

Lower T -count with faster algorithms

Vivien Vandaele^{1,2}

¹Eviden Quantum Lab, Les Clayes-sous-Bois, France

²Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France

Among the cost metrics characterizing a quantum circuit, the T -count stands out as one of the most crucial as its minimization is particularly important in various areas of quantum computation such as fault-tolerant quantum computing and quantum circuit simulation. In this work, we contribute to the T -count reduction problem by proposing efficient T -count optimizers with low execution times. In particular, we greatly improve the complexity of TODD, an algorithm currently providing the best T -count reduction on various quantum circuits. We also propose some modifications to the algorithm which are leading to a significantly lower number of T gates. In addition, we propose another algorithm which has an even lower complexity and that achieves a better or equal T -count than the state of the art on most quantum circuits evaluated. We also prove that the number of T gates in the circuit obtained after executing our algorithms on a Hadamard-free circuit composed of n qubits is upper bounded by $n(n+1)/2 + 1$, which improves on the worst-case T -count of existing optimization algorithms. From this we derive an upper bound of $(n+1)(n+2h)/2 + 1$ for the number of T gates in a Clifford+ T circuit where h is the number of internal Hadamard gates in the circuit, i.e. the number of Hadamard gates lying between the first and the last T gate of the circuit.

1 Introduction

One of the main tasks of a quantum compiler is to minimize the resources needed to execute a given quantum algorithm. This step is of considerable importance in the compilation stack as it makes quantum computation more practical and efficient. To complete this task effectively and decide which optimization to perform, we must first identify the most expensive operations hindering our way towards fast and functional quantum computation. In this regard the T gate is often targeted as it cannot be trivially implemented, for instance via transversal operations, in a fault-tolerant way in most quantum error correcting codes as opposed to Clifford gates. It implies that implementing a T gate is generally much more costly than performing a Clifford operation [1–3]. Also, numerous quantum error correcting codes can be used to perform universal fault-tolerant quantum computation with the Clifford+ T gate set. In such a setup, the depth of a quantum circuit, and so the time required to execute it, is generally directly linked to the T -depth of the circuit [4]. For this reason, much work has been put into the minimization of the T -depth in Clifford+ T circuits [5–10]. But if the number of qubits at disposal is limited, then the depth of the circuit also depends on the number of T gates within it. In addition, the depth of a circuit dictates the minimum number of physical qubits needed to encode the logical qubits utilized to perform the fault-tolerant computation. The coherence time of the logical qubits must be greater than the time required to execute the whole circuit, and so the required amount of physical qubits per logical qubits increases as the depth of the circuit increases. We can discern a feedback loop here: if the depth of the circuit is diminished then less physical qubits are required to encode the logical qubits, which frees up qubits that can be used to further lower the depth of the circuit. The optimization of the T -count can intervene at multiple stages of this process. Firstly, a lower T -count can induce a lower T -depth and so a lower circuit depth [8]. Also, the number of qubits required to implement the T gates can depend on the T -count. This is for example the case when

the T gates are implemented via magic state distillation [11]. A lower number of T gates can thus lower the number of physical qubits required to implement the circuit.

The optimization of the T -count also have important applications outside of fault-tolerant quantum computation. Numerous quantum compilers designed for NISQ devices are incorporating a step consisting in reducing the number of T gates [12–14]. It has been demonstrated by these compilers that reducing the T -count can lead to shorter circuits and can help with the minimization of other gates such as the CNOT gate.

Besides compilation, T -count minimization also plays an important role in quantum circuits simulation. As stated by the Gottesman-Knill theorem [15], circuits composed of Clifford gates and Pauli measurements can be efficiently simulated by a classical computer. Extending the gate set by adding the T gate allows the simulation of universal quantum circuits at the cost of a significant increase in computational time as no algorithm is currently known to efficiently simulate these circuits. That is why many simulation techniques have a runtime that scales exponentially with respect to the number of T gates [16–20]. Minimizing the number of T gates is then essential to exploit the full potential of these simulators and to push back the frontier of non-simulable quantum circuits.

State of the art and contributions. Any unitary gate can be implemented by the Clifford+ T gate set up to an arbitrary precision. Therefore, a compilation problem of primordial importance is to find an approximation, over the Clifford+ T gate set and that uses a little amount of T gates, of a given unitary operator to an accuracy within $\epsilon > 0$. A fundamental solution to this problem, and which can be applied to any finite universal gate set, was given by the Solovay-Kitaev theorem [21, 22]. Other approaches designed for the Clifford+ T gate set were then developed to obtain better approximations [23–25]. Further improvements have then been made by introducing measurements [26] and by using a probabilistic mixture of unitaries [27, 28]. Recently, it has been shown that these two methods can be combined with a novel approach to achieve better results [29]. Another important synthesis problem concerns the set of unitary gates which can be exactly implemented over the Clifford+ T gate set. Given one of these unitary gates, the problem then consists in finding an exact Clifford+ T implementation of it using a minimal number of T gates. For this problem, an optimal and efficient algorithm is known for the case of single-qubit unitaries [30].

Once a Clifford+ T implementation of a unitary gate has been found, whether through approximate or exact synthesis, some quantum circuit optimization methods can then be applied to reduce the number of T gates in the circuit. The algorithms developed for this purpose and achieving the best T -count reductions are foremostly designed for the restricted class of $\{\text{CNOT}, S, T\}$ quantum circuits. The problem of T -count optimization for this class of circuits has been well defined by showing its equivalence with the problem of decoding Reed-Muller codes [31]. In particular, it was demonstrated that the codewords of the punctured Reed-Muller code of length $2^n - 1$ and order $n - 4$ are generating the complete set of identities that can be used to optimize the number of T gates in $\{\text{CNOT}, S, T\}$ circuits. Reducing the number of T gates can then be done by finding relevant identities in this large set. For example it has been shown that a particular subset of identities, called spider nest identities, can be efficiently exploited to reduce the number of T gates [32–35]. An effective way to find relevant identities that can be applied to reduce the number of T gates was given by the TODD algorithm [36]. However, an important drawback of the TODD algorithm is its complexity of $\mathcal{O}(n^3 m^5)$ where n is the number of qubits and m is the number of T gates in the initial circuit, which makes it impractical for circuits of large size. In Section 3, we show how the complexity of the TODD algorithm can be reduced to $\mathcal{O}(n^4 m^3)$. In addition, we propose some modifications to the TODD algorithm which are resulting in a significantly improved reduction in the number of T gates. In the same section we propose another algorithm which has an even lower complexity of $\mathcal{O}(n^2 m^3)$ and that achieves better results than the original TODD algorithm on most quantum circuits evaluated. We also prove that the algorithms presented in Section 3 are producing Hadamard-free circuits in which the T -count is upper bounded by $(n^2 + n)/2 + 1$ where n is the number of qubits. Benchmarks are provided in Section 4 to evaluate the performances, in terms of T -count and execution time, of our algorithms on a library of reversible logic circuits and on large-scale quantum circuits. In Section 5, we extend our results for minimizing the number of $R_Z(\pi/2^d)$ gates, where d is a non-negative integer. Finally, in Section 6, we demonstrate an

upper bound for the number of $R_Z(\pi/2^d)$ gates in a Clifford+ $\{R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit. For Clifford+ T circuits we obtain an upper bound of $(n+1)(n+2h)/2+1$ for the number of T gates, which can be satisfied in polynomial time and without any ancillary qubit, and where h is the number of internal Hadamard gates in the circuit.

2 Preliminaries

2.1 T -count optimization in Hadamard-free circuits

We define the set of Pauli operators $\mathcal{P}_n$ as the set composed of all the tensor products of n Pauli matrices, which are defined as follows:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

with a multiplicative factor of ± 1 . A Pauli rotation $R_P(\theta)$ is defined as follows:

$$R_P(\theta) = \exp(-i\theta P/2) = \cos(\theta/2)I - i\sin(\theta/2)P$$

for a Pauli operator $P \in \mathcal{P}_n$ and an angle $\theta \in \mathbb{R}$. For instance, the T gate is defined as a $\pi/4$ Pauli Z rotation:

$$T = R_Z(\pi/4).$$

The Clifford group, denoted $\mathcal{C}_n$, is generated by the set of $\pi/2$ Pauli rotations acting on n qubits

$$\{R_P(\pi/2) \mid P \in \mathcal{P}_n\}.$$

A Pauli operator $P \in \mathcal{P}_n$ conjugated by a Clifford gate $U \in \mathcal{C}_n$ is always equal to another Pauli operator $P' \in \mathcal{P}_n$, i.e. $U^\dagger P U = P'$. This fact also holds when a Pauli rotation is conjugated by $U \in \mathcal{C}_n$:

$$U^\dagger R_P(\theta) U = R_{U^\dagger P U}(\theta) = R_{P'}(\theta). \quad (1)$$

That is why the operation performed by a Clifford+ T circuit acting on n qubits, represented by a unitary gate U , can always be described by a sequence of $\pi/4$ Pauli rotations and a final Clifford operator $C \in \mathcal{C}_n$ [37]:

$$U = e^{i\phi} C \left(\prod_{i=1}^m R_{P_i}(\pi/4) \right) \quad (2)$$

where m is the number of T gates in the circuit and $P_i \in \mathcal{P}_n \setminus \{\pm I^{\otimes n}\}$.

Multiple algorithms to reduce the number of T gates are foremostly designed for circuits composed of $\{\text{CNOT}, S, T\}$ gates. In order to make use of these algorithms for Clifford+ T circuits, it is necessary to circumvent the Hadamard gates in the input circuit since these algorithms cannot be directly executed on them. This can be done using one of the two following methods. The first method consists in dividing the circuit into Hadamard-free subcircuits and Clifford subcircuits containing Hadamard gates. The number of T gates in the Hadamard-free subcircuits can then be optimized using these algorithms. There exists multiple strategies that can be employed to create such a partition of the circuit. It is generally preferred to regroup the T gates in the least number of Hadamard-free subcircuits as possible to take advantage of the fact that the number of T gates in an Hadamard-free circuit can be upper bounded by $\mathcal{O}(n^2)$, where n is the number of qubits. One approach to partition the circuit is to describe the operation performed by the circuit by a sequence of Pauli rotations and a final Clifford operator, as in Equation 2, and to then reorder the Pauli rotations in the sequence by forming groups of mutually commuting Pauli rotations. This can for example be done by using the procedure whose pseudo-code is given in Algorithm 1. This algorithm has a complexity of $\mathcal{O}(nm^2)$ since checking whether or not two Pauli rotations commute takes $\mathcal{O}(n)$ operations and such commutativity checks are done at most m times at each iteration of the loop. This way of grouping the Pauli rotations is not new, an equivalent algorithm (but

Algorithm 1: Grouping of Pauli rotations

Input: A sequence $R_{P_1}, \dots, R_{P_m}$ of Pauli rotations.

Output: Equivalent sequence as layers of mutually commuting Pauli rotations.

```
1  $L \leftarrow$  list of empty sets
2 for  $i \leftarrow 1$  to  $m$  do
3    $j \leftarrow \max(\{j \mid R_{P_i} \text{ anticommutes with one element in } L_j\} \cup \{0\})$ 
4    $L_{j+1} \leftarrow L_{j+1} \cup \{R_{P_i}\}$ 
5 end
6 return  $L$ 
```

which has a worst-case complexity of $\mathcal{O}(nm^3)$ was given in Reference [38]. By using the layers of Pauli rotations produced by Algorithm 1, Equation 2 can then be rewritten as follows:

$$U = e^{i\phi} C \left(\prod_{i=1}^{|L|} \prod_{R_P \in L_i} R_P(\pi/4) \right) \quad (3)$$

where $|L|$ denotes the number of layers in L .

A Pauli operator P and a Pauli rotation $R_P(\theta)$ are diagonal if and only if P is a tensor product of the matrices I and Z , up to a multiplicative factor of ± 1 . Such a Pauli rotation can be implemented without using any Hadamard gate. Because the Pauli rotations in each layer L_i of Equation 3 are mutually commuting, we can find a Clifford operator $C_i \in \mathcal{C}_n$ such that $C_i^\dagger R_P(\pi/4) C_i$ is diagonal for all $R_P \in L_i$. We say that C_i simultaneously diagonalize the Pauli rotations in L_i . A circuit implementing the Clifford operator C_i can be found with a complexity of $\mathcal{O}(n^2m)$, where n is the number of qubits and m is the number of Pauli rotations in L_i [39]. By performing a simultaneous diagonalization for each layer of Pauli rotations, Equation 3 can then be rewritten as follows:

$$\begin{aligned} U &= e^{i\phi} C \left(\prod_{i=1}^{|L|} C_i C_i^\dagger \left(\prod_{R_P \in L_i} R_P(\pi/4) \right) C_i C_i^\dagger \right) \\ &= e^{i\phi} C \left(\prod_{i=1}^{|L|} C_i \left(\prod_{R_P \in L_i} R_{C_i^\dagger P C_i}(\pi/4) \right) C_i^\dagger \right) \end{aligned} \quad (4)$$

where $C_i \in \mathcal{C}_n$ and $C_i^\dagger P C_i$ is diagonal. Because $C_i^\dagger P C_i$ is diagonal, the associated Pauli rotations $R_{C_i^\dagger P C_i}(\pi/4)$ can be implemented using exclusively CNOT, S and T gates. The set of Pauli rotations $R_{C_i^\dagger P C_i}(\pi/4)$ for a fixed i and where $R_P \in L_i$ can then be implemented into the same Hadamard-free subcircuit in which the number of T gates can be optimized.

The second method to circumvent the Hadamard gates in the circuit relies on a measurement-based gadget which can substitute a Hadamard gate [40]. This gadget, presented in Figure 1, involves an ancilla qubit, a CZ gate and a measurement. If all the Hadamard gates in the circuits are gadgetized, then the circuit is Hadamard-free and the number of T gates can be optimized using algorithms specifically designed for Hadamard-free circuits. Only internal Hadamard gates, which are the Hadamard gates comprised between the first and the last T gates of the circuit, are required to be gadgetized. Indeed, if only the internal Hadamard gates are gadgetized then the circuit can be partitioned into an initial and final Clifford circuit and a Hadamard-free circuit in between containing all the T gates. The main drawback of this method is that one additional qubit must be used for each internal Hadamard gate that is gadgetized. This motivates the minimization of internal Hadamard gates. To do so, we will use the algorithm presented in Reference [41] which performs the synthesis of the sequence of Pauli rotations of Equation 2 with a minimal number of internal Hadamard gates.

The classically controlled X gate of Figure 1, can be commuted through the subsequent R_Z gates of the circuit by relying on the following equality:

$$R_Z(\theta)X = XR_Z(-\theta) = XR_Z(-2\theta)R_Z(\theta) \quad (5)$$

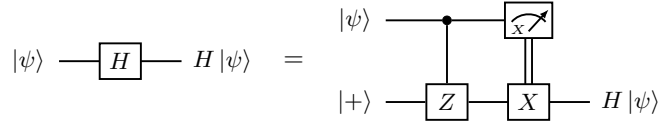


Figure 1: Circuit transformation corresponding to the gadgetization of a Hadamard gate.

In a Clifford+ T circuit, the angle θ is a multiple of $\pi/4$, and so -2θ is a multiple of $\pi/2$, which results in a rotation implementable using only Clifford gates. The classically controlled X gate is only commuted through the Pauli rotations succeeding the associated Hadamard gate, and not the Pauli rotations preceding it. Therefore, the optimized circuit in which all Hadamard gates have been gadgetized can be implemented with a measurement depth equal to $|L| - 1$, where $|L|$ corresponds to the number of layers as in Equation 4. An example of classically controlled Clifford gates resulting from the gadgetization of a Hadamard gate is provided in Figure 2.

2.2 Weighted polynomial and signature tensor

We now describe the established formalism for the optimization of the number of T gates in $\{\text{CNOT}, S, T\}$ circuits. Let C be a $\{\text{CNOT}, S, T\}$ circuit operating over n qubits. It has been demonstrated [5] that the action of C on a basis state has the form

$$|\mathbf{x}\rangle \mapsto U_f |g(\mathbf{x})\rangle \quad (6)$$

where $g : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ is a linear reversible Boolean function which can be implemented using only CNOT gates [42], and

$$U_f = \sum_{\mathbf{x} \in \mathbb{Z}_2^n} \omega^{f(\mathbf{x})} |\mathbf{x}\rangle \langle \mathbf{x}| \quad (7)$$

where $\omega = e^{i\pi/4}$ and $f : \mathbb{Z}_2^n \rightarrow \mathbb{Z}_8$ is a multilinear polynomial of degree 3 such that

$$f(\mathbf{x}) = \sum_{\alpha} l_{\alpha} x_{\alpha} - 2 \sum_{\alpha < \beta} q_{\alpha, \beta} x_{\alpha} x_{\beta} + 4 \sum_{\alpha < \beta < \gamma} c_{\alpha, \beta, \gamma} x_{\alpha} x_{\beta} x_{\gamma} \pmod{8} \quad (8)$$

where $l_{\alpha} \in \mathbb{Z}_8$, $q_{\alpha, \beta} \in \mathbb{Z}_4$ and $c_{\alpha, \beta, \gamma} \in \mathbb{Z}_2$. In the same way as in Reference [43], we will refer to the function f as a weighted polynomial due to the fact that each monomial of order m has a coefficient weighted by 2^{m-1} . It has been shown in Reference [43] that U_f belongs to the diagonal subgroup of the third level of the Clifford hierarchy [44]. We will use $\mathcal{D}_3$ to denote this group and $\mathcal{D}_3^C$ to refer to the unitaries of $\mathcal{D}_3$ which are implementable using only CCZ gates.

The parities of the coefficients for a weighted polynomial f can be described by the signature tensor $\mathcal{A}^{(U_f)} \in \mathbb{Z}_2^{(n, n, n)}$ [36], such that $\mathcal{A}^{(U_f)}$ is a symmetric tensor of order 3 satisfying

$$\begin{aligned} \mathcal{A}_{\alpha, \alpha, \alpha} &\equiv l_{\alpha} & (\text{mod } 2) \\ \mathcal{A}_{\sigma(\alpha, \beta, \beta)} &\equiv \mathcal{A}_{\sigma(\alpha, \alpha, \beta)} = q_{\alpha, \beta} & (\text{mod } 2) \\ \mathcal{A}_{\sigma(\alpha, \beta, \gamma)} &\equiv c_{\alpha, \beta, \gamma} & (\text{mod } 2) \end{aligned} \quad (9)$$

where α, β, γ are satisfying $0 \leq \alpha < \beta < \gamma < n$ and σ denotes all permutations of the indices. For convenience, we will drop the superscript (U_f) from $\mathcal{A}$ when it is clear from the context that $\mathcal{A}$ is associated with U_f . It has been proven in Reference [43] that U_{2f} can be implemented using only Clifford gates for any weighted polynomial f . It implies that two unitaries U_f and $U_{f'}$, where f and f' are weighted polynomials, are Clifford equivalent if the coefficients of f and f' all have the same parity, i.e. if $\mathcal{A}^{(U_f)}$ and $\mathcal{A}^{(U_{f'})}$ are equal.

2.3 Phase polynomial and parity table

An implementation of the U_f gate for a weighted polynomial f can be described by a phase polynomial via the circuit-polynomial correspondence [45, 46]. A phase polynomial p is a linear

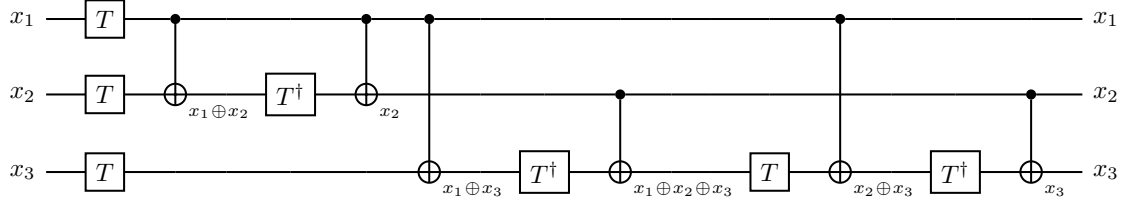


Figure 3: An implementation of the U_f gate with weighted polynomial $f(\mathbf{x}) = 4x_1x_2x_3$, which corresponds to the CCZ gate.

$q_{\alpha,\beta} = a \pmod{4}$ and $c_{\alpha,\beta,\gamma} = a \pmod{2}$. Equation 12 can then be used for each parity of a given phase polynomial to compute its associated weighted polynomial:

$$\begin{aligned}
 p(\mathbf{x}) &= \sum_{i=1}^m a_i (y_1^{(i)} x_1 \oplus \dots \oplus y_n^{(i)} x_n) \pmod{8} \\
 &= \sum_{i=1}^m a_i \left(\sum_{\alpha} y_{\alpha}^{(i)} x_{\alpha} - \sum_{\alpha < \beta} 2y_{\alpha}^{(i)} y_{\beta}^{(i)} x_{\alpha} x_{\beta} + \sum_{\alpha < \beta < \gamma} 4y_{\alpha}^{(i)} y_{\beta}^{(i)} y_{\gamma}^{(i)} x_{\alpha} x_{\beta} x_{\gamma} \right) \pmod{8} \quad (13) \\
 &= \sum_{\alpha} l_{\alpha} x_{\alpha} - \sum_{\alpha < \beta} 2q_{\alpha,\beta} x_{\alpha} x_{\beta} + \sum_{\alpha < \beta < \gamma} 4c_{\alpha,\beta,\gamma} x_{\alpha} x_{\beta} x_{\gamma} \pmod{8}
 \end{aligned}$$

where l_{α} , $q_{\alpha,\beta}$ and $c_{\alpha,\beta,\gamma}$ are satisfying

$$\begin{aligned}
 l_{\alpha} &= \sum_{i=1}^m a_i y_{\alpha}^{(i)} \pmod{8}, \\
 q_{\alpha,\beta} &= \sum_{i=1}^m a_i y_{\alpha}^{(i)} y_{\beta}^{(i)} \pmod{4}, \\
 c_{\alpha,\beta,\gamma} &= \sum_{i=1}^m a_i y_{\alpha}^{(i)} y_{\beta}^{(i)} y_{\gamma}^{(i)} \pmod{2}.
 \end{aligned} \quad (14)$$

Two weighted polynomials f and f' , with associated coefficients l_{α} , $q_{\alpha,\beta}$, $c_{\alpha,\beta,\gamma}$ and l'_{α} , $q'_{\alpha,\beta}$, $c'_{\alpha,\beta,\gamma}$ respectively, are equal if and only if $l_{\alpha} = l'_{\alpha}$, $q_{\alpha,\beta} = q'_{\alpha,\beta}$ and $c_{\alpha,\beta,\gamma} = c'_{\alpha,\beta,\gamma}$ for all α, β, γ . Indeed, if these equalities are not satisfied, then we can find some vector $\mathbf{x}$ such that $f(\mathbf{x}) \neq f'(\mathbf{x})$. Therefore, two phase polynomials p and p' are implementing the same operator if and only if their associated weighted polynomials f and f' are equal. The problem of T -count optimization then consists in finding a phase polynomial p which implements a given weighted polynomial f with a minimal number of T gates. Notice that the coefficients of the weighted polynomial represented in Equation 12 all have the same parity: $l_{\alpha} \equiv q_{\alpha,\beta} \equiv c_{\alpha,\beta,\gamma} \equiv a \pmod{2}$. And if a is even, then the associated rotation of angle $a\pi/4$ is a multiple of $\pi/2$ and can be implemented using only S gates. That is why a weighted polynomial f can be implemented using only $\{\text{CNOT}, S\}$ gates if and only if all its coefficients have an even parity, in such case we say that f is a Clifford weighted polynomial. Then, a weighted polynomial p with an associated parity table P is an implementation of a weighted polynomial f with an associated signature tensor $\mathcal{A}$ up to an operator implementable over the $\{\text{CNOT}, S\}$ gate set if and only if the equality

$$\mathcal{A}_{\alpha,\beta,\gamma} = |P_{\alpha} \wedge P_{\beta} \wedge P_{\gamma}| \pmod{2} \quad (15)$$

is satisfied for all α, β, γ satisfying $0 \leq \alpha \leq \beta \leq \gamma < n$, where n is the number of qubits. Throughout the paper, the notation $|v|$ will be used to refer to the Hamming weight of the vector v , and the symbol $\wedge$ will refer to the logical AND operation. Let f' be the weighted polynomial implemented by p , then f and f' have the same signature tensor, and f can be implemented by performing the synthesis of p and the synthesis of the Clifford operator associated with the Clifford weighted polynomial $f - f'$. The problem of finding a phase polynomial implementing a

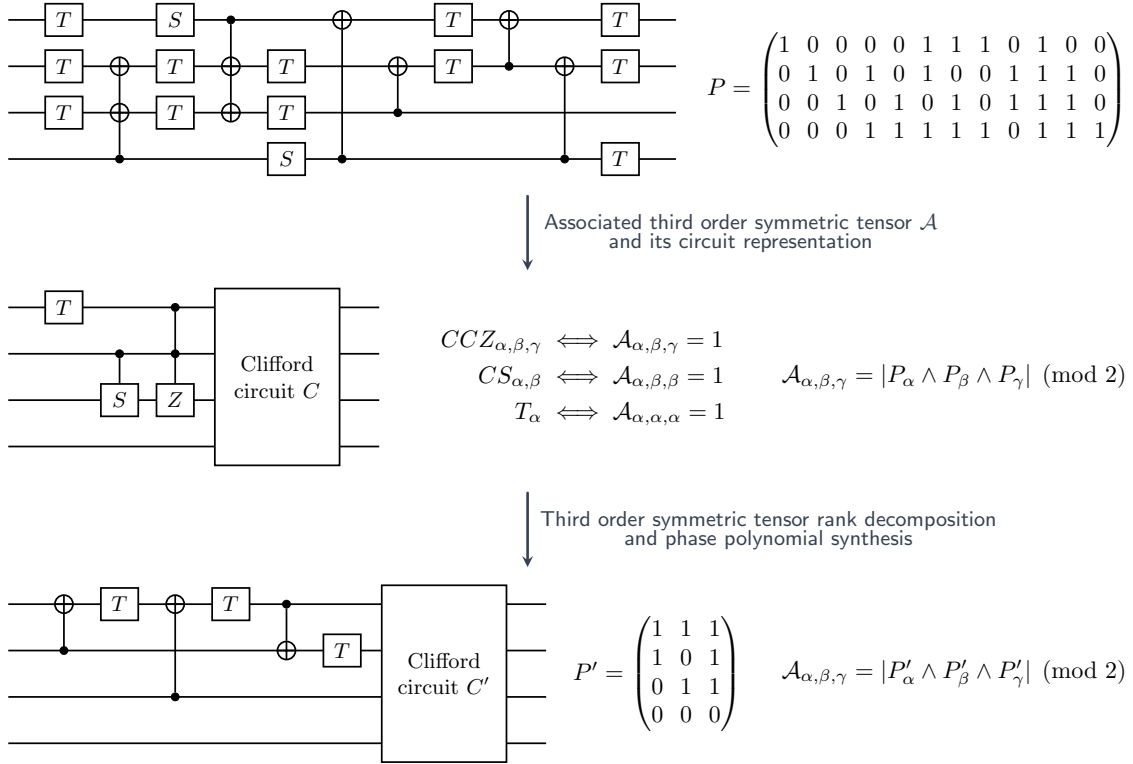


Figure 4: Overview of the process for the optimization of the number of T gates in Hadamard-free circuits. We first compute the parity table P associated with the phase polynomial implemented by the initial circuit. Then, we compute the weighted polynomial associated with P , which can be represented by the third order symmetric tensor $\mathcal{A}$ or by a $\{T, CS, CCZ\}$ circuit. We then solve the third order symmetric tensor rank decomposition problem to find another parity table P' associated with $\mathcal{A}$ but which contains a minimal number of columns. Finally, we perform the synthesis of the phase polynomial represented by P' along with the final Clifford operator to obtain the optimized circuit.

given weighted polynomial with a minimal number of T gates and up to an operator implementable over the $\{\text{CNOT}, S\}$ gate set can then be described by the following third order symmetric tensor rank decomposition (3-STR) problem [36].

Problem 1 (3-STR). Let $\mathcal{A} \in \mathbb{Z}_2^{(n,n,n)}$ be a symmetric tensor such that

$$\mathcal{A}_{\alpha,\beta,\gamma} = \mathcal{A}_{\alpha',\beta',\gamma'} \quad (16)$$

for all α, β, γ and α', β', γ' satisfying the set equality $\{\alpha, \beta, \gamma\} = \{\alpha', \beta', \gamma'\}$. Find a Boolean matrix P of size $n \times m$ such that

$$\mathcal{A}_{\alpha,\beta,\gamma} = |P_{\alpha} \wedge P_{\beta} \wedge P_{\gamma}| \pmod{2} \quad (17)$$

for all α, β, γ satisfying $0 \leq \alpha \leq \beta \leq \gamma < n$, with minimal m .

The T -count minimization problem in $\{\text{CNOT}, T, S\}$ circuits have first been shown to be equivalent to finding a minimum distance decoding in the order $n - 4$ punctured Reed-Muller code of length $2^n - 1$ [31], noted $\mathcal{RM}(n - 4, n)^*$, which is equivalent to the 3-STR problem [47]. The complexity class of the 3-STR problem is unknown, however, the related problem of finding the tensor rank of a tensor of order 3 is NP-complete [48]. And the more general problem of optimizing the number of T gates in a Clifford+ T circuit is NP-hard [49].

An illustrative overview of the process we described for the optimization of the number of T gates in Hadamard-free circuits is provided in Figure 4. In Section 3, we present two algorithms attempting to solve the 3-STR problem and we prove that the number of T gates in the circuit produced by these algorithms is upper bounded by $(n^2 + n)/2 + 1$. We evaluate the performances of our algorithms in the benchmarks of Section 4.1. And we generalize our results for the optimization of the number of $R_Z(\pi/2^d)$ gates, where d is a non-negative integer, in Section 5.

3 T -count reduction algorithms

In this section we tackle the 3-STR problem as defined in Section 2. First, we propose an algorithm for this problem in Subsection 3.1 and we prove that the number of T gates in the solution produced by this algorithm is upper bounded by $n(n+1)/2 + 1$. Then, in Subsection 3.2, we show how the complexity of the TODD algorithm of Reference [36] can be reduced and we propose some modifications to this algorithm to improve its performances.

3.1 Third order homogeneous polynomials elimination algorithm

The key mechanism used by our algorithm for reducing the T -count is based on the following theorem.

Theorem 1. *Let P be a parity table of size $n \times m$ and $P' = P \oplus \mathbf{z}\mathbf{y}^T$ where $\mathbf{z}$ and $\mathbf{y}$ are vectors of size n and m respectively such that*

$$|\mathbf{y}| \equiv 0 \pmod{2} \quad (18)$$

$$|P_\alpha \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (19)$$

$$|P_\alpha \wedge P_\beta \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (20)$$

for all $0 \leq \alpha < \beta < n$. Then we have

$$|P'_\alpha \wedge P'_\beta \wedge P'_\gamma| \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| \pmod{2} \quad (21)$$

for all $0 \leq \alpha \leq \beta \leq \gamma < n$.

Proof. For all α, β, γ satisfying $0 \leq \alpha \leq \beta \leq \gamma < n$, we have:

$$\begin{aligned} |P'_\alpha \wedge P'_\beta \wedge P'_\gamma| &= |(P_\alpha \oplus z_\alpha \mathbf{y}) \wedge (P_\beta \oplus z_\beta \mathbf{y}) \wedge (P_\gamma \oplus z_\gamma \mathbf{y})| \\ &= |[(P_\alpha \wedge P_\beta) \oplus z_\beta (P_\alpha \wedge \mathbf{y}) \oplus z_\alpha (P_\beta \wedge \mathbf{y}) \oplus z_\alpha z_\beta \mathbf{y}] \wedge (P_\gamma \oplus z_\gamma \mathbf{y})| \\ &\equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| + z_\gamma |P_\alpha \wedge P_\beta \wedge \mathbf{y}| + z_\beta |P_\alpha \wedge P_\gamma \wedge \mathbf{y}| + z_\alpha |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \\ &\quad + z_\beta z_\gamma |P_\alpha \wedge \mathbf{y}| + z_\alpha z_\gamma |P_\beta \wedge \mathbf{y}| + z_\alpha z_\beta |P_\gamma \wedge \mathbf{y}| + z_\alpha z_\beta z_\gamma |\mathbf{y}| \pmod{2} \\ &\equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| \pmod{2} \end{aligned} \quad (22)$$

□

If the conditions of Theorem 1 are satisfied, then the set of columns selected by $\mathbf{y}$ are forming a weighted polynomial that can be divided into two weighted polynomials f and f' , where f is a third order homogeneous weighted polynomial:

$$f(\mathbf{x}) = 4 \sum_{\alpha < \beta < \gamma}^n c_{\alpha, \beta, \gamma} x_\alpha x_\beta x_\gamma \pmod{8} \quad (23)$$

where $c_{\alpha, \beta, \gamma} = |P_\alpha \wedge P_\beta \wedge P_\gamma \wedge \mathbf{y}| \pmod{2}$, and where f' is a Clifford weighted polynomial:

$$f'(\mathbf{x}) = \sum_{\alpha}^n l_{\alpha} x_{\alpha} + 2 \sum_{\alpha < \beta}^n q_{\alpha, \beta} x_{\alpha} x_{\beta} \pmod{8} \quad (24)$$

where $l_{\alpha} \in \mathbb{Z}_8$, $q_{\alpha, \beta} \in \mathbb{Z}_4$ and $l_{\alpha} \equiv q_{\alpha, \beta} \equiv 0 \pmod{2}$. The unitary gate associated with the weighted polynomial f belongs to the $\mathcal{D}_3^C$ group: it is implementable using only CCZ gates [43]. It has already been shown that such weighted polynomials can be exploited to reduce the T -count, notably via the subadditivity theorem of Reference [43]. Let $U \in \mathcal{D}_3$ act on n qubits, in the following we define $\tau[U]$ as the optimal T -count to implement U without ancillary qubits:

$$\tau[U] = \min\{t \mid U = C_0 T_1 C_1 \dots T_t C_t, \{C_0, \dots, C_t\} \in \mathcal{C}_n^*\} \quad (25)$$

where $\mathcal{C}_n^*$ is the subgroup of Clifford operators which can be implemented with CNOT and S gates. The subadditivity theorem of Reference [43] states that if $\tau[U_1] \equiv 1 \pmod{2}$ and $\tau[U_2] > 0$, then $\tau[U_1 \otimes U_2] < \tau[U_1] + \tau[U_2]$ where $U_1 \in \mathcal{D}_3^C$ and $U_2 \in \mathcal{D}_3$. Based on Theorem 1, we can actually remove the condition $\tau[U_1] \equiv 1 \pmod{2}$, which gives the following theorem.

Theorem 2 (Subadditivity theorem). *Let $U_1 \in \mathcal{D}_3^C$, and $U_2 \in \mathcal{D}_3$. If $\tau[U_1], \tau[U_2] > 0$, then $\tau[U_1 \otimes U_2] < \tau[U_1] + \tau[U_2]$.*

Proof. Let $W = [P \ Q]$ be a parity table such that P and Q are the parity tables associated with the implementation of U_1 and U_2 and which have $\tau[U_1]$ and $\tau[U_2]$ columns respectively. Then, because $U_1 \in \mathcal{D}_3^C$, P satisfies the following equations:

$$|P_\alpha| \equiv 0 \pmod{2} \quad (26)$$

$$|P_\alpha \wedge P_\beta| \equiv 0 \pmod{2} \quad (27)$$

for all α, β . Let $\mathbf{z} = P_{:,i} \oplus Q_{:,j}$ for any i and j satisfying $0 \leq i < \tau[U_1]$, $0 \leq j < \tau[U_2]$, where $P_{:,i}$ denotes the i th column of the parity table P . And let P' be a parity table such that

$$P' = \begin{cases} P \oplus \mathbf{z}\mathbf{1}^T & \text{if } \tau[U_1] \equiv 0 \pmod{2}, \\ \begin{bmatrix} P \oplus \mathbf{z}\mathbf{1}^T & \mathbf{z} \end{bmatrix} & \text{otherwise.} \end{cases}$$

Then, as stated by Theorem 1, the parity table P' satisfies

$$|P'_\alpha \wedge P'_\beta \wedge P'_\gamma| \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| \pmod{2} \quad (28)$$

for all α, β, γ . And so the parity table $W' = [P' \ Q]$, which has at most one more column than W , also satisfies

$$|W'_\alpha \wedge W'_\beta \wedge W'_\gamma| \equiv |W_\alpha \wedge W_\beta \wedge W_\gamma| \pmod{2} \quad (29)$$

for all α, β, γ . However, we can notice that $P'_{:,i} = Q_{:,j}$. Therefore, by removing these two columns from W' Equation 29 still holds and W' has at least one less column than W . The parity table W' implements the unitary $U_1 \otimes U_2$ up to a Clifford operator and has at most $\tau[U_1] + \tau[U_2] - 1$ columns, thus we have $\tau[U_1 \otimes U_2] \leq \tau[U_1] + \tau[U_2] - 1 < \tau[U_1] + \tau[U_2]$. $\square$

Based on this subadditivity theorem and on Theorem 1, we can derive the following upper bound on the number of T gates in a $\{\text{CNOT}, S, T\}$ circuit.

Theorem 3. *The number of T gates in an n -qubits $\{\text{CNOT}, T, S\}$ circuit can be upper bounded by*

$$2\lfloor(n^2 + n)/4\rfloor + 1 \leq (n^2 + n)/2 + 1 \quad (30)$$

in polynomial time.

Proof. Let U be a unitary gate implementable by a $\{\text{CNOT}, S, T\}$ gate set, let P be a parity table of size $n \times m$ which implements U up to an operator implementable over the $\{\text{CNOT}, S\}$ gate set, and let L be a matrix with rows labelled by $(\alpha\beta)$ such that

$$L_{\alpha\beta} = P_\alpha \wedge P_\beta \quad (31)$$

for all α, β satisfying $0 \leq \alpha \leq \beta < n$. If P has strictly more than $(n^2 + n)/2 + 1$ columns then we can necessarily find a non-zero vector $\mathbf{y}$ satisfying $L\mathbf{y} = \mathbf{0}$ and $\mathbf{y} \neq \mathbf{1}$ because L has $(n^2 + n)/2$ rows. Note that such vector $\mathbf{y}$ necessarily satisfies Equations 19 and 20 of Theorem 1. We can then divide P into two non-empty parity tables $P^{(1)}$ and $P^{(2)}$ where the column $P_{:,i}$ belongs to $P^{(1)}$ if and only if $y_i = 1$ and to $P^{(2)}$ otherwise. The parity tables $P^{(1)}$ and $P^{(2)}$ are implementations of some unitary gates $U_1 \in \mathcal{D}_3^C$ and $U_2 \in \mathcal{D}_3$ respectively. The subadditivity theorem (Theorem 2) can then be exploited to reduce the number of columns of P . Let $\mathbf{z} = P_{:,i} \oplus P_{:,j}$ where i and j are satisfying $y_i = 1$ and $y_j = 0$, and let

$$P' = \begin{cases} P \oplus \mathbf{z}\mathbf{y}^T & \text{if } |\mathbf{y}| \equiv 0 \pmod{2}, \\ \begin{bmatrix} P \oplus \mathbf{z}\mathbf{y}^T & \mathbf{z} \end{bmatrix} & \text{otherwise.} \end{cases}$$

By Theorem 1, P' implements the same unitary gate as P up to an operator implementable over the $\{\text{CNOT}, S\}$ gate set. The parity table P' has at most one more column than P and we have $P'_{:,i} = P'_{:,j}$. Therefore, the columns i and j can be removed from P' , which entails that P'

has at least one less column than P . We showed that if the number of columns of P is strictly greater than $(n^2 + n)/2 + 1$, then the number of columns of P can be reduced by at least one in polynomial time. Moreover, if the number of columns of P is equal to $(n^2 + n)/2 + 1$ and is even, then we can necessarily find a non-zero vector $\mathbf{y}$ satisfying $L\mathbf{y} = \mathbf{0}$ because L has $(n^2 + n)/2$ rows. If there exist i such that $y_i = 0$ then we can reduce the number of columns of P as described above. Otherwise we must have $|\mathbf{y}| \equiv 0 \pmod{2}$, and so $\mathbf{y}$ satisfies the Equations of Theorem 1. Therefore, if $P' = P \oplus \mathbf{z}\mathbf{y}^T$ where $\mathbf{z}$ is equal to the i th column of P for any i , then P' implements the same unitary gate as P up to an operator implementable over the $\{\text{CNOT}, S\}$ gate set. The parity table P' has the same number of columns as P but its i th column is equal to the null vector and can therefore be removed, which leads to a parity table containing $(n^2 + n)/2$ columns. The polynomial-time procedure described above to reduce the number of columns of P can be repeated until P has a number of columns lower or equal to

$$2\lfloor(n^2 + n)/4\rfloor + 1 \leq (n^2 + n)/2 + 1 \quad (32)$$

□

Note that, for $n > 5$, this upper bound is better than the previously best known upper bound of $(n^2 + 3n - 14)/2$ [43]. The proof of Theorem 3 provides a straightforward algorithm for achieving this upper bound in polynomial time. We propose some improvements regarding this approach by providing an algorithm whose pseudo-code is given in Algorithm 2.

The algorithm starts by computing the set Z which contains all the vectors $\mathbf{z}$ which can potentially be used to transform P via Theorem 1 and reduce its number of columns:

$$Z = \{P_{:,i} \oplus P_{:,j} \mid 0 \leq i < j < m\} \cup \{P_{:,i} \mid 0 \leq i < m\}. \quad (33)$$

For each vector $\mathbf{z} \in Z$, the algorithm also computes the set $S^{(\mathbf{z})}$ of pairs of indices $\{i, j\}$ satisfying $P_{:,i} \oplus P_{:,j} = \mathbf{z}$ in the case where $i \neq j$ or satisfying $P_{:,i} = \mathbf{z}$ in the case where $i = j$. Let L be a matrix with rows labelled by $(\alpha\beta)$ such that

$$L_{\alpha\beta} = P_\alpha \wedge P_\beta \quad (34)$$

for all $\alpha < \beta$. In order to exploit the subadditivity theorem, the vector $\mathbf{y}$ must satisfy $L\mathbf{y} = \mathbf{0}$, $\mathbf{y} \neq \mathbf{0}$ and $\mathbf{y} \neq \mathbf{1}$. However, in the case where $\mathbf{y} = \mathbf{1}$ and $|\mathbf{y}| \equiv 0 \pmod{2}$ the number of columns of P can still be reduced. Indeed, in such case the vector $\mathbf{y}$ satisfies all the condition of Theorem 1, therefore the parity table P' defined as $P' = P \oplus P_{:,i}\mathbf{1}^T$ for any i is equivalent to P and its i th column is equal to the null vector and can therefore be removed, which reduces the number of columns by one. To summarize, the number of columns of P can be reduced if $\mathbf{y}$ satisfies $L\mathbf{y} = \mathbf{0}$ and $\mathbf{y} \neq \mathbf{0}$ and $\mathbf{y} \neq \mathbf{1}$ or $|\mathbf{y}| \equiv 0 \pmod{2}$. If no such $\mathbf{y}$ is found, Algorithm 2 will stop and return P . Otherwise, the algorithm will select a vector $\mathbf{z}$ such that the number of duplicated columns plus the number of all-zero columns in the parity table $P' = P \oplus \mathbf{z}\mathbf{y}^T$ is maximized. This number corresponds to the number of columns that can be removed from P if $\mathbf{z}$ is chosen (minus one in the case where $|\mathbf{y}| \equiv 1 \pmod{2}$), and is given by the following objective function:

$$-|\mathbf{y}| \pmod{2} + \sum_{\{i,j\} \in S^{(\mathbf{z})}} \begin{cases} 2(y_i \oplus y_j) & \text{if } i \neq j, \\ y_i + 2(y_i \oplus 1)(|\mathbf{y}| \pmod{2}) & \text{if } i = j. \end{cases} \quad (35)$$

The case where $i \neq j$ is straightforward, as the i th column will be equal to the j th column of P' if and only if $y_i \oplus y_j = 1$, which results in the removal of 2 columns. In the case where $i = j$, the equation

$$y_i + 2(y_i \oplus 1)(|\mathbf{y}| \pmod{2}) \quad (36)$$

is equal to 1 if $y_i = 1$ because the i th column of P' is the null vector, leading to the removal of 1 column. Otherwise, if $y_i = 0$ and $|\mathbf{y}|$ is odd, then Equation 36 is equal to 2 because the i th column of P' will be equal the $\mathbf{z}$ column vector that is added to P' in the case where $|\mathbf{y}|$ is odd, resulting in the removal of 2 columns. Once the vector $\mathbf{z}$ maximizing this objective function has been found, Algorithm 2 then computes the new parity table P' which contains fewer columns, and performs a recursive call.

Algorithm 2: Third order homogeneous polynomials elimination algorithm

Input: A parity table P of size $n \times m$.

Output: An equivalent parity table with an optimized number of columns.

```

1 procedure TOHPE( $P$ )
2    $Z \leftarrow \{P_{:,i} \oplus P_{:,j} \mid 0 \leq i < j < m\} \cup \{P_{:,i} \mid 0 \leq i < m\}$ 
3   forall  $z \in Z$  do
4      $S^{(z)} \leftarrow \{\{i, j\} \mid P_{:,i} \oplus P_{:,j} = z\} \cup \{\{i, i\} \mid P_{:,i} = z\}$ 
5   end
6    $L \leftarrow$  matrix whose rows are forming the set  $\{P_i \wedge P_j \mid 0 \leq i \leq j < n\}$ 
7   if  $\nexists \mathbf{y}$  such that  $L\mathbf{y} = \mathbf{0}$  and  $\mathbf{y} \neq \mathbf{0}$  and  $(\mathbf{y} \neq \mathbf{1} \text{ or } |\mathbf{y}| \equiv 0 \pmod{2})$  then
8     return  $P$ 
9   end
10   $\mathbf{y} \leftarrow$  any vector such that  $L\mathbf{y} = \mathbf{0}$  and  $\mathbf{y} \neq \mathbf{0}$  and  $(\mathbf{y} \neq \mathbf{1} \text{ or } |\mathbf{y}| \equiv 0 \pmod{2})$ 
11   $\mathbf{z} \leftarrow \operatorname{argmax}_{\mathbf{z} \in Z} \left\{ -|\mathbf{y}| \pmod{2} + \sum_{\{i,j\} \in S^{(\mathbf{z})}} 2(y_i \oplus y_j) + \delta_{ij} [y_i + 2(y_i \oplus 1)(|\mathbf{y}| \pmod{2})] \right\}$ 
12   $P' \leftarrow P \oplus \mathbf{z}\mathbf{y}^T$ 
13  if  $|\mathbf{y}| \equiv 1 \pmod{2}$  then
14     $P' \leftarrow [P' \quad \mathbf{z}]$ 
15  end
16   $P' \leftarrow P'$  with all its duplicated and all-zero columns removed
17  return TOHPE( $P'$ )

```

A variant of this algorithm could consist in finding both vectors $\mathbf{z}$ and $\mathbf{y}$ that are maximizing the objective function. However, our experiments on this approach showed that it significantly increases the complexity of the algorithm for a rather marginal gain in the number of T gates. The performances of Algorithm 2 are evaluated in Section 4.1.

Complexity analysis. The set Z can be computed with $\mathcal{O}(nm^2)$ operations, and all the sets $S^{(\mathbf{z})}$ can be created with the same time complexity. The matrix L has $\mathcal{O}(n^2)$ rows and a number of columns that is at most equal to m . Therefore performing a Gaussian elimination to compute a generating set of the right nullspace of L implies a complexity of $\mathcal{O}(n^2m^2)$. The vector $\mathbf{z}$ satisfying the argmax function can be computed in $\mathcal{O}(m^2)$ operations because the union of all the $S^{(\mathbf{z})}$ sets contains $\mathcal{O}(m^2)$ elements. Updating the parity table P induces $\mathcal{O}(nm)$ operations. And the algorithm is performing no more than m recursive calls. Thus, the overall complexity of Algorithm 2 is $\mathcal{O}(n^2m^3)$.

3.2 Improving the TODD algorithm

In this section we show how the TODD algorithm proposed in Reference [36] can be improved. We first describe the algorithm and demonstrate how its complexity can be reduced. We then propose a modified version of this algorithm to improve its performances.

The key mechanism of the TODD algorithm rests on the following theorem, which was first proven in Reference [36]. We provide its proof for completeness in Appendix A.

Theorem 4. Let P be a parity table of size $n \times m$ and $P' = P \oplus \mathbf{z}\mathbf{y}^T$ where $\mathbf{z}$ and $\mathbf{y}$ are vectors of size n and m respectively such that

$$|\mathbf{y}| \equiv 0 \pmod{2} \quad (37)$$

$$|P_\alpha \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (38)$$

$$|[z_\alpha(P_\beta \wedge P_\gamma) \oplus z_\beta(P_\alpha \wedge P_\gamma) \oplus z_\gamma(P_\alpha \wedge P_\beta)] \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (39)$$

for all $0 \leq \alpha < \beta < \gamma < n$. Then we have

$$|P'_\alpha \wedge P'_\beta \wedge P'_\gamma| \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| \pmod{2} \quad (40)$$

for all $0 \leq \alpha \leq \beta \leq \gamma < n$.

The TODD algorithm exploits this theorem for optimizing the number of T gates as follows. Let P be a parity table of size $n \times m$, let $\mathbf{z} = P_{:,i} \oplus P_{:,j}$ where $i \neq j$ and let χ be a matrix that contains a row equal to P_α for every α and a row equal to

$$z_\alpha(P_\beta \wedge P_\gamma) \oplus z_\beta(P_\alpha \wedge P_\gamma) \oplus z_\gamma(P_\alpha \wedge P_\beta)$$

for every triple α, β, γ satisfying $0 \leq \alpha < \beta < \gamma < n$. A vector $\mathbf{y}$ satisfying the conditions 38 and 39 of Theorem 4 can be found by computed the nullspace of χ , i.e. if $\mathbf{y}$ satisfies $\chi\mathbf{y} = \mathbf{0}$ then $\mathbf{y}$ also satisfies the conditions 38 and 39 of Theorem 4. If $\mathbf{y}$ satisfies $\chi\mathbf{y} = \mathbf{0}$ and $y_i \oplus y_j = 1$, then the number of columns of P can be reduced. Let

$$P' = \begin{cases} P \oplus \mathbf{z}\mathbf{y}^T & \text{if } |\mathbf{y}| \equiv 0 \pmod{2}, \\ \begin{bmatrix} P \oplus \mathbf{z}\mathbf{y}^T & \mathbf{z} \end{bmatrix} & \text{otherwise.} \end{cases}$$

Then, as stated by Theorem 4, P and P' are equivalent. Moreover, we have $P'_{:,i} = P'_{:,j}$ because $\mathbf{z} = P_{:,i} \oplus P_{:,j}$, therefore these two columns can be removed from P' , which results in P' having at least one less column than P .

If there doesn't exist any $\mathbf{y}$ satisfying $\chi\mathbf{y} = \mathbf{0}$ and $y_i \oplus y_j = 1$, then a different $\mathbf{z}$ vector can be considered. In the worst case the TODD algorithm performs this search for all the vectors $\mathbf{z}$ in the set $\{P_{:,i} \oplus P_{:,j} \mid 0 \leq i < j < n\}$, which contains $\mathcal{O}(m^2)$ elements. The matrix χ is composed of m columns and $\mathcal{O}(n^3)$ rows. Performing a Gaussian elimination in order to find some $\mathbf{y}$ satisfying $\chi\mathbf{y} = \mathbf{0}$ induces a complexity of $\mathcal{O}(n^3m^2)$. This procedure must be performed for the $\mathcal{O}(m^2)$ $\mathbf{z}$ vectors evaluated. And everytime a simplification is found the algorithm restart the same procedure on the new parity table $P' = P \oplus \mathbf{z}\mathbf{y}^T$, this happens at most m times. Thus, the overall complexity of the TODD algorithm is $\mathcal{O}(n^3m^5)$. The important worst-case complexity of the TODD algorithm makes it rapidly impractical for instances of increasing size. We will demonstrate how this complexity can be reduced to $\mathcal{O}(n^4m^3)$ by providing an alternative and more efficient way of checking whether or not there exists a vector $\mathbf{y}$ satisfying the conditions of Theorem 4, and therefore avoiding the expensive Gaussian elimination required to solve the system $\chi\mathbf{y} = \mathbf{0}$.

First, we can notice that Equation 39 of Theorem 4 can be greatly simplified in some cases, depending on the value of $\mathbf{z}$. For instance, if $z_\alpha = 1$ for some α and $z_\beta = 0$ for all $\beta \neq \alpha$, then Equation 39 is equal to

$$|z_\alpha(P_\beta \wedge P_\gamma) \wedge \mathbf{y}| \equiv 0 \pmod{2}. \quad (41)$$

In this specific case, a vector $\mathbf{y}$ satisfies Equations 38 and 39 of Theorem 4 if it satisfies $\chi\mathbf{y} = \mathbf{0}$ where χ contains a row equal to Equation 38 for all α and a row equal to Equation 41 for all α, β, γ satisfying $\alpha \neq \beta \neq \gamma$, $\beta < \gamma$ and $z_\alpha = 1$. In total, χ contains only $\mathcal{O}(n^2)$ rows instead of $\mathcal{O}(n^3)$ rows in the original TODD algorithm, and so the nullspace of χ can be computed in $\mathcal{O}(n^2m^2)$ operations. In the more general case where $|\mathbf{z}| > 1$, we can perform a change of basis to always end up in this specific case where $|\mathbf{z}| = 1$. A change of basis induces a complexity of $\mathcal{O}(nm)$ because $\mathcal{O}(n)$ additions must be performed between the rows of P . This must be done for all the $\mathcal{O}(m^2)$ $\mathbf{z}$ vectors evaluated, and the whole procedure is repeated $\mathcal{O}(m)$ times. Thus, this version of the TODD algorithm has a complexity of $\mathcal{O}(n^2m^5)$, which is better than the $\mathcal{O}(n^3m^5)$ complexity of the original TODD algorithm.

Instead of performing a change of basis, we can rely on the following theorem to efficiently find the vectors $\mathbf{z}$ and $\mathbf{y}$ satisfying the Equations 38 and 39 of Theorem 4.

Theorem 5. *Let P be a parity table of size $n \times m$, and let $\mathbf{z}$ and $\mathbf{y}$ be vectors of size n and m respectively and such that $|\mathbf{y}| \equiv 0 \pmod{2}$. Let L and X be matrices with rows labelled by $(\alpha\beta)$ such that*

$$L_{\alpha\beta} = P_\alpha \wedge P_\beta \quad (42)$$

and

$$X_{\alpha\beta,\gamma} = z_\alpha \delta_{\beta\gamma} \oplus z_\beta \delta_{\alpha\gamma} \quad (43)$$

for all α, β, γ satisfying $0 \leq \alpha \leq \beta < n$ and $0 \leq \gamma < n$, and where δ is the Kronecker delta defined as follows:

$$\delta_{\alpha\beta} = \begin{cases} 0 & \text{if } \alpha \neq \beta, \\ 1 & \text{if } \alpha = \beta. \end{cases} \quad (44)$$

There exists $\mathbf{y}'$ such that $L\mathbf{y} \oplus X\mathbf{y}' = \mathbf{0}$ if and only if the following conditions are satisfied:

$$|P_\alpha \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (45)$$

$$|[z_\alpha(P_\beta \wedge P_\gamma) \oplus z_\beta(P_\alpha \wedge P_\gamma) \oplus z_\gamma(P_\alpha \wedge P_\beta)] \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (46)$$

for all $0 \leq \alpha \leq \beta \leq \gamma < n$.

Proof. In the following we will use the labels $(\beta\alpha)$ and $(\alpha\beta)$ to refer to the same unique row, for instance $L_{\beta\alpha} = L_{\alpha\beta}$. For all α we have $X_{\alpha\alpha} = \mathbf{0}$, which implies that

$$L_{\alpha\alpha}\mathbf{y} \oplus X_{\alpha\alpha}\mathbf{y}' = L_{\alpha\alpha}\mathbf{y} \equiv |P_\alpha \wedge \mathbf{y}| \pmod{2} \quad (47)$$

for all α . Therefore we have

$$L_{\alpha\alpha}\mathbf{y} \oplus X_{\alpha\alpha}\mathbf{y}' = 0 \iff |P_\alpha \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (48)$$

for all α . It remains to show that there exists $\mathbf{y}'$ satisfying $L_{\alpha\beta}\mathbf{y} \oplus X_{\alpha\beta}\mathbf{y}' = 0$ for all $\alpha \neq \beta$ if and only if

$$|[z_\alpha(P_\beta \wedge P_\gamma) \oplus z_\beta(P_\alpha \wedge P_\gamma) \oplus z_\gamma(P_\alpha \wedge P_\beta)] \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (49)$$

holds for all $0 \leq \alpha < \beta < \gamma < n$.

In the case where $|\mathbf{z}| = 1$, let α be such that $z_\alpha = 1$. Then, for such vector $\mathbf{z}$, Equation 49 is equivalent to

$$|P_\beta \wedge P_\gamma \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (50)$$

for all β, γ satisfying $0 \leq \beta < \gamma < n$ and $\alpha \neq \beta \neq \gamma$. We also have $X_{\beta\gamma}\mathbf{y}' = 0$ for any $\mathbf{y}'$ and for all β, γ satisfying $\alpha \neq \beta \neq \gamma$ because $X_{\beta\gamma} = \mathbf{0}$ for all β, γ satisfying $z_\beta = 0, z_\gamma = 0$. We then have

$$L_{\beta\gamma}\mathbf{y} \oplus X_{\beta\gamma}\mathbf{y}' = L_{\beta\gamma}\mathbf{y} \equiv |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \pmod{2} \quad (51)$$

for all β, γ satisfying $0 \leq \beta < \gamma < n$ and $\alpha \neq \beta \neq \gamma$. And therefore

$$L_{\beta\gamma}\mathbf{y} \oplus X_{\beta\gamma}\mathbf{y}' = \mathbf{0} \iff |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (52)$$

for all β, γ satisfying $0 \leq \beta < \gamma < n$ and $\alpha \neq \beta \neq \gamma$. Moreover, the definition of the matrix X implies the following equality for all β, γ :

$$X_{\beta\gamma}\mathbf{y}' = \mathbf{y}'_\beta z_\gamma \oplus \mathbf{y}'_\gamma z_\beta \quad (53)$$

which entails

$$L_{\alpha\beta}\mathbf{y} \oplus X_{\alpha\beta}\mathbf{y}' = L_{\alpha\beta}\mathbf{y} \oplus \mathbf{y}'_\beta \quad (54)$$

for all β satisfying $\alpha \neq \beta$. Therefore, if $\mathbf{y}'$ satisfies $\mathbf{y}'_\beta = L_{\alpha\beta}\mathbf{y}$ for all β such that $\alpha \neq \beta$, then $L\mathbf{y} \oplus X\mathbf{y}' = \mathbf{0}$ and so Theorem 5 is true in the case where $|\mathbf{z}| = 1$.

Let B be a full rank binary matrix of size $n \times n$ which represents a change of basis, and let $\tilde{L}$ and $\tilde{X}$ be matrices constructed in the same way as L and X but with respect to BP and $B\mathbf{z}$. The proof of Theorem 5 can be completed by proving the two following propositions. First, if $\mathbf{y}$ satisfies Equations 45 and 46 for P and $\mathbf{z}$ then $\mathbf{y}$ also satisfies Equations 45 and 46 for BP and $B\mathbf{z}$. Second, if there exists $\mathbf{y}'$ such that $L\mathbf{y} \oplus X\mathbf{y}' = \mathbf{0}$ then there exists $\tilde{\mathbf{y}}'$ such that $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' = \mathbf{0}$. Indeed, if these two propositions are true, then if $\mathbf{y}$ satisfies Equations 45 and 46 for P and $\mathbf{z}$ it also satisfies Equations 45 and 46 for BP and $B\mathbf{z}$, where B is chosen such that $|B\mathbf{z}| = 1$. Then, as previously demonstrated, there exists a vector $\tilde{\mathbf{y}}'$ such that $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' = \mathbf{0}$, which would imply that there exists a vector $\mathbf{y}'$ such that $L\mathbf{y} \oplus X\mathbf{y}' = \mathbf{0}$.

Let $\tilde{\mathbf{z}}$ and $\tilde{P}$ be such that $\tilde{z}_\alpha = z_\alpha \oplus z_\beta$ and $\tilde{P}_\alpha = P_\alpha \oplus P_\beta$ for some fixed α, β satisfying $\alpha \neq \beta$ and $\tilde{z}_\gamma = z_\gamma, \tilde{P}_\gamma = P_\gamma$ for all γ such that $\gamma \neq \alpha$. If $\mathbf{y}$ satisfies Equations 45 and 46, then we have

$$\begin{aligned} |\tilde{P}_\alpha \wedge \mathbf{y}| &= |(P_\alpha \oplus P_\beta) \wedge \mathbf{y}| \\ &\equiv |P_\alpha \wedge \mathbf{y}| + |P_\beta \wedge \mathbf{y}| \pmod{2} \\ &\equiv 0 \pmod{2} \end{aligned} \quad (55)$$

and

$$\begin{aligned}
& |[\tilde{z}_\alpha(\tilde{P}_\beta \wedge \tilde{P}_\gamma) \oplus \tilde{z}_\beta(\tilde{P}_\alpha \wedge \tilde{P}_\gamma) \oplus \tilde{z}_\gamma(\tilde{P}_\alpha \wedge \tilde{P}_\beta)] \wedge \mathbf{y}| \\
&= |[(z_\alpha \oplus z_\beta)(P_\beta \wedge P_\gamma) \oplus z_\beta((P_\alpha \oplus P_\beta) \wedge P_\gamma) \oplus z_\gamma((P_\alpha \oplus P_\beta) \wedge P_\beta)] \wedge \mathbf{y}| \\
&= |[z_\alpha(P_\beta \wedge P_\gamma) \oplus z_\beta(P_\beta \wedge P_\gamma) \oplus z_\beta(P_\alpha \wedge P_\gamma) \oplus z_\beta(P_\beta \wedge P_\gamma) \oplus z_\gamma(P_\alpha \wedge P_\beta) \oplus z_\gamma P_\beta] \wedge \mathbf{y}| \quad (56) \\
&\equiv |[z_\alpha(P_\beta \wedge P_\gamma) \oplus z_\beta(P_\alpha \wedge P_\gamma) \oplus z_\gamma(P_\alpha \wedge P_\beta)] \wedge \mathbf{y}| + z_\gamma |P_\beta \wedge \mathbf{y}| \pmod{2} \\
&\equiv 0 \pmod{2}
\end{aligned}$$

and so $\mathbf{y}$ also satisfies Equations 45 and 46 for $\tilde{P}, \tilde{z}$.

Let $\tilde{L}$ and $\tilde{X}$ be matrices constructed in the same way as L and X but with respect to $\tilde{P}$ and $\tilde{z}$. Let's assume that $L\mathbf{y} \oplus X\mathbf{y}' = \mathbf{0}$, we will show that it implies $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' = \mathbf{0}$, where $\tilde{\mathbf{y}}'$ satisfies $\tilde{y}'_\alpha = y'_\alpha \oplus y'_\beta$ and $\tilde{y}'_\gamma = y'_\gamma$ for all γ such that $\gamma \neq \alpha$. By using Equation 53 we can deduce that

$$\tilde{X}_{\gamma\gamma'}\tilde{\mathbf{y}}' = \tilde{y}'_\gamma \tilde{z}_{\gamma'} \oplus \tilde{y}'_{\gamma'} \tilde{z}_\gamma = y'_\gamma z_{\gamma'} \oplus y'_{\gamma'} z_\gamma = X_{\gamma\gamma'}\mathbf{y}' \quad (57)$$

for all γ, γ' satisfying $\gamma \neq \alpha$ and $\gamma' \neq \alpha$. This entails

$$\tilde{L}_{\gamma\gamma'}\mathbf{y} \oplus \tilde{X}_{\gamma\gamma'}\tilde{\mathbf{y}}' = L_{\gamma\gamma'}\mathbf{y} \oplus X_{\gamma\gamma'}\mathbf{y}' = 0 \quad (58)$$

for all γ, γ' satisfying $\gamma \neq \alpha$ and $\gamma' \neq \alpha$. Furthermore, for all γ such that $\gamma \neq \alpha$, we have

$$\begin{aligned}
\tilde{L}_{\alpha\gamma}\mathbf{y} &\equiv |\tilde{P}_\alpha \wedge \tilde{P}_\gamma \wedge \mathbf{y}| \pmod{2} \\
&\equiv |(P_\alpha \oplus P_\beta) \wedge P_\gamma \wedge \mathbf{y}| \pmod{2} \\
&\equiv |P_\alpha \wedge P_\gamma \wedge \mathbf{y}| + |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \pmod{2} \\
&\equiv L_{\alpha\gamma}\mathbf{y} + L_{\beta\gamma}\mathbf{y} \pmod{2} \\
&\equiv X_{\alpha\gamma}\mathbf{y}' + X_{\beta\gamma}\mathbf{y}' \pmod{2}
\end{aligned} \quad (59)$$

which entails

$$\begin{aligned}
\tilde{L}_{\alpha\gamma}\mathbf{y} \oplus \tilde{X}_{\alpha\gamma}\tilde{\mathbf{y}}' &= X_{\alpha\gamma}\mathbf{y}' \oplus X_{\beta\gamma}\mathbf{y}' \oplus \tilde{X}_{\alpha\gamma}\tilde{\mathbf{y}}' \\
&= y'_\alpha z_\gamma \oplus y'_\gamma z_\alpha \oplus y'_\beta z_\gamma \oplus y'_\gamma z_\beta \oplus \tilde{y}'_\alpha \tilde{z}_\gamma \oplus \tilde{y}'_\gamma \tilde{z}_\alpha \\
&= y'_\alpha z_\gamma \oplus y'_\gamma z_\alpha \oplus y'_\beta z_\gamma \oplus y'_\gamma z_\beta \oplus (y'_\alpha \oplus y'_\beta) z_\gamma \oplus y'_\gamma (z_\alpha \oplus z_\beta) \\
&= 0
\end{aligned} \quad (60)$$

by relying on Equation 53. Finally, we have

$$\begin{aligned}
\tilde{L}_{\alpha\alpha}\mathbf{y} \oplus \tilde{X}_{\alpha\alpha}\tilde{\mathbf{y}}' &= \tilde{L}_{\alpha\alpha}\mathbf{y} \oplus \tilde{y}'_\alpha \tilde{z}_\alpha \oplus \tilde{y}'_\alpha \tilde{z}_\alpha \\
&\equiv |\tilde{P}_\alpha \wedge \mathbf{y}| \pmod{2} \\
&\equiv |(P_\alpha \oplus P_\beta) \wedge \mathbf{y}| \pmod{2} \\
&\equiv |P_\alpha \wedge \mathbf{y}| + |P_\beta \wedge \mathbf{y}| \pmod{2} \\
&\equiv 0 \pmod{2}.
\end{aligned} \quad (61)$$

Thus,

$$L\mathbf{y} \oplus X\mathbf{y}' = \mathbf{0} \implies \tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' = \mathbf{0} \quad (62)$$

which concludes the proof of Theorem 5. $\square$

Theorem 5 can be exploited to design an algorithm equivalent to the TODD algorithm, but which has a lower complexity. Before introducing this algorithm, we demonstrate how the key mechanism utilized by the TODD algorithm to reduce the T -count can be improved. The conditions given by Equations 38 and 39 of Theorem 4 are sufficient for Equation 40 to hold but they are not necessary. The following theorem gives necessary and sufficient conditions for Equation 40 to be satisfied.

Theorem 6. Let P be a parity table of size $n \times m$ and $P' = P \oplus \mathbf{z}\mathbf{y}^T$ where $\mathbf{z}$ and $\mathbf{y}$ are vectors of size n and m respectively and such that $|\mathbf{y}| \equiv 0 \pmod{2}$. And let L and X be matrices and $\mathbf{v}$ be a vector, all with rows labelled by $(\alpha\beta)$ such that

$$L_{\alpha\beta} = P_\alpha \wedge P_\beta \quad (63)$$

$$X_{\alpha\beta,\gamma} = z_\alpha \delta_{\beta\gamma} \oplus z_\beta \delta_{\alpha\gamma} \quad (64)$$

$$v_{\alpha\beta} = z_\alpha \wedge z_\beta \quad (65)$$

for all α, β, γ satisfying $0 \leq \alpha \leq \beta < n$ and $0 \leq \gamma < n$, and where δ is the Kronecker delta defined as follows:

$$\delta_{\alpha\beta} = \begin{cases} 0 & \text{if } \alpha \neq \beta, \\ 1 & \text{if } \alpha = \beta. \end{cases} \quad (66)$$

Then, the equality

$$|P'_\alpha \wedge P'_\beta \wedge P'_\gamma| \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| \pmod{2} \quad (67)$$

holds for all $0 \leq \alpha \leq \beta \leq \gamma < n$ if and only if there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$.

Proof. In the same way as in the proof of Theorem 5, we will use the labels $(\beta\alpha)$ and $(\alpha\beta)$ to refer to the same unique row, for instance $L_{\beta\alpha} = L_{\alpha\beta}$. Analogously to Equation 22, we have

$$\begin{aligned} |P'_\alpha \wedge P'_\beta \wedge P'_\gamma| &\equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| + z_\gamma |P_\alpha \wedge P_\beta \wedge \mathbf{y}| + z_\beta |P_\alpha \wedge P_\gamma \wedge \mathbf{y}| + z_\alpha |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \\ &\quad + z_\beta z_\gamma |P_\alpha \wedge \mathbf{y}| + z_\alpha z_\gamma |P_\beta \wedge \mathbf{y}| + z_\alpha z_\beta |P_\gamma \wedge \mathbf{y}| + z_\alpha z_\beta z_\gamma |\mathbf{y}| \pmod{2} \end{aligned} \quad (68)$$

for all α, β, γ satisfying $0 \leq \alpha \leq \beta \leq \gamma < n$. Because $|\mathbf{y}| \equiv 0 \pmod{2}$, we can deduce from Equation 68 that Equations 67 holds if and only if the following equation is satisfied

$$[z_\gamma (P_\alpha \wedge P_\beta) \oplus z_\beta (P_\alpha \wedge P_\gamma) \oplus z_\alpha (P_\beta \wedge P_\gamma) \oplus z_\beta z_\gamma P_\alpha \oplus z_\alpha z_\gamma P_\beta \oplus z_\alpha z_\beta P_\gamma] \wedge \mathbf{y} \equiv \mathbf{0} \pmod{2} \quad (69)$$

for all α, β, γ satisfying $0 \leq \alpha \leq \beta \leq \gamma < n$.

Let $\mathbf{z}$ be such that $|\mathbf{z}| = 1$, and let α be such that $z_\alpha = 1$. Then, for such vector $\mathbf{z}$, Equation 69 can be rewritten as

$$|P_\beta \wedge P_\gamma \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (70)$$

where β, γ are satisfying $\beta \neq \alpha, \gamma \neq \alpha$, and as

$$|P_\gamma \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (71)$$

where β, γ are satisfying $\beta = \alpha$ and $\gamma \neq \alpha$. Note that Equation 69 is necessarily satisfied in the case where $\alpha = \beta = \gamma$, and that if Equation 70 is satisfied then Equation 71 is also satisfied. Therefore, proving Theorem 6 in the case where $|\mathbf{z}| = 1$ can then be done by showing that there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ if and only if Equation 70 is satisfied. By definition we have $X_{\beta\gamma} = \mathbf{0}$ and $v_{\beta\gamma} = 0$ for all β, γ satisfying $\beta \neq \alpha$ and $\gamma \neq \alpha$. Hence,

$$L_{\beta\gamma}\mathbf{y} \oplus X_{\beta\gamma}\mathbf{y}' \oplus bv_{\beta\gamma} = L_{\beta\gamma}\mathbf{y} = |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \pmod{2} \quad (72)$$

for all β, γ satisfying $\beta \neq \alpha$ and $\gamma \neq \alpha$, and so if $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ then $L_{\beta\gamma}\mathbf{y} = 0$ which imply that Equation 70 is satisfied. Conversely, if Equation 70 is satisfied then we must have

$$L_{\beta\gamma}\mathbf{y} \oplus X_{\beta\gamma}\mathbf{y}' \oplus bv_{\beta\gamma} = L_{\beta\gamma}\mathbf{y} = 0 \quad (73)$$

for all β, γ satisfying $\beta \neq \alpha$ and $\gamma \neq \alpha$ and for any vector $\mathbf{y}'$ and Boolean b . Moreover we have

$$L_{\alpha\alpha}\mathbf{y} \oplus X_{\alpha\alpha}\mathbf{y}' \oplus bv_{\alpha\alpha} = L_{\alpha\alpha}\mathbf{y} \oplus b \quad (74)$$

because $X_{\alpha\alpha} = \mathbf{0}$ and $v_{\alpha\alpha} = 1$. And

$$L_{\alpha\beta}\mathbf{y} \oplus X_{\alpha\beta}\mathbf{y}' \oplus bv_{\alpha\beta} = L_{\alpha\beta}\mathbf{y} \oplus y'_\beta \quad (75)$$

where $\beta \neq \alpha$ by using Equation 53 and because $v_{\alpha\beta} = 0$. Let $\mathbf{y}'$ be such that $y'_\beta = L_{\alpha\beta}\mathbf{y}$ for all $\beta \neq \alpha$ and b be such that $b = L_{\alpha\alpha}\mathbf{y}$, then we have $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$. Thus, we proved that

$$L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0} \iff |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \equiv 0 \pmod{2} \quad (76)$$

for all β, γ satisfying $\beta \neq \alpha$ and $\gamma \neq \alpha$, and so Theorem 6 is true in the case where $|\mathbf{z}| = 1$.

Let B be a full rank binary matrix of size $n \times n$ which represents a change of basis, and let $\tilde{L}$, $\tilde{X}$, $\tilde{\mathbf{v}}$ be constructed in the same way as L , X and $\mathbf{v}$ but with respect to BP and $B\mathbf{z}$. The proof of Theorem 6 can be completed by proving that if there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ then there exists a vector $\tilde{\mathbf{y}}'$ and a Boolean $\tilde{b}$ such that $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{\mathbf{v}} = \mathbf{0}$. Suppose this proposition to be true and let B be such that $|B\mathbf{z}| = 1$. Then, as previously demonstrated, Equation 67 holds for BP and $B\mathbf{z}$ if and only if there exists a vector $\tilde{\mathbf{y}}'$ and a Boolean $\tilde{b}$ such that $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{\mathbf{v}} = \mathbf{0}$, which would imply that there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ if and only if Equation 67 holds for P and $\mathbf{z}$.

Let $\tilde{\mathbf{z}}$ and $\tilde{P}$ be such that $\tilde{z}_\alpha = z_\alpha \oplus z_\beta$ and $\tilde{P}_\alpha = P_\alpha \oplus P_\beta$ for some fixed α, β satisfying $\alpha \neq \beta$ and $\tilde{z}_\gamma = z_\gamma$, $\tilde{P}_\gamma = P_\gamma$ for all γ such that $\gamma \neq \alpha$. Let $\tilde{L}$, $\tilde{X}$ and $\tilde{\mathbf{v}}$ be constructed in the same way as L , X and $\mathbf{v}$ but with respect to $\tilde{P}$ and $\tilde{\mathbf{z}}$. Let's assume that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$, we will show that it implies $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{\mathbf{v}} = \mathbf{0}$, where $\tilde{\mathbf{y}}'$ satisfies $\tilde{y}'_\alpha = y'_\alpha \oplus y'_\beta$ and $\tilde{y}'_\gamma = y'_\gamma$ for all γ such that $\gamma \neq \alpha$. By using Equation 53 we can deduce that

$$\tilde{X}_{\gamma\gamma'}\tilde{\mathbf{y}}' = \tilde{y}'_\gamma\tilde{z}_{\gamma'} \oplus \tilde{y}'_{\gamma'}\tilde{z}_\gamma = y'_\gamma z_{\gamma'} \oplus y'_{\gamma'} z_\gamma = X_{\gamma\gamma'}\mathbf{y}' \quad (77)$$

for all γ, γ' satisfying $\gamma \neq \alpha$ and $\gamma' \neq \alpha$. And from the definitions of the vectors $\tilde{\mathbf{v}}$ and $\mathbf{v}$, we can deduce that

$$\tilde{v}_{\gamma\gamma'} = \tilde{z}_\gamma \wedge \tilde{z}'_{\gamma'} = z_\gamma \wedge z'_{\gamma'} = v_{\gamma\gamma'} \quad (78)$$

for all γ, γ' satisfying $\gamma \neq \alpha$ and $\gamma' \neq \alpha$. Equations 77 and 78 imply that

$$\tilde{L}_{\gamma\gamma'}\mathbf{y} \oplus \tilde{X}_{\gamma\gamma'}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{v}_{\gamma\gamma'} = L_{\gamma\gamma'}\mathbf{y} \oplus X_{\gamma\gamma'}\mathbf{y}' \oplus bv_{\gamma\gamma'} = 0 \quad (79)$$

for all γ, γ' satisfying $\gamma \neq \alpha$ and $\gamma' \neq \alpha$. Furthermore, for all γ such that $\gamma \neq \alpha$, we have

$$\begin{aligned} \tilde{L}_{\alpha\gamma}\mathbf{y} &\equiv |\tilde{P}_\alpha \wedge \tilde{P}_\gamma \wedge \mathbf{y}| \pmod{2} \\ &\equiv |(P_\alpha \oplus P_\beta) \wedge P_\gamma \wedge \mathbf{y}| \pmod{2} \\ &\equiv |P_\alpha \wedge P_\gamma \wedge \mathbf{y}| + |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \pmod{2} \\ &\equiv L_{\alpha\gamma}\mathbf{y} + L_{\beta\gamma}\mathbf{y} \pmod{2} \\ &\equiv X_{\alpha\gamma}\mathbf{y}' + bv_{\alpha\gamma} + X_{\beta\gamma}\mathbf{y}' + bv_{\beta\gamma} \pmod{2} \end{aligned} \quad (80)$$

which entails

$$\begin{aligned} &\tilde{L}_{\alpha\gamma}\mathbf{y} \oplus \tilde{X}_{\alpha\gamma}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{v}_{\alpha\gamma} \\ &= X_{\alpha\gamma}\mathbf{y}' \oplus X_{\beta\gamma}\mathbf{y}' \oplus \tilde{X}_{\alpha\gamma}\tilde{\mathbf{y}}' \oplus bv_{\alpha\gamma} \oplus bv_{\beta\gamma} \oplus \tilde{b}\tilde{v}_{\alpha\gamma} \\ &= y'_\alpha z_\gamma \oplus y'_\beta z_\alpha \oplus y'_\beta z_\gamma \oplus y'_\gamma z_\beta \oplus \tilde{y}'_\alpha \tilde{z}_\gamma \oplus \tilde{y}'_\gamma \tilde{z}_\alpha \oplus b(z_\alpha \wedge z_\gamma) \oplus b(z_\beta \wedge z_\gamma) \oplus b(\tilde{z}_\alpha \wedge z_\gamma) \\ &= y'_\alpha z_\gamma \oplus y'_\beta z_\alpha \oplus y'_\beta z_\gamma \oplus y'_\gamma z_\beta \oplus (y'_\alpha \oplus y'_\beta) z_\gamma \oplus y'_\gamma (z_\alpha \oplus z_\beta) \\ &\quad \oplus b(z_\alpha \wedge z_\gamma) \oplus b(z_\beta \wedge z_\gamma) \oplus b((z_\alpha \oplus z_\beta) \wedge z_\gamma) \\ &= 0 \end{aligned} \quad (81)$$

by relying on Equation 53. Finally, we have

$$\begin{aligned} \tilde{L}_{\alpha\alpha}\mathbf{y} \oplus \tilde{X}_{\alpha\alpha}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{v}_{\alpha\alpha} &= \tilde{L}_{\alpha\alpha}\mathbf{y} \oplus \tilde{y}'_\alpha \tilde{z}_\alpha \oplus \tilde{y}'_\alpha \tilde{z}_\alpha \oplus \tilde{b}\tilde{z}_\alpha \\ &= \tilde{L}_{\alpha\alpha}\mathbf{y} \oplus b(z_\alpha \oplus z_\beta) \\ &\equiv |\tilde{P}_\alpha \wedge \mathbf{y}| + bv_{\alpha\alpha} + bv_{\beta\beta} \pmod{2} \\ &\equiv |(P_\alpha \oplus P_\beta) \wedge \mathbf{y}| + bv_{\alpha\alpha} + bv_{\beta\beta} \pmod{2} \\ &\equiv |P_\alpha \wedge \mathbf{y}| + |P_\beta \wedge \mathbf{y}| + bv_{\alpha\alpha} + bv_{\beta\beta} \pmod{2} \\ &= L_{\alpha\alpha} \oplus bv_{\alpha\alpha} \oplus L_{\beta\beta} \oplus bv_{\beta\beta} \\ &= 0. \end{aligned} \quad (82)$$

Thus,

$$L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0} \implies \tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' \oplus b\tilde{\mathbf{v}} = \mathbf{0} \quad (83)$$

which concludes the proof of Theorem 6. $\square$

Consider the algorithm whose pseudo-code is given in Algorithm 3 and which is built upon Theorem 6. The algorithm starts by performing a call to the TOHPE algorithm (Algorithm 2). The reason for calling the TOHPE algorithm is that it is more efficient to first search for a vector $\mathbf{y}$ satisfying $L\mathbf{y} = \mathbf{0}$ and exploit Theorem 1 than to search for vectors $\mathbf{y}, \mathbf{y}'$ and a Boolean b satisfying $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$, where $L, X, \mathbf{v}$ are defined as in Theorem 6. The algorithm then proceeds by creating the matrix L and the set Z which contains all the vectors $\mathbf{z}$ that can potentially be used to transform P via Theorem 6 and reduce its number of columns. For each vector $\mathbf{z} \in Z$, the algorithm constructs the set $S^{(\mathbf{z})}$ of pairs of indices $\{i, j\}$ satisfying $P_{:,i} \oplus P_{:,j} = \mathbf{z}$ in the case where $i \neq j$ or satisfying $P_{:,i} = \mathbf{z}$ in the case where $i = j$, as well as the matrix $X^{(\mathbf{z})}$ and the vector $\mathbf{v}^{(\mathbf{z})}$ which are constructed in the same way as in Theorem 6. The set of vectors $\mathbf{y}$ satisfying $L\mathbf{y} \oplus X^{(\mathbf{z})}\mathbf{y}' \oplus b\mathbf{v}^{(\mathbf{z})} = \mathbf{0}$ for some vector $\mathbf{y}'$ and some Boolean b are stored in $N^{(\mathbf{z})}$. The algorithm then computes the pair of vectors $\mathbf{z}, \mathbf{y}$ where $\mathbf{z} \in Z$ and $\mathbf{y} \in N^{(\mathbf{z})}$ for which the number of duplicated and all-zero columns in P' is maximized, where P' is defined as follows:

$$P' = \begin{cases} P \oplus \mathbf{z}\mathbf{y}^T & \text{if } |\mathbf{y}| \equiv 0 \pmod{2}, \\ \begin{bmatrix} P \oplus \mathbf{z}\mathbf{y}^T & \mathbf{z} \end{bmatrix} & \text{otherwise.} \end{cases} \quad (84)$$

This number is given by the following objective function:

$$-|\mathbf{y}| \pmod{2} + \sum_{\{i,j\} \in S^{(\mathbf{z})}} 2(y_i \oplus y_j) + \delta_{ij} [y_i + 2(y_i \oplus 1)(|\mathbf{y}| \pmod{2})] \quad (85)$$

which is the same as the one used in Algorithm 2. After computing the vectors $\mathbf{z}$ and $\mathbf{y}$ maximizing this function, Algorithm 3 apply the transformation on the parity table and a recursive call is performed if the number of columns in the parity table has successfully been reduced, otherwise the parity table is returned.

Complexity analysis. Performing a call to the TOHPE algorithm induces a complexity of $\mathcal{O}(n^2m^3)$. The set Z and the sets $S^{(\mathbf{z})}$ can be created with $\mathcal{O}(nm^2)$ operations. We can take advantage of the fact that L doesn't depend on $\mathbf{z}$ to compute $N^{(\mathbf{z})}$ efficiently. Let $\tilde{L}$ be the reduced column echelon form of L and let B be the matrix such that $\tilde{L}B = L$. Let $\tilde{X}^{(\mathbf{z})}$ be such that

$$\tilde{X}_{:, \ell}^{(\mathbf{z})} = X_{:, \ell} \bigoplus_{j \in J^{(\ell)}} \tilde{L}_{:, j} \quad (86)$$

where $J^{(\ell)}$ is defined as follows:

$$J^{(\ell)} = \{j \mid \exists i \text{ such that } X_{i, \ell}^{(\mathbf{z})} = 1, \tilde{L}_{i, j} = 1, \tilde{L}_{i, k} = 0, \forall k \neq j\}. \quad (87)$$

And let $\tilde{\mathbf{v}}^{(\mathbf{z})}$ be such that

$$\tilde{\mathbf{v}}^{(\mathbf{z})} = \mathbf{v} \bigoplus_{j \in J} \tilde{L}_{:, j} \quad (88)$$

where J is defined as follows:

$$J = \{j \mid \exists i \text{ such that } v_i = 1, \tilde{L}_{i, j} = 1, \tilde{L}_{i, k} = 0, \forall k \neq j\}. \quad (89)$$

If there exists no vector $\mathbf{y}$ such that $\mathbf{y} \neq \mathbf{0}$ and $L\mathbf{y} = \mathbf{0}$ (which is the case after having executed the TOHPE algorithm, except potentially for $\mathbf{y} = \mathbf{1}$ which can be ignored as it cannot be used to reduce the number of columns in the parity table), then there exists a vector $\mathbf{y} \neq \mathbf{0}$, a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X^{(\mathbf{z})}\mathbf{y}' \oplus b\mathbf{v}^{(\mathbf{z})} = \mathbf{0}$ if and only if there exists a vector $\tilde{\mathbf{y}}'$ and a boolean $\tilde{b}$ such that $\tilde{X}^{(\mathbf{z})}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{\mathbf{v}}^{(\mathbf{z})} = \mathbf{0}$. If the equation $\tilde{X}^{(\mathbf{z})}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{\mathbf{v}}^{(\mathbf{z})} = \mathbf{0}$ is satisfied then the associated vector $\mathbf{y}$ is equal to

$$\mathbf{y} = \begin{bmatrix} \tilde{B}^{(X)} & \tilde{B}^{(v)} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{y}}' \\ \tilde{b} \end{bmatrix} \quad (90)$$

Algorithm 3: Faster version of the third order duplicate and destroy algorithm

Input: A parity table P of size $n \times m$.

Output: An equivalent parity table with an optimized number of columns.

```

1 procedure FastTODD( $P$ )
2    $P \leftarrow \text{TOHPE}(P)$ 
3    $Z \leftarrow \{P_{:,i} \oplus P_{:,j} \mid 0 \leq i < j < m\} \cup \{P_{:,i} \mid 0 \leq i < m\}$ 
4    $L \leftarrow$  matrix whose rows are forming the set  $\{P_i \wedge P_j \mid 0 \leq i \leq j < n\}$  labelled by  $ij$ 
5   forall  $z \in Z$  do
6      $S^{(z)} \leftarrow \{\{i, j\} \mid P_{:,i} \oplus P_{:,j} = z\} \cup \{\{i, i\} \mid P_{:,i} = z\}$ 
7      $X^{(z)} \leftarrow$  matrix such that  $X_{\alpha\beta,\gamma}^{(z)} = z_\alpha \delta_{\beta\gamma} \oplus z_\beta \delta_{\alpha\gamma}$ 
8      $\mathbf{v}^{(z)} \leftarrow$  vector such that  $v_{\alpha\beta} = z_\alpha \wedge z_\beta$ 
9      $N^{(z)} \leftarrow$  generators of the set  $\{\mathbf{y} \mid L\mathbf{y} \oplus X^{(z)}\mathbf{y}' \oplus b\mathbf{v}^{(z)} = \mathbf{0}\}$ 
10  end
11   $\mathbf{z}, \mathbf{y} \leftarrow \underset{\mathbf{z} \in Z, \mathbf{y} \in N^{(\mathbf{z})}}{\text{argmax}} \left\{ -|\mathbf{y}| \pmod{2} + \sum_{\{i,j\} \in S^{(\mathbf{z})}} 2(y_i \oplus y_j) + \delta_{ij} [y_i + 2(y_i \oplus 1)(|\mathbf{y}| \pmod{2})] \right\}$ 
12   $P' \leftarrow P \oplus \mathbf{z}\mathbf{y}^T$ 
13  if  $|\mathbf{y}| \equiv 1 \pmod{2}$  then
14     $P' \leftarrow [P' \quad \mathbf{z}]$ 
15  end
16   $P' \leftarrow P'$  with all its duplicated and all-zero columns removed
17  if  $\text{size}(P') < \text{size}(P)$  then
18    return FastTODD( $P'$ )
19  end
20  return  $P$ 

```

where $\tilde{B}^{(X)}$ is defined as follows:

$$\tilde{B}_{:, \ell}^{(X)} = \bigoplus_{j \in J^{(\ell)}} \tilde{L}_{:, j} \quad (91)$$

and $\tilde{B}^{(v)}$ is defined as follows:

$$\tilde{B}^{(v)} = \bigoplus_{j \in J} \tilde{L}_{:, j}. \quad (92)$$

Constructing the matrix $\tilde{X}^{(z)}$ can be done in $\mathcal{O}(n^4)$ operations because $X^{(z)}$ has n columns and $\mathcal{O}(n^2)$ rows, and $|X_{:, \ell}^{(z)}| < n$ which imply that the set $J^{(\ell)}$ contains no more than n elements. Constructing the vector $\tilde{\mathbf{v}}^{(z)}$ can also be done in $\mathcal{O}(n^4)$ operations because $\mathbf{v}^{(z)}$ has $\mathcal{O}(n^2)$ rows and the set J contains $\mathcal{O}(n^2)$ elements. Performing a Gaussian elimination on the matrix $[\tilde{X}^{(z)} \quad \tilde{\mathbf{v}}^{(z)}]$ to compute $\tilde{\mathbf{y}}'$ and $\tilde{b}$ requires $\mathcal{O}(n^4)$ operations because the matrix has $\mathcal{O}(n)$ columns and $\mathcal{O}(n^2)$ rows. The matrix $\tilde{B}^{(X)}$ and the vector $\tilde{B}^{(v)}$ can be computed with the same computational complexity as the matrix $\tilde{X}^{(z)}$ and the vector $\tilde{\mathbf{v}}^{(z)}$. The right nullspace of $[\tilde{X}^{(z)} \quad \tilde{\mathbf{v}}^{(z)}]$ is generated by $\mathcal{O}(n)$ pairs of vector and Boolean $\tilde{\mathbf{y}}', \tilde{b}$. For each one of these pairs, the associated vector $\mathbf{y} \in N^{(z)}$ for which there exists a vector $\mathbf{y}'$ and a Boolean b satisfying $L\mathbf{y} \oplus X^{(z)}\mathbf{y}' \oplus b\mathbf{v}^{(z)} = \mathbf{0}$ can be computed in $\mathcal{O}(n^3)$ by using Equation 90, for a total complexity of $\mathcal{O}(n^4)$. This must be done for all the $\mathcal{O}(m^2)$ vectors $\mathbf{z}$, which implies a complexity of $\mathcal{O}(n^4 m^2)$ for computing all the sets $N^{(z)}$. The vectors $\mathbf{z}$ and $\mathbf{y}$ satisfying the argmax function can be computed in $\mathcal{O}(n^3 m^2)$ operations because the union of all the $S^{(z)}$ sets contains $\mathcal{O}(m^2)$ elements, computing the value $|\mathbf{y}| \pmod{2}$ requires $\mathcal{O}(n^2)$ operations and there are $\mathcal{O}(nm^2)$ vectors $\mathbf{y}$. Updating the parity table P induces $\mathcal{O}(nm)$ operations. Finally, the algorithm is performing no more than m recursive calls. Thus, the overall worst-case complexity of Algorithm 3 is $\mathcal{O}(n^4 m^3)$.

This complexity is significantly better than the $\mathcal{O}(n^3 m^5)$ complexity of the original TODD algorithm. Note that $n \leq m$, otherwise the instance of the problem can be simplified by eliminating a row of the parity table. Indeed, if $n > m$, then a change of basis can be performed onto the parity table P so that there exists a row α satisfying $P_\alpha = \mathbf{0}$, this row can then be removed from the parity table P which leads to a smaller instance of the problem.

4 Benchmarks

In this section we evaluate the performances of the algorithms we presented, and we compare them to state-of-the-art alternatives. The benchmarks are performed on a set of circuits commonly used to compare the performances of T -count optimizers, which were obtained from References [50] and [51]. We also include a circuit implementing the block cipher DEFAULT, as given in Reference [52].

In Subsection 4.1, we benchmark the T -count optimizers presented in Section 3. We consider the case where ancillary qubits can be used, as well as the case where no ancillary qubits are used. Then, in Subsection 4.2, we compare the performance of our algorithms with state-of-the-art methods for the optimization of multiplication circuits in finite fields.

4.1 TOHPE and FastTODD algorithms

In this subsection we compare the performances of the TOHPE procedure (Algorithm 2), the FastTODD procedure (Algorithm 3), the PHAGE procedure of Reference [33], the TODD procedure of Reference [36] and the deep reinforcement learning method of Reference [53] named AlphaTensor-Quantum. The PHAGE procedure was implemented in Haskell by the authors of the method [54], while the TOHPE, FastTODD and TODD procedures were implemented in the Rust programming language. Our implementations of the TOHPE and FastTODD procedures used for the benchmarks are open source [55].

The execution times of AlphaTensor-Quantum are not reported in the benchmarks because they were not provided in Reference [53]. However, due to the nature of the method, we can expect it to have much longer execution times than all the other methods. Also, some circuits had to be split into several parts for optimizing them with AlphaTensor-Quantum in a reasonable time.

In Reference [53], it is proposed to optimize the number of T gates by implementing CCZ gates using a gadgetization technique presented in Reference [56]. This gadgetization technique uses an ancillary qubit to implement the CCZ with 4 T gates instead of 7 T gates in the case where no ancillary qubits are used. Using this gadgetization technique can therefore lead to a better T -count. However, if the purpose of optimizing the number of T gates is to use fewer resources to implement them using magic states distillation protocols, then this approach can be counterproductive. Indeed, as shown in Reference [43], an implementation of a phase polynomial associated with a third order homogeneous weighted polynomial (as in Equation 23) can benefit from a free round of distillation with quadratic error suppression. By using an ancillary qubit, the gadgetization technique for implementing the CCZ gate breaks this important property of the weighted polynomial. Such protocols for implementing and distilling a phase polynomial in a single step are referred to as synthillation protocols. The simplest example of this is that the CCZ gate can be implemented more efficiently by using the corresponding synthillation protocol rather than by distilling 4 magic states and then implementing the CCZ gate using the gadgetization technique. Concretely, for distilling n CCZ gates with quadratic error suppression, the synthillation protocol consumes $\mathcal{O}(6n)$ magic states, whereas using the gadgetization technique would consume $\mathcal{O}(12n)$ magic states by using the Bravyi-Haah protocol (which is the magic state distillation protocol with quadratic error suppression that consumes the fewest number of magic states, up to a small constant) [57]. That is why, in order to maintain consistent metrics and to assign the same cost to each T gate, we won't consider the gadgetization technique of Reference [56] for implementing the CCZ gates in the benchmarks.

Also, the only gate count metric considered in the benchmarks will be the T -count. It is important to note that all the methods considered in the benchmarks for optimization of the number of T gates can significantly increase the number of CNOT gates in the circuit. If necessary, a phase polynomial synthesis algorithm can be used to perform the synthesis of the phase polynomial produced by the T -count optimizer with an optimized number of CNOT gates [58, 59].

We first consider the case where internal Hadamard gates are gadgetized in Subsection 4.1.1, and then the case where no ancillary qubits are used in Subsection 4.1.2.

Circuit	Pre-optimization			Alpha [53]	PHAGE [33]		TOHPE		FastTODD		TODD [36]	
	n	h	T -count	T -count	T -count	t (s)	T -count	t (s)	T -count	t (s)	T -count	t (s)
Adder ₈	24	37	173	139	173	25	119	0	119	1	128	5580
Barenco Tof ₃	5	3	16	13	13	0	13	0	13	0	14	0
Barenco Tof ₄	7	7	28	23	26	1	23	0	23	0	24	0
Barenco Tof ₅	9	11	40	33	39	11	33	0	33	0	34	0
Barenco Tof ₁₀	19	31	100	83	100	27	83	0	83	0	84	237
CSLA MUX ₃	15	6	62	39	46	39	39	0	39	0	42	6
CSUM MUX ₉	30	12	84	71	84	28	71	0	71	0	72	87
DEFAULT	640	11936	39744	-	-	-	38638	-	38638	-	39744	-
Grover ₅	9	68	166	152	166	26	143	0	143	3	144	6749
Ham ₁₅ (high)	20	331	1019	773	1019	37	691	2	643	-	1001	-
Ham ₁₅ (low)	17	29	97	73	97	26	77	0	77	0	76	453
Ham ₁₅ (med)	17	54	212	156	212	28	147	0	137	19	148	21472
HWB ₆	7	20	75	51	68	117	51	0	51	0	51	29
HWB ₈	12	1103	3517	-	3517	228	2763	169	2763	-	3517	-
HWB ₁₀	16	5191	15891	-	15891	10422	12845	51781	12845	-	15891	-
HWB ₁₁	15	14441	44500	-	-	-	42643	-	42643	-	44500	-
Mod Adder ₁₀₂₄	28	304	1011	762	1010	36	657	2	575	-	957	-
Mod Mult ₅₅	9	3	35	17	21	0	17	0	17	0	18	0
Mod Red ₂₁	11	17	73	51	72	25	51	0	51	0	53	21
Mod5 ₄	5	0	8	7	7	0	7	0	7	0	8	0
QCLA Adder ₁₀	36	25	162	135	159	26	113	0	109	4	111	4216
QCLA Com ₇	24	18	95	59	91	28	59	0	59	0	60	149
QCLA Mod ₇	26	58	237	199	237	26	167	0	159	41	168	60415
QFT ₄	5	38	67	53	55	33	53	0	53	0	55	54
RC Adder ₆	14	10	47	37	44	32	37	0	37	0	38	2
Tof ₃	5	2	15	13	13	0	13	0	13	0	14	0
Tof ₄	7	4	23	19	20	0	19	0	19	0	20	0
Tof ₅	9	6	31	25	28	1	25	0	25	0	26	0
Tof ₁₀	19	16	71	55	69	26	55	0	55	0	56	24
VBE Adder ₃	10	4	24	19	22	1	19	0	19	0	20	0

Table 1: Comparison of different procedures for the optimization of the number of T gates. n represents to the number of input qubits in the circuits. h denotes the number of internal Hadamard gates in the input circuits. All internal Hadamard gates were gadgetized, resulting in h ancillary qubits. The T -count after optimization and the execution time is reported for each procedure. A blank entry in the execution time indicates that the execution couldn't be carried out in less than a day. In such cases, the reported T -count is the one obtained after 24 hours of execution. A blank entry for the AlphaTensor-Quantum column indicates that no data were reported by the authors on the corresponding circuit. A blank entry in the T -count for the PHAGE procedure indicates that the execution was halted because the memory usage exceeded 8 terabytes.

4.1.1 With ancillary qubits

We compare the performances of the algorithms in the case where all the internal Hadamard gates have been gadgetized, as explained in Section 2.1. All the circuits were pre-optimized using the **FastTMerge** procedure of Reference [60], followed by the **InternalHOpt** procedure of Reference [41] to minimize the number of internal Hadamard gates. This pre-optimization is important as reducing the number of internal Hadamard gates leads to fewer ancillary qubits when these Hadamard gates are gadgetized, and therefore improve the efficiency of the T -count optimizers.

The benchmark results are presented in Table 1. As expected from the complexity analysis of the algorithms, we can notice that the execution times of the **FastTODD** procedure are much lower than the execution times of the **TODD** procedure. Moreover, the **FastTODD** procedure is providing the best-known T -count on every circuits, except for the “Ham₁₅ (low)” circuit. While the **TODD** procedure and the **FastTODD** procedure are based on the same mechanism for reducing the T -count, the modifications that we made in the **FastTODD** procedure are resulting in significantly better T -count reduction for some circuits. For instance, for the “Mod Adder₁₀₂₄” circuit, the **FastTODD** procedure generates a circuit with a T -count that is 40% lower than the T -count of the circuit generated by the **TODD** procedure. This 40% improvement remains consistent even if we allow the algorithms to terminate by not stopping them after 24 hours.

The **TOHPE** procedure is also achieving an equivalent or better T -count than the state-of-the-art algorithms for all circuits except the “Ham₁₅ (low)” circuit. Moreover, on every circuit, the execution times of the **TOHPE** and **FastTODD** procedures are lower than the execution times of the other procedure (for the “HWB₈” and “HWB₁₀” circuits, the **PHAGE** procedure have lower

Circuit	Pre-optimization		PHAGE [33]		TOHPE		FastTODD		TODD [36]	
	n	T -count	T -count	t (s)	T -count	t (s)	T -count	t (s)	T -count	t (s)
Adder ₈	24	173	172	17	173	0	170	0	172	3
Barenco Tof ₃	5	16	16	0	16	0	16	0	16	0
Barenco Tof ₄	7	28	28	0	28	0	28	0	28	0
Barenco Tof ₅	9	40	40	0	40	0	40	0	40	0
Barenco Tof ₁₀	19	100	100	3	100	0	100	0	100	0
CSLA MUX ₃	15	62	50	2	49	0	49	0	50	0
CSUM MUX ₉	30	84	84	79	73	0	73	0	76	7
DEFAULT	640	39744	-	-	39666	-	39666	-	39744	-
Grover ₅	9	166	166	0	166	0	166	0	166	0
Ham ₁₅ (high)	20	1019	1019	5	1019	0	1019	0	1019	1
Ham ₁₅ (low)	17	97	94	0	94	0	94	0	94	0
Ham ₁₅ (med)	17	212	212	3	212	0	212	0	212	1
HWB ₆	7	75	75	0	75	0	75	0	75	0
HWB ₈	12	3517	3508	32	3498	0	3487	0	3493	2
HWB ₁₀	16	15891	15858	392	15741	1	15693	2	15698	14
HWB ₁₁	15	44500	44479	1418	44336	3	44188	5	44191	28
HWB ₁₂	20	85611	85588	8153	85362	10	85233	16	85239	135
Mod Adder ₁₀₂₄	28	1011	1010	5636	1009	0	1009	0	1010	13
Mod Mult ₅₅	9	35	28	0	28	0	28	0	28	0
Mod Red ₂₁	11	73	73	0	73	0	73	0	73	0
QCLA Adder ₁₀	36	162	161	158	161	0	161	0	161	53
QCLA Com ₇	24	95	95	16	95	0	95	0	95	2
QCLA Mod ₇	26	237	237	64	237	0	237	0	237	7
QFT ₄	5	67	67	0	67	0	66	0	66	0
RC Adder ₆	14	47	47	0	47	0	47	0	47	0
Tof ₃	5	15	15	0	15	0	15	0	15	0
Tof ₄	7	23	23	0	23	0	23	0	23	0
Tof ₅	9	31	31	0	31	0	31	0	31	0
Tof ₁₀	19	71	71	6	71	0	71	0	71	0
VBE Adder ₃	10	24	24	0	24	0	24	0	24	0

Table 2: Comparison of different procedures for the optimization of the number of T gates without using any ancillary qubits. The value n corresponds to the number of qubits in the circuits. The T -count after optimization and the execution time is reported for each procedure. A blank entry in the execution time indicates that the execution couldn't be carried out in less than a day. In such cases, the reported T -count is the one obtained after 24 hours of execution. A blank entry in the T -count for the PHAGE procedure indicates that the execution was halted because the memory usage exceeded 8 terabytes.

execution times but but does not improve the T -count at all for these circuits). This demonstrates that our algorithms are faster than state-of-the-art T -count optimizers and are producing circuits with a lower or equivalent T -count.

4.1.2 Without ancillary qubits

The benchmarks results in the case where the internal Hadamard gates are not gadgetized are presented in Table 2. All the circuits were pre-optimized using the **FastTMerge** procedure of Reference [60]. Then the circuits were divided into Hadamard-free subcircuits interposed with Clifford circuits, as described in Subsection 2.1 and using Algorithm 1 to minimize the number of Hadamard-free subcircuits.

Similarly to the case where the internal Hadamard gates are gadgetized, the **FastTODD** procedure provides the best-known T -count for all the circuits. Also, the **TOHPE** and **FastTODD** procedures have the lowest execution times. However, we can notice that the number of T gates was not reduced at all for multiple quantum circuits, and the T -count reduction achieved for the other circuits was not very substantial. This indicates the importance of the gadgetization of internal Hadamard gates for lowering the number of T gates once the **FastTMerge** procedure described in Reference [60] has been applied. A possible direction for future research work would be to develop effective strategies for optimizing the number of T gates by exploiting the gadgetization of internal Hadamard gates in cases where the number of ancillary at disposal qubits is limited.

Circuit	T -count	Alpha [53]		Ref. [61]	FastTODD		Ref. [61] & FastTODD	
		T -count	CCZ	CCZ	T -count	t (s)	T -count	t (s)
GF(2^2) Mult	18	17	3	3	18	0	17	0
GF(2^3) Mult	45	29	6	6	36	0	23	0
GF(2^4) Mult	68	39	9	9	49	0	43	0
GF(2^5) Mult	115	59	13	14	81	0	61	0
GF(2^6) Mult	150	77	18	18	113	0	83	0
GF(2^7) Mult	217	104	22	23	155	1	111	0
GF(2^8) Mult	264	123	29	27	205	1	135	0
GF(2^9) Mult	351	161	35	38	257	5	185	2
GF(2^{10}) Mult	410	196	46	42	315	17	207	3
GF(2^{16}) Mult	1040	-	-	81	797	320	425	40
GF(2^{32}) Mult	4128	-	-	243	3213	49386	1255	16485
GF(2^{64}) Mult	16448	-	-	729	12503	-	3817	-
GF(2^{128}) Mult	65664	-	-	2187	50887	-	11545	-
GF(2^{256}) Mult	262400	-	-	6561	220442	-	34731	-
GF(2^{512}) Mult	1048576	-	-	19683	1036270	-	91931	-

Table 3: Comparison of different procedures for the optimization of the number of T and CCZ gates in GF(2^n) multiplier circuits. The CCZ columns refers to the number of CCZ (or Toffoli) gates in the optimized circuit. A blank entry in the execution time for the FastTODD procedure indicates that the execution couldn't be carried out in less than a day. In such cases, the reported T -count is the one obtained after 24 hours of execution. A blank entry for the AlphaTensor-Quantum column indicates that no data were reported by the authors on the corresponding circuit.

4.2 Galois field multiplier circuits

In this subsection, we compare the performances of AlphaTensor-Quantum [53], the FastTODD procedure, and the synthesis algorithm of Reference [61] for optimizing the number of CCZ and T gates in GF(2^n) multiplier circuits. The circuits were pre-optimized using the FastTMerge procedure of Reference [60], followed by the InternalHOpt procedure of Reference [41] to minimize the number of internal Hadamard gates. After this optimization, the number of internal Hadamard gates is reduced to zero, thus eliminating the need for Hadamard gate gadgetization and ancillary qubits.

The benchmarks results are presented in Table 3. We can notice that AlphaTensor-Quantum finds the best number of CCZ gates in the case where n is equal to 5, 7 and 9. This demonstrates that, while the method of Reference [61] generates circuits with a subquadratic number of CCZ gates, the optimized circuit may not have an optimal number of CCZ gates (at least in the case where n is not a power of 2). The major drawback of AlphaTensor-Quantum is its large complexity, which imply that it can only be used on small circuits. Conversely, the FastTODD algorithm was able to optimize the number of T gates even on the largest circuit, although it did not terminate within 24 hours.

The benchmarks results are a good demonstration of the advantages and disadvantages of the AlphaTensor-Quantum method and the FastTODD procedure. Moreover, the results indicate that these two algorithms are not really in direct competition but rather represent complementary approaches. One of the differences between the two methods is that AlphaTensor-Quantum performs a complete resynthesis of the Hadamard-free parts of the quantum circuit. This implies that AlphaTensor-Quantum cannot be used to improve upon existing quantum circuits that may be already well optimized. For instance, whether or not the number of the number of T gates in the “GF(2^n) Mult” input circuit is well optimized will not change the circuit produced by AlphaTensor-Quantum. The main advantage of this resynthesis approach of AlphaTensor-Quantum is that it can be used to discover radically different circuits that have much lower number of T gates. We can clearly see that this is the case for the GF(2^n) multiplier circuits: the input circuit contains $\mathcal{O}(n^2)$ but AlphaTensor-Quantum finds a circuit with a number of CCZ gates similar to the method

of Reference [61] which produces a circuit with $\mathcal{O}(n^{1.58})$ CCZ gates. This demonstrates that AlphaTensor-Quantum could be used to give valuable insights for the design of efficient synthesis methods. On the contrary, the **FastTODD** procedure does not perform well on the “GF(2^n) Mult” circuits containing $\mathcal{O}(n^2)$ CCZ gates. However, the **FastTODD** procedure can be used to optimize the number of T gates in the circuit produced by the method given in Reference [61]. The **FastTODD** procedure could also be used to improve the solution of AlphaTensor-Quantum. For instance, as shown in Table 3, some of the circuits produced by AlphaTensor-Quantum contain an even number of T gates. In such cases, the parity table associated with the circuit satisfies all the conditions of Theorem 1. Consequently, applying the **FastTODD** procedure on these circuits will necessarily reduce the number of T gates.

5 Extension to higher levels of the Clifford hierarchy

In this section, we extend the results of Section 3 for the optimization of the number of $R_Z(\pi/2^d)$ gates in $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuits, for any non-negative integer d . These gates appear in various quantum algorithms, such as Shor’s algorithm [62], and there exists distillation protocols for implementing them fault tolerantly [63, 64]. Optimizing the number of $R_Z(\pi/2^d)$ gates may increase the number of $R_Z(2\pi/2^d)$ gates. However, this is motivated by the fact that the $R_Z(\pi/2^d)$ gate is typically more costly to implement than the $R_Z(2\pi/2^d)$ gate.

We formalize the problem of $R_Z(\pi/2^d)$ gates optimization in $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuits in Subsection 5.1. Then, in Subsection 5.2, we extend Theorem 3 to give an upper bound on the number of $R_Z(\pi/2^d)$ gates in $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuits, which can be satisfied in polynomial time by generalizing Algorithm 2. Finally, in Subsection 5.3, we show how we can construct algorithms that are similar to Algorithm 3 for optimizing the number of $R_Z(\pi/2^d)$ gates by providing a generalization of Theorem 6.

5.1 Symmetric tensor rank decomposition problem

In this subsection we formalize the problem of $R_Z(\pi/2^d)$ gates optimization in $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuits for any non-negative integer d , by showing its equivalence to the following symmetric tensor rank decomposition problem.

Problem 2 (d -STR). *Let $\mathcal{A} \in \mathbb{Z}_2^{(n, \dots, n)}$ be a symmetric tensor of order d such that*

$$\mathcal{A}_{\alpha_1, \dots, \alpha_d} = \mathcal{A}_{\beta_1, \dots, \beta_d} \quad (93)$$

for all $\alpha_1, \dots, \alpha_d$ and $\beta_1, \dots, \beta_d$ such that the set equality $\{\alpha_i, \dots, \alpha_d\} = \{\beta_i, \dots, \beta_d\}$ is satisfied. Find a Boolean matrix P of size $n \times m$ such that

$$\mathcal{A}_{\alpha_1, \dots, \alpha_d} = \left| \bigwedge_{i=1}^d P_{\alpha_i} \right| \pmod{2} \quad (94)$$

for all $\alpha_1, \dots, \alpha_d$ satisfying $0 \leq \alpha_1 \leq \dots \leq \alpha_d < n$, with minimal m .

We now prove the equivalence between the problem of minimizing the number of $R_Z(\pi/2^d)$ gates in a $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit for any non-negative integer d and the $(d+1)$ -STR problem. The following theorem was first proven in Reference [31] by showing the equivalence between the problem of optimizing $R_Z(\pi/2^d)$ gates and the minimum distance decoding problem for the punctured Reed-Muller code of order $n-d-2$ and length 2^n-1 , which is equivalent to the d -STR problem [47]. We provide a more straightforward proof of this theorem in Appendix [61].

Theorem 7. *Let d be a non-negative integer. The $R_Z(\pi/2^d)$ -count optimization problem over the $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ gate set and the $(d+1)$ -STR problem are equivalent.*

5.2 $R_Z(\pi/2^d)$ -count upper bound in $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuits

In this section we present an upper bound for the number of $R_Z(\pi/2^d)$ gates in a $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit, and we present a method for satisfying this upper bound in polynomial time. We first prove the following theorem, which is a generalization of Theorem 1.

Theorem 8. Let d be a non-negative integer, let P be a parity table of size $n \times m$ and let $P' = P \oplus \mathbf{z}\mathbf{y}^T$ where $\mathbf{z}$ and $\mathbf{y}$ are vectors of size n and m respectively such that

$$|\mathbf{y}| \equiv 0 \pmod{2} \quad (95)$$

$$\left| \bigwedge_{i=1}^d P_{\alpha_i} \wedge \mathbf{y} \right| \equiv 0 \pmod{2} \quad (96)$$

for all $0 \leq \alpha_1 \leq \dots \leq \alpha_d < n$. Then we have

$$\left| \bigwedge_{i=1}^{d+1} P'_{\alpha_i} \right| \equiv \left| \bigwedge_{i=1}^{d+1} P_{\alpha_i} \right| \pmod{2} \quad (97)$$

for all $0 \leq \alpha_1 \leq \dots \leq \alpha_{d+1} < n$.

Proof. For all $\alpha_1, \dots, \alpha_{d+1}$ satisfying $0 \leq \alpha_1 \leq \dots \leq \alpha_{d+1} < n$, we have:

$$\begin{aligned} \left| \bigwedge_{i=1}^{d+1} P'_{\alpha_i} \right| &= \left| \bigwedge_{i=1}^{d+1} (P_{\alpha_i} \oplus z_{\alpha_i} \mathbf{y}) \right| \\ &= \left| \bigoplus_{k_{d+1}=0}^1 \dots \bigoplus_{k_1=0}^1 \left[\bigwedge_{i=1}^{d+1} (k_i P_{\alpha_i} \vee (1 - k_i) z_{\alpha_i} \mathbf{y}) \right] \right| \\ &\equiv \sum_{k_{d+1}=0}^1 \dots \sum_{k_1=0}^1 \left| \bigwedge_{i=1}^{d+1} (k_i P_{\alpha_i} \vee (1 - k_i) z_{\alpha_i} \mathbf{y}) \right| \pmod{2}. \end{aligned} \quad (98)$$

The expression

$$\left| \bigwedge_{i=1}^{d+1} (k_i P_{\alpha_i} \vee (1 - k_i) z_{\alpha_i} \mathbf{y}) \right| \pmod{2} \quad (99)$$

is equal to

$$\left| \bigwedge_{i=1}^{d+1} z_{\alpha_i} \mathbf{y} \right| \pmod{2} \quad (100)$$

in the case where $k_{d+1} = \dots = k_1 = 0$, which is equal to 0 because Equation 95 is satisfied. Moreover, in the case where $\mathbf{k} \in \mathbb{Z}_2^{d+1}$ and the equations $k_{d+1} = \dots = k_1 = 1$ and $k_{d+1} = \dots = k_1 = 0$ are not satisfied, Expression 100 is equal to 0 if there exists i such that $k_i = 0$ and $z_{\alpha_i} = 0$. Otherwise it is equal to

$$\left| \bigwedge_{i=1}^d P_{\beta_i} \wedge \mathbf{y} \right| \pmod{2} \quad (101)$$

for some $\beta_1, \dots, \beta_d$ satisfying $0 \leq \beta_1 \leq \dots \leq \beta_d < n$. Expression 101 is equal to Equation 96 and is therefore equal to 0. Thus,

$$\left| \bigwedge_{i=1}^{d+1} (k_i P_{\alpha_i} \vee (1 - k_i) z_{\alpha_i} \mathbf{y}) \right| \equiv 0 \pmod{2} \quad (102)$$

for all $\mathbf{k} \in \mathbb{Z}_2^{d+1}$ such that $k_i = 0$ for some i . Finally, in the case where $k_{d+1} = \dots = k_1 = 1$, Expression 99 is equal to

$$\left| \bigwedge_{i=1}^{d+1} P_{\alpha_i} \right| \pmod{2} \quad (103)$$

Therefore,

$$\begin{aligned} \left| \bigwedge_{i=1}^{d+1} P'_{\alpha_i} \right| &\equiv \sum_{k_{d+1}=0}^1 \dots \sum_{k_1=0}^1 \left| \bigwedge_{i=1}^{d+1} (k_i P_{\alpha_i} \vee (1 - k_i) z_{\alpha_i} \mathbf{y}) \right| \pmod{2} \\ &\equiv \left| \bigwedge_{i=1}^{d+1} P_{\alpha_i} \right| \pmod{2}. \end{aligned} \quad (104)$$

□

Based on Theorem 8, we can prove the following subadditivity theorem which is a generalization of Theorem 2.

Theorem 9. *Let d be a non-negative integer, let $U_1 \in \mathcal{D}_{d+1}^C$, and let $U_2 \in \mathcal{D}_{d+1}$ where $\mathcal{D}_{d+1}$ is the diagonal subgroup of the $(d+1)$ th level of the Clifford hierarchy, and $\mathcal{D}_{d+1}^C$ is the subgroup of $\mathcal{D}_{d+1}$ which can be implemented using only $C^{\otimes d}Z$ gates. If $\tau[U_1], \tau[U_2] > 0$, then $\tau[U_1 \otimes U_2] < \tau[U_1] + \tau[U_2]$ where $\tau[U]$ is the optimal number of $R_Z(\pi/2^d)$ gates required to implement U without ancillary qubits over the $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ gate set.*

Proof. Let $W = \begin{bmatrix} P & Q \end{bmatrix}$ be a parity table such that P and Q are the parity tables associated with the implementation of U_1 and U_2 and which have $\tau[U_1]$ and $\tau[U_2]$ columns respectively. Then, because $U_1 \in \mathcal{D}_{d+1}^C$, P satisfies

$$\left| \bigwedge_{i=1}^d P_{\alpha_i} \right| \equiv 0 \pmod{2} \quad (105)$$

for all $\alpha_1, \dots, \alpha_d$ satisfying $0 \leq \alpha_1 < \dots < \alpha_d < n$ where n is the number of qubits on which U_1 is acting. Let $\mathbf{z} = P_{:,i} \oplus Q_{:,j}$ for any i and j satisfying $0 \leq i < \tau[U_1]$, $0 \leq j < \tau[U_2]$, and let P' be a parity table such that

$$P' = \begin{cases} P \oplus \mathbf{z} \mathbf{1}^T & \text{if } \tau[U_1] \equiv 0 \pmod{2}, \\ \begin{bmatrix} P \oplus \mathbf{z} \mathbf{1}^T & \mathbf{z} \end{bmatrix} & \text{otherwise.} \end{cases}$$

Then, as stated by Theorem 8, the parity table P' satisfies

$$\left| \bigwedge_{i=1}^{d+1} P'_{\alpha_i} \right| \equiv \left| \bigwedge_{i=1}^{d+1} P_{\alpha_i} \right| \pmod{2} \quad (106)$$

for all $0 \leq \alpha_1 \leq \dots \leq \alpha_{d+1} < n$. And so the parity table $W' = \begin{bmatrix} P' & Q \end{bmatrix}$, which has at most one more column than W , also satisfies

$$\left| \bigwedge_{i=1}^{d+1} W'_{\alpha_i} \right| \equiv \left| \bigwedge_{i=1}^{d+1} W_{\alpha_i} \right| \pmod{2} \quad (107)$$

for all $0 \leq \alpha_1 \leq \dots \leq \alpha_{d+1} < n$. However, we can notice that $P'_{:,i} = Q_{:,j}$. Therefore, by removing these two columns from W' Equation 107 still holds and W' has at least one less column than W . The parity table W' implements the unitary $U_1 \otimes U_2$ up to an operator implementable over the $\{\text{CNOT}, R_Z(2\pi/2^d)\}$ gate set and has at most $\tau[U_1] + \tau[U_2] - 1$ columns, thus we have $\tau[U_1 \otimes U_2] \leq \tau[U_1] + \tau[U_2] - 1 < \tau[U_1] + \tau[U_2]$. □

Based on this subadditivity theorem and on Theorem 8, we can derive the following upper bound on the number of $R_Z(\pi/2^d)$ gates in a $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit.

Theorem 10. *Let d be a non-negative integer. The number of $R_Z(\pi/2^d)$ gates in an n -qubit $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit can be upper bounded by*

$$2 \left\lceil \sum_{i=1}^d \frac{\binom{n}{i}}{2} \right\rceil + 1 \leq \sum_{i=0}^d \binom{n}{i} \quad (108)$$

in polynomial time.

Proof. Let U be a unitary gate implementable over the $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ gate set, let P be a parity table of size $n \times m$ which implements U up to an operator implementable over

the $\{\text{CNOT}, R_Z(2\pi/2^d)\}$ gate set, and let L be a matrix with rows labelled by $(\alpha_1 \dots \alpha_k)$ such that

$$L_{\alpha_1 \dots \alpha_k} = \bigwedge_{i=1}^k P_{\alpha_i} \quad (109)$$

for all $\alpha_1, \dots, \alpha_k$ satisfying $0 \leq \alpha_i < \dots < \alpha_k < n$ and k satisfying $1 \leq k \leq d$. If P has strictly more than $\sum_{i=0}^d \binom{n}{i}$ columns then we can necessarily find a non-zero vector $\mathbf{y}$ satisfying $L\mathbf{y} = \mathbf{0}$ and $\mathbf{y} \neq \mathbf{1}$ because L has $\sum_{i=1}^d \binom{n}{i}$ rows. Note that such vector $\mathbf{y}$ necessarily satisfies Equations 96 of Theorem 8. We can then divide P into two non-empty parity tables $P^{(1)}$ and $P^{(2)}$ where the column $P_{:,i}$ belongs to $P^{(1)}$ if and only if $y_i = 1$ and to $P^{(2)}$ otherwise. The parity tables $P^{(1)}$ and $P^{(2)}$ are implementations of some unitary gates $U_1 \in \mathcal{D}_{d+1}^C$ and $U_2 \in \mathcal{D}_{d+1}$ respectively. The subadditivity theorem (Theorem 9) can then be exploited to reduce the number of columns of P . Let $\mathbf{z} = P_{:,i} \oplus P_{:,j}$ where i and j are satisfying $y_i = 1$ and $y_j = 0$, and let

$$P' = \begin{cases} P \oplus \mathbf{z}\mathbf{y}^T & \text{if } |\mathbf{y}| \equiv 0 \pmod{2}, \\ \begin{bmatrix} P \oplus \mathbf{z}\mathbf{y}^T & \mathbf{z} \end{bmatrix} & \text{otherwise.} \end{cases}$$

By Theorem 8, P' implements the same unitary gate as P up to an operator implementable over the $\{\text{CNOT}, R_Z(2\pi/2^d)\}$ gate set. The parity table P' has at most one more column than P and we have $P'_{:,i} = P'_{:,j}$. Therefore, the columns i and j can be removed from P' , which entails that P' has at least one less column than P . We showed that if the number of columns of P is strictly greater than $\sum_{i=0}^d \binom{n}{i}$, then the number of columns of P can be reduced by at least one in polynomial time. Moreover, if the number of columns of P is equal to $\sum_{i=0}^d \binom{n}{i}$ and is even, then we can necessarily find a non-zero vector $\mathbf{y}$ satisfying $L\mathbf{y} = \mathbf{0}$ because L has $\sum_{i=1}^d \binom{n}{i}$ rows. If there exist i such that $y_i = 0$ then we can reduced the number of columns of P as described above. Otherwise we must have $|\mathbf{y}| \equiv 0 \pmod{2}$, and so $\mathbf{y}$ satisfies the Equations of Theorem 8. Therefore, if $P' = P \oplus \mathbf{z}\mathbf{y}^T$ where $\mathbf{z}$ is equal to the i th column of P for any i , then P' implements the same unitary gate as P up to an operator implementable over the $\{\text{CNOT}, R_Z(2\pi/2^d)\}$ gate set. The parity table P' has the same number of columns as P but its i th column is equal to the null vector and can therefore be removed, which leads to a parity table containing $\sum_{i=1}^d \binom{n}{i}$ columns.

The polynomial-time procedure described above to reduce the number of columns of P can be repeated until P has a number of columns lower or equal to

$$2 \left\lceil \sum_{i=1}^d \frac{\binom{n}{i}}{2} \right\rceil + 1 \leq \sum_{i=0}^d \binom{n}{i}. \quad (110)$$

□

The overall complexity of the algorithm described in the proof of Theorem 10 is $\mathcal{O}(n^{2d+1}m)$ where m is the number of $R_Z(\pi/2^d)$ gates in the initial circuit. For $d = 0$, the algorithm yields a parity table which contains at most one column, which is optimal. For $d = 1$, the algorithm corresponds to Lempel's matrix factorization algorithm [65], which is optimal. For $d = 2$, the algorithm is similar to the TOHPE algorithm (Algorithm 2). Note that the TOHPE algorithm has a complexity of $\mathcal{O}(n^2m^3)$ instead of $\mathcal{O}(n^5m)$ because of the heuristic used by the algorithm to better optimize the number of T gates. The same heuristic can be used in the case where $d > 2$, which would lead to an algorithm having a complexity of $\mathcal{O}(n^d m^3)$.

5.3 Generalization of the FastTODD algorithm for optimizing $R_Z(\pi/2^d)$ gates

In this subsection we generalize the FastTODD algorithm (Algorithm 3) for the optimization of the number of $R_Z(\pi/2^d)$ gates in a $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit.

Theorem 11. Let d be a non-negative integer, let P be a parity table of size $n \times m$ and $P' = P \oplus \mathbf{z}\mathbf{y}^T$ where $\mathbf{z}$ and $\mathbf{y}$ are vectors of size n and m respectively and such that $|\mathbf{y}| \equiv 0 \pmod{2}$. And let L and X be matrices and $\mathbf{v}$ be a vector, all with rows labelled by $(\alpha_1 \dots \alpha_k)$ such that

$$L_{\alpha_1 \dots \alpha_k} = \bigwedge_{i=1}^k P_i \quad (111)$$

$$X_{\alpha_1 \dots \alpha_k, \beta_1 \dots \beta_{k-1}} = \bigvee_{i=1}^k z_{\alpha_i} \delta_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k, \beta_1 \dots \beta_{k-1}} \quad (112)$$

$$v_{\alpha_1 \dots \alpha_k} = \bigwedge_{i=1}^k z_{\alpha_i} \quad (113)$$

for all $\alpha_1, \dots, \alpha_k$ and $\beta_1, \dots, \beta_{k-1}$ satisfying $0 \leq \alpha_1 < \dots < \alpha_k < n$ and $0 \leq \beta_1 < \dots < \beta_{k-1} < n$ where k satisfies $1 \leq k \leq d$, and where δ is defined as follows:

$$\delta_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k, \beta_1 \dots \beta_{k-1}} = \begin{cases} 1 & \text{if } \alpha_1 = \beta_1, \dots, \alpha_{i-1} = \beta_{i-1}, \alpha_{i+1} = \beta_i, \dots, \alpha_k = \beta_{k-1}, \\ 0 & \text{otherwise.} \end{cases} \quad (114)$$

All entries of L, X and $\mathbf{v}$ are set to zero in the case where $d = 0$ for L and in the case where $d \leq 1$ for X and $\mathbf{v}$. Then, the equality

$$\left| \bigwedge_{i=1}^k P'_{\alpha_i} \right| \equiv \left| \bigwedge_{i=1}^k P_{\alpha_i} \right| \pmod{2} \quad (115)$$

holds for all $0 \leq \alpha_1 < \dots < \alpha_k < n$ and $1 \leq k \leq d+1$ if and only if there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$.

Proof. In the case where $k = 1$, we have

$$\begin{aligned} |P'_{\alpha_1}| &= |P_{\alpha_1} \oplus z_{\alpha_1} \mathbf{y}| \\ &\equiv |P_{\alpha_1}| + |z_{\alpha_1} \mathbf{y}| \pmod{2} \\ &\equiv |P_{\alpha_1}| \pmod{2} \end{aligned} \quad (116)$$

because $|\mathbf{y}| \equiv 0 \pmod{2}$. This proves Theorem 11 in the case where $d = 0$, because in such case the equation $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ is always satisfied. We now prove Theorem 11 in the case where $d > 0$ and $\mathbf{z}$ is satisfying $|\mathbf{z}| = 1$. Without loss of generality we will assume that $z_{\alpha_1} = 1$ and $z_{\alpha_i} = 0$ for all i such that $i \neq 1$, then

$$\begin{aligned} \left| \bigwedge_{i=1}^k P'_{\alpha_i} \right| &= |(P_{\alpha_1} \oplus z_{\alpha_1} \mathbf{y}) \bigwedge_{i=2}^k P_{\alpha_i}| \\ &= |(P_{\alpha_1} \oplus \mathbf{y}) \bigwedge_{i=2}^k P_{\alpha_i}| \\ &= \left| \bigwedge_{i=1}^k P_{\alpha_i} \oplus \bigwedge_{i=2}^k P_{\alpha_i} \wedge \mathbf{y} \right| \\ &\equiv \left| \bigwedge_{i=1}^k P_{\alpha_i} \right| + \left| \bigwedge_{i=2}^k P_{\alpha_i} \wedge \mathbf{y} \right| \pmod{2} \end{aligned} \quad (117)$$

where $\alpha_1, \dots, \alpha_k$ are satisfying $0 \leq \alpha_1 < \dots < \alpha_k < n$ and k is satisfying $1 \leq k \leq d+1$. Therefore, proving Theorem 11 in the case where $d > 0$ and $|\mathbf{z}| = 1$ can be done by showing that there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ if and only if the equation

$$\left| \bigwedge_{i=2}^k P_{\alpha_i} \wedge \mathbf{y} \right| \equiv 0 \pmod{2} \quad (118)$$

holds for all $\alpha_2, \dots, \alpha_k$ satisfying $\alpha_1 < \alpha_2 < \dots < \alpha_k < n$ and k satisfying $2 \leq k \leq d+1$. By definition we have $X_{\alpha_2 \dots \alpha_k} = \mathbf{0}$ and $v_{\alpha_2 \dots \alpha_k} = 0$ for all $\alpha_2, \dots, \alpha_k$ satisfying $0 \leq \alpha_1 < \dots < \alpha_k < n$ and k satisfying $2 \leq k \leq d+1$. Hence,

$$L_{\alpha_2 \dots \alpha_k} \mathbf{y} \oplus X_{\alpha_2 \dots \alpha_k} \mathbf{y}' \oplus bv_{\alpha_2 \dots \alpha_k} = L_{\alpha_2 \dots \alpha_k} \mathbf{y} = \left| \bigwedge_{i=2}^k P_{\alpha_i} \wedge \mathbf{y} \right| \pmod{2} \quad (119)$$

and so if $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ then $L_{\alpha_2 \dots \alpha_k} \mathbf{y} = 0$ which imply that Equation 118 is satisfied. Conversely, if Equation 118 is satisfied then we must have

$$L_{\alpha_2 \dots \alpha_k} \mathbf{y} \oplus X_{\alpha_2 \dots \alpha_k} \mathbf{y}' \oplus bv_{\alpha_2 \dots \alpha_k} = L_{\alpha_2 \dots \alpha_k} = \mathbf{0} \quad (120)$$

for all $\alpha_2, \dots, \alpha_k$ satisfying $\alpha_1 < \alpha_2 < \dots < \alpha_k < n$ and k satisfying $2 \leq k \leq d+1$. Moreover we have

$$L_{\alpha_1} \mathbf{y} \oplus X_{\alpha_1} \mathbf{y}' \oplus bv_{\alpha_1} = L_{\alpha_1} \mathbf{y} \oplus b \quad (121)$$

because $X_{\alpha_1} = \mathbf{0}$ and $v_{\alpha_1} = 1$. And

$$L_{\alpha_1 \dots \alpha_k} \mathbf{y} \oplus X_{\alpha_1 \dots \alpha_k} \mathbf{y}' \oplus bv_{\alpha_1 \dots \alpha_k} = L_{\alpha_1 \dots \alpha_k} \mathbf{y} \oplus y'_{\alpha_2 \dots \alpha_k} \quad (122)$$

for all $\alpha_2, \dots, \alpha_k$ satisfying $\alpha_1 < \alpha_2 < \dots < \alpha_k < n$ and k satisfying $2 \leq k \leq d$, because $v_{\alpha_1 \dots \alpha_k} = 0$ and

$$X_{\alpha_1 \dots \alpha_k} \mathbf{y}' = \bigoplus_{i=1}^k z_{\alpha_i} y'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \quad (123)$$

for all $\alpha_1, \dots, \alpha_k$ satisfying $0 \leq \alpha_1 < \dots < \alpha_k < n$ and k satisfying $2 \leq k \leq d$ and where the rows of $\mathbf{y}'$ are labelled the same way as the columns of X . Let $\mathbf{y}'$ be such that $y'_{\alpha_2 \dots \alpha_k} = L_{\alpha_1 \dots \alpha_k} \mathbf{y}$ for all $\alpha_2, \dots, \alpha_k$ satisfying $\alpha_1 < \alpha_2 < \dots < \alpha_k < n$ and k satisfying $2 \leq k \leq d$ and b be such that $b = L_{\alpha_1} \mathbf{y}$, then we have $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$. Thus, we proved that

$$L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0} \iff \left| \bigwedge_{i=2}^k P_{\alpha_i} \wedge \mathbf{y} \right| \equiv 0 \pmod{2} \quad (124)$$

for all $\alpha_2, \dots, \alpha_k$ satisfying $\alpha_1 < \alpha_2 < \dots < \alpha_k < n$ and k satisfying $2 \leq k \leq d+1$, and so Theorem 11 is true in the case where $|z| = 1$.

Let B be a full rank binary matrix of size $n \times n$ which represents a change of basis, and let $\tilde{L}, \tilde{X}$ and $\tilde{\mathbf{v}}$ be constructed in the same way as L, X and $\mathbf{v}$ but with respect to BP and Bz . Notice that if Equation 115 is satisfied for P and z then it is also satisfied for BP and Bz . The proof of Theorem 11 can then be completed by proving that if there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ then there exists a vector $\tilde{\mathbf{y}}'$ and a Boolean $\tilde{b}$ such that $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{\mathbf{v}} = \mathbf{0}$. Suppose this proposition to be true and let B be such that $|Bz| = 1$. Then, as previously demonstrated, Equation 115 holds for BP and Bz if and only if there exists a vector $\tilde{\mathbf{y}}'$ and a Boolean $\tilde{b}$ such that $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{\mathbf{v}} = \mathbf{0}$, which would imply that there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ if and only if Equation 115 holds for P and z .

Let $\tilde{z}$ and $\tilde{P}$ be such that $\tilde{z}_{\alpha_1} = z_{\alpha_1} \oplus z_{\alpha_2}$ and $\tilde{P}_{\alpha_1} = P_{\alpha_1} \oplus P_{\alpha_2}$ for some fixed α_1, α_2 satisfying $\alpha_1 \neq \alpha_2$ and $\tilde{z}_{\beta} = z_{\beta}$, $\tilde{P}_{\beta} = P_{\beta}$ for all β such that $\beta \neq \alpha_1$. Without loss of generality, we will assume that $\alpha_1 = 0$ and $\alpha_2 = 1$. Indeed, all the other possibilities can be reduced to this case simply by permuting the rows of the parity table P and the vector z . Let $\tilde{L}, \tilde{X}$ and $\tilde{\mathbf{v}}$ be constructed in the same way as L, X and $\mathbf{v}$ but with respect to $\tilde{P}$ and $\tilde{z}$. We will now prove that if there exists a vector $\mathbf{y}'$ and a Boolean b such that $L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0}$ then $\tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' \oplus \tilde{b}\tilde{\mathbf{v}} = \mathbf{0}$, where $\tilde{\mathbf{y}}'$ satisfies $\tilde{y}'_{\alpha_1 \alpha_3 \dots \alpha_k} = y'_{\alpha_1 \alpha_3 \dots \alpha_k} \oplus y'_{\alpha_2 \dots \alpha_k}$ and $\tilde{y}'_{\alpha_3 \dots \alpha_k} = y'_{\alpha_3 \dots \alpha_k}$ such that $\alpha_i \neq \alpha_1$ for all i . By using Equation 123 we can deduce that

$$\tilde{X}_{\alpha_3 \dots \alpha_k} \tilde{\mathbf{y}}' = \bigoplus_{i=3}^k \tilde{z}_{\alpha_i} \tilde{y}'_{\alpha_3 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} = \bigoplus_{i=3}^k z_{\alpha_i} y'_{\alpha_3 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} = X_{\alpha_3 \dots \alpha_k} \mathbf{y}' \quad (125)$$

for all $\alpha_3, \dots, \alpha_k$ satisfying $\alpha_2 < \alpha_3 < \dots < \alpha_k < n$ and k satisfying $3 \leq k \leq d+2$, and where the rows of $\tilde{y}'$ and y' are labelled in the same as the columns of $\tilde{X}$ and X . And from the definitions of the vectors $\tilde{v}$ and v , we can deduce that

$$\tilde{v}_{\alpha_3 \dots \alpha_k} = \bigwedge_{i=3}^k \tilde{z}_{\alpha_i} = \bigwedge_{i=3}^k z_{\alpha_i} = v_{\alpha_3 \dots \alpha_k} \quad (126)$$

for all $\alpha_3, \dots, \alpha_k$ satisfying $\alpha_2 < \alpha_3 < \dots < \alpha_k < n$ and k satisfying $3 \leq k \leq d+2$. Equations 125 and 126 imply that

$$\tilde{L}_{\alpha_3 \dots \alpha_k} \mathbf{y} \oplus \tilde{X}_{\alpha_3 \dots \alpha_k} \tilde{\mathbf{y}}' \oplus b \tilde{v}_{\alpha_3 \dots \alpha_k} = L_{\alpha_3 \dots \alpha_k} \mathbf{y} \oplus X_{\alpha_3 \dots \alpha_k} \mathbf{y}' \oplus b v_{\alpha_3 \dots \alpha_k} = 0 \quad (127)$$

for all $\alpha_3, \dots, \alpha_k$ satisfying $\alpha_2 < \alpha_3 < \dots < \alpha_k < n$ and k satisfying $3 \leq k \leq d+2$. Furthermore, we have

$$\begin{aligned} \tilde{L}_{\alpha_1 \dots \alpha_k} \mathbf{y} &\equiv \left| \bigwedge_{i=1}^k \tilde{P}_{\alpha_i} \wedge \mathbf{y} \right| \pmod{2} \\ &\equiv \left| (P_{\alpha_1} \oplus P_{\alpha_2}) \wedge \bigwedge_{i=2}^k P_{\alpha_i} \wedge \mathbf{y} \right| \pmod{2} \\ &\equiv \left| \bigwedge_{i=1}^k P_{\alpha_i} \wedge \mathbf{y} \right| + \left| \bigwedge_{i=2}^k P_{\alpha_i} \wedge \mathbf{y} \right| \pmod{2} \\ &\equiv L_{\alpha_1 \dots \alpha_k} \mathbf{y} + L_{\alpha_2 \dots \alpha_k} \mathbf{y} \pmod{2} \\ &\equiv X_{\alpha_1 \dots \alpha_k} \mathbf{y}' + b v_{\alpha_1 \dots \alpha_k} + X_{\alpha_2 \dots \alpha_k} \mathbf{y}' + b v_{\alpha_2 \dots \alpha_k} \pmod{2} \end{aligned} \quad (128)$$

for all $\alpha_3, \dots, \alpha_k$ satisfying $\alpha_2 < \alpha_3 < \dots < \alpha_k < n$ and k satisfying $3 \leq k \leq d$, which entails

$$\begin{aligned} &\tilde{L}_{\alpha_1 \dots \alpha_k} \mathbf{y} \oplus \tilde{X}_{\alpha_1 \dots \alpha_k} \tilde{\mathbf{y}}' \oplus b \tilde{v}_{\alpha_1 \dots \alpha_k} \\ &= X_{\alpha_1 \dots \alpha_k} \mathbf{y}' \oplus X_{\alpha_2 \dots \alpha_k} \mathbf{y}' \oplus \tilde{X}_{\alpha_1 \dots \alpha_k} \tilde{\mathbf{y}}' \oplus b v_{\alpha_1 \dots \alpha_k} \oplus b v_{\alpha_2 \dots \alpha_k} \oplus b \tilde{v}_{\alpha_1 \dots \alpha_k} \\ &= \bigoplus_{i=1}^k z_{\alpha_i} y'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus \bigoplus_{i=2}^k z_{\alpha_i} y'_{\alpha_2 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus \bigoplus_{i=1}^k \tilde{z}_{\alpha_i} \tilde{y}'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \\ &\quad \oplus b \bigwedge_{i=1}^k z_{\alpha_i} \oplus b \bigwedge_{i=2}^k z_{\alpha_i} \oplus b \bigwedge_{i=1}^k \tilde{z}_{\alpha_i} \\ &= \bigoplus_{i=1}^k z_{\alpha_i} y'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus \bigoplus_{i=2}^k z_{\alpha_i} y'_{\alpha_2 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus (z_{\alpha_1} \oplus z_{\alpha_2}) y'_{\alpha_2 \dots \alpha_k} \\ &\quad \oplus \bigoplus_{i=2}^k z_{\alpha_i} \tilde{y}'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus b \bigwedge_{i=1}^k z_{\alpha_i} \oplus b \bigwedge_{i=2}^k z_{\alpha_i} \oplus b \bigwedge_{i=2}^k z_{\alpha_i} \wedge (z_{\alpha_1} \oplus z_{\alpha_2}) \\ &= \bigoplus_{i=2}^k z_{\alpha_i} (y'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus y'_{\alpha_2 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k}) \oplus z_{\alpha_2} y'_{\alpha_2 \dots \alpha_k} \\ &\quad \oplus \bigoplus_{i=2}^k z_{\alpha_i} \tilde{y}'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus b \bigwedge_{i=1}^k z_{\alpha_i} \oplus b \bigwedge_{i=2}^k z_{\alpha_i} \oplus b \bigwedge_{i=1}^k z_{\alpha_i} \oplus b \bigwedge_{i=2}^k z_{\alpha_i} \\ &= \bigoplus_{i=2}^k z_{\alpha_i} (y'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus y'_{\alpha_2 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k}) \oplus z_{\alpha_2} y'_{\alpha_2 \dots \alpha_k} \oplus z_{\alpha_2} \tilde{y}'_{\alpha_1 \alpha_3 \dots \alpha_k} \\ &\quad \oplus \bigoplus_{i=3}^k z_{\alpha_i} \tilde{y}'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \\ &= \bigoplus_{i=2}^k z_{\alpha_i} (y'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus y'_{\alpha_2 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k}) \oplus z_{\alpha_2} y'_{\alpha_2 \dots \alpha_k} \oplus z_{\alpha_2} (y'_{\alpha_1 \alpha_3 \dots \alpha_k} \oplus y'_{\alpha_2 \dots \alpha_k}) \end{aligned} \quad (129)$$

$$\oplus y'_{\alpha_3 \dots \alpha_k}) \oplus \bigoplus_{i=3}^k z_{\alpha_i}(y'_{\alpha_1 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k} \oplus y'_{\alpha_2 \dots \alpha_{i-1} \alpha_{i+1} \dots \alpha_k}) \\ = 0.$$

Finally, we have

$$\begin{aligned} \tilde{L}_{\alpha_1} \mathbf{y} \oplus \tilde{X}_{\alpha_1} \tilde{\mathbf{y}}' \oplus b\tilde{v}_{\alpha_1} &= \tilde{L}_{\alpha_1} \mathbf{y} \oplus b\tilde{z}_{\alpha_1} \\ &= \tilde{L}_{\alpha_1} \mathbf{y} \oplus b(z_{\alpha_1} \oplus z_{\alpha_2}) \\ &\equiv |\tilde{P}_{\alpha_1} \wedge \mathbf{y}| + bv_{\alpha_1} + bv_{\alpha_2} \pmod{2} \\ &\equiv |(P_{\alpha_1} \oplus P_{\alpha_2}) \wedge \mathbf{y}| + bv_{\alpha_1} + bv_{\alpha_2} \pmod{2} \\ &\equiv |P_{\alpha_1} \wedge \mathbf{y}| + |P_{\alpha_2} \wedge \mathbf{y}| + bv_{\alpha_1} + bv_{\alpha_2} \pmod{2} \\ &= L_{\alpha_1} \oplus bv_{\alpha_1} \oplus L_{\alpha_2} \oplus bv_{\alpha_2} \\ &= 0. \end{aligned} \tag{130}$$

Thus, we proved that

$$L\mathbf{y} \oplus X\mathbf{y}' \oplus b\mathbf{v} = \mathbf{0} \implies \tilde{L}\mathbf{y} \oplus \tilde{X}\tilde{\mathbf{y}}' \oplus b\tilde{\mathbf{v}} = \mathbf{0} \tag{131}$$

which concludes the proof of Theorem 11. $\square$

6 $R_Z(\pi/2^d)$ -count upper bound in universal gate sets

In this section we demonstrate an upper bound achievable in polynomial time for the $R_Z(\pi/2^d)$ -count within a $\{\text{CNOT}, H, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit where d is a non-negative integer. Note that it corresponds to an upper bound for the number of T gates in a Clifford+ T circuit in the case where $d = 2$. Our proof for the upper bound will partially rest on the following lemma, which has already been proven with a different approach in Reference [43] for the case where $d = 2$.

Lemma 1. *Let d be a non-negative integer and let $U \in \mathcal{D}_{d+1}$ act on n qubits where $\mathcal{D}_{d+1}$ is the diagonal subgroup of the $(d+1)$ th level of the Clifford hierarchy. Then U can be decomposed into two unitary gates $U = \tilde{U}\tilde{U}'$ which can be found in polynomial time and such that $\tilde{U} \in \mathcal{D}_{d+1}$ is acting on $n-1$ qubits. Furthermore, we can find a $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit implementing $\tilde{U}' \in \mathcal{D}_{d+1}$ with no more than*

$$\sum_{i=0}^{d-1} \binom{n-1}{i} + 1 \tag{132}$$

$R_Z(\pi/2^d)$ gates in polynomial time.

Proof. Let P be a parity table associated with an implementation of U , let β be the qubit on which $\tilde{U}$ is not acting, and let $\tilde{P}'$ be a parity table such that

$$\left| \bigwedge_{i=1}^d \tilde{P}'_{\alpha_i} \right| \equiv \left| \bigwedge_{i=1}^d P_{\alpha_i} \wedge P_{\beta} \right| \pmod{2} \tag{133}$$

for all $\alpha_1, \dots, \alpha_d$ satisfying $0 \leq \alpha_1 \leq \dots \leq \alpha_d < n$ and $\alpha_i \neq \beta$ for all i . The row β of $\tilde{P}'$ can be ignored, and, as stated by Theorem 10, we can optimize the number of columns of $\tilde{P}'$ such that it is lower or equal to

$$\sum_{i=0}^{d-1} \binom{n-1}{i} \tag{134}$$

and still satisfies Equation 133. If we add a null column to $\tilde{P}'$ in the case where the number of columns of $\tilde{P}'$ is not equal to $|P_{\beta}| \pmod{2}$, and then set $\tilde{P}'_{\beta} = \mathbf{1}$, then we have

$$\left| \bigwedge_{i=1}^d \tilde{P}'_{\alpha_i} \wedge \tilde{P}'_{\beta} \right| = \left| \bigwedge_{i=1}^d \tilde{P}'_{\alpha_i} \right| \equiv \left| \bigwedge_{i=1}^d P_{\alpha_i} \wedge P_{\beta} \right| \pmod{2} \tag{135}$$

for all $0 \leq \alpha_1 \leq \dots \leq \alpha_d < n$ and the number of columns of $\tilde{P}$ is at most

$$\sum_{i=0}^{d-1} \binom{n-1}{i} + 1. \quad (136)$$

Finally, we can easily find a parity table $\tilde{P}$ such that $\tilde{P}_\beta = \mathbf{0}$ and

$$\left| \bigwedge_{i=1}^{d+1} \tilde{P}_{\alpha_i} \right| \equiv \left| \bigwedge_{i=1}^{d+1} \tilde{P}'_{\alpha_i} \right| + \left| \bigwedge_{i=1}^{d+1} P_{\alpha_i} \right| \pmod{2} \quad (137)$$

for all $0 \leq \alpha_1 \leq \dots \leq \alpha_d < n$, which implies its associated unitary gate $\tilde{U}$ doesn't act on qubit β . Thus, $\tilde{U}$ and $\tilde{U}'$ are forming a decomposition of U up to an operator V implementable over $\{\text{CNOT}, R_Z(2\pi/2^d)\}$ gate set: $U = \tilde{U}\tilde{U}'V$, where $\tilde{U}'$ is the unitary gate associated with the parity table $\tilde{P}'$. The operator V can then be merged with the unitary $\tilde{U}'$ to comply with the decomposition of U established in Lemma 1. $\square$

Based on Lemma 1 and Theorem 10, we can prove the following theorem which provides an upper bound for the number of $R_Z(\pi/2^d)$ gates in a $\{\text{CNOT}, H, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit. Note that in the case where $d = 2$, we get an upper bound of $(n+1)(n+2h)/2 + 1$ for the number of T gates in a Clifford+ T circuit, which is significantly better than the previously best-known ancilla-free upper bound of $\mathcal{O}(n^2h)$ [31], where n is the number of qubits and h is the number of Hadamard gates in the circuit.

Theorem 12. *Let U be a unitary gate acting on n qubits and implementable over the $\{\text{CNOT}, H, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ gate set where d is a non-negative integer and with h internal Hadamard gates such that*

$$U = C_f U_0 \prod_{i=1}^h [H_{\alpha_i} U_i] C_{init} \quad (138)$$

where C_f, C_{init} are implementable over the $\{\text{CNOT}, H, R_Z(2\pi/2^d)\}$ gate set, H_{α_i} denotes the Hadamard gate applied on some qubit α_i , and $U_0, \dots, U_h$ are implementable over the $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ gate set. Then we can find a implementation of U over the $\{\text{CNOT}, H, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ gate set, in polynomial time, and such that the number of $R_Z(\pi/2^d)$ gates is lower or equal to

$$\sum_{i=0}^d \binom{n}{i} + h \left(\sum_{i=0}^{d-1} \binom{n-1}{i} + 1 \right). \quad (139)$$

Proof. As stated by Lemma 1, the unitary U_h can be decomposed into two unitary gates: $U_h = \tilde{U}_h \tilde{U}'_h$, such that $\tilde{U}_h$ is not acting on qubit α_h , which imply that $\tilde{U}_h$ commutes with the h th internal Hadamard gate. Let $U_i \tilde{U}_{i+1} = \tilde{U}_i \tilde{U}'_i$ for all i satisfying $0 < i < h$ where $\tilde{U}_i \tilde{U}'_i$ is a decomposition of $U_i \tilde{U}'_{i+1}$ such as given by Lemma 1 where $\tilde{U}'_i$ is not acting on qubit α_i . Then Equation 138 is equivalent to

$$U = C_f U_0 \tilde{U}_1 \prod_{i=1}^h [H_{\alpha_i} \tilde{U}'_i] C_{init} \quad (140)$$

and, as stated by Lemma 1, we can find an implementation of $\tilde{U}'_i$ in polynomial time and with no more than

$$\sum_{k=0}^{d-1} \binom{n-1}{k} + 1 \quad (141)$$

$R_Z(\pi/2^d)$ gates, for all i satisfying $0 < i \leq h$. Moreover, as stated by Theorem 10, we can find an implementation of $U_0 \tilde{U}_1$ in polynomial time and with no more than

$$\sum_{i=0}^d \binom{n}{i} \quad (142)$$

$R_Z(\pi/2^d)$ gates. Thus, U can be implemented in polynomial time over the $\{\text{CNOT}, H, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ gate set and with a number of $R_Z(\pi/2^d)$ gates that is lower or equal to

$$\sum_{i=0}^d \binom{n}{i} + h \left(\sum_{i=0}^{d-1} \binom{n-1}{i} + 1 \right). \quad (143)$$

□

7 Conclusion

We presented several polynomial-time algorithms for reducing the number of T gates in a Clifford+ T circuit. Benchmarks show that our algorithms consistently achieve the lowest execution times and provide the best T -count reduction on almost all the quantum circuits evaluated when compared to state-of-the-art T -count optimizers. As such, our algorithms not only achieve state-of-the-art T -count reduction but also offer much greater scalability, thereby allowing efficient T -count optimization on larger quantum circuits.

We proved that our algorithms are producing a circuit in which the number of T gates is upper bounded by $(n^2 + n)/2 + 1$ when they are executed on a Hadamard-free circuit, where n is the number of qubits. It has been shown that there exists an asymptotic upper bound of $n^2/2 - 1$ [66]. The question of whether or not there exists a polynomial-time algorithm satisfying this asymptotic upper bound remains open.

We also demonstrated how the number of T gates in a Clifford+ T circuit can be optimized such that it is lower or equal to $(n + 1)(n + 2h)/2 + 1$ where n is the number of qubits and h is the number of internal Hadamard gates in the circuit, without using any ancillary qubits and in polynomial time. This reinforces the interdependence between the problems of optimizing the number of internal Hadamard gates and optimizing the number of T gates. As part of future work, it would be beneficial to more clearly determine the roles that internal Hadamard gates and ancillary qubits have in the T -count optimization problem.

Acknowledgments

We acknowledge funding from the Plan France 2030 through the projects NISQ2LSQ ANR-22-PETQ-0006 and EPIQ ANR-22-PETQ-007.

References

- [1] Robert Raussendorf, Jim Harrington, and Kovid Goyal. “Topological fault-tolerance in cluster state quantum computation”. *New Journal of Physics* **9**, 199 (2007).
- [2] Austin G Fowler, Ashley M Stephens, and Peter Groszkowski. “High-threshold universal quantum computation on the surface code”. *Physical Review A* **80**, 052312 (2009).
- [3] Michael E Beverland, Aleksander Kubica, and Krysta M Svore. “Cost of universality: A comparative study of the overhead of state distillation and code switching with color codes”. *PRX Quantum* **2**, 020341 (2021).
- [4] Austin G. Fowler. “Time-optimal quantum computation” (2013). [arXiv:1210.4626](https://arxiv.org/abs/1210.4626).
- [5] Matthew Amy, Dmitri Maslov, Michele Mosca, and Martin Roetteler. “A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits”. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **32**, 818–830 (2013).
- [6] Peter Selinger. “Quantum circuits of T-depth one”. *Physical Review A* **87**, 042302 (2013).
- [7] Matthew Amy, Dmitri Maslov, and Michele Mosca. “Polynomial-time T-depth optimization of Clifford+ T circuits via matroid partitioning”. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **33**, 1476–1489 (2014).

- [8] Nabila Abdessaied, Mathias Soeken, and Rolf Drechsler. “Quantum circuit optimization by Hadamard gate reduction”. In *Reversible Computation: 6th International Conference, RC 2014, Kyoto, Japan, July 10-11, 2014. Proceedings 6*. Pages 149–162. Springer (2014).
- [9] Philipp Niemann, Anshu Gupta, and Rolf Drechsler. “T-depth optimization for fault-tolerant quantum circuits”. In *2019 IEEE 49th International Symposium on Multiple-Valued Logic (ISMVL)*. Pages 108–113. IEEE (2019).
- [10] Vlad Gheorghiu, Michele Mosca, and Priyanka Mukhopadhyay. “A (quasi-) polynomial time heuristic algorithm for synthesizing T-depth optimal circuits”. *npj Quantum Information* **8**, 110 (2022).
- [11] Sergey Bravyi and Alexei Kitaev. “Universal quantum computation with ideal Clifford gates and noisy ancillas”. *Physical Review A* **71**, 022316 (2005).
- [12] Matthew Amy and Vlad Gheorghiu. “staq—A full-stack quantum processing toolkit”. *Quantum Science and Technology* **5**, 034016 (2020).
- [13] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, and Ross Duncan. “t|ket): a retargetable compiler for NISQ devices”. *Quantum Science and Technology* **6**, 014003 (2020).
- [14] Simon Martiel and Timothée Goubault de Brugière. “Architecture aware compilation of quantum circuits via lazy synthesis”. *Quantum* **6**, 729 (2022).
- [15] Daniel Gottesman. “The Heisenberg representation of quantum computers” (1998). [arXiv:quant-ph/9807006](https://arxiv.org/abs/quant-ph/9807006).
- [16] Sergey Bravyi and David Gosset. “Improved classical simulation of quantum circuits dominated by Clifford gates”. *Physical review letters* **116**, 250501 (2016).
- [17] Sergey Bravyi, Dan Browne, Padraic Calpin, Earl Campbell, David Gosset, and Mark Howard. “Simulation of quantum circuits by low-rank stabilizer decompositions”. *Quantum* **3**, 181 (2019).
- [18] Hammam Qassim, Hakop Pashayan, and David Gosset. “Improved upper bounds on the stabilizer rank of magic states”. *Quantum* **5**, 606 (2021).
- [19] Aleks Kissinger and John van de Wetering. “Simulating quantum circuits with ZX-calculus reduced stabiliser decompositions”. *Quantum Science and Technology* (2022).
- [20] Aleks Kissinger, John van de Wetering, and Renaud Vilmart. “Classical simulation of quantum circuits with partial and graphical stabiliser decompositions”. In *17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022)*. (2022).
- [21] A. Kitaev, A. Shen, and M. Vyalyi. “Correspondence between classical and quantum computation”. Page 60–65. American Mathematical Society. (2002).
- [22] CM Dawson and MA Nielsen. “The Solovay-Kitaev algorithm”. *Quantum Information and Computation* **6**, 81–95 (2006).
- [23] Peter Selinger. “Efficient Clifford+T approximation of single-qubit operators”. *Quantum Information & Computation* **15**, 159–180 (2015).
- [24] Vadym Kliuchnikov, Dmitri Maslov, and Michele Mosca. “Asymptotically optimal approximation of single qubit unitaries by Clifford and T circuits using a constant number of ancillary qubits”. *Physical review letters* **110**, 190502 (2013).
- [25] Neil J. Ross and Peter Selinger. “Optimal ancilla-free Clifford+T approximation of z-rotations”. *Quantum Information and Computation* **16**, 901–953 (2016).
- [26] Alex Bocharov, Martin Roetteler, and Krysta M Svore. “Efficient synthesis of probabilistic quantum circuits with fallback”. *Physical Review A* **91**, 052317 (2015).
- [27] Matthew B. Hastings. “Turning gate synthesis errors into incoherent errors”. *Quantum Information and Computation* **17**, 488–494 (2017).
- [28] Earl Campbell. “Shorter gate sequences for quantum computing by mixing unitaries”. *Physical Review A* **95**, 042306 (2017).

- [29] Vadym Kliuchnikov, Kristin Lauter, Romy Minko, Adam Paetznick, and Christophe Petit. “Shorter quantum circuits via single-qubit gate approximation”. *Quantum* **7**, 1208 (2023).
- [30] Vadym Kliuchnikov, Dmitri Maslov, and Michele Mosca. “Fast and efficient exact synthesis of single-qubit unitaries generated by clifford and T gates”. *Quantum Information & Computation* **13**, 607–630 (2013).
- [31] Matthew Amy and Michele Mosca. “T-Count Optimization and Reed–Muller Codes”. *IEEE Transactions on Information Theory* **65**, 4771–4784 (2019).
- [32] Niel de Beaudrap, Xiaoning Bian, and Quanlong Wang. “Techniques to Reduce $\pi/4$ -Parity-Phase Circuits, Motivated by the ZX Calculus”. *Electronic Proceedings in Theoretical Computer Science* **318**, 131–149 (2020).
- [33] Niel de Beaudrap, Xiaoning Bian, and Quanlong Wang. “Fast and Effective Techniques for T-Count Reduction via Spider Nest Identities”. In 15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020). Volume 158, pages 11:1–11:23. (2020).
- [34] Anthony Munson, Bob Coecke, and Quanlong Wang. “AND-gates in ZX-calculus: spider nest identities and QBC-completeness”. *Electronic Proceedings in Theoretical Computer Science* **340**, 230–255 (2021).
- [35] Witalis Domitrz. “On the Verge to Improve Technique of T-count Reduction via Spider Nest Identities”. Master’s thesis. University of Oxford. (2021).
- [36] Luke E Heyfron and Earl T Campbell. “An efficient quantum compiler that reduces T count”. *Quantum Science and Technology* **4**, 015004 (2018).
- [37] David Gosset, Vadym Kliuchnikov, Michele Mosca, and Vincent Russo. “An algorithm for the T-count”. *Quantum Information & Computation* **14**, 1261–1276 (2014).
- [38] Daniel Litinski. “A game of surface codes: Large-scale quantum computing with lattice surgery”. *Quantum* **3**, 128 (2019).
- [39] Ewout Van Den Berg and Kristan Temme. “Circuit optimization of Hamiltonian simulation by simultaneous diagonalization of Pauli clusters”. *Quantum* **4**, 322 (2020).
- [40] Michael J Bremner, Richard Jozsa, and Dan J Shepherd. “Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy”. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **467**, 459–472 (2011).
- [41] Vivien Vandaele, Simon Martiel, Simon Perdrix, and Christophe Vuillot. “Optimal Hadamard gate count for Clifford+ T synthesis of Pauli rotations sequences”. *ACM Transactions on Quantum Computing* **5**, 1–29 (2024).
- [42] Scott Aaronson and Daniel Gottesman. “Improved simulation of stabilizer circuits”. *Physical Review A* **70**, 052328 (2004).
- [43] Earl T Campbell and Mark Howard. “Unified framework for magic state distillation and multiqubit gate synthesis with reduced resource cost”. *Physical Review A* **95**, 022316 (2017).
- [44] Daniel Gottesman and Isaac L Chuang. “Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations”. *Nature* **402**, 390–393 (1999).
- [45] Christopher M Dawson, Henry L Haselgrove, Andrew P Hines, Duncan Mortimer, Michael A Nielsen, and Tobias J Osborne. “Quantum computing and polynomial equations over Z_2 ”. *Quantum Information and Computation* **5**, 102–112 (2005).
- [46] Ashley Montanaro. “Quantum circuits and low-degree polynomials over $\mathbb{F}_2$ ”. *Journal of Physics A: Mathematical and Theoretical* **50**, 084002 (2017).
- [47] Gadiel Seroussi and Abraham Lempel. “Maximum likelihood decoding of certain Reed-Muller codes”. *IEEE Transactions on Information Theory* **29**, 448–450 (1983).
- [48] Johan Håstad. “Tensor rank is NP-complete”. *Journal of Algorithms* **11**, 644–654 (1990).
- [49] John van de Wetering and Matt Amy. “Optimising T-count is NP-hard” (2023). [arXiv:2310.05958](https://arxiv.org/abs/2310.05958).

- [50] Matthew Amy. “Feynman”. <https://github.com/meamy/feynman>.
- [51] Dmitri Maslov. “Reversible Logic Synthesis Benchmarks page”. <http://webhome.cs.uvic.ca/~dmaslov>. Accessed February 2023.
- [52] Kyungbae Jang, Anubhab Baksi, Jakub Breier, Hwajeong Seo, and Anupam Chattopadhyay. “Quantum Implementation and Analysis of DEFAULT”. Cryptology ePrint Archive, Paper 2022/647 (2022). <https://eprint.iacr.org/2022/647>.
- [53] Francisco JR Ruiz, Tuomas Laakkonen, Johannes Bausch, Matej Balog, Mohammadamin Barekatin, Francisco JH Heras, Alexander Novikov, Nathan Fitzpatrick, Bernardino Romera-Paredes, John van de Wetering, et al. “Quantum circuit optimization with alphasensor” (2024). [arXiv:2402.14396](https://arxiv.org/abs/2402.14396).
- [54] Xiaoning Bian. “STOMP-code”. <https://github.com/onestruggler/stomp-code/tree/8df4f46228c2f413e0cf5f8b6d25c20b6460fc0e>.
- [55] https://github.com/VivienVandaele/quantum_circuit_optimization (2023).
- [56] Cody Jones. “Low-overhead constructions for the fault-tolerant Toffoli gate”. *Physical Review A—Atomic, Molecular, and Optical Physics* **87**, 022328 (2013).
- [57] Sergey Bravyi and Jeongwan Haah. “Magic-state distillation with low overhead”. *Physical Review A—Atomic, Molecular, and Optical Physics* **86**, 052329 (2012).
- [58] Matthew Amy, Parsiad Azimzadeh, and Michele Mosca. “On the controlled-NOT complexity of controlled-NOT–phase circuits”. *Quantum Science and Technology* **4**, 015002 (2018).
- [59] Vivien Vandaele, Simon Martiel, and Timothée Goubault de Brugière. “Phase polynomials synthesis algorithms for NISQ architectures and beyond”. *Quantum Science and Technology* **7**, 045027 (2022).
- [60] Vivien Vandaele, Simon Perdrix, and Christophe Vuillot. “Optimal number of parametrized rotations and Hadamard gates in parametrized Clifford circuits with non-repeated parameters” (2024). [arXiv:2407.07846](https://arxiv.org/abs/2407.07846).
- [61] Vivien Vandaele. “Quantum binary field multiplication with subquadratic Toffoli gate count and low space-time cost” (2025). [arXiv:2501.16136](https://arxiv.org/abs/2501.16136).
- [62] Peter W Shor. “Algorithms for quantum computation: discrete logarithms and factoring”. In Proceedings 35th annual symposium on foundations of computer science. Pages 124–134. Ieee (1994).
- [63] Guillaume Duclos-Cianci and David Poulin. “Reducing the quantum-computing overhead with complex gate distillation”. *Physical Review A* **91**, 042315 (2015).
- [64] Earl T Campbell and Mark Howard. “Magic state parity-checker with pre-distilled components”. *Quantum* **2**, 56 (2018).
- [65] Abraham Lempel. “Matrix Factorization over $GF(2)$ and Trace-Orthogonal Bases of $GF(2^n)$ ”. *SIAM Journal on Computing* **4**, 175–186 (1975).
- [66] Gérard D Cohen and Simon N Litsyn. “On the covering radius of Reed-Muller codes”. *Discrete Mathematics* **106**, 147–155 (1992).

A Proof of Theorem 4

Proof of Theorem 4. Analogously to Equation 22 and using Equations 37 and 38, we have

$$\begin{aligned}
& |P'_\alpha \wedge P'_\beta \wedge P'_\gamma| \\
& \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| + z_\gamma |P_\alpha \wedge P_\beta \wedge \mathbf{y}| + z_\beta |P_\alpha \wedge P_\gamma \wedge \mathbf{y}| + z_\alpha |P_\beta \wedge P_\gamma \wedge \mathbf{y}| \\
& \quad + z_\beta z_\gamma |P_\alpha \wedge \mathbf{y}| + z_\alpha z_\gamma |P_\beta \wedge \mathbf{y}| + z_\alpha z_\beta |P_\gamma \wedge \mathbf{y}| + z_\alpha z_\beta z_\gamma |\mathbf{y}| \pmod{2} \\
& \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| + |[z_\alpha(P_\beta \wedge P_\gamma) \oplus z_\beta(P_\alpha \wedge P_\gamma) \oplus z_\gamma(P_\alpha \wedge P_\beta)] \wedge \mathbf{y}| \pmod{2}
\end{aligned} \tag{144}$$

for all α, β, γ satisfying $0 \leq \alpha \leq \beta \leq \gamma < n$. In the case where $\alpha \neq \beta \neq \gamma$, we obtain

$$|P'_\alpha \wedge P'_\beta \wedge P'_\gamma| \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| \pmod{2} \tag{145}$$

by using Equation 39. In the case where $\alpha = \beta$, we obtain

$$\begin{aligned}
|P'_\alpha \wedge P'_\beta \wedge P'_\gamma| & \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| + |[z_\alpha(P_\alpha \wedge P_\gamma) \oplus z_\alpha(P_\alpha \wedge P_\gamma) \oplus z_\gamma P_\alpha] \wedge \mathbf{y}| \pmod{2} \\
& \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| \pmod{2}
\end{aligned} \tag{146}$$

by using Equation 38. Finally, in the case where $\alpha = \beta = \gamma$, we obtain

$$\begin{aligned}
|P'_\alpha \wedge P'_\beta \wedge P'_\gamma| & \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| + |[z_\alpha P_\alpha \oplus z_\alpha P_\alpha \oplus z_\alpha P_\alpha] \wedge \mathbf{y}| \pmod{2} \\
& \equiv |P_\alpha \wedge P_\beta \wedge P_\gamma| \pmod{2}
\end{aligned} \tag{147}$$

by using Equation 38. $\square$

B Proof of Theorem 7

Proof of Theorem 7. The CNOT and $R_Z(\theta)$ gates are acting as follows on basis states $|x_1, x_2\rangle$, $|x_1\rangle$:

$$\text{CNOT}|x_1, x_2\rangle = |x_1, x_1 \oplus x_2\rangle \tag{148}$$

$$R_Z(\theta)|x_1\rangle = e^{i\theta x_1}|x_1\rangle \tag{149}$$

Therefore, the action of an n -qubits $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit on a basis state $|\mathbf{x}\rangle$ has the form

$$|\mathbf{x}\rangle \mapsto e^{i\frac{\pi}{2^d}p(\mathbf{x})}|g(\mathbf{x})\rangle \tag{150}$$

where $g : \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2^n$ is a linear reversible Boolean function which can be implemented using only CNOT gates, and p is a linear combination of linear Boolean functions:

$$p(\mathbf{x}) = \sum_{j=1}^m a_j (y_1^{(j)} x_1 \oplus \dots \oplus y_n^{(j)} x_n) \pmod{2^{d+1}} \tag{151}$$

where $\mathbf{y}^{(j)} \in \mathbb{Z}_2^n \setminus \{\mathbf{0}\}$, $\mathbf{a} \in \mathbb{Z}_{2^\ell}^n$ and $m \geq 0$. The function p is called a phase polynomial, we will refer to the Boolean vectors $\mathbf{y}^{(j)}$ as the parities of the phase polynomial p and we will refer to $\mathbf{a}$ as the weights of the phase polynomial. Notice that if a weight a_j associated to a parity $\mathbf{y}^{(j)}$ is odd, then the associated rotation can be implemented using only the $R_Z(2\pi/2^d)$ gate. Therefore, the number of $R_Z(\pi/2^d)$ gates required to implement p is equal to $|\mathbf{a} \pmod{2}|$. That is why the problem of minimizing the number of $R_Z(\pi/2^d)$ gates in a $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ circuit consists in finding a phase polynomial p' equivalent to p but with weights $\mathbf{a}'$ such that $|\mathbf{a}' \pmod{2}|$ is minimal. We now prove the following equality, which will be useful to characterize the set of phase polynomials that are equivalent:

$$x_1 \oplus \dots \oplus x_n = \sum_{k=1}^n \sum_{\alpha_1 < \dots < \alpha_k} (-2)^{k-1} \prod_{i=1}^k x_{\alpha_i} \tag{152}$$

for all $n \geq 1$ and where $\mathbf{x} \in \mathbb{Z}_2^n$. This equality is trivially true for $n = 1$, and for $n = 2$ we can easily verify by case distinction that

$$x_1 \oplus x_2 = x_1 + x_2 - 2x_1x_2. \quad (153)$$

Let assume that the equality is true for n and let $\tilde{x}_1 = x_1 \oplus x_{n+1}$, then we have

$$\begin{aligned} x_1 \oplus \dots \oplus x_{n+1} &= \tilde{x}_1 \oplus \dots \oplus x_n \\ &= \tilde{x}_1 \left(1 + \sum_{k=2}^n \sum_{1 < \alpha_1 < \dots < \alpha_{k-1}} (-2)^{k-1} \prod_{i=2}^k x_{\alpha_i} \right) \\ &\quad + \sum_{k=1}^n \sum_{1 < \alpha_1 < \dots < \alpha_k} (-2)^{k-1} \prod_{i=1}^k x_{\alpha_i} \\ &= (x_1 \oplus x_{n+1}) \left(1 + \sum_{k=2}^n \sum_{1 < \alpha_1 < \dots < \alpha_{k-1}} (-2)^{k-1} \prod_{i=2}^k x_{\alpha_i} \right) \\ &\quad + \sum_{k=1}^n \sum_{1 < \alpha_1 < \dots < \alpha_k} (-2)^{k-1} \prod_{i=1}^k x_{\alpha_i} \\ &= (x_1 + x_{n+1} - 2x_1x_{n+1}) \left(1 + \sum_{k=2}^n \sum_{1 < \alpha_1 < \dots < \alpha_{k-1}} (-2)^{k-1} \prod_{i=2}^k x_{\alpha_i} \right) \\ &\quad + \sum_{k=1}^n \sum_{1 < \alpha_1 < \dots < \alpha_k} (-2)^{k-1} \prod_{i=1}^k x_{\alpha_i} \\ &= \sum_{k=1}^{n+1} \sum_{\alpha_1 < \dots < \alpha_k} (-2)^{k-1} \prod_{i=1}^k x_{\alpha_i} \end{aligned} \quad (154)$$

Moreover, the equality holds under the modulo of any even number, and so

$$\begin{aligned} x_1 \oplus \dots \oplus x_n &= \sum_{k=1}^n \sum_{\alpha_1 < \dots < \alpha_k} (-2)^{k-1} \prod_{i=1}^k x_{\alpha_i} \pmod{2^{d+1}} \\ &= \sum_{k=1}^{d+1} \sum_{\alpha_1 < \dots < \alpha_k} (-2)^{k-1} \prod_{i=1}^k x_{\alpha_i} \pmod{2^{d+1}} \end{aligned} \quad (155)$$

where d is a non-negative integer satisfying $d + 1 \leq n$. Then we have

$$\begin{aligned} p(\mathbf{x}) &= \sum_{j=1}^m a_j (y_1^{(j)} x_1 \oplus \dots \oplus y_n^{(j)} x_n) \pmod{2^{d+1}} \\ &= \sum_{j=1}^m a_j \sum_{k=1}^{d+1} \sum_{\alpha_1 < \dots < \alpha_k} (-2)^{k-1} \prod_{i=1}^k y_{\alpha_i}^{(j)} x_{\alpha_i} \pmod{2^{d+1}} \\ &= \sum_{k=1}^d \sum_{\alpha_1 < \dots < \alpha_k} (-2)^{k-1} c_{\alpha_1, \dots, \alpha_k} \prod_{i=1}^k x_{\alpha_i} \pmod{2^{d+1}} \end{aligned} \quad (156)$$

where $c_{\alpha_1, \dots, \alpha_k} = \sum_{j=1}^m a_j \prod_{i=1}^k y_{\alpha_i}^{(j)} \pmod{2^{d-k+1}}$. Notice that two phase polynomials p and p' with parities and weights $\mathbf{y}^{(j)}$, $\mathbf{a}$ and $\mathbf{y}'^{(j)}$, $\mathbf{a}'$ respectively are equal if and only if

$$\begin{aligned} c_{\alpha_1, \dots, \alpha_k} &= \sum_{j=1}^m a_j \prod_{i=1}^k y_{\alpha_i}^{(j)} \pmod{2^{d-k+1}} \\ &= \sum_{j=1}^{m'} a'_j \prod_{i=1}^k y_{\alpha_i}'^{(j)} \pmod{2^{d-k+1}} \\ &= c'_{\alpha_1, \dots, \alpha_k} \end{aligned} \quad (157)$$

for all $\alpha_1, \dots, \alpha_k$ satisfying $\alpha_1 < \dots < \alpha_k$. Moreover, p and p' are equivalent up to an operator implementable over the $\{\text{CNOT}, R_Z(2\pi/2^d)\}$ gate set if and only if

$$c_{\alpha_1, \dots, \alpha_k} \equiv c'_{\alpha_1, \dots, \alpha_k} \pmod{2} \quad (158)$$

for all $\alpha_1, \dots, \alpha_k$ satisfying $\alpha_1 < \dots < \alpha_k$. Let P and P' be the parity tables (constructed as explained in Section 2) associated with p and p' and such that the columns of P and P' are encoding the parities $\mathbf{y}^{(j)}, \mathbf{y}'^{(j)}$ associated with an odd weight a_j or a'_j . Then we have

$$\begin{aligned} c_{\alpha_1, \dots, \alpha_k} &\equiv \sum_{j=1}^m a_j \prod_{i=1}^k y_{\alpha_i}^{(j)} \pmod{2} \\ &\equiv \left| \bigwedge_{i=1}^k P_{\alpha_i} \right| \pmod{2} \end{aligned} \quad (159)$$

for all $\alpha_1, \dots, \alpha_k$ satisfying $\alpha_1 < \dots < \alpha_k$ and k satisfying $1 \leq k \leq d+1$. And so p and p' are equivalent up to an operator implementable over the $\{\text{CNOT}, R_Z(2\pi/2^d)\}$ gate set if and only if

$$\left| \bigwedge_{i=1}^k P_{\alpha_i} \right| \equiv \left| \bigwedge_{i=1}^k P'_{\alpha_i} \right| \pmod{2} \quad (160)$$

for all $\alpha_1, \dots, \alpha_k$ satisfying $\alpha_1 < \dots < \alpha_k$ and k satisfying $1 \leq k \leq d+1$. Let $\mathcal{A} \in \mathbb{Z}_2^{(n, \dots, n)}$ be a symmetric tensor of order $d+1$ such that

$$\mathcal{A}_{\alpha_1, \dots, \alpha_{d+1}} = \left| \bigwedge_{i=1}^{d+1} P_{\alpha_i} \right| \pmod{2} \quad (161)$$

for all $\alpha_1, \dots, \alpha_{d+1}$ satisfying $0 \leq \alpha_1 \leq \dots \leq \alpha_{d+1} < n$. Then p' is equivalent to p up to an operator implementable over the $\{\text{CNOT}, R_Z(2\pi/2^d)\}$ gate set if and only if

$$\mathcal{A}_{\alpha_1, \dots, \alpha_{d+1}} = \left| \bigwedge_{i=1}^k P'_{\alpha_i} \right| \pmod{2} \quad (162)$$

for all $\alpha_1, \dots, \alpha_{d+1}$ satisfying $0 \leq \alpha_1 \leq \dots \leq \alpha_{d+1} < n$. The number of $R_Z(\pi/2^d)$ gates required to implement p' over the $\{\text{CNOT}, R_Z(\pi/2^d), R_Z(2\pi/2^d)\}$ gate set is equal to the number of column of P' . Thus, minimizing the number of $R_Z(\pi/2^d)$ gates consists in finding a parity table P' satisfying Equation 162 with a minimal number of columns, which corresponds to the $(d+1)$ -STR problem. $\square$