# The efficient transformation of $(m0|n0)$ to $(ab|cd)$ two-electron repulsion integrals

Benny G. Johnson [1], Peter M.W. Gill and John A. Pople
*Department of Chemistry, Carnegie Mellon University, Pittsburgh, PA 15213, USA*

Received 31 December 1992

A new method is presented for transforming $(m0|n0)$ integrals to $(ab|cd)$ integrals efficiently. A key feature of the new approach is the optimization of the total number of memory operations (mops) rather than floating-point operations (flops). Solutions to the inherent tree-search problem were obtained which are radically different than those following when flops are taken as the cost function. These involve new, multi-unit angular momentum transfer relations which belong to a general family having the familiar one-unit transfer relation of Head–Gordon and Pople as its simplest member. CPU timings demonstrate the new method to be clearly superior to two others considered, including optimal use of the one-unit transfer relation alone.

## 1. Introduction

With the advent of "direct" methods for large ab initio quantum chemical calculations, in which the two-electron repulsion integrals (ERIs) over atomic orbital basis functions are recomputed each time they are needed, much attention has recently been focused on efficient ERI computation. Many current methods for evaluating ERIs and their derivatives [1–12] involve the use of recurrence relations to construct the desired quantities from easily computed s-type integrals [13] via an intermediary of auxiliary integrals.

In general, when endeavoring to construct a class of ERIs from the initial s-type integrals using a given set of recurrence relations, there are a multitude of possible paths to the target, with varying associated costs in terms of CPU time required by a computer implementation. In most cases, as previous work has shown [14], not all of the intermediate integrals are required and there is potential for substantial savings by solving the inherent tree-search problems.

We focus here on one particular step in the ERI evaluation process which is incorporated in several current algorithms [5,7–10,12] – the transformation of integrals of the special form $(m0|n0)$ (zero angular momentum in the second and fourth positions) to general $(ab|cd)$ integrals. We begin by giving a brief motivation as to why such an intermediate step is beneficial in an ERI algorithm, as this offers some insight into the nature of the problem to be considered while introducing some notations which will aid in the exposition.

The shorthand notation $(ab|cd)$ represents the integral

$$\int\int \phi_a(r_1)\phi_b(r_1)\,|r_1-r_2|^{-1} \times \phi_c(r_2)\phi_d(r_2)\,dr_1\,dr_2\,. \tag{1}$$

The basis functions are

$$\phi_a(r) = (x-A_x)^{a_x}(y-A_y)^{a_y}(z-A_z)^{a_z} \times \sum_{k=1}^{K_A} D_{ak}\exp(-\alpha_{ak}|r-A|^2)\,. \tag{2}$$

This function is centered at $A=(A_x, A_y, A_z)$ and has degree of contraction $K_A$, where the $\alpha_{ak}$ and $D_{ak}$ are the primitive exponents and contraction coefficients, respectively. The Cartesian factor premultiplying the function's radial distribution determines its angular momentum, and is compactly represented by $a=(a_x, a_y, a_z)$. The total angular momentum of the basis function is $a=a_x+a_y+a_z$. Each ba-

[1] To whom correspondence should be addressed. E-mail: johnson@cmchem.chem.cmu.edu

sis function is generally part of a group of related basis functions called a *shell*. Members of the shell share the same radial function and total angular momentum [#1], but have different divisions of the total number of units amongst the components of the vector $\boldsymbol{a}$.

The electron-1 part (the function of $\boldsymbol{r}_1$) of the integrand in eq. (1) contains the product of the Cartesian factors of basis functions on centers $\boldsymbol{A}$ and $\boldsymbol{B}$. We will refer to these products as *bras* [#2], and represent them by

$$(\boldsymbol{ab}| \equiv (x-A_x)^{a_x}(y-A_y)^{a_y}(z-A_z)^{a_z} \times (x-B_x)^{b_x}(y-B_y)^{b_y}(z-B_z)^{b_z}. \quad (3)$$

Integrals arising from shells of basis functions lead to all possible bras $(\boldsymbol{ab}|$, of which there are $\binom{a+2}{2}\binom{b+2}{2}$ in number. Such a set of bras will be called a *class*, and denoted $(ab|$. It is easily shown that the members of the class $(ab|$ may be expanded as linear combinations of the classes $(m0|$, $a \leqslant m \leqslant a+b$, with the expansion coefficients depending only on the components of the vector $\boldsymbol{A}-\boldsymbol{B}$. Of course, this applies equally to the electron-2 part of the integrand, and therefore the desired integrals $(\boldsymbol{ab}|\boldsymbol{cd})$ are expressible as linear combinations of $(\boldsymbol{m0}|\boldsymbol{n0})$ integrals, $a \leqslant m \leqslant a+b$, $c \leqslant n \leqslant c+d$.

This observation alone may appear rather trifling, but it fact it can be used to great computational advantage. Since the transformation between the two sets of integrals involves nothing more than components of intercenter distances, it can be carried out *after* contraction of the primitive integrals has been performed. There are in general many fewer $(\boldsymbol{m0}|\boldsymbol{n0})$ integrals than $(\boldsymbol{ab}|\boldsymbol{cd})$ integrals, and therefore by utilizing this transformation, both the number and complexity of the primitive integrals which must be formed are diminished.

This approach was first used by Head-Gordon and Pople (HGP) [5], as an improvement on the Obara–Saika (OS) method [4]. The HGP method, which contracts the primitive integrals midway through the algorithm, represented a dramatic step forward from previous methods, most of which perform *all* transformation work on the primitives before contracting them at the very end [1–4]. At the other end of the spectrum is an older by Pople and Hehre [15] which contracts at a very early stage. The concept of introducing contraction at a strategic point has been fully generalized in the PRISM algorithm [6,7,11,12].

Since this transformation is performed on fully contracted integrals, its cost in terms of CPU time becomes insignificant compared to that of constructing the $(\boldsymbol{m0}|\boldsymbol{n0})$ integrals as the degree of contraction increases, especially if the basis set has low angular momentum. However, for moderately to weakly contracted basis sets with high angular momentum, such as the popular 6-31G* basis, the expense of this step is non-negligible, and it is important that it be carried out as efficiently as possible. This is the central topic of this Letter, and we begin our consideration of the problem by examining a previous implementation.

## 2. The one-unit transfer relation

As previously mentioned, the $(\boldsymbol{ab}|\boldsymbol{cd})$ integrals may be written as linear combinations of $(\boldsymbol{m0}|\boldsymbol{n0})$ integrals, and transforming these in the most economical way requires investigating all possible ways of evaluating these linear combinations. This is a difficult task, and in their implementation HGP did not address this issue; rather, they merely used the identity

$$(x-A_x)^{a_x}(x-B_x)^{b_x} = (x-A_x)^{a_x+1}(x-B_x)^{b_x-1} + (A_x-B_x)(x-A_x)^{a_x}(x-B_x)^{b_x-1}, \quad (4)$$

and its analogues for $y$ and $z$ to effect the transformation. Eq. (4) is a device for transferring one unit from the exponent of $x-A_x$ to the exponent of $x-B_x$, and thus we refer to (4) and its $y$ and $z$ versions as the *one-unit transfer relation*, abbreviated TR1. In

[#1] In the conventional definition of a shell, the total angular momentum is allowed to assume a range of values. For our concerns, though, it turns out that it is essentially no restriction to disregard this possibility; the cases in which this occurs and the transformation is nontrivial are relatively rare, and the problem is actually simpler in these cases than if only one angular momentum value were allowed.

[#2] This definition of a bra is inconsistent with our previous definition in ref. [7], which is quite general in that it encompasses contracted Cartesian Gaussian basis functions and their derivatives. However, the Cartesian angular momentum factors of the basis functions are the sole relevant items in the problem considered here, so for the purpose of discussion in this Letter only, it is convenient to redefine the term accordingly.

bra notation, TR1 may be concisely written as

$$(\boldsymbol{ab}| = (\boldsymbol{a}+\mathbf{1}_i, \boldsymbol{b}-\mathbf{1}_i| + (A_i - B_i)(\boldsymbol{a}, \boldsymbol{b}-\mathbf{1}_i| \,, \qquad (5)$$

where $i$ represents a Cartesian direction ($x$, $y$ or $z$), and $\mathbf{1}_i$ is a unit vector in the $i$ direction. The two bras which are combined to form $(\boldsymbol{ab}|$ are called the *i parents*, or simply *parents*, of $(\boldsymbol{ab}|$. This notation is deliberately suggestive of ERI notation, because any $|\boldsymbol{cd})$ may be attached to the three bras in eq. (5) to obtain a valid relationship between ERIs.

Historically, TR1 was simply termed the "transfer relation" [3] by Rys, Dupuis and King, who employed it to construct the $I_x$, $I_y$ and $I_z$ integrals of the Rys quadrature scheme [1]. HGP referred to it as the "horizontal recurrence relation" [5], and were the first to apply it to contracted integrals. This powerful utilization of TR1 has motivated its synthesis into subsequent algorithms [7–10,12].

In the HGP method, the $(\boldsymbol{ab}|$ (bra) and $|\boldsymbol{cd})$ (ket) transformations are carried out separately, i.e. the $(\boldsymbol{m0}|\boldsymbol{n0})$ integrals are fully transformed to $(\boldsymbol{ab}|\boldsymbol{n0})$ integrals, which are then transformed to the $(\boldsymbol{ab}|\boldsymbol{cd})$ integrals. For simplicity in the computer implementation, it is convenient to follow HGP in this regard. Thus, we will seek efficient ways to perform the one-electron transformations; these are then equally applicable to both sides of the ERIs. It turns out that this is no restriction when $b, d \leqslant 1$, but in general this represents a constraint upon the problem of how best to evaluate the linear combinations.

Since the two transformations are equivalent, we consider only the bra-transformation. Fig. 1 is a *class tree* for $(a2|$, the first class for which the transformation is nontrivial. Each class except the $(m0|$ classes is connected to two others by arrows; these arrows indicate the classes to which the parents of members of the attached class belong. A similar diagram, having $b+1$ levels, can be drawn for any class $(ab|$. TR1 is used to build the angular momentum on center $B$, starting at the bottom with the $(m0|$ classes and proceeding up the tree one level at a time, until the members of the target class $(a2|$ are obtained.

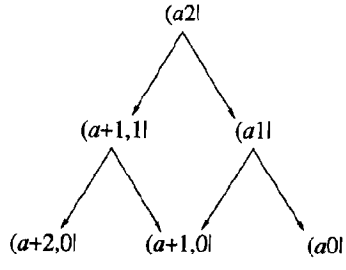


Fig. 1. Class tree for $(a2|$. The arrows from each class show the classes to which the parents of its members belong.

In assembling the target classes, HGP compute the entirety of every class in the appropriate class trees, i.e. all integrals $(\boldsymbol{ij}|\boldsymbol{n0})$, $a \leqslant i \leqslant a+j$, $0 \leqslant j \leqslant b$; $c \leqslant n \leqslant c+d$, and $(\boldsymbol{ab}|\boldsymbol{kl})$, $c \leqslant k \leqslant c+l$, $0 \leqslant l \leqslant d$ are computed. We refer to this method as the *complete* TR1 method. This is an inefficient implementation of the recurrence. We see from eq. (5) that a bra $(\boldsymbol{ab}|$ can be constructed by TR1 in up to three different ways, according to the number of nonzero components of $\boldsymbol{b}$. Of course, only one set of parents is required to form any particular bra, and it turns out that many intermediate bras are unnecessary. Therefore, a tree-search problem must be solved to determine the optimal applications of TR1 which lead to the target bras most efficiently, as is the case with other recurrence-relation-based transformations as well [14].

At this point it must be specified what is meant by "optimal". It has been the conventional wisdom that an optimal algorithm is one requiring the fewest floating-point operations, or flops [16]. We have argued [17] that this criterion is no longer well-suited to most modern computer architectures, and that a more appropriate general-purpose theoretical cost parameter is the total number of memory operations, or mops (loads+stores). Therefore, *we will seek tree-search solutions which minimize the total number of mops, rather than flops.*

Each application of TR1 (5) requires four mops and two flops in the general case $(A_i - B_i) \neq 0$. Therefore, *if TR1 is the only recurrence relation used,* minimizing the total number of mops (and flops) simply amounts to minimizing the number of intermediate bras. The TR1 tree-search problem has been previously studied by Ryu, Lee and Lindh (RLL) [18], who found approximate solutions by neglecting the intrinsic coupling of the vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ in the target class $(ab|$ via the first term of TR1. We have derived rigorously optimal solutions to this problem for $a$, $b \leqslant 3$ by various means, including exhaustive computer search for the higher angular momentum cases. We will not discuss the methods for obtaining

these, as we have found superior solutions which employ additional recurrence relations; rather, only the optimal costs will be given.

Table 1 gives the cost in mops of the $(m0|$ to $(ab|$ transformation by the complete and optimized TR1 methods, for all $a$, $b \leqslant 3$ (corresponding to s, p, d or f functions in the target class). We do not list the costs of RLL [18], as these fall between the complete and optimized TR1 costs. For $b=1$, the two methods are equivalent because in this case there are no intermediates, and hence no tree-search problem and no room for improvement. For $b>1$, there are unnecessary intermediates in the complete method, and optimizing the intermediates yields an improvement of 10%–30%, which is greatest for the larger values of $a$ and $b$. To compute from table 1 the total cost of the transformation for a particular integral class, we must account for the fact that the $|n0)$ classes act as "spectators" during the bra-transformation, and likewise for $(ab|$ during the ket-transformation. For example, the transformation cost of a $(dd|dd)$ integral class $(a=b=c=d=2)$ by the optimized TR1 method is $300\times(15+10+6)+36\times300=20100$ mops.

The transformation costs in flops are also given in table 1. These are redundant for the TR1 methods, but will prove quite interesting in comparison with those of the new method which we will propose.

## 3. Multi-unit transfer relations

Optimizing the TR1 recurrence by removing as many redundant intermediates as possible represents the last word in efficiency for a TR1-based scheme, but is this the best way overall to implement the transformation? The answer is no. Though the target bras and the parents from which they are built change at each tier of the class tree, the distance components $A_i-B_i$ which are involved are the same. This suggests that building the angular momentum of center $B$ only one unit at a time may be inefficient, since mops must be used to load one of the $A_i-B_i$ for each unit which must be transferred. From this standpoint, it would be desirable to perform more work with $A_i-B_i$ once it has been loaded before relinquishing it.

Let us consider the following new recurrence relation:

$$(\boldsymbol{ab}| = (\boldsymbol{a}+\boldsymbol{2}_i, \boldsymbol{b}-\boldsymbol{2}_i| + (A_i-B_i)[(\boldsymbol{a}+\boldsymbol{1}_i, \boldsymbol{b}-\boldsymbol{2}_i| + (\boldsymbol{a}, \boldsymbol{b}-\boldsymbol{1}_i|], \quad (6)$$

where $\boldsymbol{2}_i$ denotes two multiples of the unit vector in the $i$ direction. This relates a bra to another having two more units on center $A$ and two fewer on center $B$, and thus we call it a *two-unit transfer relation*, or TR2. What is the relative merit, if any, of this transfer relation over TR1? Its cost is five mops per ap-

Table 1
Costs in mops and flops of $(m0|$ to $(ab|$ transformation by various methods

| $a$ | $b$ | mops | | | flops | | |
|---|---|---|---|---|---|---|---|
| | | complete TR1 | optimized | | complete TR1 | optimized | |
| | | | TR1 | TR$n$ | | TR1 | TR$n$ |
| 0 | 1 | 12 | 12 | 12 | 6 | 6 | 6 |
| 1 | 1 | 36 | 36 | 36 | 18 | 18 | 18 |
| 2 | 1 | 72 | 72 | 72 | 36 | 36 | 36 |
| 3 | 1 | 120 | 120 | 120 | 60 | 60 | 60 |
| 0 | 2 | 72 | 60 | 36 | 36 | 30 | 33 |
| 1 | 2 | 180 | 152 | 108 | 90 | 76 | 99 |
| 2 | 2 | 336 | 300 | 216 | 168 | 150 | 198 |
| 3 | 2 | 540 | 492 | 360 | 270 | 246 | 330 |
| 0 | 3 | 256 | 180 | 84 | 128 | 90 | 104 |
| 1 | 3 | 564 | 396 | 252 | 282 | 198 | 312 |
| 2 | 3 | 996 | 756 | 504 | 498 | 378 | 624 |
| 3 | 3 | 1552 | 1204 | 840 | 776 | 602 | 1040 |

plication, between the cost of one and two applications of TR1. It accomplishes the same result as the following two successive applications of TR1

$$(\boldsymbol{a}+\mathbf{1}_i, \boldsymbol{b}-\mathbf{1}_i| = (\boldsymbol{a}+\mathbf{2}_i, \boldsymbol{b}-\mathbf{2}_i| + (A_i-B_i)(\boldsymbol{a}+\mathbf{1}_i, \boldsymbol{b}-\mathbf{2}_i| , \quad (7)$$

$$(\boldsymbol{ab}| = (\boldsymbol{a}+\mathbf{1}_i, \boldsymbol{b}-\mathbf{1}_i| + (A_i-B_i)(\boldsymbol{a}, \boldsymbol{b}-\mathbf{1}_i| . \quad (8)$$

Comparing eq. (6) with eqs. (7) and (8), we see that using TR2 (6) requires $A_i-B_i$ to be loaded only once instead of twice, and thus TR2 performs more transfer work than TR1 per load of an intercenter distance component. The intermediate bra $(\boldsymbol{a}+\mathbf{1}_i, \boldsymbol{b}-\mathbf{1}_i|$ is bypassed by TR2, making it possible to remove it from the transformation altogether. This is interesting because it implies the class $(ab|$ can be built form the $(m0|$ classes using fewer intermediates than the minimum possible using TR1 alone.

It is important to point out that just because eq. (6) achieves the same result as two TR1 transfers in fewer mops, it does not necessarily mean that using it instead of TR1 automatically results in a lower *total* cost. An intermediate bra represents an arithmetic subexpression occurring in the linear expansion of a particular target bra, and such a subexpression can be common to more than one expansion. For example, a bra can be used as a parent for generating up to three other bras by TR1, but of course the cost of forming the parent is exacted only once. Therefore, the price paid for its inclusion in the tree can potentially be recovered by allowing as many as three other bras to be formed by a less expensive TR than would otherwise be possible. On the other hand, if we imagine, e.g., an $(a2|$ tree in which an intermediate serves as a parent of only one bra, it is obvious that removing the intermediate is a winning proposition, because although this dictates that the target bra be formed by the TR2 instead of TR1, raising its cost by one mop, avoiding the evaluation of the parent by TR1 saves four mops, a net improvement.

This illustrates the opposing factors which must be considered when optimizing transformations with more than one TR. The cost of incorporating any intermediate must be weighed against its potential to lower the cost of other bras. Since the TRs involved vary in cost, it is no longer necessarily true that minimizing the number of mops is equivalent to minimizing the number of intermediates. It is now evident that this leads to a new, more difficult tree-search problem.

Given that there exists a TR2 which shows promise for improving the efficiency of TR1-only transformations, it should first be asked if there are also other multi-unit TRs which could be useful before attacking the new tree-search problem. To answer this question, a systematic way to derive all possible TRs is needed. HGP derived TR1 by judiciously combining two versions of the OS eight-term recurrence relation to obtain a massive cancellation of terms; however, the basic essence of TR1 is most easily arrived at by canceling all common factors from both sides of eq. (4), yielding (for $x$).

$$(x-B_x) = (x-A_x) + (A_x-B_x) . \quad (9)$$

This shows that TR1 should be interpreted as the "expansion" of $x-B_x$ about the point $A_x$, and more significantly, as the simplest (linear) member of the family of Cartesian multinomial expansions. By considering expansions of higher-order Cartesian multinomials, multi-unit TRs are obtained.

To facilitate the derivation of an exhaustive set of TRs, it is convenient to represent them graphically as paths on a class tree. The diagram begins at the top or root node of the class tree, which represents the target bra. The first step is to expand a particular linear factor in the target using eq. (9) (i.e. apply TR1 in reverse to replace a bra by its parents). This process is indicated by drawing lines from the root node to nodes in the two parent classes immediately below it in the class tree, and marking the expanded node with the appropriate Cartesian direction. Therefore, the diagrammatic representation of TR1 is


. (10)

The left and right nodes are, respectively, the first and second terms on the right-hand side of eq. (5). Higher-order expansions are constructed by recursive expansion of the end nodes in exactly the same manner, operating on linear factors one at a time. The order of a TR is given by the maximum number of levels descended by its graph. Following a left-

pointing branch marked $i$ transfers one unit from $B$ to $A$ in the $i$ direction (again, note here the TR is being constructed in reverse), while following a right-pointing branch removes one unit from $B$ and multiplies by a factor of $A_i - B_i$. Distinct paths from the root node to the same class will coincide at the same node if the set of Cartesian indices labeling nodes where a right branch is taken is the same for each path. An end node accessible by multiple paths signifies that the corresponding term in the TR has a multiplicity equal to the total number of distinct paths to the node. This is sufficient information for constructing a TR from its graph, and since all possible graphs of a given order can be straightforwardly generated by hand, this method does indeed allow for easy systematic determination of all TRs of a desired order.

Fig. 2 shows the graphs of the six different TR2s which, when taken together with TR1, constitute the entire set of TRs which are applicable to the $(a2|$ tree; the TR2 presented in eq. (6) corresponds to the third graph. (By following the above procedure the reader should be able to verify that this is indeed the case). The costs shown are easily obtained from the graphs as the number of end nodes (which is the number of bras on the right-hand side of the TR), plus the number of distinct Cartesian indices (which is the number of intercenter distance components), plus one (the store of the target bra).

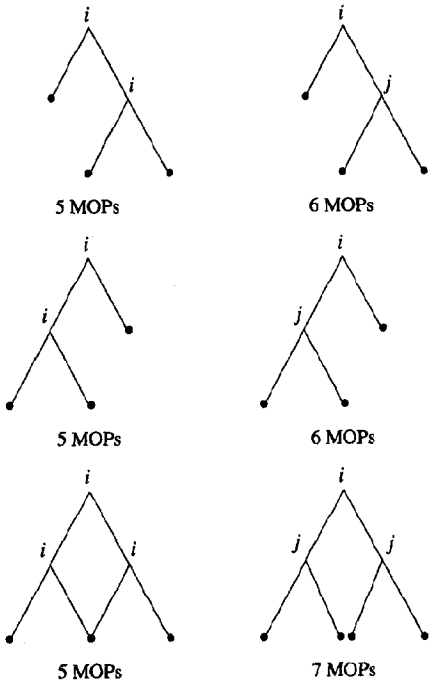


Fig. 2. Graphical representation of the set of all TR2s applicable to the $(a2|$ tree-search problem.

A preliminary simplification of the tree-search problem can be made by observing that not all of the seven TRs are needed. Al three unidirectional TR2s in fig. 2 have the same cost (five mops), but the third performs the entire two-unit transformation while the other two do not. Therefore, the first two TR2s are strictly inferior to the third and need not be considered any further. The tree-search problem involving the remaining five TRs was approached by answering the following question: Is there any intermediate or set of intermediates whose inclusion in the tree is strategic enough to overcome its own cost? It can be shown for $(a2|$ that, perhaps somewhat surprisingly, the answer is no. (A recursion-based proof of this is straightforward but tedious, and hence is omitted here.) Therefore, the optimal transformation uses only the two TR2s at the bottom of fig. 2, which when written out [#3] are

$$(\boldsymbol{ab}| = (\boldsymbol{a}+2_i, \boldsymbol{b}-2_i| + (A_i - B_i)[2(\boldsymbol{a}+1_i, \boldsymbol{b}-2_i| + (A_i - B_i)(\boldsymbol{a}, \boldsymbol{b}-2_i|\,]\,, \tag{11}$$

$$(\boldsymbol{ab}| = (\boldsymbol{a}+1_i+1_j, \boldsymbol{b}-1_i-1_j| + (A_i - B_i)[(\boldsymbol{a}+1_j, \boldsymbol{b}-1_i-1_j| + (A_j - B_j)(\boldsymbol{a}, \boldsymbol{b}-1_i-1_j|\,] + (A_j - B_j)(\boldsymbol{a}+1_i, \boldsymbol{b}-1_i-1_j|\,. \tag{12}$$

[#3] The TRs are written with maximum factorization of the intercenter distance components from the bras and integer factors, which is appropriate for minimizing the number of mops required. It is important to note that a TR may need to be expressed differently for mop optimality than for flop optimality. For example, implementing eq. (11) with the fewest flops would involve writing the linear combination of bras with no factorization; since the coefficients $2(A_i - B_i)$ and $(A_i - B_i)^2$ involve only nuclear positions, they could be precomputed at insignificant cost and then reused as needed, giving an effective cost of four flops per application rather than five as implied by eq. (11). However, this is not useful for saving mops, because it would require two intercenter distance terms to be loaded instead of one. The expression in eq. (11) is therefore the best one, even though it requires one more multiplication than the minimum possible.

As this is a very special type of transformation, we shall in general refer to the use of only the TRs derived from complete expansion of the $n$th-order multinomials (i.e. no intermediates) as the TR$n$ method.

Why does the minimum-mops solution have no intermediates? As previously mentioned, the utility of a given TR relative to the others available is governed by a trade-off between its cost versus the extent of the transformation it accomplishes. When the mop is the unit of cost, then eqs. (11) and (12) are quite attractive, because they perform as much work as three applications of TR1, the least expensive TR, for less than twice its cost (five and seven mops, respectively, versus four for TR1). Since the cost of TR1 is the cost of an intermediate, it is therefore qualitatively reasonable that number of intermediates in the optimal tree would be few or perhaps none. On the other hand, if flops are counted the balance changes drastically, with TR1 costing only two flops while eqs. (11) and (12), at four and six flops, respectively, have become relatively more expensive. Since intermediates are now very cheap, it would be expected that higher-order TRs would not be extremely helpful, and that not much gain would be possible beyond optimized TR1.

Upon proceeding to $b=3$, a similar analysis did not reveal any solution of lower cost than that of the TR$n$ method. Though not rigorously proven optimal, it can be said that TR$n$ is near optimal for $b=3$ in the sense that the addition of any number of bras to a single intermediate class cannot improve the total cost. The three relevant TR3s are

$$
\begin{aligned}
(\boldsymbol{ab}| &= (\boldsymbol{a}+3_i, \boldsymbol{b}-3_i| + (A_i-B_i)\{3(\boldsymbol{a}+2_i, \boldsymbol{b}-3_i| \\
&+ (A_i-B_i)[3(\boldsymbol{a}+1_i, \boldsymbol{b}-3_i| \\
&+ (A_i-B_i)(\boldsymbol{a}, \boldsymbol{b}-3_i|]\}, \qquad (13)\\
(\boldsymbol{ab}| &= (\boldsymbol{a}+2_i+1_j, \boldsymbol{b}-2_i-1_j| \\
&+ (A_i-B_i)\{2[(\boldsymbol{a}+1_i+1_j, \boldsymbol{b}-2_i-1_j| \\
&+ (A_j-B_j)(\boldsymbol{a}+1_i, \boldsymbol{b}-2_i-1_j|] \\
&+ (A_i-B_i)[(\boldsymbol{a}+1_j, \boldsymbol{b}-2_i-1_j| \\
&+ (A_j-B_j)(\boldsymbol{a}, \boldsymbol{b}-2_i-1_j|]\} \\
&+ (A_j-B_j)(\boldsymbol{a}+2_i, \boldsymbol{b}-2_i-1_j|, \qquad (14)
\end{aligned}
$$

$$
\begin{aligned}
(\boldsymbol{ab}| &= (\boldsymbol{a}+1_i+1_j+1_k, \boldsymbol{b}-1_i-1_j-1_k| \\
&+ (A_i-B_i)\{(\boldsymbol{a}+1_j+1_k, \boldsymbol{b}-1_i-1_j-1_k| \\
&+ (A_j-B_j)[(\boldsymbol{a}+1_k, \boldsymbol{b}-1_i-1_j-1_k| \\
&+ (A_k-B_k)(\boldsymbol{a}, \boldsymbol{b}-1_i-1_j-1_k|] \\
&+ (A_k-B_k)(\boldsymbol{a}+1_j, \boldsymbol{b}-1_i-1_j-1_k|\} \\
&+ (A_j-B_j)[(\boldsymbol{a}+1_i+1_k, \boldsymbol{b}-1_i-1_j-1_k| \\
&+ (A_k-B_k)(\boldsymbol{a}+1_i, \boldsymbol{b}-1_i-1_j-1_k|] \\
&+ (A_k-B_k)(\boldsymbol{a}+1_i+1_j, \boldsymbol{b}-1_i-1_j-1_k|\,. \qquad (15)
\end{aligned}
$$

In table 1 the costs in mops and flops of the TR$n$ method are given for the classes considered. Note that the improvement in theoretical cost over the complete method is 33%–67%, substantially greater than that obtained by optimizing the TR1 recurrence. The most interesting feature of table 1, though, is the marked disparity between the cost of the TR$n$ method in mops and flops. When mops are counted, TR$n$ is predicted to be by far the most efficient of the three methods, with the advantage increasing with increasing angular momentum. However, in flops the TR$n$ method is predicted to be an extremely poor one, becoming progressively worse as the angular momentum increases and often even exceeding the complete TR1 method in cost. Given this discrepancy, it is imperative that the performance of the theoretical solutions be tested in a real application.

## 4. Practical performance

Table 2 gives Cray Y-MP/832 CPU timings for the transformation by the three methods, via the HGP-TCCTT path of the PRISM algorithm [11,12] as implemented in GAUSSIAN 92 [19]. Timings for optimized TR1 are estimated from the complete TR1 timings from the ratios of the theoretical costs in table 1; this method does not appear in G92, but would involve executing the same code in the innermost loop as for the complete method, only fewer times according to the number of intermediates omitted, and hence the estimates are accurate.

The first part of table 2 contains timings for a bicubic arrangement of p, d or f shells (there is no transformation for s), and it should be noted that for each system the two timings are proportional to the

Table 2
CPU time for $(m0|n0)$ to $(ab|cd)$ transformation by various methods

| System | CPU time [a] (s) | | |
|---|---|---|---|
| | complete TR1 | optimized | |
| | | TR1 [b] | TR$n$ |
| bicube of p shells [c] | 0.0086 | 0.0086 | 0.0086 |
| bicube of d shells [c] | 0.29 | 0.26 | 0.20 |
| bicube of f shells [c] | 3.52 | 2.73 | 2.37 |
| benzene, 6-31G* | 0.11 | - | 0.099 |

[a] Cray Y-MP/832, one processor, vector mode; one SCF cycle; no point group symmetry was used.
[b] Estimated based on ratio of costs of optimized and complete methods.
[c] Twelve uncontracted shells with exponent 0.8 in bicubic array with side length 1.4 Å.

corresponding mop-costs in table 1, except for the f bicube, where the improvement of TR$n$ over the complete method is a little less than predicted by the mop costs. This is due to the fact that the TR3 relations are rate-limited by multiplications on the Cray In particular, eq. (13) is the next-to-last entry in table 1 of ref. [17] (with $m=n=3$), in which the Cray. performance of this formula was discussed.

There is no direct correspondence between the timings and the flop costs in table 1. As mentioned before, TR$n$ would never be advocated on flop considerations: for example, RLL, whose analysis was based on flops, rejected the possibility of direct summation of the $(m0|$ bras except when $\boldsymbol{b}$ is unidirectional. Yet, the timings for optimized TR1, the most flop-efficient method, are above those for TR$n$, the method clearly prescribed by mops. It is obvious that flops give a particularly poor assessment of the quality of the various methods, and that the use of an appropriate theoretical cost measure is absolutely crucial in approaching this particular problem.

The last row of table 2 gives timings for a more typical example: benzene, 6-31G*, which contains a range of integral classes. For some classes, such as (pp|pp), there is no improvement over the complete method, while the best improvement is obtained in this case on the class (dd|dd). The improvements for the latter two methods are weighted averages of the individual class improvements. Empirically, it is again seen that the TR$n$ method yields the lowest CPU time.

Another aspect of the practical performance which would be briefly discussed is memory usage. Since the TR$n$ method uses no intermediates, only the $(ab|$ and $(m0|$ classes must be stored. PRISM uses a general algorithm for storage allocation which attempts to make optimal use of memory by reassigning locations for temporary quantities as soon as they are no longer needed in the calculation. Since the number of bras in $(ab|$ is always greater than or equal to the number of bras in the $(m0|$ classes, the entire transformation is carried out within the memory used to store the final $(ab|$ bras. This is of course the absolute minimum amount of memory which could be used, and both the HGP and RLL methods require more than this.

## 5. Concluding remarks

We have presented a new method, the TR$n$ method, for transforming $(\boldsymbol{m0}|\boldsymbol{n0})$ integrals to $(\boldsymbol{ab}|\boldsymbol{cd})$ integrals efficiently. This method was arrived at by taking the total number of mops as the cost to be minimized rather than the total number of flops as is usually done. The new approach led to radically different tree-search solutions involving no intermediates and new transfer relations, most of which have not previously been used in ERI calculations. The appropriateness of the mops-based approach was borne out by the close correlation of CPU timings of the three methods examined to their costs in mops, with TR$n$ clearly superior both in theory and in practice; on the other hand, this method is predicted to be disastrous by flops.

Although this work was done in the context of improving the efficiency of ERI computation, its scope is not limited to this area. The TRs presented here were derived solely from consideration of Cartesian multinomials, and hence their applicability is entirely independent of the radial distribution of the basis functions (i.e. whether they are Gaussian or not) and also of the one- or two-electron operator involved. These TRs can be used to generate any integrals over one-electron products of Cartesian basis functions situated on two different centers. Some important examples include electrostatic integrals

[20], potential-fitting integrals required by some density functional methods (TR1 has already been applied here [21]), and overlap integrals, whose calculation is the dominant step in some semiempirical methods.

It was shown that by examining the complete family of Cartesian multinomials, useful recurrence relations could be obtained in addition to the simplest such recurrence relation (TR1) already in use. This notion can, of course, be utilized elsewhere as well. For example, we have previously addressed [14] the optimal generation of the one-center integrals of the McMurchie and Davidson ERI method [2]. In this case, the fundamental set of polynomials is the Hermite polynomials, and the recurrence relation is simply an expression of the familiar homogeneous recurrence relation for these polynomials. From expansions of three-dimensional products of Hermite polynomials, additional recurrence relations can be derived which could potentially increase the efficiency of this transformation.

One motivation for the TR$n$ method was to save mops by increasing the amount of work performed per load of an intercenter distance component, which was achieved by maximizing the transformation work on each individual target bra. An alternate approach would be to use an available distance to transform more than one bra at a time. This is less preferable to TR$n$, and we have not considered it, for the following reasons: The innermost loop of the implementation runs over a bath of like-type integrals, executing the same TR for all members of the batch. In this way, excellent vector performance is obtained. Including multiple instances of various TRs within the innermost loop would require a special-case coding for each different class, and furthermore, though this may improve performance marginally for small classes, as the number of individual TR statements in the inner loop increases vector performance will eventually decline, due to insufficient registers to ensure that each distinct quantity need be loaded only once.

Finally, a comment on higher angular momentum cases. The TR$n$ method was proven rigorously optimal for $b=2$, and clearly shown empirically to be best method for $b=3$. As $b$ increases, however, the expense of the higher-order TRs will eventually become great enough relative to low-order TRs so that intermediates will become necessary in minimizing the cost and TR$n$ will be non-optimal. Given the attractive simplicity and excellent practical performance of TR$n$ for lower angular momentum, a pragmatic strategy for higher values of $b$ would be to carry out the transformation in successive TR$n$-like stages of two or three units each using the TR2s and TR3s given here. This will be much more efficient than the complete TR1 method and indeed, most likely more efficient than optimized TR1 were the solutions known, which they are not. The difference in efficiency between this simple procedure and the rigorously optimal solutions is almost certainly not worth the effort which would be required to find them.

**Acknowledgement**

This work was supported by the National Science Foundation under grant number CHEM-8918623. BGJ thanks the Mellon College of Science for a Graduate Fellowship.

**References**

[1] H.F. King and M. Dupuis, J. Comput. Phys. 21 (1976) 144.
[2] L.E. McMurchie and E.R. Davidson, J. Comput. Phys. 26 (1978) 218.
[3] J. Rys, M. Dupuis and H.F. King, J. Comput. Chem. 4 (1983) 154.
[4] S. Obara and A. Saika, J. Chem. Phys. 84 (1986) 3963.
[5] M. Head-Gordon and J.A. Pople, J. Chem. Phys. 89 (1988) 5777.
[6] P.M.W. Gill, M. Head-Gordon and J.A. Pople, Intern. J. Quantum Chem. Symp. 23 (1989) 269.
[7] P.M.W. Gill, M. Head-Gordon and J.A. Pople, J. Phys. Chem. 94 (1990) 5564.
[8] T.P. Hamilton and H.F. Schaefer III, Chem. Phys. 150 (1991) 163.
[9] R. Lindh, U. Ryu and B. Liu, J. Chem. Phys. 95 (1991) 5889.
[10] I. Panas, Chem. Phys. Letters 184 (1991) 86.
[11] P.M.W. Gill and J.A. Pople, Intern. J. Quantum Chem. 40 (1991) 753.
[12] P.M.W. Gill, Advan. Quantum Chem., in press.
[13] P.M.W. Gill, B.G. Johnson and J.A. Pople, Intern. J. Quantum Chem. 40 (1991) 745.
[14] B.G. Johnson, P.M.W. Gill and J.A. Pople, Intern. J. Quantum Chem. 40 (1991) 809.

[15] J.A. Pople and W.J. Hehre, J. Comput. Phys. 27 (1978) 161.
[16] D. Hegarty and G. Van der Velde, Intern. J. Quantum Chem. 23 (1983) 1135.
[17] M.J. Frisch, B.G. Johnson, P.W.M. Gill, D.J. Fox and R.H. Nobes, Chem. Phys. Letters 206 (1993) 225.
[18] U. Ryu, Y.S. Lee and R. Lindh, Chem. Phys. Letters 185 (1991) 562.
[19] M.J. Frisch, G.W. Trucks, M. Head-Gordon, P.M.W. Gill, M.W. Wong, J.B. Foresman, B.G. Johnson, H.B. Schlegel, M.A. Robb, E.S. Replogle, R. Gomperts, J.L. Andres, K. Raghavachari, J.S. Binkley, C. Gonzalez, R.L. Martin, D.J. Fox, C.J. DeFrees, J. Baker, J.J.P. Stewart and J.A. Pople, GAUSSIAN 92 (Gaussian, Inc., Pittsburgh, PA, 1992).
[20] B.G. Johnson, P.M.W. Gill, J.A. Pople and D.J. Fox, Chem. Phys. Letters 206 (1993) 239.
[21] R. Fournier, J. Andzelm and D.R. Salahub, J. Chem. Phys. 90 (1989) 6371.