A mixed-integer linear program formulation for fast matrix multiplication

Laurent Sorber, Marc Van Barel

April 30, 2017

1 Introduction

Multiplying $N \times N$ matrices naively costs N^3 floating point (FP) multiplications. Strassen formulated a *multiplication tensor* $\mathcal{T}$ of size $N^2 \times N^2 \times N^2$ whose polyadic decomposition of rank R in factor matrices with elements in $\{-1, 0, 1\}$ corresponds to multiplying $N \times N$ matrices with only R FP multiplications. Furthermore, Strassen found a solution for $N = 2$ with $R = 7$ [9], which was later shown to be both a canonical polyadic decomposition [2, 5, 10] (there is no polyadic decomposition with smaller R) and essentially unique [3] (all other solutions of the same rank are equivalent to Strassen's solution).

Strassen's algorithm is most effective when applied at the block level instead of the scalar level, thereby saving one block-matrix multiplication in the multiplication of 2×2 block matrices. Applying this recursively leads to a complexity of $O(N^{\log_2(7) \approx 2.80735\ldots})$ floating point operations for matrix multiplication. In practice, only a few iterations of Strassen's algorithm would be applied before the additional sums and differences are more expensive than the block-multiplication savings.

Multiplying 3×3 matrices naively costs 27 multiplications, while the current best known polyadic decompositions do it in only $R = 23$ multiplications. However, recursive application of such solutions lead to a complexity of $O(N^{\log_3(23) \approx 2.85405\ldots})$, which is worse than Strassen's 2×2 solution. We would have to find a solution of at most $R = 21$ to improve on Strassen's algorithm. It is currently not known whether $R = 23$ is the lowest rank that can be attained, nor is it the case that solutions of that rank are essentially unique. This leads us to some interesting questions. Do lower rank solutions exist? If so, are those essentially unique? If not, is the most sparse solution (i.e., the solution with the least number of sums and differences) essentially unique?

While local optimization methods can be useful to generate candidate solutions, they cannot be used to prove that no solutions of a certain rank exist, nor that no better solutions exist (whether it be of lower rank, or more sparse solutions). In this note, we translate the polyadic decomposition into a mixed-integer linear program (MILP), which, if tractable, can provide answers to the above questions.

2 Fast matrix multiplication as a nonlinear program

2.1 Problem statement

The $\{0, 1\}$ multiplication tensor $\mathcal{T}$ associated with the matrix multiplication $A \cdot B = C$ where A is of size $M \times P$, B is of size $P \times N$ and C is of size $M \times N$ is defined by the equation

$$\mathcal{T} \cdot_1 \operatorname{vec}(A^{\mathrm{T}}) \cdot_2 \operatorname{vec}(B^{\mathrm{T}}) = \operatorname{vec}(C) \tag{1}$$

where $\cdot_n$ is the mode-n tensor-vector product and $\operatorname{vec}(\cdot)$ is the column-wise vectorization operator. The multiplication tensor is sometimes written as $\langle M, P, N\rangle$ to specify the problem dimensions. In this note, we consider only square matrix multiplication and set $M = P = N$, although much of our discussion also applies to rectangular matrices and other bilinear operators.

The tensor $\mathcal{T}$ can be decomposed into a sum of R rank-one tensors:

$$\mathcal{T}_{ijk} = \sum_{r=1}^{R} U_{ir} \cdot V_{jr} \cdot W_{kr} \quad \forall i, j, k = 1 \dots N^2 \tag{2}$$

where U, V, and W are the so-called factor matrices of size $N^2 \times R$. Equation (2) is called a polyadic decomposition (PD) of $\mathcal{T}$, and specifically a canonical polyadic decomposition (CPD) if R is minimal. The naive matrix multiplication algorithm corresponds to a PD of rank $R = N^3$, where each rank-one tensor contributes one nonzero element to the sum.

Each rank-one matrix $U_{:r} \cdot V_{:r}^{\mathrm{T}}$ corresponds to a single multiplication of a linear combination of elements of the matrix A and a linear combination of the elements of B. The entry W_{kr} then determines how much the resulting scalar contributes to the kth entry of $\operatorname{vec}(C)$. To see this, we rewrite (1) using (2) as:

$$\begin{aligned}
\operatorname{vec}(C)_k &= \mathcal{T}_{::k} \cdot_1 \operatorname{vec}(A^{\mathrm{T}}) \cdot_2 \operatorname{vec}(B^{\mathrm{T}}) && (3)\\
&= \left(\sum_{r=1}^{R} (U_{:r} \cdot V_{:r}^{\mathrm{T}}) \cdot W_{kr} \right) \cdot_1 \operatorname{vec}(A^{\mathrm{T}}) \cdot_2 \operatorname{vec}(B^{\mathrm{T}}) && (4)\\
&= \sum_{r=1}^{R} \langle \operatorname{vec}(A^{\mathrm{T}}), U_{:r} \rangle \cdot \langle V_{:r}, \operatorname{vec}(B^{\mathrm{T}}) \rangle \cdot W_{kr} && (5)
\end{aligned}$$

With factor matrices U, V, and W that satisfy the equality (2) and a rank $R < N^3$, we refer to (5) as a fast matrix multiplication algorithm.

2.2 Formulation as an optimization problem

An optimal polyadic decomposition of the multiplication tensor (2) can be formulated as the discrete non-linear program (NLP)

$$\begin{aligned} \underset{U,V,W}{\text{minimize}} \quad & 3N^2R \sum_{i,j,k=1}^{N^2} \left| T_{ijk} - \sum_{r=1}^{R} U_{ir} \cdot V_{jr} \cdot W_{kr} \right| \\ & + \sum_{i,r} |U_{ir}| + \sum_{j,r} |V_{jr}| + \sum_{k,r} |W_{kr}| \\ \text{s.t.} \quad & U, V, W \in \{-1, 0, 1\}^{N^2 \times R} \end{aligned} \tag{6}$$

where the entries of the factor matrices U, V, and W are constrained to be -1, 0 or 1 so that they act as selection operators in the fast matrix multiplication algorithm (5). The objective function's primary goal is to minimize the L1 norm of the residual and is multiplied by a factor of $3N^2R$ so that it doesn't conflict with its secondary objective, which is to maximize sparsity of the solution. Other norms, distance metrics, or even equalities, could be considered for the primary goal. However, we choose the L1 norm here since it will be closest to our translation into a MILP.

3 Fast matrix multiplication as a mixed-integer linear program

3.1 Motivation

Much like eigenvalue decomposition algorithms, MILP solvers have seen decades of progress and are now able to solve problems that were previously considered unsolvable [7]. For example, the commercial solver Gurobi v7.0 claims a 43x speedup since it was released 7 years ago [8]. IBM's CPLEX v12.7 solver claims a 37x speedup since its v10 release 10 years prior [6]. Modern MILP solvers use a number of building blocks to achieve those speedups:

1. Presolve: Tighten formulation and reduce problem size.
2. Solve continuous relaxations: Ignores integrality and gives a bound on the optimal integral objective.
3. Cutting planes: Cut off relaxation solutions.
4. Branching variable selection: Intelligently explore search space.
5. Heuristics: Find integer feasible solutions.

A MILP's convex relaxation as a linear program (LP) is what allows us to explore the search space of candidate solutions effectively and find a globally optimal solution. The root LP can be solved efficiently in polynomial time

and will provide us with both a solution that is (hopefully) close to feasible, and a lower bound on the objective function. In the subsequent branch and bound phase of the solver, variables that were not integral but should be can be selected for branching. Nodes in this search tree can provide better upper bounds (when new feasible solutions are found) and lower bounds (when the best unexplored node's LP objective value is higher than the current lower bound) on the objective value. When the gap between the upper and lower bound is reduced to zero, we have obtained a globally optimal solution in the sense that there are no solutions with a better objective value.

3.2 MILP formulation

We parametrize the factor matrices with two binary components that represent whether the factor matrices' entries are strictly negative, strictly positive, or neither. With these components in hand, we show that linear constraints can emulate the multilinear character of the objective function. To improve convergence to a global minimum, additional constraints could be formulated. However, let us begin by describing compact MILP with the same solutions as the NLP (6):

$$
\begin{aligned}
\text{minimize} \quad & 3N^2R \sum_{i,j,k=1}^{N^2} COST_{ijk} \\
& + \sum_{i,r} ABSALTU_{ir} + \sum_{j,r} ABSALTV_{jr} + \sum_{k,r} ABSALTW_{kr} \\
& + \frac{1}{3N^2R} \left(\sum_{i,r} ABSDIFFU_{ir} + \sum_{j,r} ABSDIFFV_{jr} + \sum_{k,r} ABSDIFFW_{kr} \right) \\
\text{s.t.} \quad & \text{// } COST \text{ is the absolute value of } \mathcal{T} - [\![U, V, W]\!] \\
& -COST_{ijk} \leq T_{ijk} - \sum_{r=1}^{R} VAL_{ijkr} \leq COST_{ijk} \\
& \text{// } U = [U > 0] - [U < 0] \\
& U = POSU - NEGU \\
& V = POSV - NEGV \\
& W = POSW - NEGW \\
& \text{// } |U| = [U > 0] + [U < 0] \\
& ABSU = POSU + NEGU \\
& ABSV = POSV + NEGV \\
& ABSW = POSW + NEGW \\
& \text{// Alternate definition of } |U|, |V|, \text{ and } |W| \\
& -ABSALTU \leq U \leq ABSALTU \\
& -ABSALTV \leq V \leq ABSALTV \\
& -ABSALTW \leq W \leq ABSALTW \\
& \text{// Distance between two measures of absolute values} \\
& -ABSDIFFU \leq ABSU - ABSALTU \leq ABSDIFFU \\
& -ABSDIFFV \leq ABSV - ABSALTV \leq ABSDIFFV \\
& -ABSDIFFW \leq ABSW - ABSALTW \leq ABSDIFFW \\
& \text{// Force } VAL \text{ for the case (1,1,1) and (-1,-1,-1)} \\
& -2 \leq VAL_{ijkr} - U_{ir} - V_{jr} - W_{kr} \leq 2 \\
& \text{// Force } VAL \text{ for the case (1,1,-1) and (1,-1,-1)} \\
& -2 \leq VAL_{ijkr} - U_{ir} + V_{jr} + W_{kr} \leq 2 \\
& -2 \leq VAL_{ijkr} + U_{ir} - V_{jr} + W_{kr} \leq 2 \\
& -2 \leq VAL_{ijkr} + U_{ir} + V_{jr} - W_{kr} \leq 2 \\
& \text{// Force } VAL \text{ for the cases (0,*,*), (*,0,*), and (*,*,0)} \\
& -ABSU_{ir} \leq VAL_{ijkr} \leq ABSU_{ir} \\
& -ABSV_{jr} \leq VAL_{ijkr} \leq ABSV_{jr} \\
& -ABSW_{kr} \leq VAL_{ijkr} \leq ABSW_{kr} \\
& \text{// Variable bounds} \\
& -1 \leq U, V, W, VAL \leq 1 \\
& 0 \leq NEGU, NEGV, NEGW, POSU, POSV, POSW \leq 1 \\
& 0 \leq ABSU, ABSV, ABSW, ABSALTU, ABSALTV, ABSALTW \leq 1 \\
& 0 \leq COST, ABSDIFFU, ABSDIFFV, ABSDIFFW \leq 1 \\
& \text{// Integer variables} \\
& NEGU, NEGV, NEGW, POSU, POSV, POSW \in \mathbb{Z}^{N^2 \times R}
\end{aligned}
\tag{7}
$$

The idea of the MILP (7) is to introduce variables $VALUE_{ijkr}$ and force them to be equal to the multilinear products $U_{ir} \cdot V_{jr} \cdot W_{kr}$ when the entries of U, V, and W are in $\{-1, 0, 1\}$. Herein, forcing $VALUE_{ijkr}$ to be zero when one of U_{ir}, V_{jr}, or W_{kr} is zero is the most difficult constraint since it requires having access to the absolute values of the factor matrices. To illustrate the problem, set $NEGU_{ir}$ and $POSU_{ir}$ equal to 0.5. According to our formulation, $ABSU_{ir}$ will be $0.5 + 0.5 = 1$ while U_{ir} itself will be $0.5 - 0.5 = 0$, leading to $VALUE_{ijkr}$ being free to take on any value in $[-1, 1]$ even though it should clearly be zero.

To help mitigate this problem, we introduced an alternative measure of the absolute value $ABSALTU_{ir}$ which is at least as large as $|U_{ir}|$. Minimizing $ABSALTU$ as a secondary goal in the objective function serves two purposes: on the one hand we make sure that $ABSALTU_{ir}$ is indeed equal to $|U_{ir}|$, and on the other hand we will obtain the most sparse solution among those that have a residual of zero. The third and final goal of the objective function is then to also keep the distance between $ABSU$ and $ABSALTU$ as small as possible. We have noticed that this does indeed lead to a smaller number of non-integer variables in the solution of the LP relaxation.

3.3 Dimensionality and tractability

Table 1 shows the dimensionality of the MILP (7) for some choices of the matrix order N and the number of multiplications R. The MIPLIB 2010 benchmark [4] lists some (mixed) binary problems with up to 1.5M rows, 125k binary variables and 28M nonzeros that have been solved. Of course that does not necessarily mean this particular problem can be solved by the current solvers, but at least the size of the problem still seems to be within reason.

		$N = 2$ $R = 7$	$N = 3$ $R = 23$	$N = 4$ $R = 46$
Variables (columns)	$O(N^6R)$	448	16767	188416
Binary variables	$6N^2R$	168	1242	4416
Constraints (rows)	$O(14N^6R)$	6272	235k	2.638M
Nonzeros	$O(44N^6R)$	19712	738k	8.290M

Table 1: MILP problem dimensions for some choices of the matrix order N and the number of multiplications R.

3.4 Optimizations

Reducing the gap: upper bound. To help speed up the optimization process, one effective technique is to try to reduce the gap between the upper and lower bound on the objective function. The upper bound can be lowered by providing high quality initial solutions to the solver. For example, we could use local optimization techniques to generate MILP feasible solutions, or take an

existing solution such as for $N = 2, R = 7$ or $N = 3, R = 23$ and optionally truncate it to a lower rank if desired.

Reducing the gap: lower bound. Assuming a solution with zero residual exists, the lower bound is determined by the number of nonzeros in the factor matrices. Since the multiplication tensor contains exactly N^3 ones in nontrivial locations, the factor matrices would need to contain at least $3N^3$ nonzeros to fill up those ones. Since $R < N^3$, we expect to see at least one sum or difference and so we can raise the lower bound to $3N^3 + 1$.

Cyclic symmetry. Multiplication tensors $\mathcal{T}$ corresponding to square matrix multiplication have the cyclic symmetry property $\mathcal{T}_{ijk} = \mathcal{T}_{kij} = \mathcal{T}_{jki}$. This means that if $[\![U, V, W]\!]$ is a PD then $[\![W, U, V]\!]$ and $[\![V, W, U]\!]$ are also PDs of $\mathcal{T}$. Ballard proposed to parametrize the factor matrices in the cyclic invariant form $U := [A\, B\, C\, D]$, $V := [A\, D\, B\, C]$, and $W := [A\, C\, D\, B]$, where $A \in \{-1, 0, 1\}^{N^2 \times S}$ and $B, C, D \in \{-1, 0, 1\}^{N^2 \times T}$ [1]. The rank is therefore equal to $R = S + 3T$. This reduces the number of integer variables by a factor of three. However, there is no guarantee that the minimal rank of the cyclic invariant parametrized PD is equal to the rank of the CPD. Still, we are somewhat optimistic that the two ranks may coincide given that (1) they coincide for the case $N = 2, R = 7$, (2) there are cyclic invariant solutions for $N = 3, R = 23$, and (3) that Comon's conjecture on the symmetric PD rank being equal to the CPD rank has been proven to be true under certain assumptions on the tensor or its rank.

Factor matrix structure. Some additional forms of structure can be imposed as linear constraints on the factor matrices. For example, we could require the sum of absolute values of each factor matrix column to be at least one so that the solver does not look for solutions of rank smaller than R. The same constraint can be applied to the rows of the factor matrices since there can be no all-zero slices in the multiplication tensor. Finally, we could additionally conjecture that all factor matrices have an equal number of nonzeros (since this is the case for $N = 2, R = 7$ and for the cyclically invariant solutions for $N = 3, R = 23$). Furthermore, this would raise the lower bound on the objective function from $3N^3 + 1$ to the next nearest multiple of three.

Acknowledgements

We thank Nick Vannieuwenhoven for the insightful discussions which have helped improve this manuscript.

References

[1] G. Ballard. Discovering Fast Matrix Multiplication Algorithms via Tensor Decomposition. http://users.wfu.edu/ballard/pdfs/CSE17.pdf, March 2017. [Online].

[2] R. W. Brockett and D. Dobkin. On the optimal evaluation of a set of bilinear forms. *Linear Algebra and Its Applications*, 19(3):207–235, 1978.

[3] H. F. de Groote. On varieties of optimal algorithms for the computation of bilinear mappings ii. optimal algorithms for 2×2-matrix multiplication. *Theoretical Computer Science*, 7(2):127–148, 1978.

[4] G. G. et al. MIPLIB 2010 - Mixed-binary programs. http://miplib.zib.de/miplib2010-MBP.php, February 2017. [Online].

[5] J. E. Hopcroft and L. R. Kerr. On minimizing the number of multiplications necessary for matrix multiplication. *SIAM Journal on Applied Mathematics*, 20(1):30–36, 1971.

[6] IBM. CPLEX MIP performance evolution. http://www-01.ibm.com/software/commerce/optimization/cplex-performance/, November 2016. [Online].

[7] J. Linderoth. Mixed-Integer (Linear) Programming: Recent Advances, and Future Research Directions. http://www.focapo-cpc.org/pdf/Linderoth.pdf, January 2017. [Online].

[8] G. Optimization. Gurobi 7.0 Performance Benchmarks. http://www.gurobi.com/pdfs/benchmarks.pdf, March 2017. [Online].

[9] V. Strassen. Gaussian elimination is not optimal. *Numer. Math.*, 13:354–356, 1969.

[10] S. Winograd. On multiplication of 2×2 matrices. *Linear algebra and its applications*, 4(4):381–388, 1971.