Application of Algebraic Approximation to Three Dimensional Multibody Coulomb Problem: Implementation of GPGPU

Shun-ichi OIKAWA, Koichiro HIGASHI and Yutaka MATSUMOTO
Graduate School of Engineering, Hokkaido University, N-13, W-8, Sapporo 060-8628, Japan
(Received 7 December 2009 / Accepted 5 March 2010)

The algebraic model (ALG) proposed by the authors has sufficiently high accuracy in calculating the motion of a test particle with all the field particles at rest. When all the field particles are moving, however, the ALG has relatively poor prediction ability on the motion of the test particle initially at rest. Nonetheless, the ALG approximation gives a good results for the statistical quantities, such as variance of velocity changes or the scattering cross section, for a sufficiently large number of Monte Carlo trials. We have implemented a graphics processing unit (GPU) using NVIDIA’s CUDA architecture into the ALG scheme for Coulomb multibody problems. For N=28-body problem, the ALG calculations on the GPU is several times faster than on a typical CPU. The achieved speedup ratios on an NVIDIA GTX-285 are 10.5 and 2500 against the ALG-CPU and the DIM-CPU, respectively both on an Intel Celeron @3.06 GHz.

© 2010 The Japan Society of Plasma Science and Nuclear Fusion Research

Keywords: multibody problem, algebraic approximation, diffusion, General Purpose Graphics Processing Unit (GPGPU)
DOI: 10.1585/pfr.5.S2026

1. Introduction

Since it is difficult to rigorously deal with multibody Coulomb collisions, the current classical theory \cite{1, 2} considers them as a series of temporally-isolated binary Coulomb collisions within the Debye sphere. We have proposed the algebraic analysis approximation (ALG) for Coulomb multibody interactions \cite{3–5}. When all the field particles are moving, the ALG has relatively poor prediction ability \cite{4, 5} on the two-dimensional motion of the test particle initially at rest. Nonetheless, the ALG approximation gives excellent results for the statistical quantities \cite{5}, such as variance of velocity changes or the scattering cross section, for a sufficiently large number of Monte Carlo trials.

In addition, the ALG scheme conserves the angular momentum, as well as linear momentum, of the entire system. The conventional Coulomb collision models, such as the Takizuka-Abe binary collision model \cite{6}, treats Coulomb interactions only in the velocity space. These models cannot be applied to problems in which the numerical conservation of total angular momentum is crucial. One of such problems includes the spontaneous plasma rotation \cite{7, 8}.

In this paper, we will report implementation of the general purpose graphics processing unit (GPGPU) using CUDA \cite{9} architecture into our ALG scheme. CUDA is a general purpose parallel computing architecture developed by NVIDIA Corporation on several NVIDIA graphics processing units (GPUs).

author’s e-mail: oikawa@qe.eng.hokudai.ac.jp

2. Algebraic Approximation

As shown in Fig. 1 the scattering angle, \( \chi = \pi - 2 \theta \), for a binary interaction is given by \( b = b_0 \tan \theta \), where \( b \) is the impact parameter, \( b_0 \equiv e^2/4\pi \epsilon_0 \mu g_0^2 \) corresponds to \( \chi = \pi/2 \) scattering, and \( g_0 \) the initial relative speed at \( r = \infty \) and \( \theta = -\theta_0 \). The angular component of the equation motion gives the well-known invariant of

\[
\frac{r^2}{dr^2} = \text{const} = \frac{b_{0 \theta}}{r},
\]

and the radial component is given by

\[
\frac{d^2 r}{dr^2} = \frac{g_0^2 b_0}{r^2} \left( 1 + \frac{b_0}{r} \tan^2 \theta_0 \right).
\]

The first term in the parentheses on the right hand side of Eq. (2) stands for the Coulomb force \( F_C \propto r^{-2} \). This force is much smaller for small angle scattering, i.e., \( \chi \ll 1 \), than the second term. This latter force scales as \( F_s \propto r^{-3} \) and results from the conservation of angular momentum Eq. (1). At the closest point \( r_{\text{min}} = r (\theta = 0) \) shown in Fig. 1, we have

\[
\frac{b_0 \tan^2 \theta_0}{r_{\text{min}}} = \frac{2}{\chi} \gg 1.
\]

As shown above, the \( r \)-dependence of \( F_s \propto r^{-3} \) is steeper than that of \( F_c \propto r^{-2} \), the change in momentum \( p \equiv \mu v \) is almost due solely to \( F_s \) near \( r = r_{\text{min}} \). As a consequence, the exact hyperbolic (or occasionally elliptic) trajectory for the particle can be approximated as a broken line with an impulse force of

\[
\mu \Delta v = 2 \mu b_0 \cos \theta_0 e_x = 2 \mu b_0 \sin \frac{\chi}{2} e_x,
\]
2.1 Coordinate transformation

In order to apply the above binary interaction approximation (ALG) to three-dimensional multibody cases, first we seek for a field particle that gives the test particle an impulse force at the earliest time. For this purpose, it is convenient to transform the coordinate system from the original three dimensional \((x, y, z)\) coordinates to a two dimensional \((\xi, \eta)\) ones, in such a way that the third axis, say \(\zeta\), is set parallel to the angular momentum \(L = r \times \mu g = Le_\zeta\), and the relative velocity \(g = (g_\xi, g_\eta) = (0, g) = ge_\eta\), as shown in Fig. 2.

Then the relative position \(r \equiv r_i - r_j\) between particle-\(i\) and particle-\(j\) has a \(\xi\)-, an \(\eta\)-, and a \(\zeta\)-coordinates of

\[
\xi_{ij} = r \cdot e_\xi, \quad \eta_{ij} = \frac{r \cdot g}{g}, \quad \zeta_{ij} = 0,
\]

where the unit vector \(e_\xi\) is defined as \(e_\xi \equiv e_\zeta \times e_\eta \equiv L/L \times g/g\). When \(\eta_{ij} < 0\) the two particles are approaching, while they are moving away for \(\eta_{ij} > 0\).

The particle moves in the direction parallel to the \(\eta\)-axis with a constant velocity of \(g\), and is to interact at \((\xi_{ij}, 0)\) with this field particle in a time interval of

\[
\Delta t_{ij} = \frac{\eta_{ij}}{g}.
\]

Accordingly, the field particle that the test particle is given an impulse force at the earliest time has the smallest positive \(\Delta t_{ij}\), i.e.,

\[
\Delta t_{\text{min}} \equiv \min \left(\max \left(0, -\frac{\eta_{ij}}{g}\right)\right), \quad \text{for } 1 \leq i, j \leq N.
\]

We have ignored the effect of field particles with \(\eta_{ij} > 0\), since the interaction is completed at \(\eta = 0\) in our approximation. In other words, such field particles have already interacted with the test particle in the past.

When the test particle moves to the position of \((\xi_{ij}, 0)\), it changes the relative velocity by \(\Delta g_{ij}\) as

\[
\Delta g_{ij} = 2g \sin \frac{\chi_{ij}}{2} e_\zeta, \quad \frac{\chi_{ij}}{2} = \arctan \frac{b_0}{\xi_{ij}},
\]

where the pair \(i\) and \(j\) satisfies Eq. (7), and we have approximated that the impact parameter is given by \(b = \xi_{ij}\) in Eq. (4) as shown in Fig. 2. Thus, in the \((\xi, \eta)\) coordinate system, the field particle position \(\xi_{ij}\) and \(\eta_{ij}\) correspond to the velocity change \(\Delta g_{ij}\) and the time of the interaction \(\Delta t_{ij}\), respectively.

2.2 Changes in position and momentum

In an \(N\)-body problem, changes in the position of each particle-\(I\), \((1 \leq I \leq N)\), during the time interval \(\Delta t_{ij}\) for a interacting particle pair \((i, j)\), is given by

\[
\Delta r_I = v_I \Delta t_{ij},
\]

irrespective of whether they are interacting, \(I \in [i, j]\), or not, since they are assumed to be moving straight in the ALG approximation.

On the other hand, changes in linear momenta of the interacting particle pair \((i, j)\), during \(\Delta t_{ij} = \Delta t_{\text{min}}\), are given by

\[
\Delta p_i \equiv m_i \Delta v_i = +\mu_{ij} \Delta g_{ij},
\]

\[
\Delta p_j \equiv m_j \Delta v_j = -\mu_{ij} \Delta g_{ij},
\]

since \(\mu_{ij} = \mu_{ji}\) and \(\Delta g_{ij} = -\Delta g_{ij}\) by definition. Therefore the ALG approximation conserves the binary momentum,
Relative errors in invariants with the ALG for the number of particles \(N = 28, 1332,\) and 9292.

<table>
<thead>
<tr>
<th>(N)</th>
<th>(#) of particles</th>
<th>Relative errors in (P)</th>
<th>(L)</th>
<th>(E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>(3 \times 10^{-16})</td>
<td>(4 \times 10^{-16})</td>
<td>(2 \times 10^{-7})</td>
<td></td>
</tr>
<tr>
<td>1,332</td>
<td>(4 \times 10^{-16})</td>
<td>(3 \times 10^{-14})</td>
<td>(7 \times 10^{-7})</td>
<td></td>
</tr>
<tr>
<td>9,262</td>
<td>(3 \times 10^{-15})</td>
<td>(2 \times 10^{-13})</td>
<td>(4 \times 10^{-7})</td>
<td></td>
</tr>
</tbody>
</table>

\(p_{ij} \equiv p_i + p_j, \) since, from Eqs. (11) and (12),

\[
\Delta p_{ij} = \Delta p_i + \Delta p_j = 0,
\]

in the system in principle. Furthermore, the interaction between \(i\) and \(j\) conserves the binary angular momentum, \(L_{ij} \equiv r_{ij} \times p_{ij},\) as

\[
\Delta L_{ij} \equiv \Delta r_{ij} \times p_{ij} + (r_{ij} + \Delta r_{ij}) \times \Delta p_{ij} = 0,
\]

since both cross products are those between two vectors with the same direction, as shown in Fig. 2.

Any other particle-\(k\) \((k \neq i, j)\) does not change its velocity, i.e.,

\[
\Delta p_k = m_k \Delta v_k = 0, \quad \Delta L_k = v_k \Delta r_k \times p_k = 0,
\]

since they are not interacting during the interval of \(\Delta t_{ij}.\)

We have, thus far, shown above that the binary interaction between particles \(i\) and \(j\) conserves linear and angular momenta, i.e., \(p_{ij}(t + \Delta t_{ij}) = p_{ij}(t)\) and \(L_{ij}(t + \Delta t_{ij}) = L_{ij}(t)\). It is easy to show the total linear momentum and total angular momentum of the entire system are conserved with our model. Table 1 lists the relative errors in invariants with the ALG for the number of particles \(N = 28, 1332,\) and 9292. The invariants include the linear momentum \(P = \sum_{i=1}^{N} m_i p_i\), the angular momentum \(L = \sum_{i=1}^{N} r_i \times m_i p_i\), and the total energy \(E = \sum_{i=1}^{N} m_i \gamma_i^2 / 2 + \sum_{i} \sum_{j} e_i e_j / 4\pi \varepsilon_0 r_{ij}\) of the entire system. The relative errors in the total momenta \(P\) and \(L\) are due to the finite-bit calculation on the computer. The relative errors in total energy \(E,\) of order of \(10^{-6}\) are large compared with those of the total momenta, since most particles move freely ignoring the interparticle forces in our model.

### 3. Test Calculation

In the following test calculations, protons in a hydrogen plasma with temperature of \(T = 10\) keV and number density of \(n = 10^{20} \text{m}^{-3}\) will be analyzed.

#### 3.1 Convergence with \(N\)

Figure 3 shows the \(N\)-dependence on changes in position \(\Delta r\) and in velocity \(\Delta v\) of a test particle. From the figure, changes in position converge at around \(N \sim 700,\) and those in velocity are almost independent from \(N.\) Therefore, in order to make statistical calculations concerning velocity changes, such as the scattering cross section, the number of particles \(N\) needs not be large; \(N = 28\) will be adopted in the next section.

#### 3.2 Variance in velocity changes

Figure 4 shows the variance of change in velocity,

\[
\langle \Delta v^2 \rangle \equiv \frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} \frac{1}{N} \sum_{i=1}^{N} (\Delta v_i)^2,
\]

of all the particles for three-dimensional Coulomb \(N = 28\)-body problem, where \(N_{MC}\) stands for the number of Monte Carlo trials. With the direct integration method, DIM, calculation ends up to the number of Monte Carlo trials of \(N_{MC} = 10^7,\) since it needs much more CPU times than the ALG. The agreement between the ALG and the DIM is excellent. As the DIM in this study, we have used the six-stage fifth-order Runge-Kutta-Fehlberg method known as the RKF65 [10, 11] with the absolute numerical error tolerance of \(10^{-6}.\) Several jumps seen in Fig. 4, such as those at \(N_{MC} \sim 7 \times 10^5,\) \(9 \times 10^5,\) and \(3 \times 10^6\) are due to close encounters, i.e., large angle scattering with \(\Delta t_{ij} / \gamma_0 \sim O(1).\) If a head-on collision of \(\Delta g_{ij} = 2 \gamma \sim 2 \gamma_0\) between the pair \((i, j)\) occurred during the next, i.e., the \(5 \times 10^{10} + 1\)st, Monte Carlo trial in this 28-body system, it alone could raise the dimensional variance by

\[
\frac{2 \times \Delta v^2_{i}}{28 \times (5 \times 10^{10} + 1)} \sim 0.14 \times 10^{-11} \gamma_{0}^2.
\]

Thus the variance by using the ALG, at the number of Monte Carlo trials \(N_{MC} = 5 \times 10^{10},\) can be expressed as

\[
\langle (\Delta v_{ALG})^2 \rangle \sim (1.88 \pm 0.14) \times 10^{-11} \gamma_{0}^2.
\]
Fig. 4 Normalized variance of velocity changes for $N = 28$ by using the ALG with $N_{MC} = 5 \times 10^{10}$, while up to $N_{MC} = 10^{9}$ for the DIM.

Table 2 CPU times for $N = 28$ with and without GPU.

<table>
<thead>
<tr>
<th>method</th>
<th>device</th>
<th>CPU time [ms]</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIM</td>
<td>CPU</td>
<td>31.0</td>
<td>2500</td>
</tr>
<tr>
<td>ALG</td>
<td>CPU</td>
<td>0.13</td>
<td>10.5</td>
</tr>
<tr>
<td>ALG</td>
<td>GPU</td>
<td>0.0124</td>
<td>1</td>
</tr>
</tbody>
</table>

The most CPU-time-consuming part of the ALG scheme is finding $\Delta t_{\text{min}}$ in Eq. (7), which can be executed in parallel on GPUs. As shown in Table 2, CPU/GPU time for $N = 28$ by using the ALG with $N_{MC} = 5 \times 10^{10}$ is 180 hr (6.4 $\times$ 10$^{5}$ sec) on a GPU; NVIDIA GTX-285 with the RAM of 1 GB, the number of cores of 240 at the clock rate of 1.48 GHz. The average calculation time on the GPU is around 0.0124 ms per one ALG calculation, so that the speedup ratios of the GPU calculation are 10.5 and 2500 against the ALG-CPU and the DIM-CPU, respectively.

3.3 CPU time with and without GPU

Figure 5 shows the CPU time for the ALG and the DIM. The number of particles dependence of the CPU time for the ALG approximation, $\tau_{\text{CPU}}^{\text{ALG}}$, is in proportion to $N^{3.02}$, as

$$\tau_{\text{CPU}}^{\text{ALG}} \sim 5.56 \times 10^{-9} N^{3.02}. \quad (20)$$

Similarly, $\tau_{\text{CPU}}^{\text{DIM}}$ for the DIM is:

$$\tau_{\text{CPU}}^{\text{DIM}} \sim 3.06 \times 10^{-5} N^{2.34}. \quad (21)$$

from which the ALG is suited for multibody problems for moderate number of particles of $N \leq 3 \times 10^5$. Also depicted in Fig. 5 is the calculation time, $\tau_{\text{CPU}}^{\text{DIM}}$ on a GPU, specifically NVIDIA GTX-295, which scales as

$$\tau_{\text{CPU}}^{\text{DIM}} \sim 1.88 \times 10^{-9} N^{2.75}. \quad (22)$$

The ALG on the GPU is suited for multibody problems for huge numbers of particles of $N \leq 2 \times 10^{10}$ which is several orders of magnitudes larger than the total number of particles in the Debye sphere in typical fusion plasmas.

4. Conclusion

We have implemented a graphics processing unit (GPU) using NVIDIA’s CUDA architecture into the ALG scheme for Coulomb multibody problems. For $N$=28-body problem, the ALG calculations on a GPU is several times faster than on a typical CPU. The achieved speedup ratios on NVIDIA GTX-285 are 10.5 and 2500 against the ALG-on-CPU and the DIM-on-CPU, respectively both on an Intel Celeron @3.06 GHz.

Acknowledgement

The authors would like to thank Dr. A. Wakasa and Prof. M. Itagaki for their fruitful discussions on the subject. The author would also acknowledge the continuous encouragement of the late Prof. T. Yamashina. Part of the GPU/CUDA coding was done by Mr. R. Ueda. This research was partially supported by a Grant-in-Aid for Scientific Research (C), 21560061.