

Polynomial Verification of 2-Affine Spaces

Anna Bernasconi
Dipartimento di Informatica
Università di Pisa, Italy
anna.bernasconi@unipi.it

Valentina Ciriani
Dipartimento di Informatica
Università degli Studi di Milano, Italy
valentina.ciriani@unimi.it

Gianmarco Cuciniello
Dipartimento di Informatica
Università degli Studi di Milano, Italy
gianmarco.cuciniello@unimi.it

Caroline Dominik
Institute of Computer Science
University of Bremen, Germany
cardom@uni-bremen.de

Rolf Drechsler
Institute of Computer Science
University of Bremen/DFKI, Germany
drechsler@uni-bremen.de

Abstract—Polynomial Formal Verification (PFV) ensures that a class of circuits can be verified efficiently by calculating polynomial upper bounds for the resource demands of the verification process. In this paper, we address the PFV of Boolean affine spaces represented by a 2-XOR sum of products. We show that time and space resources remain quadratic in the number of input variables during the entire verification process. Specifically, we prove that the dimensions of ROBDDs and QRBDDs representing a 2-affine space are linear. Furthermore, we prove that all ROBDDs generated during the symbolic simulation of the circuit can be computed in linear time. Finally, we provide an overall quadratic upper bound for the formal verification of QRBDD-based circuits. The experimental results confirm the given bounds.

I. INTRODUCTION

Formal verification has become a central part of system design, as it is necessary to guarantee their correct behavior. For example, a circuit can be verified by assigning all possible combinations of inputs and comparing the result to a reference, but this is only feasible for very small designs. Using formal methods, on the other hand, symbolic inputs can be used, which are simulated in the same manner. The resulting outputs are formal expressions, which describe the behavior of the circuit for all input combinations. The efficiency of this process can be increased, by using *Binary Decision Diagrams* (BDDs) to represent the symbolic values. But the drawback of mathematical approaches, such as this *symbolic simulation*, is their computational complexity. Depending on the inspected circuit, the resource demands in time and memory can grow exponentially, and in addition they are hard to predict upfront. With *Polynomial Formal Verification* (PFV) [9], [11] the feasibility of the verification is guaranteed by giving polynomial upper bounds for the required runtime and space used by the entire verification process. Consequently, a broader use of formal methods in practice is possible as their application becomes more reliable. In the case of symbolic simulation, it is not sufficient to inspect the size of the final BDDs, which represent the outputs of the considered circuit. In fact, it is necessary to analyze each BDD that is computed during the symbolic simulation. PFV has already been proven for several types of functions and verification methods, such as

symmetric functions [10], adders or multipliers [11]. With the aim of identifying new classes of Boolean functions that can be verified in polynomial time, this paper investigates the characteristic functions of affine spaces. Their algebraic representation takes the form of an AND of XORs of literals, commonly referred to as *canonical expressions* (CEX). In particular, we focus on 2-CEX forms, where XORs contain at most two literals. These forms are particularly favored in logic synthesis due to technological constraints, as 2-XOR gates are preferred over unbounded XOR gates. 2-CEXs are frequently exploited in three-level logic minimization, such as 2-SPP forms [7], or for managing regular functions, including autosymmetric functions [1]. Moreover, the XOR operator plays an important role in current new technologies, such as reversible and quantum computing [1], [14], as well as cryptographic protocols such as secure multiparty computation [16].

In this paper, we prove that the time and space requirements for the PFV of 2-CEX forms are quadratic with respect to the number of input variables throughout the overall verification process. Specifically, we show that the dimensions of the ROBDDs and QRBDDs corresponding to a 2-CEX exhibit linear growth. Moreover, we prove that all ROBDDs produced during the symbolic simulation of the circuit can be computed in linear time. Finally, we provide a quadratic upper bound for the formal verification of QRBDD-based circuits. Our experimental results support the theoretical bounds.

II. PRELIMINARIES

A. Affine spaces

Let us consider the n Boolean variables x_1, \dots, x_n on the Boolean space $\{0, 1\}^n$ where each point is described by a binary vector of n elements. An *XOR factor* is an XOR, i.e., \oplus , of one or more variables, one of which possibly complemented. Moreover, $(\{0, 1\}^n, \oplus)$ is the Boolean space of vectors with n components and the elementwise XOR operator \oplus . Let α be a vector in $\{0, 1\}^n$ and let V be a vector subspace of $(\{0, 1\}^n, \oplus)$, we can define $A = \alpha \oplus V$ as the *affine space* over V with vector α , by applying the XOR between α and each point of V . Given an affine space A , we can partition the input variables into two classes: canonical and

non-canonical variables. The canonical variables are the truly independent variables in A , which can assume all possible combinations of 0-1 values. In contrast, the non-canonical variables are not independent because they can be defined as XORs of the canonical ones or have a constant value. Any affine space A can be represented by an algebraic expression (CEX) that is an AND of XOR factors [2], [3], [7]. In particular, a 2-CEX contains XOR factors of at most two literals, i.e., it is an AND of XOR factors that contain at most two variables: a non-canonical variable and possibly a canonical one. For example, the CEX $(x_1 \oplus \bar{x}_2)(x_1 \oplus \bar{x}_3)x_4(x_5 \oplus x_7)\bar{x}_8$ in $\{0, 1\}^9$ is a 2-CEX, with two canonical variables, x_1 and x_5 . Notice that the non-canonical variables are either constant ($x_4 = 1$, $x_8 = 0$) or depend on the canonical ones (e.g., $x_2 = x_1$ and $x_7 \neq x_5$).

B. QRBDD-based circuit

A *Binary Decision Diagram* (BDD) is a directed, acyclic graph, where each internal node is labeled by a Boolean variable x_i and has exactly two outgoing edges, the 0-edge and the 1-edge, pointing to two nodes called the 0-child and the 1-child, respectively. The terminal nodes are labeled 0 or 1. BDDs are typically used to efficiently represent and manipulate Boolean functions [5], [6].

A BDD is *ordered* if all variables are sorted according to a total order x_1, \dots, x_n . A BDD is *reduced* if no redundant nodes or isomorphic subgraphs exist. A reduced and ordered BDD is called *ROBDD*. The *size* of a BDD is defined as the number of its internal and terminal nodes, while the *width* is the maximum number of nodes on the same level. Finally, an ordered BDD is *Quasi-Reduced* (QRBDD) if only isomorphic subgraphs are reduced, while redundant nodes are kept in the diagram (see, e.g., Fig. 1) [12]. In a QRBDD every path from the root node to a terminal node contains every input variable. QRBDDs can be exploited to construct small-depth circuits, following the synthesis method presented in [12]. The idea is to build a circuit C_f that describes the reachability between the root-node of the QRBDD for a function f and the two terminal nodes. This is done by first using *Reachability Matrices* (RMs) to describe the reachability between the nodes of adjacent levels. For levels ℓ and $\ell + 1$ with width w_ℓ and $w_{\ell+1}$, respectively, this is a matrix $RM_{\ell, \ell+1}$ of dimension $w_\ell \times w_{\ell+1}$ whose entry (i, j) is a literal or a constant that describes how the j -th node on level $\ell + 1$ is reached from the i -th node on level ℓ : through a negative edge (\bar{x}_ℓ), a positive edge (x_ℓ), both edges (1) or no edge (0). Then, all RMs are multiplied in pairs using *Binary Matrix Multiplications* (BMMs), e.g., $BMM_{\ell, m}$ then describes the reachability of nodes on level m from nodes on level ℓ . The BMMs are aligned in a tree-shaped manner to reduce the depth (see Fig. 2). In particular, the matrix $BMM_{0, n}$ describes the reachability between the root and the terminal nodes and therefore represents f .

In practice, C_f can be implemented as follows. Each RM contains only literals and/or constants and is simply implemented using connections without gates. As the entries of BMMs are *sums of products* (SOPs), they are implemented by

first combining the inputs using a level of fan-in 2 AND-gates and then using levels of fan-in 2 OR-gates to combine their results. For a QRBDD with maximum width w , this results in a number of gates in $O(w^3)$ per BMM and a depth in $O(\log w)$ due to the tree-like sorting of the OR gates. The overall circuit C_f has a maximum size in $O(nw^3)$, since the circuit consists of $n - 1$ BMMs, and the maximum depth is $O(\log n \log w)$, as stated in Theorem 3 of [12]. Thus, if the QRBDD for f has a maximum width that is polynomial in n , the circuit C_f has polynomial size and logarithmic depth.

III. POLYNOMIAL VERIFICATION OF 2-AFFINE SPACES

We investigate the size of the ROBDD and QRBDD representation of 2-XOR affine spaces, and prove that the size of the QRBDD is linear in the number of input variables. This allows us to derive a compact QRBDD-based circuit C_f for the given 2-XOR affine space. We then show that the size of all ROBDDs constructed during the symbolic simulation of the circuit C_f is linear in the number of variables.

A. ROBDDs and QRBDDs for 2-Affine Spaces

The optimal ROBDD representation of 2-XOR Boolean affine spaces is studied in [4]. In the sequel, we show that the optimal representation is linear, in the number of input variables, for both ROBDDs and QRBDDs. Moreover, we prove that the maximum widths of the ROBDD and QRBDD, representing a 2-affine space, are constant. In this section, we always refer to ROBDDs and QRBDDs in the optimal variable ordering described in [4]. For the sake of clarity, we only consider 2-CEXs without complemented variables in the 2-XORs (all results can be easily extended to general 2-CEXs).

Lemma 1: The size of the ROBDD that represents a 2-affine space is at most $2(n + 1) - k$, where k is the dimension of the affine space. The maximum width of the ROBDD for a 2-affine space is at most 2.

Proof. Follows from [4]. ■

Lemma 2: The size of the QRBDD representing a 2-affine space is at most $3n - k + 1$. The maximum width of the QRBDD representing a 2-affine space is at most 3.

Proof. Recall that we can partition the input variables into two subsets: the subset of the k canonical (i.e., independent) variables and the remaining $n - k$ non-canonical ones. In the ROBDD, each non-canonical variable is directly connected to the variable in the next level and to the 0 terminal with the other branch. Therefore, to make the ROBDD quasi-reduced, we need to insert one chain of new redundant nodes containing all the variables (in the ordering) from the upper non-canonical variable to the 0-terminal. Each non-canonical variable instead of being directly connected to 0, is connected to the next variable in this redundant chain. The nodes corresponding to the canonical variables are always connected to the variables in the next level and do not originate any chain in the QRBDD. Therefore, each QRBDD level has a width of at most 3 (the 2 nodes of the corresponding ROBDD plus the additional redundant node in the chain). Recalling that the number of levels is n , we have two cases: 1) if the root is a non-canonical

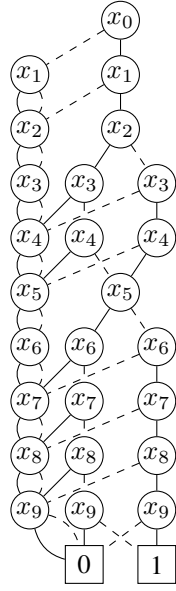


Fig. 1. QRBDD of the function $f = x_0 x_1 (x_2 \oplus x_3) (x_2 \oplus x_4) (x_5 \oplus x_6) (x_5 \oplus x_7) (x_5 \oplus x_8) (x_5 \oplus x_9)$

variable, then the additional chain contains all the variables but one, since in the ROBDD it is directly connected to 0; 2) if the root is a canonical variable, then the additional chain contains all the variables but two, since in the ROBDD it is not connected to 0. Therefore, by Lemma 1, the total number of nodes is at most $n - 1 + 2(n + 1) - k = 3n - k + 1$. ■

By Lemmas 1 and 2 we obtain the following proposition.

Proposition 1: The size of ROBDDs and QRBDDs representing 2-affine spaces is linear (that is, is in $O(n)$). The maximum widths of ROBDDs and QRBDDs representing a 2-affine space are constant.

Finally, combining Proposition 1 with Theorem 3 of [12], we obtain the following result.

Proposition 2: Let C_f be the QRBDD-based circuit for a function f representing a 2-XOR affine space, with n variables. C_f has linear size in $O(n)$ and depth in $O(\log n)$.

Example 1: Let us consider the CEX $f(x_0, \dots, x_9) = x_0 x_1 (x_2 \oplus x_3) (x_2 \oplus x_4) \cdot (x_5 \oplus x_6) (x_5 \oplus x_7) (x_5 \oplus x_8) (x_5 \oplus x_9)$ representing a 2-XOR affine space with canonical variables x_2 and x_5 . The QRBDD representing f is shown in Fig. 1, with the chain of redundant nodes represented on the left side. Note how nodes in this chain can only reach other nodes within the chain. Also, note how all levels contain at most three nodes.

B. Symbolic Simulation

In the previous subsection, we have shown that the QRBDD representing a 2-affine space contains a linear number of nodes and has a constant width. Thus, C_f has linear size and logarithmic depth. In this section, we show that the symbolic simulation of C_f can be performed in polynomial time. Recall that we consider 2-CEXs without complementations. Moreover, without loss of generality, we assume that all singleton non-canonical variables are the first in the variable

ordering for the BDDs. First, let us discuss the size of all ROBDDs constructed during the symbolic simulation of C_f . The following proposition holds.

Proposition 3: Let C_f be the QRBDD-based circuit for a function f , which is the characteristic function of a 2-affine space. Each output signal of the $BMM_{i,j}$ subcircuits of C_f can be computed with at most 3 AND gates and 2 OR gates of fan-in 2. Moreover, one output signal of each row of $BMM_{i,j}$ can be computed after the other signals on the same row, with one inverter and at most one fan-in 2 OR gate.

Proof. First, observe that the maximum width of $QRBDD_f$ is 3. Thus, each output signal $BMM_{i,j}$ subcircuit of C_f corresponds to a matrix of dimension at most 3×3 . Since each output of $BMM_{i,j}$ is the inner product of at most three inputs, then it can be computed with an SOP that contains at most 3 AND gates and 2 OR gates of fan-in 2. Moreover, since the sum of all entries in the rows of any BMM matrix must be equal to 1, we can alternatively compute one output signal in each row of $BMM_{i,j}$ as the complement of the other entry or as the complement of the sum of the other two entries in the same row. This strategy requires first to compute the other signals in the same row, which are at most 2, and then to take the complement of their OR, as shown in Fig. 3. ■

We now prove that all ROBDDs constructed during the symbolic simulation of C_f have linear size.

Lemma 3: Let C_f be the QRBDD-based circuit for a function f with n variables, which is the characteristic function of a 2-affine space. Let $QRBDD_f$ be the QRBDD for f . Then, the size of all ROBDDs constructed during the symbolic simulation of C_f is in $O(n)$.

Proof. Each ROBDD constructed during the symbolic simulation of C_f is either for an output signal or for an internal signal of a binary matrix multiplication circuit, $BMM_{i,j}$, with $0 \leq i < j < n$. We consider these two cases separately.

Output signals. Let us consider an ROBDD for an output signal of a $BMM_{i,j}$ subcircuit. Recall that these signals describe the reachability of nodes on level j of $QRBDD_f$ starting from nodes on level i . We then study the QRBDD that describes the reachability between two nodes $x_{i,k}$ and $x_{j,h}$, where $0 \leq h, k \leq 2$ since the maximum width of $QRBDD_f$ is 3, as observed in Proposition 3. Notice that these reachabilities are formally described by the entries of the matrix representing the output signals of $BMM_{i,j}$. We suppose that the chain of redundant nodes is on the left side of the $QRBDD_f$. Thus, let 0 be the index of the nodes in the chain. Notice that nodes in the chain can only reach other nodes in the chain. Since we have at most 3 nodes in each level, we must consider $0 \leq k, h \leq 2$. This can be divided into three cases:

(1) $k = 0$ (unless $x_{i,k}$ is the root): In this case the node $x_{i,k}$ lays on the chain. Therefore, only nodes in the chain can be reached, and the ROBDDs representing these signals are the terminal 1 for $h = 0$, and the terminal 0 for $h = 1, 2$.

(2) $k = 1, 2, h = 1, 2$ or $x_{i,k}$ is the root: Here, the constructed ROBDD represents a CEX derived from the original one and therefore has linear dimensions. For this,

all singletons and XORs containing only variables preceding x_i and following x_j in the variable ordering are eliminated. Then, it consists of three parts: (i) A preceding product of literals is computed, if $x_{i,k}$ is non-canonical associated to some canonical variable x_s . It describes how $x_{i,k}$ reaches the closest following canonical variable x_r or $x_{j,h}$. This is done by eliminating x_s from all XOR factors, including and following x_i . If $x_{i,k}$ belongs to the branch where x_s assumes the value 1, all literals in the product are complemented. Note that this gives 0 if $x_{i,k}$ and $x_{j,h}$ are non-canonicals associated with the same canonical, but on different branches of the canonical. (ii) Following the variable order, all singletons and XOR factors containing x_i, \dots, x_{j-1} are kept, excluding XOR factors that contain x_s if $x_{i,k}$ is a non-canonical associated with x_s , or ones that contain x_r if $x_{j,h}$ is a non-canonical associated with x_r . This can be none if $x_{i,k}$ and $x_{j,h}$ correspond to the same canonical variable, $x_{j,h}$ corresponds to $x_{i,k}$ or there are no such XORs in between. (iii) A succeeding product of literals is computed if $x_{j,h}$ is a non-canonical associated to some canonical x_r and $x_{i,k}$ is not associated to x_r as well. It describes how $x_{j,h}$ is reached from x_r and is obtained by taking all XOR factors that contain x_r but no variable following x_j in the variable order, and eliminating x_r . Again, if $x_{j,h}$ belongs to the branch where x_r assumes value 1, all literals are complemented. Note that x_r can be $x_{i,k}$. Finally, we multiply the CEX by x_r or \bar{x}_r , depending on whether $x_{j,h}$ belongs to the path starting from the 1- or 0-edge of x_r .

(3) $h = 0$: Here, the node $x_{j,h}$ lays on the chain or is the 0 terminal. We first show that the sum (OR) B_S of the ROBDDs describing the reachability of the other nodes on the level j with $h = 1$ and $h = 2$, represents a CEX and is therefore of linear size. This automatically holds if $w_j = 2$. As seen in Case 2, the two CEX obtained for $h = 1$ and $h = 2$ are identical except for the complementation of all literals in subcase (iii). For their sum, simply all XOR factors that contain x_r , but no variable following x_j , are kept without eliminating x_r . Complementing B_S , we obtain the ROBDD representing the reachability of node $x_{j,0}$, which must then also be of linear size.

Internal signals. There are two possible cases.

(1) (AND) The internal signal is the output of an AND gate. Notice that the input functions of this AND gate are the output signals already discussed, each represented with a linear size ROBDD. The size of their AND is therefore linear, since the two ROBDDs are defined on disjoint sets of variables.

(2) (OR) Notice that there are at most three AND gates (i.e., the maximum number of rows and columns of the BMM matrices is three). Therefore, to sum up their outputs, we need at most two OR gates, one internal OR and one external. Furthermore, we can observe that the internal OR operation occurs exclusively during the multiplication of a matrix $BMM_{i,\ell}$ with three columns and a matrix $BMM_{\ell,j}$ with three rows. The first matrix $BMM_{i,\ell}$ describes the reachability of nodes associated with a non-canonical variable x_ℓ on level ℓ , starting from any level $i < \ell$ in $QRBDD_f$, including the level 0 of the root. In contrast, the matrix $BMM_{\ell,j}$ describes the paths that

originate from the nodes on level ℓ , corresponding to the same non-canonical variable x_ℓ , and reach the nodes on level $j > \ell$. Moreover, the internal OR in such a matrix product can only be relevant if the node $x_{i,k}$ on level i is not in the chain and corresponds to a canonical variable, a non-canonical variable such that there is at least a canonical variable between levels i and j , and the root; and if the target node $x_{j,h}$ on level j is in the chain, i.e., $h = 0$, or is the 0-terminal. Indeed, only in these cases the node $x_{i,k}$ can reach all nodes on the level ℓ , corresponding to the non-canonical variable x_ℓ , and all nodes on the level ℓ can reach the target node $x_{j,h}$ on the level j . Otherwise, at least one of the results of the three AND gates is 0 and thus, the internal OR is either 0 or equal to the external OR (which corresponds to an external signal, already studied above). In other words, during matrix multiplication, the computation of a relevant internal OR is needed only to derive the entries of the first column of $BMM_{i,j}$. The entries in all other columns are the result of only external ORs, and we already know that they can be computed by ROBDDs of linear size. Since the sum of all entries in the rows of any BMM matrix must be equal to 1, we can now compute the entries of the first column of $BMM_{i,j}[k, 0]$ for $k \geq 0$, replacing the inner product computation, which requires two ORs, with an alternative strategy: we first compute the other entries $BMM_{i,j}[k, h]$ with $h > 0$, and then we complement their sum. More precisely, we compute $BMM_{i,j}[k, 0]$ complementing the entry $BMM_{i,j}[k, 1]$ if level j contains only two nodes, or complementing the sum of the other two entries $BMM_{i,j}[k, 1] + BMM_{i,j}[k, 2]$ when level j corresponds to a non-canonical variable x_j in a 2-XOR factor and contains three nodes. The first case simply requires to complement a CEX represented by a ROBDD of linear size, obviously resulting in a ROBDD of linear size. In the second case, we first need to compute a ROBDD for $BMM_{i,j}[k, 1] + BMM_{i,j}[k, 2]$. We know from the analysis conducted for external signals, that these two entries are CEXs that can differ only for the complementation of the succeeding product of literals. Thus, their sum is again a CEX, represented by a ROBDD of linear size. Complementing this ROBDD, we eventually obtain a linear size ROBDD representing the entry $BMM_{i,j}[k, 0]$. ■

Example 2: Let us consider the QRBDD depicted in Fig. 1, representing the 2-affine space $f = x_0x_1(x_2 \oplus x_3)(x_2 \oplus x_4)(x_5 \oplus x_6)(x_5 \oplus x_7)(x_5 \oplus x_8)(x_5 \oplus x_9)$, in particular the following RMs for $QRBDD_f$:

$$RM_{4,5} = \begin{pmatrix} 1 & 0 \\ x_4 & \bar{x}_4 \\ \bar{x}_4 & x_4 \end{pmatrix} \quad RM_{5,6} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & x_5 & \bar{x}_5 \end{pmatrix}$$

$$RM_{6,7} = \begin{pmatrix} 1 & 0 & 0 \\ x_6 & \bar{x}_6 & 0 \\ \bar{x}_6 & 0 & x_6 \end{pmatrix} \quad RM_{7,8} = \begin{pmatrix} 1 & 0 & 0 \\ x_7 & \bar{x}_7 & 0 \\ \bar{x}_7 & 0 & x_7 \end{pmatrix}$$

All RMs are multiplied in a tree-like arrangement, as depicted in Fig. 2. This gives the following BMMs:

$$BMM_{4,6} = RM_{4,5} \cdot RM_{5,6} = \begin{pmatrix} 1 & 0 & 0 \\ x_4 & \bar{x}_4 x_5 & \bar{x}_4 \bar{x}_5 \\ \bar{x}_4 & x_4 x_5 & x_4 \bar{x}_5 \end{pmatrix}$$

$$BMM_{6,8} = RM_{6,7} \cdot RM_{7,8} = \begin{pmatrix} 1 & 0 & 0 \\ x_6 + x_7 & \bar{x}_6 \bar{x}_7 & 0 \\ \bar{x}_6 + \bar{x}_7 & 0 & x_6 x_7 \end{pmatrix}$$

$$BMM_{4,8} = \begin{pmatrix} 1 & 0 & 0 \\ x_4 + (x_5 \oplus \bar{x}_6) + (x_5 \oplus \bar{x}_7) & \bar{x}_4 x_5 \bar{x}_6 \bar{x}_7 & \bar{x}_4 \bar{x}_5 x_6 x_7 \\ \bar{x}_4 + (x_5 \oplus \bar{x}_6) + (x_5 \oplus \bar{x}_7) & x_4 x_5 \bar{x}_6 \bar{x}_7 & x_4 \bar{x}_5 x_6 x_7 \end{pmatrix}$$

The entries of these matrices have been simplified and are represented as CEXs, or the complement of CEXs, rather than as SOPs, to enhance readability. Based on these BMMs, we now provide explicit examples for the proof of Lemma 3. We first consider the three cases for the output signals of the $RM_{i,j}$ and of the $BMM_{i,j}$ sub-circuits which describe the reachability between two nodes $x_{i,k}$ and $x_{j,h}$, where $0 \leq h, k < 2$, and $0 \leq i < j \leq 9$:

(1) $k = 0$ (unless $x_{i,k}$ is the root): Here, the entries in the BMMs are 1 for $h = 0$ and 0 for $h = 1, 2$, which can be seen in the first row of $BMM_{4,6}$, $BMM_{6,8}$, and $BMM_{4,8}$.

(2) $k = 1, 2, h = 1, 2$ or $x_{i,k}$ is the root: For example entry $\bar{x}_4 x_5 \bar{x}_6 \bar{x}_7$ of $BMM_{4,8}$ is again a CEX constructed by reducing the original CEX. It describes the reachability between two non-canonical variables x_4 and x_8 , that are associated with different canonical variables x_2 and x_5 , respectively. In this case, all paths between x_4 and x_8 lie outside the chain and pass through x_5 . The preceding product of literals of (i) is \bar{x}_4 , the succeeding product of literals of (ii) is $x_5 \bar{x}_6 \bar{x}_7$. No part for (ii) is necessary, as no canonical variables are in between x_2 and x_5 , which means no XORs in between have to be kept. We can notice that the CEX $\bar{x}_4 \bar{x}_5 x_6 x_7$ occurring in the same row, differs in the complementation of the literals in the succeeding product of literals, as described in (iii). Meanwhile, the CEXs below those two CEXs, differ in the complementation of product of literals, as described in (i). Another example can be seen in $BMM_{6,8}$: The two non-canonical variables x_6 and x_8 are associated with the same canonical variable x_5 . As stated in (i), if $x_{i,k}$ and $x_{j,h}$ belong to different branches (i.e., $k \neq h$) there are no paths between them, and the corresponding entry in $BMM_{6,8}$ is 0. However, if the two nodes are in the same branch then the path is described by the product $\bar{x}_6 \bar{x}_7$ for $h = k = 1$ (left branch), and $x_6 x_7$ for $h = k = 2$ (right branch).

(3) $h = 0$: This case, where the target node lies on the redundant chain, is represented, for instance, by the first column of $BMM_{4,8}$. The paths are now described by the complementation of the union of the other two CEXs in the same row. Notice that these two expressions differ only in the complementations of the last literals, thus their union is again a CEX representing a 2-affine space. For instance, in the second row of $BMM_{4,8}$, the sum of the two last entries is $\bar{x}_4 x_5 \bar{x}_6 \bar{x}_7 + \bar{x}_4 \bar{x}_5 x_6 x_7$, which is equivalent to the CEX $\bar{x}_4 (x_5 \oplus x_6)(x_5 \oplus x_7)$. The first entry of the second row of $BMM_{4,8}$ is the complement of this: $x_4 + (x_5 \oplus \bar{x}_6) + (x_5 \oplus \bar{x}_7)$.

Let us now consider the ROBDDs that are used in the computation of the internal signals. An example for the disjoint sets of variables, which causes the output of an AND gate to be a linear size BDD, can be seen when comparing the two matrices $BMM_{4,6}$ and $BMM_{6,8}$. An example for internal signals that are the output of an OR gate and are computed as the complement of the sum of the other entries in the same row is the first column of $BMM_{4,8}$, already considered for Case 3. The corresponding implementation is depicted in Fig. 3, and requires an overall number of 7 fan-in 2 gates and one inverter, organized in a depth 4 sub-circuit. The standard computation requires instead 10 fan-in 2 gates organized in a depth 3 sub-circuit.

C. Circuit Synthesis based on QRBDDs

We now evaluate the runtime of each ITE-operation carried out during symbolic simulation for AND and OR gates, essential for the overall runtime. For the AND operation, we recall from [8] the following lemma.

Lemma 4 ([8]): Let C_f be the QRBDD-based circuit for a function f with n variables. Let A_i be an AND gate in C_f and F and G the ROBDDs of the inputs of A_i . Then the runtime of the ITE operator that constructs the output-ROBDD of A_i during the symbolic simulation of C_f is in $O(1)$.

The proof of this lemma exploits the fact that the supports of the inputs of the AND gates are disjoint. Meanwhile, for

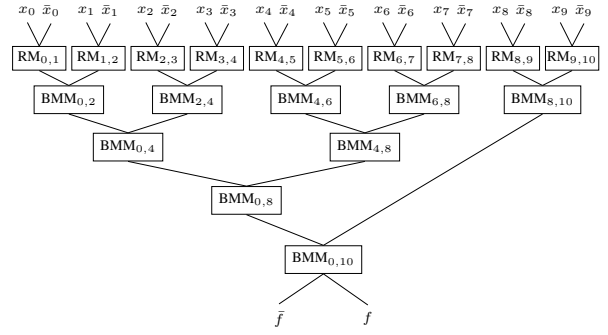


Fig. 2. Structure of a QRBDD-based circuit C_f for the function f of the running example.

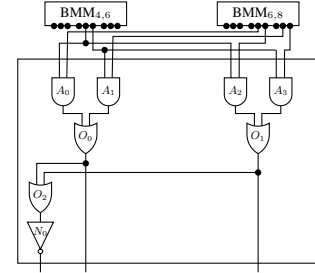


Fig. 3. Computation of the second row of outputs of $BMM_{4,8}$ exploiting the fact that the sum of all entries in a row must be 1.

the OR operation, we exploit the fact that the inputs of all OR gates required for the computation of the BMMs (for both internal and external signals) are always 2-CEXs that differ only for the polarities of their last terms, as observed in the proof of Lemma 3. In this case, the OR of the two ROBDDs can be computed in linear time, that is, $O(n)$ (the proof will be inserted in the extended version of the paper):

Lemma 5: Let C_f be the QRBDD-based circuit for a function f with n variables. Let O_i be any OR gate in C_f and F and G the ROBDDs of the inputs of O_i . Since F and G represent 2-CEXs that differ only in the polarities of their last factors, then the runtime of the ITE-operator constructing the output-ROBDD of O_i during the symbolic simulation of C_f is in $O(n)$.

Combining all results derived or reviewed so far, we can finally provide an overall upper bound for the formal verification of QRBDD-based circuits for 2-affine spaces.

Theorem 1: Let C_f be the QRBDD-based circuit for a function f with n variables, which is the characteristic function of a 2-affine space. Let $QRBDD_f$ be the QRBDD for f . Then the symbolic simulation of C_f has time complexity in $O(n^2)$ and space complexity in $O(n^2)$.

Proof. The overall runtime depends on the number of ITE-operations during the symbolic simulation of C_f and on their runtime. From Lemma 3 it follows that the ROBDD constructed after each gate has a maximum size of $O(n)$. Moreover, Lemma 4 implies that the ITE operator for AND-gates needs at most $O(1)$ steps, while Lemma 5 implies that the ITE operator for OR-gates needs $O(n)$ steps. Recall from

TABLE I
EXPERIMENTAL RESULTS FOR EACH TESTCASE.

| 2-CEX | n | #Gates | #Nodes QRBDD | #Nodes ROBDD | Runtime |
|---|----|--------|--------------------|------------------|---------|
| $x_5(x_2 \oplus x_7)(x_1 \oplus x_6)$ | 5 | 48 | 13 (≤ 15) | 9 (≤ 10) | 0.0622s |
| $(x_1 \oplus x_5)(x_1 \oplus x_4)(x_0 \oplus x_7)(x_1 \oplus x_3)(x_0 \oplus x_6)(x_1 \oplus x_2)$ | 8 | 96 | 22 (≤ 24) | 16 (≤ 16) | 0.0680s |
| $(x_7 \oplus x_8)(x_7 \oplus x_9)x_2x_4(x_5 \oplus x_0)(x_5 \oplus x_3)(x_5 \oplus x_1)(x_5 \oplus x_6)$ | 10 | 130 | 27 (≤ 30) | 18 (≤ 20) | 0.0842s |
| $x_5(x_{12} \oplus x_8)x_3x_{13}(x_1 \oplus x_9)(x_7 \oplus x_4)(x_{12} \oplus x_2)(x_7 \oplus x_{11})(x_7 \oplus x_{14})(x_1 \oplus x_6)$ $(x_1 \oplus x_{10})(x_7 \oplus x_6)$ | 15 | 203 | 40 (≤ 45) | 26 (≤ 30) | 0.0835s |
| $(x_0 \oplus x_7)(x_2 \oplus x_6)(x_1 \oplus x_5)(x_2 \oplus x_3)(x_0 \oplus x_4)(x_8 \oplus x_{15})(x_9 \oplus x_{12})(x_8 \oplus x_{11})$ $(x_{10} \oplus x_{13})(x_{10} \oplus x_{16})(x_9 \oplus x_{18})(x_1 \oplus x_{14})(x_2 \oplus x_{19})(x_{10} \oplus x_{17})$ | 20 | 291 | 54 (≤ 60) | 36 (≤ 40) | 0.0995s |
| $(x_{12} \oplus x_9)(x_5 \oplus x_{15})(x_5 \oplus x_{19})(x_5 \oplus x_{25})(x_{12} \oplus x_2)(x_{21} \oplus x_{16})x_3x_7x_{23}x_8x_{26}$ $(x_5 \oplus x_{24})(x_{14} \oplus x_0)(x_1 \oplus x_6)(x_1 \oplus x_4)(x_1 \oplus x_{27})(x_1 \oplus x_{17})(x_{28} \oplus x_{29})$ $(x_1 \oplus x_{22})(x_5 \oplus x_{20})(x_5 \oplus x_{18})(x_5 \oplus x_{10})(x_{12} \oplus x_{13})(x_5 \oplus x_{11})$ | 30 | 467 | 80 (≤ 90) | 51 (≤ 60) | 0.1428s |
| $(x_1 \oplus x_{22})(x_2 \oplus x_{39})(x_4 \oplus x_{18})(x_{35} \oplus x_5)(x_{28} \oplus x_{30})(x_6 \oplus x_{20})(x_1 \oplus x_{36})$ $(x_7 \oplus x_{29})(x_{28} \oplus x_{23})(x_8 \oplus x_{14})(x_6 \oplus x_{27})(x_{28} \oplus x_0)x_9x_3x_{12}x_{21}(x_2 \oplus x_{19})$ $x_{10}x_{25}(x_{11} \oplus x_{37})(x_{28} \oplus x_{13})(x_{15} \oplus x_{26})(x_6 \oplus x_{33})(x_{16} \oplus x_{38})(x_{28} \oplus x_{17})$ $(x_{35} \oplus x_{24})(x_{11} \oplus x_{32})(x_7 \oplus x_{34})(x_{28} \oplus x_{31})$ | 40 | 591 | 104 (≤ 120) | 65 (≤ 80) | 0.2230s |
| Total average (on the set of 1000 2-CEXs) | 6 | 55 | 16 (≤ 19) | 11 (≤ 13) | 0.0715s |

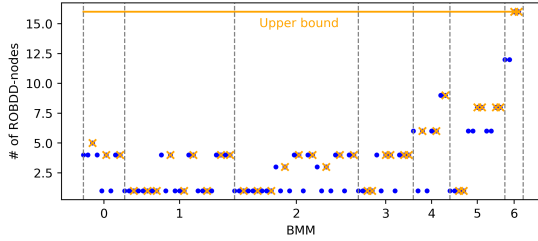


Fig. 4. BDD-sizes during verification of $(x_1 \oplus x_5)(x_1 \oplus x_4)(x_0 \oplus x_7)(x_1 \oplus x_3)(x_0 \oplus x_6)(x_1 \oplus x_2)$.

Proposition 2 that the size of C_f is in $O(n)$, thus implying at most $O(n)$ ITE-operations of at most $O(n)$ runtime. Thus, we get an overall runtime of $O(n^2)$. The space complexity depends on the number of ROBDDs that are needed simultaneously. C_f consists of levels of RMs or BMMs, and the BMMs consist of levels of AND and OR-gates. All ROBDDs constructed on the same level need to be built simultaneously to simulate the next level. The number of BMMs on a level is at most $\lfloor n/2 \rfloor$. Since $QRBDD_f$ has a maximum of 3 nodes on each level, both dimensions of each RM are bounded by 3. Therefore, each BMM computes at most nine matrix cells on the same level, and each cell requires at most eight gates (i.e., 4 AND-gates, three ORs and an inverter; see Fig. 3). It follows that overall at most $\lfloor n/2 \rfloor \cdot O(1) \in O(n)$ gates are on the same level. Since the maximum size of the ROBDDs constructed during the simulation is in $O(n)$, the overall space complexity is in $O(n^2)$. ■

IV. EXPERIMENTAL RESULTS

The symbolic simulation of 2-CEXs was implemented in C, using CUDD [15] for ROBDD operations and a version of the ROBDD package *iBDD*¹ [13] for QRBDD operations. This was executed for 7 testcases with up to $n = 40$ on a 3.8-4.8 GHz Intel Core Ultra 7 155U CPU with 64 GB RAM. The results can be seen in Table I. Each 2-CEX is given in the first column, n in the second column, and the number

¹R. Krauß, FrEDDY (previously iBDD) [Source code], <https://github.com/runekrauss/freddy>, 2025 (accessed on 2025-07-22)

of gates of the resulting circuit is given in the third column. The number of nodes of the respective QRBDD and ROBDD are in the fourth and fifth column, and the execution time of the symbolic simulation can be found in the last column. For comparison, the values $3 \cdot n$ and $2 \cdot n$ are added in brackets next to the QRBDD size and the ROBDD size, respectively. As the sizes never exceed both of these linear bounds, this supports Proposition 1. The measured runtimes never exceed 0.3s which underlines the efficiency of the proposed approach. Further, a set of 1000 2-CEXs used in [4] was verified. The rounded average result is given in the last row of Table I and is similar to the results of the other testcases.

To further analyze blowups during the verification, the size of each ROBDD constructed during the symbolic simulation was measured. The ROBDD size per BMM can be seen in Fig. 4 for the testcase with $n = 8$. The ROBDDs constructed at an output signal are marked with orange crosses, and the ROBDDs constructed inside a BMM are marked with blue dots. For comparison, the upper bound $2 \cdot n$ is marked as an orange line. This line is never exceeded by any measured ROBDD size, which underlines the bound of $O(n)$ given in Lemma 3 for the size of all ROBDDs. The results for the remaining testcases listed in Table I are similar.

V. CONCLUSION

In this paper, we have proved a quadratic upper bound on the resources needed for PFV of 2-affine spaces. In this research area, a main future goal is to further identify classes of functions that can be verified in polynomial time and space.

ACKNOWLEDGEMENTS

This work was supported in part by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the Italian MUR. Neither the European Union nor the Italian MUR can be held responsible for them. Further, this work was supported by the German Research Foundation (DFG) within the Reinhart Koselleck Project *PolyVer* (DR 287/36-1).

The first two authors are members of the Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM), which provided partial support for this work.

REFERENCES

- [1] A. Bernasconi, A. Berti, V. Ciriani, G. M. D. Corso, and I. Fulginiti, "XOR-AND-XOR logic forms for autosymmetric functions and applications to quantum computing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 6, pp. 1861–1872, 2023.
- [2] A. Bernasconi, S. Cimato, V. Ciriani, and M. C. Molteni, "Multiplicative complexity of XOR based regular functions," *IEEE Transactions on Computers*, vol. 71, no. 11, pp. 2927–2939, 2022.
- [3] A. Bernasconi and V. Ciriani, "Autosymmetry of incompletely specified functions," in *Design, Automation & Test in Europe Conference & Exhibition*, 2021, pp. 360–365.
- [4] A. Bernasconi, V. Ciriani, and M. Longhi, "On the optimal OBDD representation of 2-XOR Boolean affine spaces," in *Design, Automation & Test in Europe Conference & Exhibition*, 2022, pp. 1437–1442.
- [5] R. Bryant, "Graph based algorithm for Boolean function manipulation," *IEEE Transactions on Computers*, vol. 35, no. 9, pp. 667–691, 1986.
- [6] —, "Symbolic Boolean manipulation with ordered binary decision diagrams," *ACM Computing Surveys*, vol. 24, no. 3, pp. 293–318, 1992.
- [7] V. Ciriani, "Synthesis of SPP three-level logic networks using affine spaces," *IEEE Transactions on CAD of Integrated Circuits and Systems*, vol. 22, no. 10, pp. 1310–1323, 2003.
- [8] C. Dominik and R. Drechsler, "A class of polynomially verifiable circuits of logarithmic depth," in *13th International Conference on Cloud Computing, Data Science & Engineering*, 2023, pp. 271–276.
- [9] R. Drechsler, "PolyAdd: Polynomial formal verification of adder circuits," in *IEEE Design and Diagnostics of Electronic Circuits and Systems*, 2021, pp. 99–104.
- [10] R. Drechsler and C. Dominik, "Edge verification: Ensuring correctness under resource constraints," in *Symposium on Integrated Circuits and Systems Design*, 2021, pp. 1–6.
- [11] R. Drechsler and A. Mahzoon, "Polynomial formal verification: Ensuring correctness under resource constraints," in *IEEE International Conference on Computer-Aided Design*, 2022, pp. 1–9.
- [12] N. Ishiura, "Synthesis of multilevel logic circuits from binary decision diagrams," *IEICE Transactions on Information and Systems*, vol. 76, pp. 1085–1092, 1993.
- [13] R. Krauss, J. Zielasko, and R. Drechsler, "FrEDDY: Modular and efficient framework to engineer decision diagrams yourself," in *Design, Automation & Test in Europe Conference & Exhibition*, 2025, pp. 1–2.
- [14] B. Schmitt, F. Mozafari, G. Meuli, H. Riener, and G. D. Micheli, "From Boolean functions to quantum circuits: A scalable quantum compilation flow in C++," in *Design, Automation & Test in Europe Conference & Exhibition*, 2021, pp. 1044–1049.
- [15] F. Somenzi, "CUDD: CU decision diagram package release 2.5.0," 2012. [Online]. Available: <https://github.com/ivmai/cudd>
- [16] E. Testa, M. Soeken, H. Riener, L. Amaru, and G. D. Micheli, "A logic synthesis toolbox for reducing the multiplicative complexity in logic networks," in *Design, Automation & Test in Europe Conference & Exhibition*, 2020, pp. 568–573.