

A New Model for the Multiple Constant Multiplication Problem

Rémi Garcia, Anastasia Volkova, Alexandre Goldsztejn

▶ **To cite this version:**

Rémi Garcia, Anastasia Volkova, Alexandre Goldsztejn. A New Model for the Multiple Constant Multiplication Problem. 2021. hal-03454510

HAL Id: hal-03454510

https://nantes-universite.hal.science/hal-03454510

Preprint submitted on 29 Nov 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A New Model for the Multiple Constant Multiplication Problem

Rémi Garcia[1], Anastasia Volkova[1], Alexandre Goldsztejn[2]

[1] Univ. Nantes, CNRS, LS2N, Nantes, France
[2] CNRS, Univ. Nantes, LS2N, Nantes, France
firstname.lastname@univ-nantes.fr

Keywords : *multiple constant multiplication, mixed-integer linear programming*

1 Introduction

The multiplications by several integer constants is a frequent operation in many algorithms implemented in embedded systems, *e. g.*, the evaluation of digital filters. Instead of using costly generic multipliers, circuit designers usually lean towards multiplierless implementations, where a series of bit shifts and additions/subtractions, which come at lesser cost, is used instead. For example, rewriting $7x$ as $2^3x - x$, a generic multiplier is replaced by just one adder and one bit-shift, whose cost is negligible as these are hardwired during circuit design. Moreover, when multiplying by several constants, intermediate results can be shared, *e. g.* for multiplication by both 7 and 23, the latter can be computed as $23x = 7x + 2^4x$. Figure 1 presents the so-called *adder graph* describing the multiplierless solution.

Aiming at reducing the hardware cost, the problem of finding a multiplierless implementation with the least number of adders, for a set of given target constants, is known as a Multiple Constant Multiplication (MCM) problem and conjectured to be NP-Hard [11]. Another important metric, used in practice, is the delay, which is directly correlated to the *adder depth* (AD), *i. e.* the maximum number of cascaded adders (*e. g.* adder depth is two in Figure 1). When the adder depth is limited *a priori*, the problem is called a Bounded MCM (BMCM).

The current state-of-the-art optimal approach for the MCM problem [9] is based on ILP. However, it is stated not as an optimization but a satisfaction problem, deciding whether a set of target constants can be realized using exactly N_A adders. The BMCM problem is also optimally solved based on an ILP model from [8], this time allowing to directly minimize the number of adders. The bottleneck of this approach are heavy precomputations which, with the increasing coefficient word length, lead to impractically big models.

With this work we first propose a new ILP model for MCM as a *minimization* problem, allowing for better versatility. We also demonstrate a limitation of the state-of-the-art MCM model [9], which misses optimal solutions in certain cases, and correct it within our model. We further extend our model to solve the BMCM problem by adding necessary constraints into the ILP model, avoiding any heavy precomputations and neutralizing the main performance bottleneck. Moreover, we combine the minimization of both adder depth and the adder count into one objective function, offering a new complete $\text{MCM}_{\min \text{AD}}$model. Finally, we propose new symmetry breaking constraints, which significantly improve the resolution process.

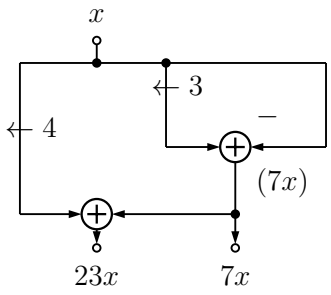


FIG. 1: Optimal adder graph for $7x$ and $23x$

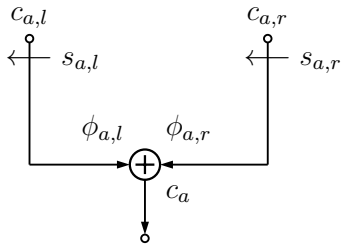


FIG. 2: Adder model from [9]

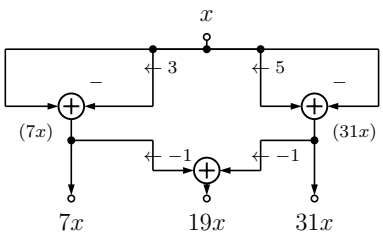


FIG. 3: Optimal adder graph for computing $7x$, $19x$ and $31x$

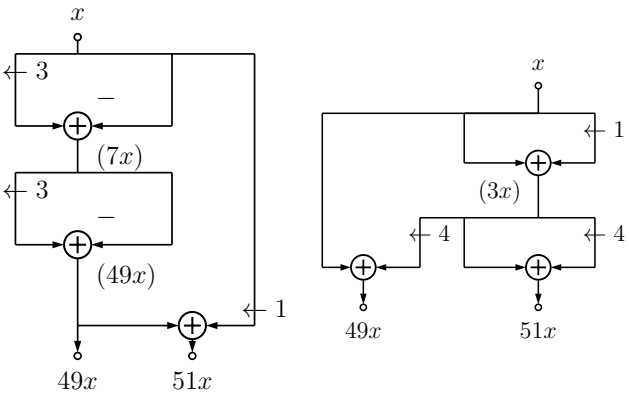


FIG. 4: Optimal adder graphs with different AD

2 State of the art

First formulations of MCM date back to 1986 [3] and, since then, the problem has been well studied and can be tackled using greedy algorithms [2], heuristics, bounds [6], optimal approaches [1, 7, 9, 11], etc. The current state-of-the-art optimal method [9] is based on an ILP model for the adder graphs describing MCM. The adder graphs have the following properties: they are directed acyclic graphs (DAG), for which each node, except for the input node, corresponds to an adder. Nodes have an in-degree of two and each edge weight represents a bit shift, which is noted along an arrow in figures. No weight or zero means the absence of a bit shift. Each adder is annotated by a constant called *fundamental*, which corresponds to the constant by which the input is multiplied. Without any loss of generality [4], we use only odd fundamentals. Additionally, any MCM instance can be transformed into an equivalent MCM problem with only unique odd positive target constants.

The ILP model [9] solves a decision problem and minimizing the number of adders requires an outside loop on the number of adders, starting with a known lower bound [6] and continuing while no feasible solution is found. The model relies on the linearization of the relation between an adder and its input in the adder graph (see Figure 2). Yet, we discovered that the model does not permit negative shifts, which are necessary for reaching optimal solutions for some target constants sets as $[7, 19, 31]$. In Figure 3, we show the optimal adder graph which is not obtained by the existing ILP model. Not allowing negative shifts leads to adder graphs with at least four adders, which are therefore not optimal.

If the adder depth is *a priori* bounded, all possible combinations of adders can be generated, knowing that the integer constants and shifts are bounded [4]. This idea is the basis of a second ILP model [8], which solves the BMCM problem. This model requires heavy precomputations that enumerate all possible adders for each adder stage. Hence, the size of the model directly depends on the word length of the coefficients and on the maximum considered AD. This leads to an important performance bottleneck, limiting the applicability of the model only to low ADs (typically 3 or 4). Usually, it is desirable to obtain an adder graph with the smallest AD possible as a second objective. Solving BMCM with multiple AD values can theoretically solve this problem, which we will refer to as $\text{MCM}_{\min \text{AD}}$, but this approach fails in practice as the existing model is limited to low ADs.

3 Modeling for MCM and BMCM

We though of two straightforward approaches to address the issue of the existing ILP model to solve the MCM problem [9]. However, both have limitations, either because of potential numerical instabilities, or because of an important search space expansion. We solve the MCM problem by proposing a new model, better adapted to handle the negative shifts (Section 3.1). The adder depth information is included in this model in Section 3.2, which allows solving both BMCM and $\text{MCM}_{\min \text{AD}}$ homogeneously. Finally, symmetry breaking constraints are included in the model in Section 3.3.

Constants/Variables and their meaning
$N_A \in \mathbb{N}$: number of adders;
$N_O \in \mathbb{N}$: number of outputs;
$C \in \mathbb{N}^{N_O}$: target constants;
$S_{\min}, S_{\max} \in \mathbb{Z}$: minimum and maximum shift;
$w \in \mathbb{N}$: word length.
$c_a \in [\![0, 2^w]\!]$, $\forall a \in [\![0, N_A]\!]$: constant obtained in adder a with c_0 fixed to the value 1;
$c_a^{\text{nsh}} \in [\![0, 2^{w+1}]\!]$, $\forall a \in [\![1, N_A]\!]$: constant obtained in adder a before the negative shift;
$c_a^{\text{odd}} \in \mathbb{N}$, $\forall a \in [\![1, N_A]\!]$: variable used to ensure that c_a is odd;
$c_{a,i} \in [\![0, 2^w]\!]$, $\forall a \in [\![1, N_A]\!]$, $i \in \{l, r\}$: constant of adder from input i before adder a;
$c_{a,l}^{\text{sh}} \in [\![0, 2^{w+1}]\!]$, $\forall a \in [\![1, N_A]\!]$: constant of adder from left input before adder a and after the left shift; for simplification $c_{a,r}^{\text{sh}}$ is an alias of $c_{a,r}$;
$c_{a,i}^{\text{sh,sg}} \in [\![-2^{w+1}, 2^{w+1}]\!]$, $\forall a \in [\![1, N_A]\!]$, $i \in \{l, r\}$: signed constant of adder from input i before adder a and after the shift;
$\Phi_{a,i} \in \{0, 1\}$, $\forall a \in [\![1, N_A]\!]$, $i \in \{l, r\}$: sign of i input of adder a. 0 for + and 1 for −;
$c_{a,i,k} \in \{0, 1\}$, $\forall a \in [\![1, N_A]\!]$, $i \in \{l, r\}$, $k \in [\![0, N_A - 1]\!]$: 1 if input i of adder a is adder k;
$\sigma_{a,s} \in \{0, 1\}$, $\forall a \in [\![1, N_A]\!]$, $s \in [\![0, S_{\max}]\!]$: 1 if left shift before adder a is equal to s;
$\Psi_{a,s} \in \{0, 1\}$, $\forall a \in [\![1, N_A]\!]$, $s \in [\![S_{\min}, 0]\!]$: 1 if negative shift of adder a is equal to s;
$o_{a,j} \in \{0, 1\}$, $\forall a \in [\![1, N_A]\!]$, $j \in [\![1, N_O]\!]$: 1 if adder a is equal to the j-th target constant.

TAB. 1: Constants (top) and variables (bottom) used for the ILP formulation

3.1 ILP Model for MCM

In adder graphs that only involve odd fundamentals, adders are linked one with another by the nonlinear equation

$$c_a = 2^{-s_1} \left((-1)^{\phi_{a,l}} 2^{s_2} c_{a,l} + (-1)^{\phi_{a,r}} c_{a,r} \right), \tag{1}$$

where c_a is the fundamental computed by adder a, the $c_{a,i}$ are its non-shifted and non-signed inputs and, signs and shifts are handled by $\phi_{a,i}$, s_1 and s_2. To linearize that equation, we have to include many binary variables and indicator or big M constraints. For conciseness, in the following we will only present the indicator constraints. Necessary variables and their meaning are presented in Table 1 and constraints are as follows:

$$c_a^{\text{nsh}} = c_{a,l}^{\text{sh,sg}} + c_{a,r}^{\text{sh,sg}} \qquad \forall a \in [\![1, N_A]\!] \tag{C1}$$

$$c_a^{\text{nsh}} = 2^{-s} c_a \quad \text{if } \Psi_{a,s} = 1 \qquad \forall a \in [\![1, N_A]\!], s \in [\![S_{\min}, 0]\!] \tag{C2}$$

$$\sum_{s=S_{\min}}^{0} \Psi_{a,s} = 1 \qquad \forall a \in [\![1, N_A]\!] \tag{C3}$$

$$\sigma_{a,0} = \sum_{s=S_{\min}}^{-1} \Psi_{a,s} \qquad \forall a \in [\![1, N_A]\!] \tag{C4}$$

$$c_a = 2c_a^{\text{odd}} + 1 \qquad \forall a \in [\![1, N_A]\!] \tag{C5}$$

$$c_{a,i} = c_k \quad \text{if } c_{a,i,k} = 1 \qquad \forall a \in [\![1, N_A]\!], i \in \{l, r\}, \forall k \in [\![0, a-1]\!] \tag{C6}$$

$$\sum_{k=0}^{a-1} c_{a,i,k} = 1 \qquad \forall a \in [\![1, N_A]\!], i \in \{l, r\} \tag{C7}$$

$$c_{a,l}^{\text{sh}} = 2^s c_{a,l} \quad \text{if } \sigma_{a,s} = 1 \qquad \forall a \in [\![1, N_A]\!], s \in [\![0, S_{\max}]\!] \tag{C8}$$

$$\sum_{s=0}^{S_{\max}} \sigma_{a,s} = 1 \qquad \forall a \in [\![1, N_A]\!] \tag{C9}$$

$$c_{a,i}^{\text{sh,sg}} = -c_{a,i}^{\text{sh}} \quad \text{if } \Phi_{a,i} = 1 \qquad \forall a \in [\![1, N_A]\!], i \in \{l, r\} \tag{C10}$$

$$c_{a,i}^{\text{sh,sg}} = c_{a,i}^{\text{sh}} \quad \text{if } \Phi_{a,i} = 0 \qquad \forall a \in [\![1, N_A]\!], i \in \{l, r\} \tag{C11}$$

$$c_a = C_j \quad \text{if } o_{a,j} = 1 \qquad \forall a \in [\![0, N_A]\!], j \in [\![1, N_O]\!] \tag{C12}$$

$$\sum_{a=0}^{N_A} o_{a,j} = 1 \qquad \forall j \in [\![1, N_O]\!] \tag{C13}$$

Constraint (C1) states that the value of an adder before a potential negative shift is equal to the sum of its shifted and signed inputs. Constraints (C2), (C3) and (C5) apply the negative shift to variables c_a and ensure that the computed fundamental is odd. It can be noticed that a potential negative shift after c_a^{nsh} only makes sense if the sum of the inputs is even, which can only happen if no left shift is applied to the left input: constraint (C4) specifically states that in order to speed up the solving. The link between an adder and its inputs is enforced by constraints (C6) and (C7). Constraints (C8) and (C9) permit to apply the shifts to the inputs of an adder while constraints (C10), (C11) apply the sign. Finally, every target constant is computed once and only once thanks to constraints (C12) and (C13). These constraints define a decision problem that can be solved with no objective function.

Thanks to good heuristics that give a tight upper bound on the optimal number of adders, it is now possible, and efficient, to tackle MCM as a minimization problem, instead of relying on an outside loop on the number of adders and a satisfaction model. By fixing N_A to an upper bound, obtained using a heuristic solution [10], we are able to minimize the number of effectively *used adders*. A binary variable, $u_a \in \{0, 1\}$, $\forall a \in [\![1, N_A]\!]$, permits to deactivate an adder if not used: $c_a = 1$ if $u_a = 0$. Then, adding to the model the objective function $\min \sum u_a$ permits to solve the MCM problem using the full potential of MILP solvers and not only their satisfiability part.

3.2 Support of Limited Adder Depth: BMCM and $\text{MCM}_{\min \text{AD}}$

In order to propagate the adder depth, we introduce two sets of integer variables, ad_a and $ad_{a,i}$, $i \in l, r$, representing the AD of the adder a and of its left and right inputs, respectively. Naturally, $ad_0 = 0$ and the AD propagation is handled with the following constraints

$$ad_a = \max\left(ad_{a,l} + 1, ad_{a,r} + 1\right) \qquad \forall a \in [\![1, N_A]\!], \tag{2}$$

$$ad_{a,i} = ad_k \quad \text{if } c_{a,i,k} = 1 \qquad \forall a \in [\![1, N_A]\!], i \in \{l, r\}, \forall k \in [\![0, a-1]\!]. \tag{3}$$

Note that the max in (2) can be linearized adding a set of binary variables to the model.

Then, constraint $\overline{ad} \geq ad_a$ permits to bound the AD by a user-given constant, $\overline{ad}$, tackling the BMCM problem.

Our BMCM model can actually be extended towards $\text{MCM}_{\min \text{AD}}$in order to optimize for both adder count and adder depth. By introducing a variable $ad_{\max}$ with the constraints $ad_{\max} \geq ad_a$, so that $ad_{\max}$ is an upper bound on the AD. Then, the objective function $\min \sum (N_A u_a) + ad_{\max}$ is a weighted sum that enforces a lexicographic optimization with the number of adders as first objective and the AD as second. Indeed, reducing the number of adders is unconditionally stronger than increasing the AD because $N_A \geq ad_{\max}$.

Solving the $\text{MCM}_{\min \text{AD}}$ permits to select from the set of solutions with minimal number of adders those which yield the smallest delay in a hardware implementation. For example, for target constants $\{49, 51\}$ our new $\text{MCM}_{\min \text{AD}}$ yields an optimal solution with an AD = 2 and $N_A = 3$, with the intermediate fundamental 3, instead of AD = 3 for other solutions with $N_A = 3$ but using an intermediate fundamental such as 7 (see Figure 4).

In order to speed up the computations, Gustafsonn [6] proposed lower bounds on the AD for target constants, which we use to guide the solver. This leads to the following new constraints:

$$ad_a \geq o_{a,j} \times \underline{ad}_j \qquad \forall a \in [\![1, N_A]\!], j \in [\![1, N_O]\!], \tag{4}$$

where $\underline{ad}_j$ is the lower bound on the adder depth of the output j.

			MCM			BMCM_3			$\text{MCM}_{\min \text{AD}}$		
Benchmark	**WL**	**#coeff.**	T	N_A	AD	T	N_A	AD	T	N_A	AD
GAUSSIAN_3	8	3	< 1	4	2	< 1	4	2	< 1	4	2
GAUSSIAN_5	11	3	3	5	4	5	6	3	16	5	4
HIGHPASS_5	7	4	< 1	4	2	< 1	4	2	< 1	4	2
HIGHPASS_9	7	5	< 1	5	2	< 1	5	2	< 1	5	2
HIGHPASS_15	9	12	< 1	12	2	< 1	12	2	< 1	12	2
LAPLACIAN_3	7	3	< 1	3	3	< 1	3	3	< 1	3	3
LOWPASS_5	7	5	2	6	5	3	6	3	3	6	3
LOWPASS_9	9	12	47	12	6	2	12	3	43	12	3
LOWPASS_15	11	25	TO	–	–	TO	–	–	TO	–	–
UNSHARP_3-1	7	3	< 1	4	2	< 1	4	2	< 1	4	2
UNSHARP_3-2	11	3	3	5	4	2	5	3	2	5	3

TAB. 2: Results and comparison of the proposed MCM models. T is the solving time in seconds, N_A and AD stand for the number of adders and the adder depth, respectively.

3.3 Symmetry Breaking Constraints

The model contains some symmetries that can be broken in order to reduce the search space. One can note that, if two neighboring adders, a and $a+1$ do not directly interact, then their order in the solution has no impact. If they do interact, their AD has to differ. Ordering the adders first by AD, and secondly by value, inside a same AD group, enforces an order on the adders. Finding additional constraints that permit to enforce that order is a classic way to remove symmetric solutions, keeping at least one of them.Breaking these symmetry means finding additional constraints, called symmetry breaking constraints, that remove some symmetric solutions but keep at least one of them.

The first constraint that ensures the right order is straightforward:

$$ad_a \leq ad_{a+1} \quad \text{if } u_{a+1} = 1, \qquad \forall a \in [\![1, N_A - 1]\!]. \tag{5}$$

To enforce the correct order inside each adder depth group, we ensure that

$$c_a \leq c_{a+1} + 2^w \left(ad_{a+1} - ad_a\right) \quad \text{if } u_{a+1} = 1, \qquad \forall a \in [\![1, N_A - 1]\!]. \tag{6}$$

For both above constraints, (5) and (6), the indicator constraint can be dropped if the adder is known to be used thanks to a known lower bound [6].

More symmetries can be broken as it could be desirable to have the unused adders stored on the last indices: $u_a \leq u_{a-1}$, $\forall a \in [\![2, N_A]\!]$. This constraint forces adder $a-1$ to be used if adder a is. Thus, used adders are first indices and a few u_a can be fixed to 1 as a known lower bound can be computed [6].

4 Experiments

We implemented our MILP model in `julia` using the `JuMP` library and provide an open-source tool for reproducibility[1]. All experiments were performed on a Linux laptop with i7-10810U processor and 32 GB RAM using the solver Gurobi 9.1.1 with a time limit of 30 minutes. We used a set of benchmarks commonly used with MCM [8] and report in Table 2 the resolution of the problems MCM, BMCM and $\text{MCM}_{\min \text{AD}}$ using our model.

Except for the instance `LOWPASS_15`, optimization times are small and the minimization of the adder depth does not lead to significant time overhead showing the robustness of our model. Minimizing the adder depth led to a reduction from 6 to 3 when solving `LOWPASS_9`. Solving the BMCM problem with our model has permitted a large reduction of the solving time for the instance `LOWPASS_9`. It is clear that solving the model without any bound on the AD led to the exploration of a larger search space since the obtained solution has an AD of 6.

[1] `https://gitlab.univ-nantes.fr/remi-garcia/tool-mcm-roadef2022`

5 Conclusion

The MCM is a cornerstone operation of numerous computing applications and its efficiency is critical for the design of resource-constrained hardware circuits. With this work we introduced new ILP-based models for the MCM problem with three flavors: classic minimization of total number of adders, MCM under bounded adder depth, and a combined minimization of the adder count and depth. These models resolve several issues of the current state-of-the-art and offer novel features. The benchmarks demonstrate that better results can be obtained with our complete models, and in a reasonable time, permitting digital circuit designers to cover a larger design-space while relying on optimality of the results. The impact of the proposed models goes beyond the standalone MCM, as these are used as basic bricks in an ILP-based digital filter design [5].

References

[1] Levent Aksoy, Ece Olcay Güneş, and Paulo Flores. Search algorithms for the multiple constant multiplications problem: Exact and approximate. 34(5):151–162, August 2010.

[2] Erik Backenius and Erik Säll. Two's Complement Conversion to Minimal Signed Digit Code, 2005.

[3] Robert Bernstein. Multiplication by integer constants. *Software: Practice and Experience*, 16(7):641–652, 1986.

[4] Andrew G. Dempster and Malcolm D. Macleod. Constant integer multiplication using minimum adders. *IEE Proceedings - Circuits, Devices and Systems*, October 1994.

[5] Rémi Garcia, Anastasia Volkova, Martin Kumm, Alexandre Goldsztejn, and Jonas Kühle. Hardware-aware Design of Multiplierless Second-Order IIR Filters with Minimum Adders. working paper or preprint, August 2021.

[6] Oscar Gustafsson. Lower Bounds for Constant Multiplication Problems. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 54(11):974–978, November 2007.

[7] Oscar Gustafsson. Towards optimal multiple constant multiplication: A hypergraph approach. In *2008 42nd Asilomar Conference on Signals, Systems and Computers*, pages 1805–1809, October 2008.

[8] Martin Kumm. *Multiple Constant Multiplication Optimizations for Field Programmable Gate Arrays*. Springer Fachmedien Wiesbaden, Wiesbaden, 2016.

[9] Martin Kumm. Optimal Constant Multiplication Using Integer Linear Programming. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(5):567–571, 2018.

[10] Martin Kumm, Peter Zipf, Mathias Faust, and Chip-Hong Chang. Pipelined adder graph optimization for high speed multiple constant multiplication. In *2012 IEEE International Symposium on Circuits and Systems*. IEEE, May 2012.

[11] Jason Thong and Nicola Nicolici. An Optimal and Practical Approach to Single Constant Multiplication. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(9):1373–1386, 2011.