OPTIMIZATION OF THE MULTIPOLE TO LOCAL TRANSLATION OPERATOR IN THE ADAPTIVE FAST MULTIPOLE METHOD

S. Abeyratne[^], B. Erdelyi[*,^]
[^]Department of Physics, Northern Illinois University, DeKalb, IL 60115
*Physics Division, Argonne National Laboratory, Argonne, IL 60439

Abstract

The Fast Multipole Method (FMM) is an accurate and fast way to calculate potentials/fields created by a very large number of particles. The runtime of the FMM is significantly less than that of the pairwise calculation if the particle number, N, is sufficiently large. Two major parts in the FMM are the upward pass and the downward pass. The upward pass calculates multipole expansions and then performs multipole-to-multipole translations. The downward pass calculates multipole-to-local expansions and local-to-local expansions. The multipole-to-local translation in the downward pass is the most time consuming translation in the FMM. In order to make the FMM more efficient, we sought to minimize the time taken by the multipole-to-local translation. The promising and practical strategy to minimize the multipole-to-local translation time is to replace the 3D multipole-to-local translation with a 1D multipole-to-local translation in conjunction with rotations of the coordinate axes. In this paper we show how to perform the 1D multipole-to-local translation and the time comparisons between the two FMM variants.

INTRODUCTION

The traditional method to calculate the Coulomb interaction between charged particles is pairwise calculation. The computational power needed for this method is of the order O(N^2) and it is not practical if the number of particles is very large. The FMM can reduce the computational power to O(N).

The details of the two major components of the FMM, data structuring [1, 2] and calculation of the potential/field [3], are discussed in our previous papers. If the number of particle is large, the data structuring time scales linearly [2]. We have also shown that the total runtime, the addition of data structuring time and the time taken to calculate the potential, scales linearly with the number of particles, if the number is large enough [3].

The major components of runtime with optimized settings are the time taken by upward pass (~3%), the downward pass (~50%) and the direct calculation in the final summation (~47%). To decrease the runtime we need to decrease the downward pass time or the direct calculation time. As the direct calculation time solely depends on the optimum q (the parameter q has its usual meaning, the maximum number of sources allowed in the neighbourhood of a given target), it is no longer possible to reduce the direct calculation time. Hence, we focus on reducing the downward pass time. Further analysis showed that the multipole-to-local translation (M2L) used in the downward pass is the bottleneck. This paper places emphasis on the M2L operator and shows that the FMM runtime can be significantly reduced if the 3D M2L translation is replaced by the 1D M2L translation.

ALGORITHM

In the data structuring we create the C-forest (the collection of trees) and the D-tree (parent-child relations of all boxes which contain target particles) [1, 2] at the optimum q. Once the octree-type data structure is created we can implement the FMM proper to calculate the potentials/fields.

The FMM implementation consists of two key components, the upward pass and the downward pass [3]. The downward pass starts at level two and it has three steps. The first step is the M2L translation of the multipole expansions of the boxes in the interaction list of a particular box or leaf node, b. In the second step, we move to the next finer level and compute the local expansion of child nodes of b around their centers from the multipole expansions of the boxes in the interaction list of child nodes. In the third step, we re-expand the result in the first step to evaluate the contribution from the local expansion of the parent box and add that to the result obtained in the second step. In order to complete the downward pass the above three steps must be repeated until we reach the finest level.

After the upward pass and downward pass we move to the final summation [3]. In the second step of the final summation we calculate the potential at each target in b due to the sources in the neighbourhood of b. This is a pairwise calculation. We have noticed that the M2L translation in downward pass and direct calculation in final summation take a considerably longer amount of time compared to all the other operations in the FMM process. Since this pairwise calculation is unavoidable and we cannot reduce that time it becomes apparent that the FMM runtime can be further reduced by optimizing the M2L translation operator. At higher orders M2L translation takes more than 50% of the total FMM runtime.

In this translation, we perform a 3D M2L translation in all three directions, x, y and z. In 3D, the number of boxes in the interaction list varies from 27 to 189. Therefore, the M2L translation takes a longer time. If we rotate the coordinate system of the box with the multipole expansion to be translated in such a way that the z-axis of the box aligns with the line joining the centers of the box and the interaction list boxes, the 3D transformation of the M2L becomes a 1D transformation. Hence, the time taken

ISBN 978-3-95450-138-0

for M2L translation can be considerably reduced as shown in Table 1 and 2.

1D M2L TRANSLATION

There are four coordinate systems involved in changing 3D M2L translation to 1D.

- Original multipole expansion in $(X, Y, Z)_{m,\,o}$
- Rotated multipole expansion in $(X, Y, Z)_{m,\,r}$
- Original local expansion in $(x, y, z)_{l,\,o}$
- Rotated local expansion in $(x, y, z)_{l,\,r}$,

where the subscripts l, m, o, and r denote local, multipole, original, and rotated, respectively.

The rotation about any arbitrary axis $\hat{\boldsymbol{\omega}}$ by an angle θ can be described by the matrix R,

$$R = \begin{pmatrix} t\omega_x^2 C & t\omega_x\omega_y - S\omega_z & t\omega_x\omega_z + S\omega_y \\ t\omega_x\omega_y + S\omega_z & t\omega_y^2 C & t\omega_y\omega_z - S\omega_x \\ t\omega_x\omega_z - S\omega_y & t\omega_y\omega_z + S\omega_x & t\omega_z^2 C \end{pmatrix}.$$

C, S and t denote cos θ, sin θ and (1- cos θ), respectively.

In order to make a 1D translation we need to rotate the coordinate system of the interaction list box such that the z axis lies along the line joining the center of an interaction list box and the center of a D-tree node, which is denoted by **r** (Fig. 1). In our implementation, we avoid this rotation if the z axis is already aligned with **r.** The unit vector along the axis of rotation, $\hat{\boldsymbol{\omega}}$, is then given by the cross product between **z** and **r**.

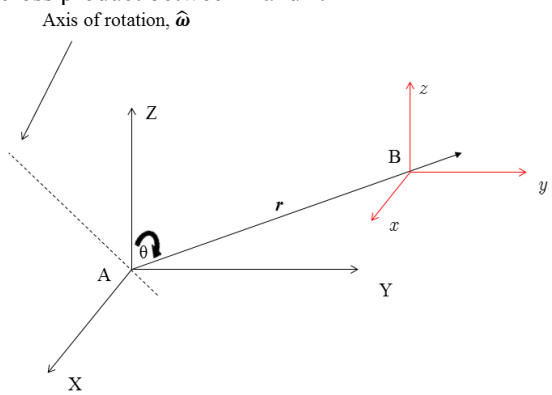

Figure 1: Rotation of the coordinate system.

If $\boldsymbol{r} = (\boldsymbol{r_x}, \boldsymbol{r_y}, \boldsymbol{r_z})$, $\hat{\boldsymbol{\omega}} = (\boldsymbol{\omega_x}, \boldsymbol{\omega_y}, \boldsymbol{\omega_z})$ and $\boldsymbol{z} = (0,0,1)$, then $\omega_x = -\frac{r_y}{r}$, $\omega_y = \frac{r_x}{r}$, $\omega_z = 0$.

Once the coordinate system is rotated, we transfer the multipole expansion into a local expansion. Finally, this rotated local expansion must be rotated back to the original coordinate system. This process is explained below.

The relationship between rotations can be given as follows:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_{m,r} = R \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_{m,o} \text{ and } \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{l,r} = R \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{l,o}.$$

The translation of the multipole expansion M_o to local expansion L_o can be expressed as a composition between M_o and M2L.

$$L_o \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{l,o} = M_o \circ M2L_o \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{l,o}.$$

Since r is the distance between the centers of the boxes (invariant under R), we can establish the following relationships:

$$\frac{1}{r^2} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_{m,o} = M2L_o \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{l,o};$$

Similarly, the new operator $M2L_n$ can be introduced as,

$$\frac{1}{r^2} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_{m,r} = M2L_n \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{l,r},$$

where the subscript n stands for *new*.

The two operators, $M2L_o$ and $M2L_n$, must give the same local expansion around centers of the nodes. Hence,

$$L_n \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{l,o} = L_o \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{l,o}.$$

By further simplification, it can be shown that

$$L = M \circ R^{-1} \circ M2L_o \circ R,$$

where $L_o = L_n = L$ and $M_o = M$. Since the rotation matrix is orthogonal $R^{-1} = R^T$.

PERFORMANCE ANALYSIS

The 3D M2L translation requires a dense function composition operator in three variables while the 1D M2L translation requires a sparse function composition operator in the variables x and y and a dense operator in z. The 3D translation is computationally more expensive compared to the 1D translation. Both translations are order-dependent and hence the difference in cost between these translations becomes order-dependent. The two rotations needed for the 1D translation are considered order-independent overhead. Empirically, we notice that the time saving by employing the 1D operator instead of 3D operator becomes significant around order five, independent of the type of distribution. Therefore, based on our results, it is clear that below order five 3D M2L translation is advantageous, while above order five 1D translation is advantageous, independent of the distribution (Figs. 2 and 3). In order to achieve high accuracy, it is essential to perform calculations at higher orders [3, 4].

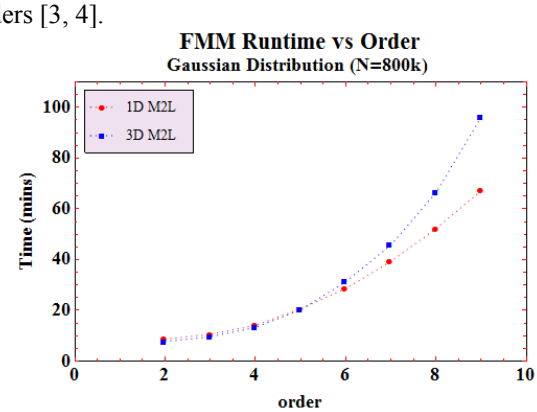

Figure 2: Runtime measured for normally distributed particles using 1D and 3D M2L operators.

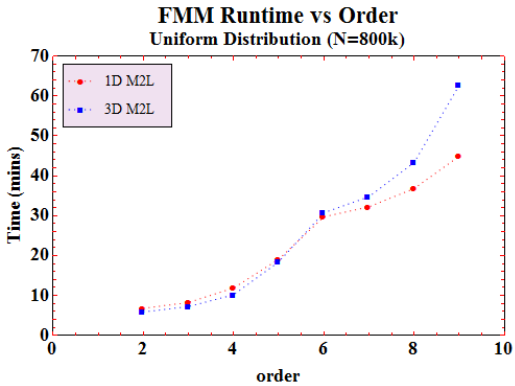

Figure 3: Runtime measured for uniformly distributed particles using 1D and 3D M2L operators.

Tables 1 and 2 summarize the runtime measured for normally and uniformly distributed particles, respectively, at different orders at optimum q (in each case identical sources and targets are used).

We studied the relationship between the FMM runtime (t) and the expansion order (p) in the region where the order is greater than five. For normally distributed particles (N=800k), t is proportional to the nth power of p and the value of n is 3.4 and 2.4 for 3D M2L and 1D M2L, respectively. For uniformly distributed particles (N=800k), the corresponding values are 7.2 and 5.5.

Table 1: Runtime measured for normally distributed particles using 1D and 3D M2L operators.

Order	Time(mins)	
	1D	3D
2	8.724	7.765
3	10.589	9.621
4	14.155	13.314
5	20.582	20.235
6	28.620	31.342
7	39.294	45.760
8	52.292	66.442
9	67.178	96.185

Table 2: Runtime measured for uniformly distributed particles using 1D and 3D M2L operators.

Order	Time(mins)	
	1D	3D
2	6.800	5.886
3	8.265	7.236
4	11.900	10.204
5	19.107	18.450
6	29.732	30.752
7	32.240	34.767
8	36.809	43.323
9	44.926	62.701

The runtimes are machine dependent. All the tests were run on a single core Intel® Core™ i5-2410M @ (2.30GHz) computer (no SSE). The machine had 8GB of RAM. The code for data structuring is written in C++ and compiled under Cygwin on Windows 7, and used the g++ compiler with optimization flags –O3 and –funroll-loops. The FMM is implemented in COSY Infinity [4].

SUMMARY

In this paper, we showed that the use of 1D M2L operator (for expansion orders greater than five) in the downward pass to calculate the potentials/fields reduces the overall FMM runtime in both normally and uniformly distributed particles. As shown in Table 3, when run at optimum q, 1D M2L operator saves the downward pass time by 40% for uniformly distributed particles and by 35% for normally distributed particles. Hence, the overall FMM runtime is reduced by 30% and 29% for normal and uniform distributions, respectively.

Table 3: Downward pass times at order 9 (N=800k).

Distribution	Dimension	Optimum q	Time (mins)
Uniform	1D	6000	9.40
	3D	6000	15.70
Normal	1D	12000	31.08
	3D	20000	47.75

ACKNOWLEDGMENT

This work was supported by the U.S. Department of Energy, Office of Nuclear Physics, under Contract No. DE-SC0005823.

REFERENCES

[1] Nail A. Gumerov and Ramani Duraiswami, “Fast Multipole Methods for the Helmholtz Equation in Three Dimensions,” Elsevier, 2004.

[2] S. Abeyratne, S.Manikonda and B. Erdelyi, “Optimal Fast Multipole Method Data Structures,” Conf. Proc. C1205201 (2012) 352-354.

[3] S. Abeyratne, S.Manikonda and B. Erdelyi, “A Novel Differential Algebraic Adaptive Fast Multipole Method,” Conf. Proc. (2013).

[4] H. Zhang and M. Berz, “The Fast Multipole Method in Differential Algebra Framework,” Nuclear Instruments and Methods A 645 (2011) 338.

ISBN 978-3-95450-138-0