted to data analysis, within a GRID distributed environment, done for the LHC and other large scale experiments at which JINR does participate. Later on, the possibility of its use for solving parallel problems as well, under substantial upgrade of the supervising control system, was defined. The measured parallel computing performance is encouraging, however. It shows that ways do exist for dynamic resource reallocation if urgent alternative needs are arising.

Second, for a given number $n_c$ of CPUs inside a parallel system, there are magic values of the system orders $N$. For such $N$ values, which are multiples of $n_c$, all the existing CPUs solve subproblems of exactly the same magnitudes. Hence all the parallel tasks are ended within exactly the same CPU time, resulting thus in minimum overall parallel computing time. Now, an inspection of the $N$ values for the SIMFAP cluster shows that all of them are magic numbers. Therefore, the SIMFAP cluster outputs correspond to maximum possible cluster performances for the given $N$ sampling. Quite the opposite happens for the CICC system sampling, where none of the existing $N$ values is a magic number. The measured CICC system performance could therefore be still improved provided sets of carefully chosen $N$ orders are considered.

• The Figs. 1 and 2 point to the decrease of the cluster performances under the solution of lower size problems. The details of the two curves for the SIMFAP and CICC clusters differ significantly, however. In the case of the SIMFAP cluster, there is a rather wide low slope (near plateau) region which points to the existence of a range of problems the solution of which can be done at the upper possibilities of the cluster. In the case of the CICC cluster, the occurrence of an overall large slope may be taken for a hint to the need of further improvement of the approach to parallel computing under multi-core architectures.

• We end the discussion with the consideration of a least squares fit ($L_2$-norm) of the computing times obtained on the two parallel systems. The equation (1) points to the increase of the HPL benchmark computing time with
the third power of the order of the solved systems. The third power law should hold true for the accumulated experimental data as well. In order to check this expectation, we solved the $L_2$-problem described in Appendix A for the SIMFAP and CICC data under two different assumptions on the *a priori* uncertainties: $\sigma_i = 1$ (uniform distribution of the uncertainties) and $\sigma_i = \sqrt{f_i}$ (Poisson distribution of the uncertainties) respectively (Table 2).

**Table 2**

Fitting coefficients of the optimum polynomial fit of Appendix A and data dispersion around theoretical fitting polynomial of degree $m = 3$

<table>
<thead>
<tr>
<th>$\text{SIMFAP IFIN-HH}$</th>
<th>$\text{CICC LIT-JINR}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>$\sigma_i = 1$</td>
</tr>
<tr>
<td>0</td>
<td>0.783 + 03</td>
</tr>
<tr>
<td>1</td>
<td>0.4350 – 01</td>
</tr>
<tr>
<td>2</td>
<td>0.1038 – 05</td>
</tr>
<tr>
<td>3</td>
<td>0.8592 – 11</td>
</tr>
<tr>
<td>$s_3$</td>
<td>0.5449</td>
</tr>
</tbody>
</table>

(*) The code yielded the optimum value $m = 4$.

For the SIMFAP data, the code yielded in both cases the optimal order $m = 3$ of the fitting polynomial $f_m(x_i)$, with the $s_3$ values pointing to a dispersion of the data smaller by an order of magnitude under the assumption of an *a priori* Poisson distribution of the uncertainties.

For the CICC data, the code yielded the optimum value $m = 4$ under $\sigma_i = 1$, while $m = 3$ under $\sigma_i = \sqrt{f_i}$. The $m = 4$ output at $\sigma_i = 1$ points a manifestly wrong character of the uniform initial *a priori* uncertainty hypothesis. Once again, the estimated dispersion under Poisson distribution of the *a priori* uncertainties is one order of magnitude smaller than under the alternative hypothesis of uniform initial uncertainties.

An independent measure of the quality of the least-squares fits is given by the relative errors {$e_i(i,m) = |z_i,m|/f_i |, i = 0,1,\cdots,n$}, where $z_i,m$ denote the residuals (9). The data reported in Table 3 point to the well-known drawback of the least squares fit under the uniform initial uncertainty hypothesis: it overemphasizes the importance of the large magnitude abscissas, with monotonically decreasing values of the relative errors across the $N$ sampling.

Under the *a priori* Poisson distribution uncertainty, the SIMFAP data show excellent agreement between the fitting third degree polynomial and the measured data. In the case of the CICC data, the agreement is worse, providing further argument on the possibility of the performance improvement.

Figs. 4 to 7 provide the same conclusions in graphic form.
Table 3

Quality of the third order polynomial fit of the computing times reported in Table 1, measured by the relative errors $e_i(i,m) = |\frac{z_i - f_i}{f_i}|$

<table>
<thead>
<tr>
<th>SIMFAP IFIN-HH</th>
<th>CICC LIT-JINR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>$\sigma_i = 1$</td>
</tr>
<tr>
<td>10 000</td>
<td>0.038</td>
</tr>
<tr>
<td>20 000</td>
<td>0.017</td>
</tr>
<tr>
<td>30 000</td>
<td>0.004</td>
</tr>
<tr>
<td>40 000</td>
<td>0.002</td>
</tr>
<tr>
<td>50 000</td>
<td>0.001</td>
</tr>
<tr>
<td>60 000</td>
<td>0.0002</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 4 – Fitting SIMFAP computing time data with second degree polynomials (7) under $\sigma_i = 1$ (left) and $\sigma_i = \sqrt{f_i}$ (right).

Fig. 5 – Same as Fig. 4 for third degree fitting polynomials (7).
4. TWO CASE STUDIES ON SIMFAP CLUSTER

Another useful approach to the cluster performance assessment is the investigation of the dependence of the output parameters on the number of processors involved in the solution of given fixed-size problems. In Fig. 8 we report outputs of two such case studies: (i) matrix multiplication ($N = 1000$); (ii) the calculation of interaction electrostatic potentials in “protein-liquid” systems.

The second problem uses a multigrid algorithm, based on the continuous analog of Newton’s method, for the numerical solution of boundary value problems for the nonlinear Poisson-Boltzmann equation,

$$
-\nabla \cdot [\varepsilon(r) \nabla u(r)] + R^2 \sinh[u(r)] = \frac{4\pi e^2}{k_B T} \sum_{\alpha=1}^{N} \delta(r - r_\alpha) \quad \text{for} \quad u(\infty) = 0.
$$

(6)
Fig. 8 – Computing time (left) and Acceleration (right) of parallel computer solution of two problems: (i) matrix multiplication (open circles); (ii) the calculation of interaction electrostatic potentials in “protein-liquid” systems (full triangles). The interrupted straight line provides the $p/2$ plot, above which the experimental data are placed.

Here $u(r) = e \psi(r)/(k_B T)$ is the dimensionless electrostatic potential. The quantity $\psi(r)$ denotes the electrostatic potential at a field position $r$, $e$ is the electron charge, $k_B$ is the Boltzmann constant, $T$ is the absolute temperature. The protein molecule itself represented by $N_p$ point charges $q_\alpha$ at the positions $r_\alpha$, yielding the delta functions in (6). The dielectric constant $\varepsilon(r)$ varies by about two orders of magnitude across the interface between the molecule and the surrounding solvent. The coefficient $\tilde{\kappa}$ is the modified Debye-Hückel parameter, which is proportional to the ionic strength of the solution. The coefficient $\tilde{\kappa}$ is also sharply varying at the interface between the solvent region and an ion exclusion layer surrounding the molecular surface.

Calculations have been done for the archetypal Crambin molecule [5], the PDB (Protein Data Bank) description of the 327 atoms of which can be found at [6, 7].

The scrutiny of the data shown in Fig. 8 points to the following features of the cluster operation:

- Under proper code implementation, the computing time needed to get parallel code solutions monotonically decreases with $p$, the number of parallel processors involved in the parallel solution. Indeed, the addition of further processors to the problem solution replaces the disk swap memory-transfer of temporary data in favour of the much more effective RAM-transfer.

- The fractional gain in the speed of execution gradually decreases with the increase of $p$, eventually resulting in a plateau-like shape (with a
small negative slope) behaviour of $T(p)$. This simply shows that the weight of the disk swap memory data transfer needing longer computation interrupts at the processor level is decreased towards zero, such that the weight of the inter-processor communications becomes sizable.

- At given $p$, the execution time depends on the inner problem complexity. The solution of the case study (ii) is simply more demanding than that of the case study (i).

- The *acceleration of the calculations*, defined as the ratio $T(1)/T(p)$, depends on the nature of the solved problem. In the case study (i), the larger weight of the inter-processor communications slows down the acceleration of the calculations.

5. CONCLUSIONS

The solution of models asking for very large numbers of knots in the discretization mesh needs the migration to high performance computing based on parallel cluster architectures. The acquisition of ready-to-use parallel computing facilities being beyond limited budgetary resources, the solution at IFIN-HH was to buy the hardware and the inter-processor network, and to implement by own efforts the open software concerning both the operating system and the parallel computing standard.

The present paper provides a report demonstrating the successful solution of these tasks. The implementation of the well-known HPL Benchmark points to the effective and reliable operation of the cluster.

The comparison of HPL outputs obtained on parallel clusters of different magnitudes shows that there is an optimum range of the order $N$ of the linear algebraic system over which a given parallel cluster provides *optimum parallel solutions*. For the SIMFAP cluster, this range can be inferred to correspond to about $(1 \pm 2) \times 10^4$ linear algebraic equations.

For an algorithm of polynomial complexity $N^a$, the task sharing among $p$ processors within a parallel solution mainly follows an $(N/p)^a$ behaviour under peak performance achievement. Thus, while the problem complexity remains the same, a substantial decrease of the coefficient of the leading order of the polynomial complexity is achieved.

Acknowledgments. The present work was done within the Hulubei-Meshcheryakov programme of cooperation Romania–LIT-JINR. Romanian authors acknowledge partial financial support from the Romanian Authority for Scientific Research (Project 7/2006 – SIMFAP), A. Ayriyan, E. Hayryan and O. Streletsova acknowledge partial financial support from RFBR 05-01-00645 and RFBR-BRFBR 06-01-81014 Grants.
APPENDIX A

Let \( f : [a, b] \rightarrow \mathbb{R} \), \( f = f(x) \), denote a polynomial dependence to be determined from a sampling \( \{ x_i \in [a, b] \mid i = 0, 1, \ldots, n \} \) of exactly known abscissas, where the values \( \{ f_i \mid i = 0, 1, \ldots, n \} \) have been measured with the a priori uncertainties \( \{ \sigma_i \mid i = 0, 1, \ldots, n \} \).

We seek \( f(x) \) as a discrete polynomial,

\[
    f_m(x_i) = \sum_{k=0}^{m} a_k p_k(x_i), \quad m < n, \tag{7}
\]

spanned by the set \( \{ p_k(x_i) \mid k = 0, 1, \ldots, m \mid i = 0, 1, \ldots, n \} \) of basis monic polynomials satisfying the weighted orthogonality conditions

\[
    \sum_{i=0}^{n} p_k(x_i) p_l(x_i)/\sigma_i^2 = \delta_{kl} \sum_{i=0}^{n} [ p_k(x_i)/\sigma_i ]^2 = \delta_{kl} \| p_k \|^2. \tag{8}
\]

Given the set of residuals

\[
    z_{i,m} = f_i - f_m(x_i), \tag{9}
\]

we build the \( L_2 \)-functional

\[
    S(a_0, \ldots, a_m) = \sum_{i=0}^{n} (z_{i,m}/\sigma_i)^2, \tag{10}
\]

the minimization of which yields the coefficients \( a_k \),

\[
    a_k = \sum_{i=0}^{n} f_i p_k(x_i)/\sigma_i^2 \| p_k \|_2, \quad k = 0, 1, \ldots, m. \tag{11}
\]

The basis polynomials \( p_k(x_i) \) are got by Gram-Schmidt orthogonalization,

\[
    p_k(x_i) = \delta_i^k - \sum_{l=0}^{k-1} c_{kl} p_l(x_i); \quad c_{kl} = \sum_{i=0}^{n} [ \delta_i^k p_l(x_i)/\sigma_i^2 ]^2, \quad k = 1, 2, \ldots, m, \tag{12}
\]

\[
    \delta_i = x_i - \bar{x}, \quad \bar{x} = \frac{1}{n+1} \sum_{i=0}^{n} x_i, \tag{13}
\]

with the initial condition \( p_0(x_i) = 1, \ \forall \ i \in \{ 0, 1, \ldots, n \} \).

The dispersion of the fitted values \( f_m(x_i) \) around the measured ones \( f_i \),

\[
    s_m = \left\{ \frac{1}{m(m-1)} \sum_{i=0}^{n} \left( \frac{z_{i,m}}{\sigma_i} \right)^2 \right\}^{1/2}, \tag{14}
\]
provides the $L_2$-measure of the quality of the fitting polynomial $f_m(x_i)$.

To fix the optimal value of the degree $m$ of the fitting polynomial (7), we ask the fulfilment of either of the following two criteria:

1. The two successive sequences of residuals, $\{z_{i,m-1} \mid i = 0, 1, \ldots, n\}$ and $\{z_{i,m} \mid i = 0, 1, \ldots, n\}$, represent pure noise, i.e., in both sequences there are no more than two successive residuals of the same sign [4].
2. The sequence $\{z_{i,m} \mid i = 0, 1, \ldots, n\}$ represents pure noise, while $s_m < \tau$, where we have taken for $\tau$ the five percent criterion ($\tau = 0.05$).

REFERENCES

1. A. Petitet, R. C. Whaley, J. Dongarra, A. Cleary, HPL – A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers
   +http://www.netlib.org/benchmark/hpl/