Memory & Cognition
1986, 14 (6), 478-487

Cognitive addition and multiplication: Evidence for a single memory network

DAVID C. GEARY, KEITH F. WIDAMAN, and TODD D. LITTLE
University of California, Riverside, California

In an experiment using verification task procedures, 100 subjects responded to simple and complex problems of addition and multiplication. Identical structural parameters were found to model reaction time accurately to both addition and multiplication problems. Slope estimates for a memory network parameter did not differ significantly between simple and complex problems within an operation or between addition and multiplication problems. Both complex addition and complex multiplication problems were processed columnwise, with column sums or products being retrieved from an interrated memory network. The two types of complex problems included similar processes for carrying and for encoding of single digits, and both were self-terminated when an error in the units column was encountered. Addition and multiplication facts appear to be retrieved from a single interrelated memory network. A conceptual model for this interrelated network is discussed.

Chronometric analyses of response latencies to arithmetic problems enable the modeling of the processing components involved in solving such problems. Early research found reaction time (RT) for addition (Restle, 1970) and number comparison (Moyer & Landauer, 1967) tasks to be a function of the difference between the two numbers. Restle argued that simple addition problems are processed by transforming the numbers to be added into analog magnitudes, which are represented as distances along an internal number line. According to Restle, addition involves the concatenation of the shorter line segment onto the longer line segment, and the sum is represented by the end point of the concatenated line segments.

Parkman and Groen (1971) found that adult RT to simple addition problems was best predicted by the smaller (minimum, or *min*) of the two addends. The hypothesized process consistent with this result involves setting an internal counter to the larger addend and then incrementing the counter a number of times equal to the smaller addend until a sum is obtained (Groen & Parkman, 1972). However, in Parkman and Groen's study, the sum of the two addends explained nearly as much RT variance as did the smaller addend. Groen and Parkman interpreted this result, coupled with their finding of uniform RT to tie problems (problems with two identical integers), as reflecting direct memory access for most addition facts, with occasional memory retrieval failure for some problems. When memory retrieval failed, adults reverted to the more reliable *min* counting strategy.

Ashcraft and Battaglia (1978) empirically tested the above counting and direct-access/counting models, using verification tasks for simple addition problems. Adult RT to these problems, except when the difference between the stated sum and the correct sum was large ("unreasonable incorrect" split), was best predicted by the square of the correct sum (sum^2). This finding was inconsistent with both the counting and direct-access/counting models. Ashcraft and Battaglia interpreted their results as suggesting that the correct sum for a simple addition problem is obtained through retrieval of the sum from a memory network of addition facts. They conceptualized the memory network as a square matrix with column and row entry nodes for the integers 0 through 9. The correct sum for a given simple addition problem is stored at the intersection of the entry nodal values corresponding to the two addends. Because in their study RT increased exponentially with the size of the correct sum, Ashcraft and Battaglia argued that the matrix is "stretched" in the region of larger sums, resulting in longer vector distances and therefore in longer RTs. RT patterns for primed problems and for more complex addition problems were interpreted as consistent with this network model (Ashcraft & Stazyk, 1981).

Recently, Miller, Perlmutter, and Keating (1984) reported that RT for simple addition and simple multiplication problems was better predicted by the correct product (*prod*) of the problem digits than by the sum^2. This result, and analyses of errors, suggests that both addition and multiplication facts are retrieved from a similar memory network. Widaman, Geary, and Cormier (1986) replicated the finding that the correct product was the better predictor of RT to simple and complex addition problems and argued that the product structural variable is also consistent with retrieval of addition facts from a memory network. In summary, the use of chronomet-

The authors wish to thank Cindy Slominski and Debbie Retter for their assistance with data collection. The present study was supported in part by Grants No. HD-14688 and HD-04612 from the National Institute of Child Health and Human Development to the second author, and by a grant from the Academic Computing Center of the University of California, Riverside. Requests for reprints should be addressed to either the first author, who is now at the Department of Psychology, University of Texas, El Paso, TX 79968, or to the second author, Department of Psychology, University of California, Riverside, CA 92521.

ric procedures in the cognitive arithmetic area has led to the identification of a memory retrieval process involved in solving arithmetic problems.

However, processing of arithmetic problems involves components other than retrieval of facts, such as a carrying operation for complex problems (Ashcraft & Stazyk, 1981). Recently, Widaman et al. (1986) outlined a general theoretical model for the processing components for both simple and complex addition problems. Widaman et al.'s model is an elaboration of Ashcraft's (1982) model, and includes the same basic processing stages: encode, search/compute, decide, and respond. The first stage involves encoding of the problem's operation (e.g., addition) and the initial two integers (e.g., in the units column in complex problems) to be summed. Once the encoded integers are in working memory (Case, 1985), a sum for these integers is obtained through either a counting process or a memory search process. For simple addition involving two single-digit addends, the obtained sum is then compared with the stated sum (for verification tasks), and a decision ("true" or "false") is made and executed. For more complex problems, recycling loops that correspond to the summing of more than one column of numbers and/or more than two rows of integers are included in the model. For multicolumn problems, the encoding and search/compute processes for simple problems are recycled until sums are obtained for each column. This recycling loop may be modified in two ways: First, a carrying operation (Ashcraft & Stazyk, 1981) is required if the preceding column's sum is greater than 9; and second, complex multicolumn problems may be self-terminated if a column error is encountered before the entire problem is processed. That is, if the stated sum for the units column is incorrect, the problem will be exited and the response "false" will immediately be executed.

Widaman et al. (1986) used multiple regression techniques to model the above process strategies. Regression equations were fit to RT data for simple and complex addition problems, with independent structural variables specified for each component process. Widaman et al. found that these regression equations predicted RT to addition problems extremely well. The product and column-wise product for simple and complex problems, respectively, always provided better fit to RT data than did alternative search/compute parameters. Furthermore, variables specifying added elementary component processes (e.g., encoding of digits) always resulted in significant increases in RT variance explained. Finally, Widaman et al. tested the assumption that subjects would use a self-terminating strategy by comparing full-model R^2s for statistical models representing self-terminating strategies with the R^2s for models for exhaustive strategies. The self-terminating strategy models always provided a better fit to RT than did exhaustive strategy models.

Although Widaman et al. (1986) tested their model with addition problems only, the same process components (e.g., carrying), and perhaps the same search/compute parameter (i.e., product), may represent process strategies for other arithmetic operations. Indeed, Parkman (1972) found that RT to simple multiplication problems was best predicted by the same structural variables that best fit RT to simple addition problems (i.e., *min*, *sum*). Parkman argued that multiplication problems are solved through retrieval of the product from a memory network. Furthermore, he hypothesized that the network for multiplication facts is hierarchically related to the network for addition facts (Parkman, 1972).

Results from experiments that presented confusion problems, in which the stated sum is correct for a different operation (e.g., addition) but incorrect for the given operation (e.g., multiplication), supported Parkman's conclusion (Stazyk, Ashcraft, & Hamann, 1982, Experiment 3; Winkelman & Schmidt, 1974). Winkelman and Schmidt presented certain multiplication problems with stated products that were incorrect for multiplication but correct for addition, and other multiplication problems with stated products that were incorrect for both multiplication and addition. RTs and error rates for the former (confusion) problems were higher than those for the latter (nonconfusion) problems. Results from Miller et al.'s (1984) experiment and from confusion experiments (Stazyk et al., 1982; Winkelman & Schmidt, 1974) are consistent with the hypothesis that there is an interrelated memory network for addition and multiplication facts.

However, it has not been shown that *added* elementary processes involved in the solving of addition problems (e.g., encoding and carrying) are necessary to the mental solving of multiplication problems. In fact, no information is available at present regarding the process components involved in the solving of complex multiplication problems. Therefore, in the present study, we sought (1) to further test the hypothesis that addition and multiplication facts are stored in an interrelated memory network, (2) to provide empirical information on process strategies involved in complex multiplication, and (3) to test Widaman et al.'s (1986) model concurrently for simple and complex multiplication problems.

METHOD

Subjects

One hundred undergraduates (45 male, 55 female) who were enrolled in psychology courses at the University of California, Riverside, served as subjects. All subjects received $3 or course credit for participating in this experiment.

Stimuli

A total of 320 arithmetic problems served as stimuli. The global set consisted of 80 problems of each of four types of arithmetic: simple addition, complex addition, simple multiplication, and complex multiplication. These four sets were presented independently and in the above order.

Simple addition. The 80 simple addition problems consisted of two vertically presented integers with a stated sum. Forty of the problems (the "true" problems) were selected from the 90 possible nontie pairwise combinations of the integers 0 through 9 and were presented with the correct sum. The frequency and placement of all integers was counterbalanced. That is, each integer (0 through 9) appeared eight times across the 40 problems and appeared equally often as the first addend and as the second addend. The remaining simple addition problems (the "false" problems) were the same 40

pairs of integers presented with a stated sum incorrect by ±1 or ±2. The magnitude of the error was counterbalanced across the 40 "false" problems. No repetition of either integer or of the stated sum was allowed across consecutive trials, and no more than four consecutive presentations of "true" or "false" problems were allowed.

Complex addition. The 80 complex addition problems consisted of two vertically placed double-digit integers with a stated sum. The 40 "true" problems were constructed from 80 of the 90 possible integers 10 through 99. The larger integer was the first addend for one half of the problems, and the frequency of individual digits 0 through 9 was counterbalanced for position. Within a given problem, all four digits were unique. Finally, the stated sum for the units column was greater than 9, and therefore required a carrying operation, for one half of the problems. The 40 "false" problems were the same 40 pairs of integers presented with a stated sum incorrect by ±1, ±2, ±10, or ±20. The placement of the error was counterbalanced; that is, each value of the error (e.g., +1 or −2) occurred five times, with the constraint that when the sum for the units column was greater than 9, one half of the errors occurred in the units column and one half of the errors occurred in the tens column. No repetition of either addend or of the stated sum was allowed across consecutive trials, and no more than four consecutive presentations of "true" or "false" problems were allowed.

Simple multiplication. The 80 simple multiplication problems consisted of two vertically presented single-digit integers with a stated product. Problems with two identical integers (ties) and problems including the integer 0 were excluded because of inconsistent performance with these problems (e.g., see Stazyk et al., 1982). Accordingly, the "true" simple multiplication problems consisted of the remaining 36 unique combinations of nontie and nonzero problems and 4 randomly selected inverted (e.g., 3 × 7, 7 × 3) problems. The larger value integer was placed in the top position for half of the problems; thus, each unique integer 1 through 9 appeared four or five times in the top position and four or five times in the bottom position across the 40 problems. The 40 "false" problems consisted of the same 40 pairs of integers, but with the stated product deviating from the correct product by ±1, ±2, or ±10. Across the 40 problems, the stated product deviated from the correct product by ±1 or ±2 for 24 problems, and by ±10 for 16 problems. No repetition of either integer or of the stated product was allowed across consecutive trials, and no more than four consecutive presentations of "true" or "false" problems were allowed.

Complex multiplication. The 80 complex multiplication problems consisted of a double-digit multiplicand placed vertically over a single-digit multiplier, presented with a stated product. Multiplicands consisted of a random sample of 40 of the 90 integers from 10 through 99. The integers 1 through 9 served as multipliers. Across the 40 stimuli, the integers 1 through 9 served as the multiplier four or five times each; the units place for the multiplicand contained the integers 0 through 9 four times each; and the tens place for the multiplicand contained the integers 1 through 9 four or five times each. Within each problem all digits were unique. The "false" problems consisted of the same 40 pairs of multiplicands and multipliers, presented with a stated product deviating from the correct product by ±1, ±2, ±10, ±20, or ±100. The placement of these errors was counterbalanced across the stated product columns. That is, there were 14 errors in the units column, 14 errors in the tens column, and 12 errors in the hundreds column. No repetition of the stated product or of multiplicands or multipliers was allowed across consecutive trials, and no more than four consecutive presentations of "true" or "false" problems were allowed.

Apparatus

The arithmetic problems were presented at the center of a 30 cm × 30 cm video screen controlled by an Apple II+ microcomputer. A Cognitive Testing Station clocking mechanism ensured the collection of RTs with ±1 msec accuracy. The subjects were seated approximately 70 cm from the video screen and responded by depressing one of two buttons located on a board directly in front of them. Each subject responded "true" by depressing the button on the side of his/her preferred hand and responded "false" using his/her nonpreferred hand.[1]

For each problem, a READY prompt appeared at the center of the video screen for 500 msec, followed by a 1,000-msec period during which the screen was blank. Then an arithmetic problem appeared on the screen and remained until the subject responded, at which time the problem was removed. If the subject responded correctly, the screen was blank for 1,000 msec, and the READY prompt for the next problem then appeared. If the subject responded incorrectly, a WRONG prompt of 1,000 msec duration followed the removal of the stimulus and preceded the 1,000-msec interproblem blank period.

Procedure

The subjects were tested individually in a quiet room. They were told that they were going to be presented with four individual sets of arithmetic problems in a set order: simple addition, complex addition, simple multiplication, and complex multiplication.[2] They were told that their task was to respond "true" or "false" to the presented problem by pressing the appropriate key. Equal emphasis was placed on speed and accuracy. Subjects were told the type of problem to be presented before each set, and a practice set of eight problems was presented at the beginning of each set. A short rest period followed each set. The entire testing session lasted approximately 45 min.

RESULTS

Overall error rate in the matrix of 32,000 RTs was 4.9% (range 3.2% to 8.5%, across sets), and fewer than 1.0% of the RTs were deleted as outliers (using Dixon's test; Wike, 1971). All analyses excluded these error and outlier RTs. Processing models for simple addition and multiplication were fit to average RT data using multiple regression techniques. Models for the search/compute process fit to RT data included Parkman and Groen's (1971) five counting-based models, Ashcraft's (1982) true-sum-squared parameter, and the correct product (Miller et al., 1984). A truth parameter (coded 0 for correct problems and 1 for incorrect problems) was included in the regression equations. The truth parameter represents RT intercept differences between "true" and "false" problems. For simple addition and multiplication, equations were fit using each of the above search/compute parameters and the truth parameter.

Regression equations for complex addition and multiplication problems were fit according to the processing stages described by Widaman et al. (1986). Specifically, the above search/compute parameters were fit separately for each column sum or column product, and parameters for the number of items encoded (NI) and a self-terminating carrying operation (carryst), as well as the truth parameter, were included in each equation. The NI parameter was coded equal to the total number of digits in the problem, including the stated sum (e.g., 6 or 7 for complex addition problems, but 3 for complex addition problems that were self-terminated because of an error in the units column). The carryst parameter was coded 0 if the units sum was < 10, or if the stated units sum was incorrect. If the stated units sum was correct and

≥ 10, then the carryst parameter was coded 1. All processes following a units-column error were coded 0. The fit of each regression equation was based on the value of the R^2 and the significance of the partial F ratio for each parameter. A partial F ratio tests the significance of the independent variance explained for each structural variable in a regression equation (Ashcraft & Stazyk, 1981; Cohen & Cohen, 1983).

Addition

Simple addition. The three best-fitting regression models for simple addition problems are presented in the top half of Table 1. Inspection of Table 1 reveals that RT to simple addition problems was best predicted by the correct product (*prod*) of the addends along with a truth parameter (R^2 = .737).[3] The *sum*² and *min* structural variables, each accompanied by the truth parameter, were the next best predictors for these problems. Initially, the NI variable was included in each of these equations; however, the partial F ratios were not significant for this parameter, so the NI variable was not included in any of the final equations. The nonsignificance of the NI variable likely resulted from a lack of variance in the number of integers in simple addition problems (i.e., 3 or 4). Dropping the NI variable from the equations resulted in the incorporation of encoding speed into the intercept value. Across the three equations, intercept values were highly similar and the regression weights for the truth variable were identical. Identical regression weights for truth suggest that this parameter was orthogonal to the search/compute process. The independence of components included in each equation was supported by the finding that no interaction (e.g., *prod* × truth) was significant for any of the equations [for equations including the *prod*, *sum*², and *min* variables, respectively, $F(1,76)$ = 0.00, 0.05, and 0.01; $p > .50$ for all F ratios].

Complex addition. In the bottom half of Table 1, the three best-fitting equations for complex addition problems are presented. Inspection of Table 1 reveals that all of the processing components (NI, carryst, truth) proposed by Widaman et al. (1986) had highly significant ($p < .01$) F ratios for all three equations. The equation that specified columnwise processing of problems with the columnwise product as the search parameter, along with the above process components, provided the best fit to complex addition RT (R^2 = .867). In each of the three equations, the search/compute parameter was initially estimated separately for each column. Inspection of these results revealed highly similar columnwise slope estimates. Accordingly, the columnwise slope estimates for the units and tens columns were constrained to be equal. We evaluated the significance of this equality constraint by using an incremental F test (Cohen & Cohen, 1983) to test the decrease in R^2 associated with enforcing the equality constraints. Constraining columnwise slope estimates to be equal resulted in nonsignificant decreases in the full-model R^2 for all three equations [for equations including *prod*, *sum*², and *min* variables, respectively, $F(1,75)$ = 0.17,

Table 1
Statistical Summaries of Regression Analyses: Addition

Equation	R^2	F	*df*	*MSe*
Simple Addition				
RT = 888 + 9.87 (prod) + 156 (truth) Partial F: 180.54, 35.0	.737	107.77	2,77	118.41
RT = 832 + 2.54 (sum²) + 156 (truth) Partial F: 141.28, 29.66	.689	85.47	2,77	128.62
RT = 866 + 79 (min) + 156 (truth) Partial F: 134.95, 28.80	.681	81.88	2,77	130.52
$\overline{RT}$ = 1,150.98 msec				
Complex Addition				
RT = 727 + 179 (NI) + 8.04 (unitprod) + 367 (carryst) + 8.04 (tenprod) + 232 (truth) Partial F: 91.40, 54.46, 91.40, 54.46, 18.82	.867	122.38	4,75	197.79
RT = 743 + 165 (NI) + 2.09 (unitsum²) + 317 (carryst) + 2.09 (tensum²) + 223 (truth) Partial F: 71.91, 52.91, 23.59, 52.91, 17.20	.866	120.68	4,75	198.98
RT = 721 + 170 (NI) + 67 (unitmin) + 378 (carryst) + 67 (tenmin) + 231 (truth) Partial F: 73.98, 46.93, 43.47, 46.93, 17.65	.859	114.17	4,75	203.80
$\overline{RT}$ = 2,271.80 msec				

Note—All models significant at the $p < .0001$ level. All partial F ratios significant at the $p < .01$ level. Prod = product; truth = intercept differences between "true" and "false" problems; sum² = square of the correct sum; min = smaller of the two addends; NI = number of items encoded; unitprod = units-column product; carryst = self-terminating carrying operation; tenprod = tens-column product; unitsum² = units-column sum²; tensum² = tens-column sum²; unitmin = units-column min; tenmin = tens-column min.

0.50, and 0.16; $p > .50$ for all F ratios]. Identical slope estimates are therefore presented in Table 1 for the units and tens columns.

Across the three equations, the component processes required in the model were the same, and the regression estimates were highly similar. The search/compute strategy × truth interactions were estimated separately for each column slope, and the significance of these interactions was tested using an incremental F test (Cohen & Cohen, 1983). The addition of these interactions to the three regression equations failed to increase the full-model R^2 significantly for any equation [for equations including *prod*, *sum*^{2}, and *min* variables, respectively, $F(2,72) = 2.00$, $p > .10$; $F(2,72) = 1.12$, $p > .25$; $F(2,72) = 1.88$, $p > .10$].

Multiplication

Simple multiplication. The three best-fitting equations for simple multiplication problems are presented in the top half of Table 2. Inspection of Table 2 reveals that RT to simple multiplication problems was best predicted by the *prod*, along with the truth parameter ($R^2 = .721$). The smaller of the multiplicand and multiplier (*min*) and the *sum*^{2}, each accompanied by the truth parameter, were the next best predictors for these problems. The NI parameter was initially included in each of these equations. However, as with simple addition problems, the partial F ratios for the NI parameter were not significant, so NI was not included in the final equations. Speed of encoding was therefore encompassed within the intercept value. All verification (truth) processes again appeared to be orthogonal to the search/compute process; this conclusion was supported by the finding that, once again, no interaction was significant for any of the equations [for equations including the *prod*, *sum*^{2}, and *min* variables, respectively, $F(1,76) = 1.27$, 0.93, and 1.25; $p > .25$ for all F ratios].

Complex multiplication. The three best-fitting equations for complex multiplication problems are presented in the bottom half of Table 2. Inspection of Table 2 reveals that all but one of the processing components (NI, for the last equation) proposed by Widaman et al. (1986) for complex addition problems had significant partial F ratios ($p < .05$) across these three equations. In addition to the above components, a carrying remainder parameter (carrem, coded the value of the remainder following the units-column multiplication) was fit in each of the equations. The carrem reflects the number of units that must be incremented onto the product of the multiplier and the tens-column digit of the multiplicand in order to give the correct answer in the tens and hundreds columns of the problem. Consider the problem 27 × 6; following the units-column multiplication (7 × 6), the remainder of this operation (4) must be held in working memory during the tens-column multiplication (2 × 6), and then this remainder (4; the carrem) must be added to the provisional

Table 2
Statistical Summaries of Regression Analyses: Multiplication

Equation	R^2	F	df	MSe
Simple Multiplication				
RT = 903 + 10.06 (prod) + 155 (truth) Partial F: 170.92, 23.09	.721	99.50	2,77	131.44
RT = 838 + 92 (min) + 155 (truth) Partial F: 159.64, 26.81	.707	93.23	2,77	134.53
RT = 864 + 2.55 (sum^2) + 155 (truth) Partial F: 141.96, 24.80	.684	83.38	2,77	139.86
$\overline{RT}$ = 1,231.62 msec				
Complex Multiplication				
RT = 1,143 + 105 (NI) + 13.53 (unitprod) + 481 (carryst) + 13.53 (tenprod) + 160 (carrem) + 191 (truth) Partial F: 5.17, 31.38, 12.88, 31.38, 14.84, 4.57	.878	104.78	5,74	362.02
RT = 1,086 + 103 (NI) + 2.89 (unitsum^2) + 541 (carryst) + 2.89 (tensum^2) + 180 (carrem) + 209 (truth) Partial F: 4.41, 22.38, 14.59, 22.38, 17.21, 4.97	.866	94.48	5,74	378.72
RT = 1,152 + 83 (NI) + 114 (unitmin) + 361 (carryst) + 114 (tenmin) + 201 (carrem) + 182 (truth) Partial F: 2.71, 21.52, 6.50, 21.52, 24.02, 3.75	.865	93.50	5,74	380.44
$\overline{RT}$ = 2,839.56 msec				

Note—All models significant at the $p < .0001$ level. All partial F ratios, except the F ratio for NI in the final equation, significant at the $p < .05$ level. Prod = product; truth = intercept differences between "true" and "false" problems; min = smaller of the multiplicand and multiplier; sum^2 = square of the correct sum; NI = number of items encoded; unitprod = units-column product; carryst = self-terminating carrying operation; tenprod = tens-column product; carrem = value of the remainder following the units-column multiplication; unitsum^2 = units-column sum^2; tensum^2 = tens-column sum^2; unitmin = units-column min; tenmin = tens-column min.

tens-column product (12) to complete the problem. The carrem parameter showed highly significant partial F ratios for all three equations.

The equation fit with the columnwise product and the above process components provided the best fit to complex multiplication RT ($R^2 = .878$). The equations were initially fit with the search/compute parameter estimated separately for each column; again, results revealed highly similar columnwise slope estimates in each analysis. Constraining the columnwise slope estimates to be equal resulted in a nonsignificant decrease in the full-model R^2 for each of the three equations [for equations including *prod*, *sum*^{2}, and *min* variables, respectively, $F(1,74) = 0.02, 0.16$, and 0.93; $p > .25$ for all F ratios]. Identical slope estimates are therefore presented in Table 2 for the units and the tens columns.

Process component and intercept estimates showed somewhat greater variability across the three equations than did the comparable estimates from complex addition problems. However, despite some variability in component speed estimates across equations, the same encoding, truth, and carrying components were required across the three equations. The preceding component processes for complex multiplication were identical to the processes fit for complex addition. The search/compute strategy × truth interactions were estimated separately for each column slope, and the significance of the interactions was tested using an incremental F test. The addition of these interactions to the regression equation did not significantly increase the full-model R^2 for any equation [for equations including *prod*, *sum*^{2}, and *min* variables, respectively, $F(2,71) = 2.29$, $p > .05$; $F(2,71) = 0.82$, $p > .25$; $F(2,71) = 0.82$, $p > .25$].[4]

Split Effects

Previous results using verification procedures have indicated that RT may vary as a function of the difference in magnitude, or split, between the correct sum and the stated sum in "false" problems (Ashcraft & Battaglia, 1978; Krueger & Hallford, 1984). Ashcraft and Battaglia reported a monotonic decrease in RT as the size of the split increased (i.e., for "unreasonable incorrect" stated sums). Furthermore, the split effect may also be influenced by the use of an odd-even heuristic (Krueger & Hallford, 1984). In the present study, only columnwise stated sums or products with "reasonable incorrect" values (i.e., ± 1 or ± 2 per column) were used. Therefore, the split effect found in previous research (Ashcraft & Battaglia, 1978; Ashcraft & Stazyk, 1981) might not be evident in responses to these stimuli, as Krueger and Hallford (1984) reported no difference between splits of ± 1 and splits of ± 2. To test the significance of the magnitude of the columnwise split, we added independent structural variables that were coded the size of the split for each column to the full-model regression equations for each problem type. To test each split parameter, we tested the significance of the increase in the full-model R^2 associated with each added split variable. These results indicated that the size of the columnwise split for all four problem types and across all columns was never significant (all $ps > .10$).

Combined Analysis

As hypothesized, inspection of Tables 1 and 2 reveals that identical structural variables and highly similar slope estimates predicted RT to both simple and complex addition and multiplication problems. If these estimates do not differ significantly, this would suggest identical encoding, memory search, and verification processes for both addition and multiplication, and for both simple and complex problems. To test this hypothesis, we combined data from simple and complex addition and multiplication problems and evaluated a series of regression models.

Specifically, we combined data sets two at a time and conducted significance tests of the difference in the regression weights for identical structural variables. For the combined analyses, a dummy coded structural variable, type, was added to the more complex of the two regression equations. The partial F ratio for the type parameter tested intercept differences between the two problem types. Codes for all other structural variables remained the same as those reported for the independent analyses. Partial F ratios for the interaction between the type parameter and the remaining structural variables provided a significance test of the difference between regression weights for each parameter for the two problem types in a given combined analysis. Because of the large number of variables and interactions tested with these analyses, a significance level of $p < .01$ was adopted.

Simple operations. Inspection of Tables 1 and 2 reveals highly similar intercept and slope estimates for simple addition and multiplication. The first combined analyses compared regression weight differences for simple addition and multiplication. In this combined set, the type parameter (coded 0 for addition and 1 for multiplication) was added to the equation with the *prod* and the truth parameters. The partial F ratio for the type parameter tested the intercept difference between addition and multiplication. The partial F ratio for the type × *prod* interaction tested the slope difference between addition and multiplication for the *prod* parameter, and the partial F ratio for the type × truth interaction tested the truth parameter difference.

The regression equation for the combined simple addition and multiplication data sets is presented in the top portion of Table 3. Inspection of the equation reveals that both the *prod* parameter [$F(1,157) = 371.17$, $p < .0001$] and the truth parameter [$F(1,157) = 63.21$, $p < .0001$] were highly significant. The addition of the type parameter and its interactions revealed that the type [$F(1,154) = 0.17$, $p > .25$], type × *prod* [$F(1,154) = 0.03$, $p > .25$], and type × truth [$F(1,154) = 0.00$, $p > .25$] parameters were not significant.

The first equation presented in Table 3 forced the intercept, *prod*, and truth values to be equal for simple addition and multiplication and provided a highly significant fit to the combined data ($R^2 = .735$) [$F(2,157) = 217.19$, $p < .0001$]. Furthermore, allowing the intercept,

prod, and truth estimates to differ for addition and multiplication did not significantly increase the full-model R^2 [$F(3,154) = 0.05, p > .25$]. These results suggest identical encoding, truth, and memory search processes and identical memory search rates for simple addition and multiplication problems.

Simple and complex addition. For a comparison of simple with complex addition, the type parameter was coded 0 for simple addition and 1 for complex addition, and the carryst parameter was coded 0 for simple addition. The NI parameter for simple addition was coded the number of digits in the problem (3 or 4). The regression equation for the combined analysis of simple and complex addition is presented as the second equation in Table 3.

As with simple operations, the addition of the type parameter and the appropriate interactions revealed that the partial F ratios for the type [$F(1,151) = 0.50$, $p > .25$], type × *prod* [$F(1,151) = 0.07, p > .50$], and type × truth [$F(1,151) = 0.80$, $p > .25$] parameters were not significant. However, the type × NI interaction was significant [$F(1,151) = 8.49, p < .01$]. In all, these results suggest highly similar truth parameter and memory search processes and memory search rates for simple and complex addition problems. Widaman et al. (1986) found that speed of encoding integers increased linearly as the number of integers in the problem increased. On the basis of Widaman et al.'s finding and the value estimated for speed of encoding integers for simple addition (50 msec) and complex addition (179 msec) in the present study, we coded the NI parameter in the combined analysis so as to allow encoding speed for complex addition to be estimated as three times the value estimated for simple addition encoding speed. The slower encoding rate for complex addition (64 × 3, or 192 msec per digit) than for simple addition (64 msec per digit) replicates the finding of Widaman et al. (1986).

Simple and complex multiplication. Identical procedures were used to compare simple and complex multiplication problems, with results similar to those reported above. Inspection of the third equation in Table 3 again reveals a single equation that provides a good representation of the combined multiplication RT. Adding the type parameter and interactions revealed nonsignificant partial F ratios for the type [$F(1,150) = 1.40, p > .10$], type × *prod* [$F(1,150) = 2.25$, $p > .10$], type × truth [$F(1,150) = 0.17, p > .50$], and type × NI [$F(1,150)$

Table 3
Statistical Summaries of Regression Analyses: Combined

Equation	R^2	F	*df*	*MSe*
Simple Addition and Multiplication				
RT = 895 + 10.05 (prod) + 156 (truth) Partial *F*: 371.17, 63.21	.735	217.19	2,157	124.2
Simple and Complex Addition				
RT = 694 + 64 (NI)* + 8.01 (prod) +333 (carryst) + 199 (truth) Partial *F*: 763.93, 132.68, 52.30, 53.56	.942	634.14	5,155	169.0
Simple and Complex Multiplication				
RT = 455 + 62 (NI)† + 9.73 (prod) + 419 (carryst) + 202 (carrem) + 228 (truth) Partial *F*: 219.49, 58.86, 15.49, 48.94, 24.05	.931	410.33	5,154	290.3
Complex Addition and Multiplication				
RT = 998 + 141 (NI) + 9.35 (prod) + 352 (carryst) + 250 (carrem) + 188 (truth) Partial *F*: 38.29, 56.31, 27.41, 153.55, 11.35	.874	211.32	5,154	305.5
Simple Addition and Complex Multiplication				
RT = 476 + 61 (NI)† + 9.74 (prod) + 419 (carryst) + 202 (carrem) + 225 (truth) Partial *F*: 222.95, 54.41, 15.83, 48.71, 23.99	.935	443.19	5,154	287.4
Simple Multiplication and Complex Addition				
RT = 704 + 63 (NI)* + 8.26 (prod) + 329 (carryst) + 197 (truth) Partial *F*: 715.52, 145.19, 48.16, 49.89	.934	545.29	5,154	173.8

Note—All models significant at the $p < .0001$ level. All partial F ratios significant at the $p < .01$ level. Structural variables are identical to those described for the individual analyses, with the coding changes. Prod = product; truth = intercept differences between "true" and "false" problems; NI = number of items encoded; carryst = self-terminating carrying operation; carrem = value of the remainder following units-column multiplication. *The regression parameter estimate presented refers to encoding time per digit for the simple operation; thus, encoding time per digit for complex addition is exactly three times the estimate shown. †The regression parameter estimate presented refers to encoding time per digit for the simple operation; thus, encoding time per digit for complex multiplication is exactly two times the estimate shown.

$= 0.32, p > .50$] parameters. As for addition problems, these results suggest highly similar, if not identical, truth parameter and memory search processes and memory search rates for simple and complex multiplication. Furthermore, initial encoding speeds for the two types of multiplication did not appear to differ. However, forcing encoding speeds to be equal resulted in an unreasonably low intercept value for the combined equation. Therefore, on the basis of initial NI estimates (50 msec for simple, 105 msec for complex), we coded the NI parameter in the combined equation so as to allow encoding speed for complex multiplication to be estimated as double the value estimated for simple multiplication encoding speed. Allowing these NI estimate differences yielded a reasonable intercept value for the combined equation presented in Table 3.

Complex operations. Intercept and slope estimates were compared for complex addition and complex multiplication. Table 3 presents a single regression equation for complex addition and multiplication RT. Here, the addition of the type parameter and interactions revealed nonsignificant F ratios for the type [$F(1,149) = 1.83$, $p > .10$], type × NI [$F(1,149) = 2.14, p > .10$], type × carryst [$F(1,149) = 0.93, p > .25$], type × truth [$F(1,149) = 0.04, p > .25$], and type × *prod* [$F(1,149) = 4.91, p > .01$] parameters. These results suggest that identical process components are involved in the solving of both complex addition and complex multiplication problems, and that the speed of execution of these components does not differ significantly between the two operations.

Simple addition and complex multiplication. A procedure identical to that used to compare simple multiplication with complex multiplication was used in this analysis. Inspection of the fifth regression equation in Table 3 reveals that a single equation provided a good fit to the combined RT data for simple addition and complex multiplication ($R^2 = .935$). Adding the type parameter and interactions revealed nonsignificant partial F ratios for the type [$F(1,150) = 1.34, p > .10$], type × *prod* [$F(1,150) = 2.38, p > .10$], type × truth [$F(1,150) = 0.16, p > .50$], and type × NI [$F(1,150) = 0.38, p > .50$] parameters. Although the type × NI interaction was nonsignificant, it was necessary to specify a linear increase in encoding time per digit across problem type (as previously found by Widaman et al., 1986) in order to allow estimation of an intercept term of reasonable magnitude.[5] This equation and the equation for simple and complex multiplication produced nearly identical regression weights for the same parameters, and nearly identical R^2s.

Simple multiplication and complex addition. A procedure identical to that used to compare simple addition with complex addition was used in this analysis. Inspection of the final equation in Table 3 again reveals that a single regression equation provided a good fit to the combined RT data ($R^2 = .927$). The addition of the type parameter and the appropriate interactions revealed that the partial F ratios for the type [$F(1,151) = 0.47, p > .50$], type × *prod* [$F(1,151) = 0.36, p > .50$], and type × truth [$F(1,151) = 0.75, p > .25$] parameters were not significant. The type × NI interaction was, again, significant [$F(1,151) = 9.70, p < .01$], necessitating a linear constraint for the NI parameter across problem types.[6] This equation and the equation for simple and complex addition produced highly similar regression weights for the same parameters, as well as comparable amounts of RT variance explained.

Self-terminating versus exhaustive processing

A separate structural variable for the self-terminating strategy was not included in the regression equations for complex problems. Rather, we tested the validity of this strategy by comparing the fit of equations representing exhaustive processing of problems with the fit of equations reflecting self-terminating processing. If a self-terminating strategy was used, the only process executed following a units-column error would be the response "false." Accordingly, for equations reflecting a self-terminating strategy, in problems with a units-column error, structural variables representing any processes (e.g., carrying, carrem) following the units column were recoded to 0 values. For equations reflecting exhaustive processing, codes of structural variables were left unchanged, thereby representing execution of the processes following a units-column error.

The same structural variables were used in the regression equations for both exhaustive and self-terminating strategies, given the coding changes for the self-terminating processes. As a result, nested comparisons and F tests could not be derived to compare the two types of strategies. However, our assumption that subjects used the self-terminating strategies was based on the change in the full-model R^2s compared with the full-model R^2s under exhaustive processing. Table 4 presents full-model R^2s for complex addition and multiplication problems under exhaustive and self-terminating strategies. Inspection of Table 4 reveals that the self-terminating strategy, compared with the exhaustive strategy, resulted in a mean increase in RT variance explained of 42.2% for complex addition problems and 19.3% for complex multiplication problems. The assumption that subjects used a self-terminating strategy is thus supported for both addition and multiplication problems.

Table 4
Full-Model R^2s Comparing Exhaustive with Self-Terminating Strategies

Column Parameter	Exhaustive	Self-Terminating	Percent Increase In RT Variance Explained
		Addition	
prod	.438	.867	42.9
sum²	.449	.866	41.7
min	.439	.859	42.0
		Multiplication	
prod	.679	.878	19.9
sum²	.674	.866	19.2
min	.676	.865	18.9

Note—Prod = product; sum² = square of the correct sum; min = smaller of the two addends (or multiplicand and multiplier).

DISCUSSION

Converging evidence has suggested the existence of an interrelated memory network for addition and multiplication facts (Miller et al., 1984; Parkman, 1972; Stazyk et al., 1982; Winkelman & Schmidt, 1974). The results of the present study further support this hypothesis, and suggest that it is true for both simple and complex problems. The *prod* parameter, which has been interpreted as representing search distance in long-term memory for multiplication facts (Stazyk et al., 1982) and addition facts (Miller et al., 1984; Widaman et al., 1986), provided the best representation of RT data for both simple and complex forms of addition and multiplication problems. Although the *prod* variable provided the best fit to RT across all four problem types, its fit was only slightly better, in several cases, than that of the *min* and sum^2 parameters. Thus, a strong argument favoring the *prod* variable over the *min* or sum^2 variables cannot be made. Nevertheless, slope estimates for the *prod* structural variable did not differ significantly, either between simple and complex problems within operations or across addition and multiplication operations. The memory retrieval process, therefore, appears to have highly similar, if not identical, search speeds for both addition and multiplication facts.

The *prod* structural variable allows for a conceptual model of the memory network search that is simpler than models proposed previously. Widaman et al. (1986) showed that the *prod* was compatible with a geometric matrix representing a memory network similar to that proposed by Ashcraft and Battaglia (1978). Like Ashcraft and Battaglia, Widaman et al. conceptualized the memory network as a square symmetric matrix, with two orthogonal axes representing nodes for the two integers to be added. However, in Widaman et al.'s model the distance between the nodal values is assumed to be constant, not "stretched" in the region of larger sums, as in Ashcraft and Battaglia's model. The network is entered at the origin, and the rate of activation of the network is assumed to be a constant function of the area of the network activated. The *prod* structural variable represents the total area of the matrix activated, and the *prod* is then linearly related to the search time required to arrive at the correct answer.

A third axis, representing the different operations of addition and multiplication, is a reasonable addition to the model of the memory network for arithmetic facts. The third axis results in a three-dimensional model of simple arithmetic facts. The three dimensions include one dimension, or node, for the first addend (or multiplicand); a second dimension, or node, for the second addend (or multiplier); and a third dimension that specifies the operation performed (addition vs. multiplication). Such a model allows for the representation of arithmetic facts for different operations at different levels in the long-term memory network. Thus, declarative knowledge of the correct answer for the problems $7 + 3$ and 7×3 would be represented in the same region of the matrix, but at different levels. The activation of the three-dimensional matrix would commence with the encoding of presented integers (e.g., 7 and 3) and with the activation of information at all levels of the operator axis. The specification of the correct operation for the stated problem (e.g., addition) would result in preferential activation of the matrix at this level, but remaining levels of the operation axis would be activated in accord with the similarity of the operator to the correct operation for the problem. The retrieved arithmetic fact would then be represented by the activated area of the matrix at the level of the operator axis given preferential activation.

The preceding conceptual model for the long-term memory network of arithmetic facts is an elaboration of previous conceptual models (Ashcraft & Battaglia, 1982; Parkman, 1972; Widaman et al., 1986) that allows specifically for confusion effects (Stazyk et al., 1982; Winkelman & Schmidt, 1974). Confusion effects would occur with the simultaneous activation of differing levels of the operator axis in the same region of the memory network. Accordingly, the probability of retrieving certain arithmetic facts (e.g., $7 + 3 = 10$, or $7 \times 3 = 21$) is greater than 0 for any single operation at the area of the region represented by the product of the integers. The greater the probability of retrieving conflicting information from the same region of the network, the greater the confusion effect.

The second purpose of this study was to assess the fit of Widaman et al.'s (1986) model for cognitive multiplication. We found support for the idea of identical process components for both simple and complex forms of addition and multiplication. Complex multiplication problems appear to be processed columnwise, and the best-fitting model included the same structural variables for encoding, memory search, carrying to the next column, and for the differences between "true" and "false" problems as did the model for complex addition problems. Regression weight estimates did not differ significantly for the search (product) and verification (truth) processes, or for the carrying operation.

The combined analyses suggested that encoding time per digit may increase as the number of digits in the addends or multiplier and multiplicand in a problem increases. Such a finding was previously reported by Widaman et al. (1986) for different types of addition problems, and is similar to a finding by Poltrock and Schwartz (1984) for multidigit number comparison. In the present study, simple problems had two digits in the addends or multiplier and multiplicand, complex multiplication problems had three digits, and complex addition problems had four digits. Statistical tests of encoding time per digit comparing the above three problem types were not always significant. However, a linear increase of 60 to 65 msec in the per-digit encoding time was observed with each increase of one digit per problem. The above finding is consistent with Widaman et al.'s report of a linear increase of 57 msec in per-digit encoding time as the number of digits per problem increased.

Finally, Widaman et al.'s (1986) finding of self-terminating processing of arithmetic problems was replicated for addition problems and verified for multiplica-

tion problems. Appropriate self-termination requires the action of a metacognitive process that monitors the course of problem solution, selecting and executing proper process components at each step. Thus, the processing of tens-column information under a self-terminating strategy is contingent upon whether an error is encountered in the units column; under an exhaustive strategy, the complete problem is processed regardless of whether errors are encountered. Under a self-terminating strategy, the component process most efficient in solving the problem is executed; that is, either the response "false" is made following a units-column error, or the tens-column information is processed if the units-column answer is correct. The R^2 differences between models reflecting self-terminating and exhaustive processing were larger for addition than for multiplication. This result likely reflects physical, rather than operational, differences between these two problem types (see Widaman et al., 1986).

To conclude, the present study provides further evidence for a single memory network for addition and multiplication facts. Furthermore, identical processing components, including a metacognitive component, appear to be used for the processing of both addition and multiplication problems, and these processing components may prove important for the study of other arithmetic operations. Finally, the combined regression analyses described provide an adjunct methodology, complementing the use of confusion experiments, for the comparison of component processes within operations and across arithmetic operations.

REFERENCES

ASHCRAFT, M. H. (1982). The development of mental arithmetic: A chronometric approach. *Developmental Review*, **2**, 213-236.

ASHCRAFT, M. H., & BATTAGLIA, J. (1978). Cognitive arithmetic: Evidence for retrieval and decision processes in mental addition. *Journal of Experimental Psychology: Human Learning & Memory*, **4**, 527-538.

ASHCRAFT, M. H., & STAZYK, E. H. (1981). Mental addition: A test of three verification models. *Memory & Cognition*, **9**, 185-196.

CASE, R. (1985). *Intellectual development: Birth to adulthood*. New York: Academic Press.

COHEN, J., & COHEN, P. (1983). *Applied multiple regression/correlation analysis for the behavioral sciences*. (2nd ed.). Hillsdale, NJ: Erlbaum.

GROEN, G. J., & PARKMAN, J. M. (1972). A chronometric analysis of simple addition. *Psychological Review*, **79**, 329-343.

KRUEGER, L. E., & HALLFORD, E. W. (1984). Why 2 + 2 = 5 looks so wrong: On the odd-even rule in sum verification. *Memory & Cognition*, **12**, 171-180.

MILLER, K., PERLMUTTER, M., & KEATING, D. (1984). Cognitive arithmetic: Comparison of operations. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, **10**, 46-60.

MOYER, R. S., & LANDAUER, T. K. (1967). Time required for judgment of numerical inequality. *Nature*, **215**, 1519-1520.

PARKMAN, J. M. (1972). Temporal aspects of simple multiplication and comparison. *Journal of Experimental Psychology*, **95**, 437-444.

PARKMAN, J. M., & GROEN, G. J. (1971). Temporal aspects of simple addition and comparison. *Journal of Experimental Psychology*, **89**, 335-342.

POLTROCK, S. E., & SCHWARTZ, D. R. (1984). Comparative judgments of multidigit numbers. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, **10**, 32-45.

RESTLE, F. (1970). Speed of adding and comparing numbers. *Journal of Experimental Psychology*, **85**, 274-278.

STAZYK, E. H., ASHCRAFT, M. H., & HAMANN, M. S. (1982). A network approach to mental multiplication. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, **8**, 320-335.

WIDAMAN, K. F., GEARY, D. C., & CORMIER, P. (1986). *A general model for mental addition: Evidence for self-terminating, columnwise processing*. Manuscript submitted for publication.

WIKE, E. L. (1971). *Data analysis*. Chicago: Aldine-Atherton.

WINKELMAN, J. H., & SCHMIDT, J. (1974). Associative confusions in mental arithmetic. *Journal of Experimental Psychology*, **102**, 734-736.

NOTES

1. Hand assignment was not counterbalanced, because previous studies in our laboratory have indicated that the magnitude and direction of the "true"/"false" intercept differences are not different regardless of whether or not hand assignment is counterbalanced.

2. Order of presentation of problem type was not counterbalanced, because of our concern that repeated transitions from one operation to the other might be confusing for some subjects. The period between the second addition set and the first multiplication set was at least 5 min, and subjects were told that they would be presented with multiplication problems before beginning this new operation. These precautions were taken to ensure that subjects did not confuse addition with multiplication, and multiplication with addition.

3. The determination of best fit was made in terms of variance explained. Thus, the *prod* structural variable provided a better fit than did the *min* and *sum*^{2} variables. However, the R^2 differences for equations including each of these variables are small; therefore, it cannot be argued that the *prod* variable is absolutely the best search/compute predictor for these data. Rather, the *prod* was better than all other search/compute variables tested.

4. Miller et al. (1984) reported that simple multiplication problems that included a 1 as the multiplicand or multiplier were processed rather quickly, relative to problems not containing a 1 (or a 0). In the present study, this effect was found only for simple multiplication problems when the multiplicand was a 1, and only for complex multiplication problems when the multiplier was a 1. However, analyses excluding the above multiplication problems yielded results that were identical in all essential respects to the results of analyses including these problems. That is, excluding the above problems made no difference in terms of the relative fit of particular regression equations, and made no noticeable difference in terms of the size of the regression estimates for variables in each regression equation. Therefore, results from the full set of stimuli were presented.

5. The NI parameter for the combined regression equation (in Table 3) was coded in the same manner as the NI parameter for the combined simple and complex multiplication analysis.

6. The NI parameter for the combined regression equation (in Table 3) was coded in the same manner as the NI parameter for the combined simple and complex addition analysis.

(Manuscript received December 30, 1985;
revision accepted for publication May 13, 1986.)