JOURNAL OF COMPUTATIONAL PHYSICS 36, 271–279 (1980)

Note

On Evaluating Strategies for the Computation of DWBA Integrals

1. Introduction

Heavy ion reaction studies require exact calculations of transition amplitudes which involve two- or higher-dimensional radial integrals. The computation of these integrals is quite time-consuming because of two factors: the numerical approximation of the integral itself, and the number of such integrals required. Most distorted-wave-Born-approximation (DWBA) calculations follow the method of Austern *et al.* [1] or are adaptations, modifications, or approximations of it (see, for example, Refs. [2–4]). The inclusion of recoil effects, which is essential for high-energy heavy-ion interactions [4], affects both the dimensionality of the radial integrals, as well as the number of integrals required. The latter is due to the selection rules for angular momenta, introduced by the inclusion of recoil [3]. Because of computer storage limitations, the complete processing of a reaction requires the computation of various functions (such as bound state and optical model wave functions) and their storage in permanent files [3]. When the radial integrals are computed, the selection rules determine which table of function values must be retrieved from files. Since the tables needed for a specific integral are not necessarily stored sequentially, these reading operations contribute significantly to the overall computation time.

In this paper we present a method for analyzing the complexity of algorithms, and in particular of those involved in DWBA calculations. We explore certain computational schemes and show how they can be analyzed and evaluated, a priori, by determining for each one of them a "cost." Our purpose is not to give computational recipes for certain specific reaction theories, but to demonstrate the usefulness of evaluating computational schemes before choosing one for implementation, and to provide a method for doing it. For this reason we chose to illustrate this method by analyzing the computation of radial integrals for knock-out reactions, according to a rather simple model given by Park [5].

In Section 2 we describe a method of analysis by applying it to obtain the complexity for a straightforward "simple-minded" computation of radial integrals. In Section 3 we analyze two other strategies and we discuss the effect of rearranging certain of their subalgorithms.

0021-9991/80/080271–09$02.00/0

2. Analysis of Computational Schemes

We shall compare computational methods by obtaining for each of them an estimate of a quantity that is proportional to the computation time of the corresponding algorithms. We emphasize that we are interested in an a priori analysis of a computational method and not in an evaluation of the actual execution time. For this reason, it will suffice to define our quantity in terms of a model of computation which, although oversimplified, incorporates all the features of the method, but excludes any computation related to the control and management of the execution of the algorithm. We shall define as a straight-line algorithm [6] a sequence of assignment statements of the form $A \leftarrow B\omega C$, $A \leftarrow \mathscr{R}(B)$, or $A \leftarrow \mathscr{F}(B)$. Here A, B, and C are variable names, constants or subscripted variables, ω is one of the operators $+$, $-$, $*$ or $/$, $\mathscr{R}$ represents an input (read) operation from some external file, and $\mathscr{F}$ represents a function evaluation which for some reason we may prefer not to express explicitly in terms of operations ω. We assign to each operation, including the function $\mathscr{F}$, a cost parameter the value of which is proportional to the execution time for that operation or for the function, respectively. We define as the complexity of a straight-line algorithm the sum of the cost parameters of all assignment statements that constitute the algorithm. For the sake of notational convenience, instead of writing

$$\begin{aligned} A_1 &\leftarrow B_1\omega_1 C_1\,, \\ A_2 &\leftarrow B_2\omega_2 C_2\,, \\ &\vdots \\ A_n &\leftarrow B_n\omega_n C_n\,, \end{aligned}$$

we shall write $A_i \leftarrow B_i\omega_i C_i$ $(i = 1, 2, \ldots, n)$.

We consider now a direct knock-out reaction, which we represent symbolically as

$$a + A \to a + (C + b) \to (a + C) + b \to B + b.$$

Here a and b are the incident and outgoing particles (or clusters), respectively, A and B are the target and residual nuclei, respectively, and C is a heavy closed shell core which is considered to be part of the cluster structure of both the target and the residual nuclei. According to the model considered here, the core is assumed infinitely heavy and the radial integrals associated with the DWBA transition amplitude have the form [5]

$$\begin{aligned} I(l, l', l''; j, j'; L, L') = \int_0^\infty \int_0^\infty & r_a^2 dr_a\, r_b^2 dr_b\, \xi_{nL}(r_b)\, R^*_{n'l'j'}(r_b) \\ & \times \zeta^*_{n'L'}(r_a)\, R_{nlj}(r_a) f_{l''}(r_a, r_b), \end{aligned} \tag{2.1}$$

where $\zeta_{n'L'}(r_a)$ and $\xi_{nL}(r_b)$ are the radial parts of the bound state wavefunctions for particles a and b, respectively, and $R_{nlj}(r_a)$ and $R_{n'l'j'}(r_b)$ are elastic scattering wavefunctions which are solutions of the radial Schrödinger equation with optical potentials. The functions $f_{l''}(r_a, r_b)$ are the expansion coefficients of the direct interaction which for simplicity we consider here spin independent and therefore we can write

$$V_{ab}(\mathbf{r}_a, \mathbf{r}_b) = \sum_{l''} f_{l''}(r_a, r_b) P_{l''}(\cos\theta). \tag{2.2}$$

The Schrödinger equation can be solved numerically easier, if it is written in the form [8, 9]

$$u''(r) = A(r)\,u(r), \tag{2.3}$$

where $u(r) = r\psi(r)$. Here $\psi(r)$ is the solution of the Schrödinger equation in its canonical form. If each one of the wavefunctions of the integrand of (2.1) is expressed in terms of solutions of the corresponding modified equations of the form (2.3), the factors r_a^2 and r_b^2 are removed from (2.1). For the sake of simplicity we change the notation in an obvious manner, and we write the integral (2.1) in the form

$$I_{klm} = \int_0^{R_a}\int_0^{R_b} g^*(x)\,G_k(x)\,h(y)\,H_l^*(y) f_m(x, y)\,dx\,dy + T_{klm}, \tag{2.4}$$

where

$$T_{klm} = \int_{R_a}^{\infty}\int_{R_b}^{\infty} g^*(x)\,G_k(x)\,h(y)\,H_l^*(y) f_m(x, y)\,dx\,dy.$$

In practice the integral I_{klm} is taken equal to the first term of (2.4), after choosing appropriate values for R_a and R_b so that $|T_{klm}| < e$, where $e > 0$ is some acceptable bound for the truncation error committed. The values of R_a and R_b depend on the potentials of the Schrödinger equations used to generate the wavefunctions of the integrand of (2.4).

In order to approximate the integral (2.4), we apply a product rule [7], either of the Newton–Côtes type or of the Gauss type, and we obtain

$$I_{klm} = \sum_{i=1}^{a} c_i g^*(x_i)\,G_k(x_i) \sum_{j=1}^{b} c_j h(y_j)\,H_l^*(y_j) f_m(x_i, y_j) + E, \tag{2.5}$$

where E is the appropriate error term. The vectors $(c_1 g^*(x_1),\ldots, c_a g^*(x_a))^T$ and $(c_1 h(x_1),\ldots, c_b h(x_b))^T$ can be computed and stored once and for all in main storage. We shall represent their components by cg_i^* and ch_j, respectively. Likewise the vectors $(x_1,\ldots, x_a)^T$ and $(y_1,\ldots, y_b)^T$ are stored in main storage. Their values may be equally spaced or chosen according to the requirements of Gaussian integration. The optical model wavefunctions are stored as the vectors $G_k \equiv (G_{k1}, G_{k2},\ldots, G_{ka})^T$ and $H_l \equiv (H_{l1}, H_{l2},\ldots, H_{lb})^T$ in permanent (external) files and are retrieved by specifying the orbital angular momentum quantum number (k or l). Since a vector, e.g., G_k, is stored in internal form as one record, the corresponding input operation will be written in the form $G_k \leftarrow \mathscr{R}(k)$. For simplicity we shall consider only spinless particles. The inclusion of spin simply doubles the number of wavefunctions and multiplies the number of integrals to be computed, by a constant factor.

Our objective is the discussion of the computation process of the sums in Eq. (2.5). A straight-line algorithm for the numerical evaluation of one integral I_{klm} according to this equation is the following:

```
 1. $G_k \leftarrow \mathscr{R}(k)$
 2. $H_l \leftarrow \mathscr{R}(l)$
 3. $I_{klm} \leftarrow 0$
 4. $s \leftarrow 0$                                      ⎫
 5. $p \leftarrow cg_i * G_{ki}$                          ⎪
 6. $q \leftarrow ch_j * H_{lj}$         ⎫                ⎪
 7. $f \leftarrow \mathscr{F}(m, x_i, y_j)$ ⎬ $(j = 1, 2,..., b)$ ⎬ $(i = 1, 2,..., a)$.     (2.6)
 8. $q \leftarrow q * f$                 ⎪                ⎪
 9. $s \leftarrow s + q$                 ⎭                ⎪
10. $t \leftarrow p * s$                                  ⎪
11. $I_{klm} \leftarrow I_{klm} + t$                      ⎭
```

We shall represent the cost parameters for the operations $+$, $*$, $\mathscr{R}$, and $\mathscr{F}$, by α, μ, ρ, and f, respectively. We shall assign a zero cost parameter to initializations of the form $s \leftarrow 0$. The complexity for the above algorithm can be easily found to be

$$\begin{aligned} T(1) &= 2\rho + \{(2\mu + \alpha + f)b + 2\mu + \alpha\}a \\ &= 2\rho + (2\mu + \alpha + f)ab + (2\mu + \alpha)a. \end{aligned} \tag{2.7}$$

The complexity for the entire computation, i.e., for all the values of k, l, and m, compatible with the selection rules, is

$$T(N) = \sum_k \sum_l \sum_m T(1) = NT(1), \tag{2.8}$$

where N is the total number of integrals, which can be obtained in principle from the relation

$$N = \sum_{k=0}^{K} \sum_{l=0}^{L} \sum_{m=m_{\min}}^{m_{\max}} 1 \tag{2.9}$$

subject to the constraints $k + m + B =$ even and $l + m + A =$ even. Here A and B are the quantum numbers for the bound state functions, denoted in (2.1) by L and L', respectively [5]. From the triangle relations that must be satisfied simultaneously by (k, m, B) and (l, m, A) we obtain

$$m_{\min} = \max(|k - B|, |l - A|), \qquad m_{\max} = \min(k + B, l + A). \tag{2.10}$$

Due to these selection rules the summation of (2.9) is not easy to evaluate. We can, however, obtain a rather conservative bound on N [10] which is

$$N \leqslant \text{floor}(\tfrac{1}{2}(\lambda + 1)^2(L + 1) - \tfrac{1}{6}\lambda(\lambda + 1)(\lambda + 2)). \tag{2.11}$$

Here $\lambda = 2\max(A, B)$, floor(x) = integ(x) and we have taken $K = L$. Relation (2.11) shows that $N = O(L)$, i.e., the number N of integrals is linear in the largest number of partial waves; however, although the parameter λ is usually small, the factor $(\lambda + 1)^2/2$ that multiplies $L + 1$ can make the number of integrals quite large. Combining (2.7) and (2.11) and retaining the dominant terms only, we obtain

$$T(N) = O(\rho\lambda^2 L + f\lambda^2 L n^2), \tag{2.12}$$

where $n = \max(a, b)$. Knowledge of the precise values of ρ and f is not important here, especially since this is an a priori analysis. One may conjecture that ρ may be larger than f (say $\rho \simeq fn^2$) and therefore both terms should be retained in (2.12). In any case, comparisons of the relative efficiencies of computational schemes can be made by using relation (2.12) without any assumption on ρ and f. For example, a hypothetical method of time complexity $O(\rho\lambda^2 L + f\lambda^2 Ln)$ would be definitely superior to the method based on algorithm (2.6), whereas a method of time complexity $O(\rho\lambda^2 L^2 + f\lambda^2 Ln^2)$ would be inferior.

Algorithm (2.6) for the computation of one I_{klm} value is nested within the three loops controlled by k, l, and m. One may wonder whether removal of the read operations 1 and 2 from the inner loop would improve the complexity of the overall computation. The result can be seen to be

$$T(N) = \sum_{k=0}^{K}\left\{\rho + \sum_{l=0}^{L}\left\{\rho + \sum_{m=m_{\min}}^{m_{\max}}\left\{\sum_{i=1}^{a}\left(2\mu + \sum_{j=1}^{b}(2\mu + \alpha + f) + \alpha\right)\right\}\right\}\right\}.$$

Expansion of this expression and use of the selection rules implied by the summation process (2.9) yield the result

$$T(N) = O(\rho L^2 + f\lambda^2 L n^2) \tag{2.13}$$

which cannot be considered an improvement over (2.12) since λ is definitely smaller than L. On the contrary, for usual values of L, the cost implied by (2.13) is much

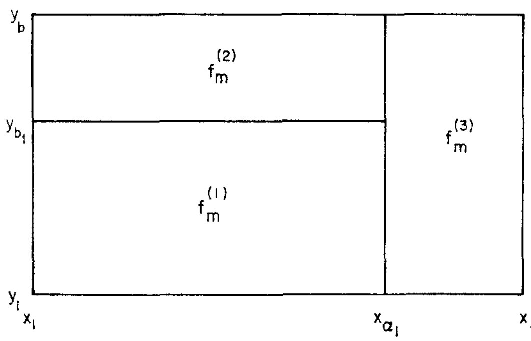

FIG. 1. Formulas (2.5) and (3.1) are strictly equivalent. One can be convinced by noticing the way the interval $[x_1, x_a] \times [y_1, y_b]$ is partitioned. The additional notation $f_m^{(p)}$, is being used to help keep in mind this partition.

higher than that implied by (2.12). This is due to the fact that by removing the read operation from the m-loop, we removed them also from the scope of the selection rules, and therefore they are executed for each value of k and l.

3. Improved Strategies

The fact that in formula (2.12) f multiplies the product Ln^2, which can be very large, indicates that the computing time for the execution of the function $\mathscr{F}$ greatly affects the complexity of the whole computation. From algorithm (2.6) it can be seen that the same value of this function is computed several times. One obvious way to decrease the computation time is to compute and store all function values $f_m(x_i, y_j)$ so that repetitions can be avoided. This, however, requires too much storage. A compromise can be reached by choosing two integers $a_1 < a$ and $b_1 < b$, and rewriting formula (2.5) for the numerical evaluation of the integral I_{klm}, in the form (see Fig. 1)

$$\begin{aligned} I_{klm} = &\sum_{i=1}^{a_1} cg_i^* G_k(x_i) \sum_{j=1}^{b_1} ch_j H_l^*(y_j) f_m^{(1)}(x_i, y_j) \\ &+ \sum_{i=1}^{a_1} cg_i^* G_k(x_i) \sum_{j=b_1}^{b} ch_j H_l^*(y_j) f^{(2)}(x_i, y_j) \\ &+ \sum_{i=a_1}^{a} cg_i^* G_k(x_i) \sum_{j=1}^{b} ch_j H_l^*(y_j) f_m^{(3)}(x_i, y_j), \end{aligned} \tag{3.1}$$

where $f_m^{(1)}(x, y) \equiv f_m(x, y)$ for $x_1 \leqslant x \leqslant x_{a_1}$, $y_1 \leqslant y \leqslant y_{b_1}$, $f_m^{(2)}(x, y) \equiv f_m(x, y)$ for $x_1 \leqslant x \leqslant x_{a_1}$, $y_{b_1} \leqslant y \leqslant y_b$, and $f_m^{(3)}(x, y) \equiv f_m(x, y)$ for $x_{a_1} \leqslant x \leqslant x_a$, $y_1 \leqslant y \leqslant y_b$. The function values $f_m^{(1)}(x_i, y_j)$ ($i = 1, 2,..., a_1$, $j = 1, 2,..., b_1$, $m = 0, 1,..., \min(K + B, L + A)$) are computed only once and stored as a three-dimensional array with elements F_{mij}. However, the function values $f_m^{(2)}(x_i, y_j)$ and $f_m^{(3)}(x_i, y_j)$ are evaluated as needed. Thus the first summation of Eq. (3.1) is obtained with an algorithm that is analogous to (2.6) in which, however, statement 7 is removed and statement 8 is replaced by the statement $q \leftarrow q * F_{mij}$. The values of the elements F_{mij} are computed by the straight-line algorithm

$$F_{mij} \leftarrow \mathscr{F}(m, x_i, y_j) \qquad \left\{ \begin{aligned} m &= 0, 1,..., \min(K + B, L + A) \\ i &= 1, 2,..., a_1 \\ j &= 1, 2,..., b_1 \end{aligned} \right\}$$

that is executed before the main algorithm. The two additional summations of Eq. (3.1) are computed with algorithms which are essentially similar to (2.6). The complexity for this computational scheme is easily obtained as in the case discussed in Section 2, and the corresponding bound is given by

$$T(N) = O(\rho\lambda^2 L + f\lambda^2 L(n^2 - n_1^2) + fLn_1^2), \tag{3.2}$$

where $n_1 \equiv \max(a_1, b_1)$. The improvement is due to the fact that the product $f\lambda^2 L$ is not multiplied anymore by n^2 but by $n^2 - n_1^2$. A far greater benefit can be derived from this method if the choice of integers a_1 and b_1 can be made so that $x_{a_1} \simeq R_a$ and $y_{b_1} \simeq R_b$ for a comprehensive class of nuclei and incident particle energies. We shall call such problems "medium size problems." In these cases Eq. (3.1) contains only the first summation and the corresponding complexity is

$$T(N) = O(\rho\lambda^2 L + fLn_1^2) \tag{3.3}$$

which represents a substantial improvement over both (2.12) and (3.2). Therefore, a system could be designed which would give a fast computation for medium size problems but would take additional time for large size problems.

An even better computational scheme results when, instead of using k and l as independent parameters, m is made independent, and k and l dependent through the selection rules. Indeed, from the triangle relations $\Delta(k, m, B)$ and $\Delta(l, m, A)$ imposed by the presence of the appropriate Clebsch–Gordan coefficients in the transition amplitude for the nuclear reaction [5], we obtain

$$| m - B | \leqslant k \leqslant m + B,$$
$$| m - A | \leqslant l \leqslant m + A$$

from which we deduce that for a given value of m there correspond several values of the parameters (or subscripts) k and l. Specifically, there are at most $2B + 1$ values of k and $2A + 1$ values of l. It is, therefore, advantageous to use two matrices of sizes $(2B + 1) \times a$ and $(2A + 1) \times b$, in order to store the vectors $G_k(\mathbf{x})$ ($| m - B | \leqslant k \leqslant m + B$) and $H_l(\mathbf{y})$ ($| m - A | \leqslant l \leqslant m + A$), respectively, for as long as they are needed, so that repeated read operations may be avoided. As the value of m is incremented, one of the vectors $G_k(\mathbf{x})$ is deleted and a new one is read into the aforementioned matrix. Something similar happens to the vectors $H_l(\mathbf{y})$. In practice, this data manipulation is conveniently achieved by operating the two storage matrices as circular queues [11] in which one vector constitutes one node (or record) of the queue [10]. The function values $f_m(x_i, y_j)$ do not have to be stored, since m is an independent parameter and takes its values sequentially. Only a matrix F of size $a \times b$ is needed for storing the function values of $f_m(x, y)$ for a given value of m. Writing a straight-line algorithm for this computational scheme is a rather straightforward task [10] and we shall not do it here. The complexity for such an algorithm is found to be

$$\begin{aligned} T(N) = {} & fab + 2\rho + \sum_{i=1}^{a} \left\{ \mu + \sum_{j=1}^{b} (2\mu + \alpha) + \mu + \alpha \right\} \\ & + 2\rho B + \rho(M - 2B) + 2\rho A + \rho(M - 2A) \\ & + \sum_{m=1}^{M} \left\{ fab + \sum_{k=|m-B|}^{m+B} \left\{ \mu\alpha + \sum_{l=|m-A|}^{m+A} \{(2\mu + \alpha)ab + (\mu + \alpha)\alpha\} \right\} \right\}, \end{aligned}$$

where

$$M \equiv \max(K + B, L + A) \leqslant \max(K, L) + \max(A, B).$$

Using the same definitions and bounds as before, we obtain

$$T(N) = O(\rho L + fLn^2) \tag{3.4}$$

which constitutes an improvement over all previous cases because the parameter λ^2 does not appear as a multiplier either of ρL or of fLn^2. For large values of n (e.g., $n > 100$), the first term in all complexity formulas can be neglected. At any rate, it is the absence of the factor λ^2 that makes the difference in the fourth case.

In order to obtain a feeling for how the four complexity formulas (2.12), (2.13), (3.2), and (3.4) compare, we shall assign certain plausible numerical values to the cost parameters. Thus we take $\mu = 1$ and $\rho = 1000$. The value of f depends on the form of the interaction potential and on the method of computation of the expansion coefficients in (2.2). We have estimated that a value of 10 is neither large nor small. We take $\lambda = 4$, which means that at least one of the bound states (initial or final) has the orbital angular momentum quantum number equal to 2, and $n_1/n = 70/100$. Then the four complexity formulas become, respectively,

$$\begin{aligned} T(N) &= O(16000L + 160Ln^2), \\ T(N) &= O(1000L^2 + 160Ln^2), \\ T(N) &= O(16000L + 85Ln^2), \\ T(N) &= O(1000L + 10Ln^2). \end{aligned}$$

The two extreme cases differ by a factor of 16 (i.e., λ^2) which is substantial.

4. Conclusions

The discussion of the previous section suggests that the improvements in efficiency of the last computational scheme are significant enough (especially for higher values of the quantum numbers L and L') to warrant the development of a program based on this scheme. Furthermore, it is correct to conjecture that in computations such as DWBA analysis of nuclear reactions or similar ones, it is advisible to avoid computational schemes that use several independent parameters (provided that this is possible). However, the most important conclusion from this discussion is the need for a theoretical analysis of the computational scheme used to solve a physical problem Obtaining the analytic expressions that represent the solution to a problem, and even identifying some numerical method for handling those expressions, does not constitute the complete solution. An a priori theoretical analysis of the whole computational process can provide invaluable information, concerning the relative costs of certain parts of the computation, the structuring of data, the retrieval of data from files, etc. The method of analysis used in this paper is very simple, but nevertheless sufficient to allow an evaluation of a computational scheme and the making of appropriate design decisions.

REFERENCES

1. N. AUSTERN, R. M. DRISKO, E. C. HALBERT, AND G. R. SATCHLER, *Phys. Rev.* **133B** (1964), 3.
2. A. J. BALTZ AND S. KAHANA, *Phys. Rev. Lett.* **29** (1972), 1267.
3. R. M. DEVRIES, *Phys. Rev. C* **8** (1973), 951.
4. K. S. LOW AND T. TAMURA, *Phys. Rev. C* **11** (1975), 789.
5. J. Y. PARK, *Prog. Theoret. Phys.* **30** (1963), 45.
6. A. V. AHO, J. E. HOPCROFT, AND J. D. ULLMAN, "The Design and Analysis of Computer Algorithms," Addison-Wesley, Reading, Mass., 1974.
7. P. J. DAVIS AND P. RABINOWITZ, "Numerical Integration," Blaisdell, Waltham Mass., 1967.
8. M. A. MELKANOFF, T. SAWADA, AND J. RAYNAL, "Nuclear Optical Model Calculations," Methods in Computational Physics (B. Alder, S. Fernbach and M. Rotenberg, Eds.), Vol. 6, pp. 2–79, Academic Press, New York, 1966.
9. J. M. BLATT, *J. Computational Phys.* **1** (1967), 382.
10. S. D. DANIELOPOULOS, unpublished Technical Memorandum, North Carolina State University, Computer Science, revised version (1978).
11. D. E. KNUTH, "The Art of Computer Programming, Vol. 1, Fundamental Algorithms," Addison-Wesley, Reading, Mass., 1973.

RECEIVED: September 19, 1978; REVISED: July 2, 1979

STYLIANOS D. DANIELOPOULOS*

North Carolina State University,
Department of Computer Science, Raleigh, North Carolina

* Present address: University of Ioannina, Department of Mathematics, Ioannina, Greece.