The Prism Algorithm for Two-Electron Integrals

PETER M.W. GILL AND JOHN A. POPLE
Department of Chemistry, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213

Abstract

A new approach to the evaluation of two-electron repulsion integrals over contracted Gaussian basis functions is developed. The new scheme encompasses 20 distinct, but interrelated, paths from simple shell-quartet parameters to the target integrals, and, for any given integral class, the path requiring the fewest floating-point operations (FLOPS) is that used. Both theoretical (FLOP counting) and practical (CPU timing) measures indicate that the method represents a substantial improvement over the HGP algorithm.

Introduction

Because of their large number, the evaluation and manipulation of two-electron integrals is the major difficulty in a Hartree-Fock calculation.

A. Szabo and N.S. Ostlund [1]

Over the decades, this realization has been the single most important driving force in formulating improvements to practical implementations of the Hartree-Fock self-consistent field (SCF) method. By 1980, several ingenious algorithms [2–4] for the evaluation of two-electron repulsion integrals (ERIS) were available, and since (in a conventional SCF calculation) the ERIS need be computed only once (after which they are stored and retrieved on each iteration of the SCF procedure), there appeared to be little to be gained by further improving the ERIS evaluation algorithms.

The balance, however, was shifted substantially when, in 1982, Almlöf and co-workers introduced the "direct SCF" method [5] in which ERIS are recomputed on each iteration of the SCF. Recently, too, the "direct" approach has been extended to MP2 calculations [5, 6]. Direct methods allow very large calculations to be performed without prohibitively large disk requirements, but, naturally, they cost much more than do their conventional analogs. Indeed, the cost of a direct SCF or direct MP2 calculation is essentially some multiple of the cost of the associated ERIS evaluation. Clearly, within such a framework, it is crucial that highly efficient methods for ERIS computation be utilized.

After lying dormant for several years, the study of novel ERIS algorithms was invigorated by the discovery of the Obara–Saika (os) recurrence relation [7] (which had been implicit in earlier work [8] by Schlegel) in 1986. Two years later, Head-Gordon and Pople (HGP) suggested [9] that the os methodology is improved
if the os recurrence relation is used in judicious combination with another that they called the horizontal recurrence relation. Recently, Hamilton and Schaefer showed that the addition of a third recurrence relation to the HGP scheme results in even greater efficiency \[10\], and we have argued that a hybrid of the McMurchie–Davidson (MD) scheme \[4\] and HGP methodology is superior to either \[11\].

Any ERI may be characterized by two quantities (to be defined later), its angular momentum \(L_{\text{tot}} \geq 0\) and its degree of contraction \(K_{\text{tot}} \geq 1\), and, given these, it is possible to make qualitative statements about the performance of each of the algorithms above in computing the ERI. For example, the Pople–Hehre (PH) axis-switching technique \[2\] is particularly effective when applied to ERIS with large \(K_{\text{tot}}\) but is otherwise very expensive. It is most commonly used for small-\(L_{\text{tot}}\) ERIS. Conversely, the Rys quadrature \[3\] is best suited to small-\(K_{\text{tot}}\) ERIS and is most commonly used in large-\(L_{\text{tot}}\) cases. The MD \[4\] and os \[7\] formulations function well for all values of \(L_{\text{tot}}\) provided that \(K_{\text{tot}}\) is rather small, but become excessively expensive when this is not so. The more recent algorithms \[9–11\] behave better than does os as \(K_{\text{tot}}\) increases, but are still inferior to the PH method for moderate to large \(K_{\text{tot}}\) values.

Is there any pattern to these algorithmic behaviors that might suggest how to construct a new methodology that performs well for all \(L_{\text{tot}}\) and \(K_{\text{tot}}\)? Fortunately, there is, and a very useful trichotomy is revealed:

(a) The algorithms that achieve their best results when \(K_{\text{tot}}\) is large but that become inefficient elsewhere are those in which the contraction step occurs very early in the methodology. The archetype of such algorithms is the PH axis-switch method \[2\].

(b) The defining characteristic of the algorithms that perform very well when \(K_{\text{tot}}\) is small but that lose their effectiveness elsewhere is that their contraction step occurs as the last stage in the methodology. The Rys \[3\] and os \[7\] algorithms exemplify this category.

(c) Between these two extremes, we find the algorithms in which contraction is introduced at some intermediate stage. Although inferior to the PH method when \(K_{\text{tot}}\) is large and inferior to the os method when \(K_{\text{tot}}\) is small, the algorithms in this category, which include the MD- and HGP-based \[4, 9–11\] methodologies, offer a generally useful performance compromise.

In recognition of this trichotomy, and in order to perform satisfactorily for all combinations of \(L_{\text{tot}}\) and \(K_{\text{tot}}\), current ab initio programs have had to include several, quite distinct, ERI subprograms. For example, faced with the computation of a certain ERI, GAUSSIAN 90 \[12\] uses simple heuristic rules to select an algorithm from a repertoire consisting of PH, Rys, os, and HGP codes. However, such code proliferation, and the inaccuracies associated with the automatic selection process, are clearly undesirable.

The PRISM algorithm, on the other hand, is explicitly designed to permit the contraction steps to occur precisely where it is most efficient for them to do so. It is hoped that this intrinsic flexibility will enable the PRISM algorithm to perform well for a wide range of classes of ERI.
Notation and Definitions

An unnormalized primitive Cartesian Gaussian function,

$$\varphi_{ak}(r) = (x - A)_a^a(y - A)_a^a(z - A)_a^a \exp[-\alpha_k(r - A)^2]$$,  \hspace{1cm} (1)

is defined by its angular momentum vector \( \mathbf{a} = (a_x, a_y, a_z) \), by its position vector \( \mathbf{A} \), and by its exponent \( \alpha_k \). Its angular momentum \( \mathbf{a} = (a_x + a_y + a_z) \). We will refer to a set of primitive functions on a given center and with a given exponent as a primitive shell.

Primitive functions are often linearly combined to form a contracted Cartesian Gaussian function

$$\phi_{a}(r) = \sum_{k=1}^{K_A} D_{ak} \varphi_{ak}(r),$$  \hspace{1cm} (2)

where the \( D_{ak} \) are known as contraction coefficients and \( K_A \) is known as the degree of contraction of \( \phi_a \).

The primitive four-center Gaussian ERI is the integral

$$[ab | cd] = \int \varphi_{a1}(r_1) \varphi_{b1}(r_1) r_1^{-1} \varphi_{c2}(r_2) \varphi_{d2}(r_2) dr_1 dr_2.$$

The left-hand subscripts are rarely of particular interest, and it is common to denote the integral (3) by \([ab | cd] \).

Combining (2) and (3) leads to a contracted four-center Gaussian ERI

$$\langle ab | cd \rangle = \sum_{kA} \sum_{kB} \sum_{kC} \sum_{KD} \sum_{D_{akA} D_{bkB} D_{ckC} D_{dkD}} [a_k b_k | c_k d_k],$$

which we distinguish from a primitive ERI by the use of parentheses instead of square brackets. A class of ERIs is defined as the set of all \( \langle ab | cd \rangle \) associated with a given shell-quartet.

The total angular momentum of the ERI (4) is \( L_{tot} = (a + b + c + d) \). Its bra degree of contraction and ket degree of contraction are \( K_{bra} = K_A K_B \) and \( K_{ket} = K_C K_D \), respectively. Its total degree of contraction is \( K_{tot} = K_{bra} K_{ket} \).

Associated with two primitive Gaussian functions \( \varphi_a \) and \( \varphi_b \) (centered at \( A \) and \( B \), with angular momenta \( a \) and \( b \), with exponents \( \alpha \) and \( \beta \), and with contraction coefficients \( D_A \) and \( D_B \)) is another center \( P \) and two parameters \( \sigma_P \) and \( U_P \) defined by

$$\sigma_P = 1/(2\alpha + 2\beta)$$  \hspace{1cm} (5)

$$U_P = (8\pi^3)^{1/2} \sigma_P^{a+b} D_A D_B \exp[-2\alpha\beta\sigma_P (A - B)^2]$$  \hspace{1cm} (6)

$$P = (2\alpha A + 2\beta B)\sigma_P,$$  \hspace{1cm} (7)

and, analogously associated with the primitive functions \( \varphi_c \) and \( \varphi_d \), we define another center \( Q \) and two parameters \( \sigma_Q \) and \( U_Q \).
As we have previously shown [11, 13], to compute the class (4), we form

$$R = Q - P$$

$$R^2 = R_x^2 + R_y^2 + R_z^2$$

$$\delta^2 = 1/(2\sigma_P + 2\sigma_Q)$$

$$T = \delta^2 R^2$$

$$U = U_P U_Q$$

$$[0]^m = U(2\delta^2)^{m+1/2} \left( \frac{2}{\pi} \right)^{1/2} \int_0^1 t^{2m} \exp(-Tt^2) dt.$$  

Given the set of $[0]^m$ ($0 \leq m \leq L_{\text{red}}$), as MD and we [4,11] have shown, it is possible to use a recurrence relation to form the set of intermediate Hermite $\text{ERIS} [r]$ ($0 \leq r \leq L_{\text{red}}$). The $[r]$ are equal, within a change of sign, to $\text{ERIS}$ that we designate by $[p|q]$ and that represent the electrostatic interaction between one primitive (Hermite) function on center $P$ and another on center $Q$. We refer to the function on center $P$, which we symbolize by $[p]$, as a $p$-$\text{bra}$. Likewise, we refer to the function on center $Q$, which we symbolize by $|q]$, as a $q$-$\text{ket}$. MD and we have shown [4,11] that various recurrence relations may then be used to $\text{bra}$-transform the $[p|q]$ to $[ab|q]$ and to $\text{ket}$-transform the $[ab]|q]$ to the desired $[ab|cd]$. Moreover, as we have argued [11], the $\text{bra}$- and $\text{ket}$-transformation steps can be generalized to produce not only $\text{ERIS}$ like $[ab|cd]$ but, also, their derivatives with respect to $A$, $B$, $C$, and $D$. In this very general framework, we symbolize the results of the $\text{bra}$-transformation by $[\text{bra}|q]$ and those of the $\text{ket}$-transformation by $[\text{bra}|\text{ket}]$.

The issue of contraction, which is central to the PRISM algorithm, may now be discussed. Straightforward contraction is exemplified by

$$[\text{bra}|\text{ket}] = \sum_{K_{\text{bra}}} [\text{bra}|\text{ket}]$$

$$[\text{bra}|\text{ket}] = \sum_{K_{\text{ket}}} (\text{bra}|\text{ket})$$.

and we note that a contraction like (14) requires exactly $K_{\text{bra}} - 1$ additions. However, as we have previously suggested [11], it is frequently beneficial to generalize this concept to include simultaneous scaling of the uncontracted quantities by exponent ratios. Examples of this type of contraction are

$$a^{\alpha} b^{\beta} [r] = \sum_{K_{\text{bra}}} (2\alpha)^{\alpha} (2\beta)^{\beta} [r]$$

$$[0]_{abcd}^{(m)} = \sum_{K_{\text{bra}}} (2\gamma)^{\gamma} (2\delta)^{\delta} [0]^{(m)}.$$

Note that the contraction (16) requires $K_{bra} - 1$ additions and $K_{bra}$ multiplies.

Throughout this paper, we will use the acronym FLOP for a floating-point operation, that is, a floating-point add, subtract, multiply, or divide. The number of
FLOPS required by a given algorithm to compute a class of ERIS (its FLOP-count) is usually a good theoretical indication of the practical performance of the algorithm when implemented on a computer, and this fact lies at the heart of the PRISM method.

**A New Algorithm for Two-Electron Integrals**

In Figure 1, we symbolically represent a number of interrelated paths from uncontracted \([0]^{(m)}\) integrals (at the top left corner) to fully contracted \((\text{bra} | \text{ket})\)'s (at the bottom right). The figure is structured so that an arrow to the right corresponds to a contraction step, while a downward pointing arrow indicates a transformation step, i.e., one in which recurrence relations are used to build or shift angular momentum. Since it is clear that any path from \([0]^{(m)}\) to \((\text{bra} | \text{ket})\) must involve three transformation and two contraction steps in some order, it is possible to label each path uniquely by an acronym indicating the order in which the transformation and contraction steps are performed. For example, the path that passes through the \((0)^{(m)}\) would be termed CCTTT, while, at the other ex-

![Diagram of PRISM Paths](image-url)
treme, the path that passes through $[\text{bra} | \text{ket}]$ would be termed TTTCC. It should be apparent that there are exactly 10 paths through Figure 1.

We note that TTCTC is very similar to the standard MD algorithm, which, as MD indicated [4], can be shown to be always at least as efficient as TTTCC. We also note that, in the context of the BRAKET algorithm [11], we have previously termed TCTCT and TCCTT the “late contraction” and “early contraction” paths, respectively.

To perform the transformation steps (T1 to T9) in Figure 1, appropriate recurrence relations must first be derived. Many years ago, MD developed ones appropriate for uncontracted bra- and ket-transformations (T4, T7, and T8) and for the uncontracted $r$-transformation (T1). More recently, we have presented a variety of recurrence relations [11] that are applicable to both uncontracted and contracted bra- and ket-transformations (T4 – T9). The recurrence relations for the remaining $r$-transformations (T2 and T3) can be deduced easily from that for the T1 transformation:

$$[r]^{(m)} = R_i [r - 1,]^{(m+1)} - (r_i - 1) [r - 2,]^{(m+1)},$$

by expanding $R_i = Q_i - P_i$ in various ways. For the T2 transformation, we use

$$R_i = \frac{2\alpha}{2\zeta} (B_i - A_i) + (Q_i - B_i),$$

when

$$a' b' p' [r]^{(m)} = (B_i - A_i) a' b' (r' + 1)(r - 1,]^{(m+1)} + (Q_i - B_i) a' b' p' (r - 1,]^{(m+1)}$$

$$- (r_i - 1) a' b' p' (r - 2,]^{(m+1)},$$

while, for the T3 transformation, we use

$$R_i = \frac{2\alpha}{2\zeta} (B_i - A_i) + \frac{2\gamma}{2\eta} (C_i - D_i) + (D_i - B_i),$$

from which it follows that

$$a' b' p' [r]_{c'd'q}^{(m)} = (B_i - A_i) a' b' (r' + 1)(r - 1,]_{c'd'q}^{(m+1)} + (C_i - D_i) a' b' p' (r - 1,]_{c'd'q}^{(m+1)}$$

$$+ (D_i - B_i) a' b' p' (r - 1,]_{c'd'q}^{(m+1)} - (r_i - 1) a' b' p' (r - 2,]_{c'd'q}^{(m+1)}.$$  

As we have noted before [11], it is possible to improve the efficiency of the T1, T2, and T3 transformations by solving the tree-search problem that is implicit in the T1 transformation when $L_{tot} > 2$. We have examined this problem in detail and our results are presented elsewhere [14].

We now have at our disposal all of the recurrence relations necessary to form any desired braket from $[0]^{(m)}$ integrals using any of the 10 paths in Figure 1. We note, however, that Figure 1 can be generalized further. The constraint that the bra-contraction precedes the ket-contraction and that the bra-transformation precedes the ket-transformation is unnecessary and, in Figure 2, it is removed. Figure 2 (which is, of course, a rectangular prism) contains 20 paths from $[0]^{(m)}$ to $[\text{bra} | \text{ket}]$ and, henceforth, we will refer to these as PRISM paths.
At this point, it is instructive to present detailed examples of how a class of ERIS might be formed on various PRISM paths. Specifically, we will consider the problem of forming a class of \((p_s \mid p_s)\) ERIS and we will use the CCTTT and TCTTC paths as examples.

**Example 1: Forming \((p_s \mid p_s)\) ERIS on the CCTTT Path**

The discussion is facilitated if we work *backward* along the CCTTT path. Thus, we begin by considering the *ket*-transformation step, i.e., the formation of \((p_i s \mid p_i s)\) from \((p_i s \mid q)_{c'd'q'}\). From Eq. (45) in [11b], we have

\[
(p_i s \mid p_i s) = (D_i - C_i)(p_i s \mid s)_{011} + (p_i s \mid p_i s)_{001},
\]

and, as (23) costs two FLOPS for each of the nine \((p_i s \mid p_i s)\), the step-cost of the *ket*-transformation \(T_9 = 18\) FLOPS.

Continuing backward, we next consider the *bra*-transformation step, i.e., the formation of \((p_i s \mid q)_{c'd'q'}\) from \(\phi(p \mid q)_{c'd'q'}\). As above, we have

\[
(p_i s \mid q)_{c'd'q'} = (B_i - A_i)(s \mid q)_{c'd'q'} + \omega(p_i \mid q)_{c'd'q'},
\]
and, as (24) costs two FLOPS per \((p|q)_{c:d}\) and we have just found that there are 12 of these to form, the bra-transformation step-cost \(T_b = 24\) FLOPS.

Knowing the four \(p\)-bras and four \(q\)-kets that are needed and the Hermite identity \([4,11]\]

\[
a^{r'}p'(p|q)c^{d'}q' = (-1)^q a^{r'}p'(p + q)c^{d'}q' \tag{25}
\]

enables us to list, as follows, the 13 \(a^{r'}p'(r)c^{d'}q'\) that are needed:

\[
go11(0)_{011}, \ g01(1)_{001}, \ g01(1)_{101}, \ g01(2)_{001}. \tag{26}
\]

Using the \(T_3\) recurrence relation (22), these integrals can be reduced to \(a^{r'}p'(0)_{c:d}\) as follows:

\[
o11(1)_{001} = (B_i - A_i)_{12}(0)_{001}^{01} + (C_i - D_i)_{011}(0)_{102}^{00} + (D_i - B_i)_{001}(0)_{001}^{01} \tag{27}
\]

\[
o01(1)_{011} = (B_i - A_i)_{021}(0)_{011}^{00} + (C_i - D_i)_{001}(0)_{112}^{01} + (D_i - B_i)_{001}(0)_{011}^{00} \tag{28}
\]

\[
o01(2)_{001} = (B_i - A_i)_{021}(0)_{001}^{00} + (C_i - D_i)_{001}(1)_{012}^{01} + (D_i - B_i)_{001}(1)_{012}^{01} \tag{29}
\]

\[
o12(1)_{101} = (B_j - A_j)_{2001}(0)_{001}^{00} + (C_j - D_j)_{001}(0)_{102}^{00} + (D_j - B_j)_{001}(0)_{001}^{00} \tag{30}
\]

\[
o01(1)_{102} = (B_j - A_j)_{0201}(0)_{102}^{00} + (C_j - D_j)_{001}(0)_{203}^{00} + (D_j - B_j)_{001}(0)_{102}^{00} \tag{31}
\]

\[
o01(1)_{001} = (B_j - A_j)_{0201}(0)_{001}^{00} + (C_j - D_j)_{001}(0)_{001}^{00} + (D_j - B_j)_{001}(0)_{001}^{00}, \tag{32}
\]

and (27)–(32) imply that the \(r\)-transformation step-cost \(T_3 = 108\) FLOPS.

We have worked backward through the three transformation steps on the CCTTT path and, in this way, have reduced the nine target \(\text{ERIS}\) to the following 14 \(a^{r'}p'(0)_{c:d}\) integrals:

\[
\begin{align*}
go11(0)_{001}, & \ g01(0)_{001}, \ g01(0)_{011}, \ g01(0)_{101}, \ g01(0)_{112}, \ g01(0)_{102}, \ g01(0)_{001}, \\
go12(0)_{101}, & \ g01(0)_{001}, \ g01(0)_{011}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}. \tag{33}
\end{align*}
\]

In a \(ket\)-contraction costing \(C_2 = (22K_{ket} - 14)\) FLOPS, these integrals can be formed from the following eight \(a^{r'}p'(0)_{c:d}\) integrals:

\[
\begin{align*}
o11(0)_{001}, & \ g01(0)_{001}, \ g01(0)_{011}, \ g01(0)_{101}, \ g01(0)_{112}, \ g01(0)_{102}, \ g01(0)_{001}, \\
o12(0)_{101}, & \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}, \ g01(0)_{001}. \tag{34}
\end{align*}
\]

which, in turn, can be formed from the following three \(\text{ERIS}\) integrals:

\[
\begin{align*}
o01(0)_{001}, & \ g01(0)_{001}, \ g01(0)_{001}, \\
o01(0)_{001}, & \ g01(0)_{001}, \ g01(0)_{001}. \tag{35}
\end{align*}
\]

in a \(bra\)-contraction step costing \(C_1 = (14K_{bra} - 8)K_{ket}\) FLOPS.

It turns out [13] that a set of \(\text{ERIS}\) \((0 \leq m \leq 2)\) can be formed from the corresponding set of \(F_m(T)\) integrals at a cost \(X = 6K_{bra}K_{ket}\) FLOPS (and \(K_{bra}K_{ket}\) square roots). Finally, to find the total cost of forming a class of \((ps|ps)\) \(\text{ERIS}\) from \(F_m(T)\) integrals along CCTTT, it suffices simply to add together the six
step-costs that we have just computed. Thus,
\[
\text{CCTTT-cost} = X + C_1 + C_2 + T_3 + T_6 + T_9
\]
\[
= 6K_{\text{bra}}K_{\text{ket}} + (14K_{\text{bra}} - 8)K_{\text{ket}} + (22K_{\text{ket}} - 14) + 108 + 24 + 18
\]
\[
= 20K_{\text{bra}}K_{\text{ket}} + 14K_{\text{ket}} + 136. \quad (36)
\]

Henceforth, we will often represent the FLOP-cost incurred along a given path when forming a given braket class by expressing it in this form, i.e.,
\[
\text{Path-cost} = xK_{\text{bra}}K_{\text{ket}} + yK_{\text{ket}} + z,
\]
and we will refer to \(x\), \(y\), and \(z\) as the path-cost parameters for that path and braket class. Note that, for uniform contraction \(K\), this becomes the familiar
\[
\text{Path-cost} = xK^4 + yK^2 + z. \quad (38)
\]

**Example 2: Forming \((ps|ps)\) ERIS on the TCTTC Path**

As before, we will work backward along the TCTTC path. Thus, we begin by considering the ket-contraction step, i.e., the formation of \((ps|ps)\) from \((ps|ps)\), which is defined by the simple contraction formula
\[
(ps|ps) = \sum (p_i|s_p). \quad (39)
\]
The summation clearly involves \(K_{\text{ket}} - 1\) additions and, since we are forming nine \((ps|ps)\), the ket-contraction cost \(C_9 = 9K_{\text{ket}} - 9\) FLOPS.

Continuing up the PRISM, we next consider the ket-transformation step in which \((ps|ps)\) are formed from \((ps|q)\). From Eq. (45) in [11b], we have
\[
(ps|ps) = (2\delta)(D_j - C_j)(ps|s)_{001} + (ps|p)_{001}, \quad (40)
\]
and, as this recurrence relation costs two FLOPS per \(ps|ps\) and there are 9\(K_{\text{ket}}\) of these, the ket-transformation step-cost \(T_8 = 18K_{\text{ket}}\) FLOPS.

Proceeding to the bra-transformation step, in which \((ps|q)_{001}\) are formed from \(a'b'n(p|q)_{001}\), we have (as in the previous example)
\[
(ps|q)_{001} = (B_i - A_i)_{011}(s|q)_{001} + 001(p|q)_{001}, \quad (41)
\]
and, as this recurrence relation costs two FLOPS per \(ps|q)_{001}\) and we have just discovered that there are 12\(K_{\text{ket}}\) of these to form, the bra-transformation step-cost \(T_5 = 24K_{\text{ket}}\) FLOPS.

Knowing the four \(p\)-bras and four \(q\)-kets that are needed and the Hermite identity [4,11],
\[
a'b'n(p|q)_{k'd'q'} = (-1)^q a'b'n(p + q)_{k'd'q'}, \quad (42)
\]
enables us to list, as follows, the 13 \(a'b'n(r)\) that are needed:
\[
011(0)_{001}, 011(1)_{001}, 001(1)_{001}, 001(2)_{001}. \quad (43)
\]
These \( \{ r \} \) can be formed, in the bra-contraction step, as follows:

\[
0_{11}(0)_{001} = \sum (2\beta)_{001}[0]_{001} \tag{44}
\]

\[
0_{11}(1)_{001} = \sum (2\beta)_{001}[1]_{001} \tag{45}
\]

\[
0_{01}(1)_{001} = \sum 0_{01}[1]_{001} \tag{46}
\]

\[
0_{01}(2)_{001} = \sum 0_{01}[2]_{001} \tag{47}
\]

To form the four \( 0_{01}(r)_{001} \) requires \( 4(K_{bra} - 1)K_{ket} \) additions and \( 4K_{bra}K_{ket} \) multiplies, while formation of the 9 \( 0_{01}(r)_{001} \) requires \( 9(K_{bra} - 1)K_{ket} \) adds. Thus, the cost of the bra-contraction step is \( C_3 = 17K_{bra}K_{ket} - 13K_{ket} \) FLOPS.

The required \( \{ r \} \) are formed in the \( T_1 \) step (18) as follows:

\[
0_{01}[1]_{001} = R_{100}[0]_{001}^{(1)} \tag{48}
\]

\[
0_{01}[1]_{001}^{(2)} = R_{100}[0]_{001}^{(2)} \tag{49}
\]

\[
0_{01}[2]_{001} = R_{100}[1]_{001}^{(1)} - \delta_{ij} 0_{01}[0]_{001}^{(i)} \tag{50}
\]

which, it is easily verified, costs \( T_1 = 15K_{bra}K_{ket} \) FLOPS.

As before, the set of \( 0_{01}[0]_{001} \) \((0 \leq m \leq 2)\) can be formed from the \( F_m(T) \) integrals at a cost \( X = 6K_{bra}K_{ket} \) FLOPS (and \( K_{bra}K_{ket} \) square roots). Thus, the total cost of forming a class of \( (ps | ps) \) ERIS from \( F_m(T) \) integrals along TCTTC is

\[
X + T_1 + C_3 + T_3 + T_8 + C_8 = 6K_{bra}K_{ket} + 15K_{bra}K_{ket} + (17K_{bra} - 13)K_{ket} + 24K_{ket} + 18K_{ket} + (9K_{ket} - 9) = 38K_{bra}K_{ket} + 38K_{ket} - 9. \tag{51}
\]

Thus, the \( x, y, \) and \( z \) path-cost parameters for \( (ps | ps) \) formation on the TCTTC path are 38, 38, and -9, respectively.

The Theoretical Performance of the PRISM Algorithm

Throughout this paper, our principal interest is the efficient generation of \( (bra | ket) \) from \( [0]^{(m)} \), and we will not discuss how the \( [0]^{(m)} \) may be formed in the first place. Elsewhere [14], however, we have addressed this question in detail and have presented an optimized scheme for the formation of \( [0]^{(m)} \) from shell-pair data, and this scheme is easily modified to compute \( [ss | ss]^{(m)} \) if these are required. In order to compare the FLOP-costs of methods that use \( [0]^{(m)} \) integrals with methods that use \( [ss | ss]^{(m)} \) integrals, we must establish a "common denominator." We have shown [13] that a complete set of \( [0]^{(m)} \) \((0 \leq m \leq L_{tot})\) can be formed from the corresponding \( F_m(T) \) integrals in \( 2L_{tot} + 2 \) FLOPS and one square root, and it is straightforward to deduce that the analogous cost for forming a set of \( [ss | ss]^{(m)} \) \( L_{tot} + 2 \) FLOPS and one square root. Therefore, for FLOP-counting purposes in the present paper, we will assume that all \( F_m(T) \) have
already been computed and any path-cost parameters that we will discuss will pertain to the cost of forming ERIS from these \( F_n(T) \) integrals. We will simply ignore the square root since this is common to all algorithms that we will consider.

Analyses analogous to that performed in the foregoing section for \((ps\mid ps)\) along the CCTTT and TCTTC paths can be conducted for other ERIs classes and other paths. Proceeding in this way for \((ps\mid ps), (pp\mid pp), \) and \((dd\mid dd)\) classes leads to the associated PRISM step-costs (Figs. 3–5) and path-cost parameters (Table I).

Table I contains path-cost parameters for PRISM paths and for three other methods. The PH axis-switch technique and HGP algorithm have been described in detail elsewhere [2, 9]. os did not provide a prescription for the use of their recurrence relation [7], and, for this reason, it was necessary first for us to design one. We have adopted the following “left-to-right” approach: The recurrence relation is first used to reduce the angular momentum at A to zero, then it is simi-
larly applied to B, then to C and, finally, to D. When a function can be reduced in more than one way (e.g., $d_{xy}$), we choose the reduction direction from the optimal $T_1$ transformation that we have examined elsewhere [14]. Specifically, we reduce $d_{xy}$, $d_{zz}$, and $d_{yz}$ in the $x$, $z$, and $y$ directions, respectively.

Similarly, as we have discussed previously [11], there is often more than one way in which the various one-electron recurrence relations that we have constructed [11] can be used to form a desired class of bras (or kets) in the PRISM approach. The $(ps \mid bras)$ that we needed in the foregoing section can be formed in only one way, namely,

$$ (p|s] = (B_i - A_i)|_{001}(s] + |_{001}(p]. $$  \hspace{1cm} (52)

In general, our program reduces more complicated bras (e.g., $dd$ etc.) by “left-to-right” application of the $(p \rightarrow a)$ recurrence relation (Eq. (45) in [11b]) and its analogs, $(p \rightarrow b)$, $(q \rightarrow c)$, and $(q \rightarrow d)$.
Although we do not expect the "left-to-right" recipe to lead to optimal reduction schemes for either 0s or PRISM, its performance appears to be adequate. In one important special case, however, that of \((pp|bras)\), the PRISM program uses the following, more carefully optimized, reduction scheme:

\[
\begin{align*}
012(p_i) &= 001(p_i) - 012(p_i) \\
101(p_is) &= (B_i - A_i)112(s) + 002(p_i) \\
(p_ip_j) &= (A_j - B_j)011(p_ip_j) + (B_i - A_i)012(p_i) + 002(d_{ij}) + \delta_{ij}001(s). 
\end{align*}
\]

It is clear from Table I that the most efficient path to any given ERI class will be very dependent on the \(K_{bra}\) and \(K_{ker}\) values of the class. For a particular class and particular values of \(K_{bra}\) and \(K_{ker}\), the path-costs can be evaluated using the formulae in Table I and the optimal PRISM path thereby selected. The results of such analyses are displayed in Figures 6–8 that we term "Choice Diagrams" for
Table 1. Path-cost parameters for ERI formation.

<table>
<thead>
<tr>
<th>Path</th>
<th>$x$</th>
<th>$y$</th>
<th>$z$</th>
<th>$x$</th>
<th>$y$</th>
<th>$z$</th>
<th>$x$</th>
<th>$y$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCTTT</td>
<td>20</td>
<td>14</td>
<td>136</td>
<td>70</td>
<td>168</td>
<td>2904</td>
<td>575</td>
<td>5,506</td>
<td>159,624</td>
</tr>
<tr>
<td>CTCTT</td>
<td>20</td>
<td>66</td>
<td>29</td>
<td>70</td>
<td>771</td>
<td>976</td>
<td>575</td>
<td>18,079</td>
<td>65,212</td>
</tr>
<tr>
<td>CTTCT</td>
<td>20</td>
<td>88</td>
<td>6</td>
<td>70</td>
<td>1240</td>
<td>306</td>
<td>575</td>
<td>39,289</td>
<td>19,080</td>
</tr>
<tr>
<td>CTTTC</td>
<td>20</td>
<td>100</td>
<td>-9</td>
<td>70</td>
<td>1537</td>
<td>-81</td>
<td>575</td>
<td>55,597</td>
<td>-1,296</td>
</tr>
<tr>
<td>TCCTT</td>
<td>38</td>
<td>4</td>
<td>29</td>
<td>225</td>
<td>129</td>
<td>976</td>
<td>2,655</td>
<td>3,852</td>
<td>65,212</td>
</tr>
<tr>
<td>TCTCT</td>
<td>38</td>
<td>26</td>
<td>6</td>
<td>225</td>
<td>598</td>
<td>306</td>
<td>2,655</td>
<td>25,062</td>
<td>19,080</td>
</tr>
<tr>
<td>TCTTC</td>
<td>38</td>
<td>38</td>
<td>-9</td>
<td>225</td>
<td>895</td>
<td>-81</td>
<td>2,655</td>
<td>41,370</td>
<td>-1,296</td>
</tr>
<tr>
<td>TTTCC</td>
<td>57</td>
<td>3</td>
<td>6</td>
<td>646</td>
<td>108</td>
<td>306</td>
<td>21,290</td>
<td>3,672</td>
<td>19,080</td>
</tr>
<tr>
<td>TTCTC</td>
<td>57</td>
<td>15</td>
<td>-9</td>
<td>646</td>
<td>405</td>
<td>-81</td>
<td>21,290</td>
<td>19,980</td>
<td>-1,296</td>
</tr>
<tr>
<td>TTTCC</td>
<td>72</td>
<td>0</td>
<td>-9</td>
<td>1051</td>
<td>0</td>
<td>-81</td>
<td>41,270</td>
<td>0</td>
<td>-1,296</td>
</tr>
</tbody>
</table>

It is assumed that all necessary $F_n(T)$ integrals are available (see text).

The Pople-Hehre axis-switch algorithm [2].

The Head-Gordon-Pople algorithm [9].

The Obara-Saika algorithm [7] using a “left-to-right” reduction scheme (see text).

the PRISM paths. On the basis of the three ERI classes analyzed, we can make a number of useful observations:

(a) Large values of $K_{tot} = K_{bra} K_{ket}$ favor paths in which contraction is introduced early, whereas small values favor paths with late contraction.

\[
K_{ket}
\]

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 \\
\end{array}
\]

\[
K_{bra}
\]

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 \\
\end{array}
\]

Figure 6. Choice diagram for $(ps | ps)$ formation.
Figure 7. Choice diagram for $(pp|pp)$ formation.

Figure 8. Choice diagram for $(dd|dd)$ formation.
(b) Generally speaking, paths in which contraction is introduced early become less favorable as $L_{ot}$ increases. For example, it is clear that the CCTTT path will always become optimal when $K_{bra} \geq K_{crit}$ and $K_{ket} \geq K_{crit}$ for some class-dependent critical value $K_{crit}$. However, while $K_{crit} = 1$ (trivially) for $(ss | ss)$, we find that $K_{crit} = 3$ for $(ps | ps)$, $K_{crit} = 4$ for $(pp | pp)$ and $K_{crit} = 8$ for $(dd | dd)$.

(c) The path-cost parameters for HGP (which could be viewed as a TCCT method) most closely resemble those of the TTCCT PRISM path.

(d) HGP is much cheaper than is OS for uncontracted $(dd | dd)$. This important fact has not previously been recognized in the literature.

(e) The path-cost parameters of the PRISM paths range from being comparable to those of the PH axis-switch technique to being similar to those of the OS method. This suggests that PRISM has the flexibility to perform well for a wide range of ERI classes but that it will be poorest for uncontracted, high-angular momentum classes.

The Practical Performance of the PRISM Algorithm

To compare the practical performance of our implementation of PRISM with implementations of other methods, we have measured CPU timings on MicroVax-II and VAXStation-3100 computers. Although our PRISM program is vectorizable (it is modeled on our BRAKET program [11] and its loop structure is shown in Fig. 9), these machines are both scalar computers. We plan to publish CPU timing comparisons on vector computers (Alliant FX-8 and Cray Y-MP) in the near future.

Head-Gordon and Pople have previously discussed the practical performance of their implementation of the HGP algorithm [9] relative to the GAUSSIAN 86 implementations of the PH [2] and Rys [3] algorithms. They found that HGP was uniformly faster than was Rys for all ERI classes and faster than PH except for highly contracted classes. These results were consistent with FLOP counts that they had also computed. It appears, therefore, that the best standards against which to compare our PRISM program are the HGP and PH programs.

Following HGP's example [9], we first examine the relative performances of the ERI programs on "pure" systems, i.e., ones that contain only one type of shell. Each case consists of 12 shells in a bicubic arrangement. In the $(ss | ss)$ case, the shells are those from the STO-4G basis set for hydrogen [15] and the edge length of the bicube is 0.8 Å. In the $(pp | pp)$ and $(sp, sp | sp, sp)$ cases, the shells are those from the STO-2G basis set for carbon [15] and the edge length is 1.4 Å. In the $(dd | dd)$ case, the shells are uncontracted with exponent 0.8 and, again, the edge length is 1.4 Å. In Table II, we compare the times taken by the HGP, PH, and PRISM codes to compute all of the ERIS in each of the four systems. No point-group symmetry was used, and the requested accuracy was $10^{-10}$ for HGP and PRISM and about $10^{-8}$ for PH. The computer used was a MicroVAX-II running VMS.

The $(ss | ss)$ results reflect the different approaches of the three codes to the task of computing the incomplete gamma function: HGP uses 6th-order Taylor in-
Call **CalcS2** and **CalcS3** to compute and order shell-pair information
For \( L = 0 \) to \( \text{Max}L \)
  - Call **TabGmT** to set up table for interpolating \( G_{L}(T) \)
  - Call **MakMD2** to form driver for \( (0)^{(m)} \rightarrow [r] \) transformation
  - For each bra angular momentum type
    - Call **MakleT** to form drivers for bra transformation
  - For each ket angular momentum type
    - Call **MakleT** to form drivers for ket transformation
    - Call various routines to form contraction drivers etc.
  - For each bra degree of contraction
    - For each ket degree of contraction
      - Call **Choose** to select the optimal PRISM Path
      - Call **CalcSF** to compute two-centre scaling factors
      - Call **CalcBS** to find maximum possible batch size
        - For each batch of shell-quartets, until finished
          - Call **PickS4** to select the batch
          - Call **CalcS4** to form basic shell-quartet parameters
          - Call **Calc0m** to form \( (0)^{(m)} \)
          - Call **DoCont** to do the bra-contraction
          - Call **DoCont** to do the ket-contraction
          - Call **DoMD4** to do the \( (0)^{(m)} \rightarrow [r] \) transformation
          - Call **DoShuf** to form \( (p | q) = (-1)^{I_{r}}(p + q) \)
          - Call **DoTran** to transform \( (p | q) \rightarrow (bra | ket) \)
        - Next batch of shell-quartets
        - Next bra degree of contraction
        - Next ket degree of contraction
        - Next ket angular momentum type
        - Next bra angular momentum type
  - Next \( L \)

Figure 9. Program loop structure for bra-ket formation using the CCTTT path.
(Other paths result from permuting the calls to DoCont, DoMD4, and DoTran.)

Interpolation, as recommended by MD [4] and Harris [16]; PH uses Everett’s Formula with Throwback [17]; PRISM employs 3rd-order Chebyshev interpolation [13]. The high efficiency of Chebyshev interpolation is well known.

In the \((pp | pp)\) example, the PRISM program selected the CCTTT path. (Note that this could have been anticipated from Fig. 7 with \( K_{bra} = K_{ket} = 4 \).) It is easy to deduce, from the path-cost parameters in Table I, that CCTTT needs less than half as many FLOPS as does HGP for \( K = 2 \) \((pp | pp)\) and, accordingly, the ratio of the HGP and PRISM timings is greater than 2. The performance of the PH program is intermediate between HGP and PRISM, but the comparison here is not quite fair: PH begins by assuming that each of the \( p \) shells is an \( sp \) shell and, therefore, computes many more ERIS than are required. Only afterward does the PH program discard the unwanted ERIS.
A fairer comparison between PRISM and PH is afforded by the \((sp, sp \mid sp, sp)\) example. Here, the TCCTT path was chosen, and as can be seen, there is little difference between the performances of the PRISM and axis-switch codes. The HGP program, on the other hand, is clearly the slowest of the three. It is interesting to note that the HGP/PRISM ratio is smaller for \((sp, sp \mid sp, sp)\) than for \((pp \mid pp)\).

The axis-switch technique is not implemented in GAUSSIAN 90 for \(d\) shells, but it would be expected to perform poorly in the uncontracted example that we have selected: Axis-switching is worthwhile only when the degree of contraction is sufficiently large. It is particularly interesting to note, however, that HGP is considerably faster than is PRISM (which selected the TTTCC path) in the \((dd \mid dd)\) example in Table II. Again, reference to Table I reveals that this could have been predicted on the basis of their relative FLOP counts. In the uncontracted case, the total FLOP cost is simply the sum of the \(x, y,\) and \(z\) parameters, which is 23,761 FLOPS for HGP and 39,974 FLOPS for TTTCC. This reveals the major deficiency of the PRISM algorithm: Even those paths that are best suited to uncontracted ERI classes (e.g., TTTCC) become increasingly less and less competitive with HGP as the ERI angular momentum increases. Thus, although TTTCC, OS, and HGP are comparable for uncontracted \((pp \mid pp)\), TTTCC and OS are roughly 70% more expensive than is HGP for uncontracted \((dd \mid dd)\). We project that the difference is even larger for uncontracted \((ff \mid ff)\) classes.

We are now in a position to examine some timing results on naphthalene—a "real" molecule. All \(C-C\) and \(C-H\) bond lengths, respectively, were 1.4 and 1.1 Å. All angles were 120 degrees. Because it cannot be used in direct \(SCF\) calculations in GAUSSIAN 90, the axis-switch program could not be fairly compared with the HGP and PRISM programs. In Table III, we present timings for a single direct \(SCF\) iteration, i.e., the timings are composed of a contribution from computing the ERS and a contribution from assimilating them into Fock matrices. The relative importance of the assimilation component is greatest (25%–30%) when the basis set used is least contracted (3-21G) and is negligible for strongly contracted basis sets (STO-3G). The calculations were performed on a VAXStation-3100 running VMS.
With its ability to match the performance of the axis-switch method in highly contracted ERI classes, PRISM fares best in the STO-3G calculation where it required only 44% of the time needed by HGP. This fraction rises to 54% for 6-31G, to 81% for 6-31G*, and to 78% for 3-21G. The explanation for this trend is twofold: First, weakly contracted basis sets offer PRISM little opportunity to utilize paths that are substantially more efficient than is HGP, and, second, the assimilation phase (which is essentially a constant added to both the PRISM and HGP ERI-evaluation times) is most expensive for these same basis sets.

Conclusions

The overall performance of the PRISM algorithm is superior to that of the PH axis-switch technique (which is poor for weakly contracted ERI classes) and to that of the HGP methodology (which is poor for strongly contracted ERI classes). PRISM is substantially inferior to HGP; however, for weakly contracted classes of high angular momentum, a modified version of PRISM that does not suffer from this defect needs to be developed.

Acknowledgments

We thank Elizabeth Gill for assistance with the figures in this paper and Benny Johnson for useful comments on its content. This work was supported by the National Science Foundation under grant CHEM-8918623 and was presented at the Argonne National Laboratory's Methodology of the Evaluation of Integrals in LCAO Calculations workshop in August 1990.

Bibliography


Received November 20, 1990
Accepted for publication December 11, 1990