Factoring with Qutrits: Shor's Algorithm on Ternary and Metaplectic Quantum Architectures

Alex Bocharov*, Martin Roetteler*, and Krysta M. Svore*
*Quantum Architectures and Computation Group,
Microsoft Research, Redmond, WA (USA)

We determine the cost of performing Shor's algorithm for integer factorization on a ternary quantum computer, using two natural models of universal fault tolerant computing on ternary quantum systems: (i) a model based on magic state distillation that assumes the availability of the ternary Clifford gates, projective measurements, classical control and (ii) a model based on a metaplectic topological quantum computer (MTQC). Arguably, a natural choice to implement Shor's algorithm on a ternary quantum computer is to translate the entire arithmetic into a ternary form. However, it is also possible to simply emulate the standard binary version of the algorithm by encoding each qubit in a three level system. In this paper we address this emulation approach and analyze the complexity of implementing Shor's period finding function in both models, (i) and (ii). We compare the costs in terms of magic state counts required in each mode and find that a binary emulation implementation of Shor's algorithm on a ternary quantum computer requires slightly smaller circuit depth than the corresponding implementation in the binary Clifford+T framework. The reason for this are simplifications for binary arithmetic that can be leveraged over ternary gate sets. We also highlight that magic state preparation on MTQC requires magic state preprocessor of asymptotically smaller size which gives the MTQC solution a significant advantage over the binary framework.

I. INTRODUCTION

Shor's quantum algorithm for integer factorization [28] is a striking case of the exponential speed-up promised by a quantum computer over the best-known classical algorithms. Since Shor's original paper, many explicit circuit constructions over qubits for performing the algorithm have been developed and analyzed. This includes the automated synthesis of the underlying quantum circuits for the binary case (see the following and references therein: [3, 4, 12, 20, 21, 30, 31, 33, 35]).

It has been previously noted that multi-valued logic beyond binary encoding may be a more natural embedding of some computations and potentially offer a cost savings. Experimental implementation of computation with ternary logic, for example with Josephson junctions, dates back to 1989 [22, 23]. More recently, multi-valued logic has been proposed for linear ion traps [24], cold atoms [29], and entangled photons [19]. In topological quantum computing it has been shown that metaplectic non-Abelian anyons [14] naturally align with ternary, and not binary, logic. These anyons offer a natively topologically protected universal set of quantum gates (see, for example, [25]), in turn requiring little to no quantum error correction.

Transferring the wealth of multi-qubit circuits worked out for quantum algorithms to a multi-qutrit framework is not straightforward. Some of the binary primitives, for example the binary Hadamard gate and the two-qubit CNOT gate, do not remain Clifford operations in the ternary case. Therefore, they cannot be emulated by ternary Clifford circuits. We resolve this complication by developing circuits for a *generic* ternary quantum computer. We extend the solution to the Metaplectic Topological Quantum Computer (MTQC) platform [14], which further reduces the cost of implementation.

A generic ternary framework that supports the full ternary Clifford group, measurement, and classical control [11], also supports a magic state distillation protocol that prepares the P_9 gate:

$$P_9 = \omega_9^{-1} |0\rangle\langle 0| + |1\rangle\langle 1| + \omega_9 |2\rangle\langle 2|, \; \omega_9 = e^{2\pi i/9}. \tag{1}$$

The Clifford+P_9 basis is universal for quantum computation and serves a similar functional role in ternary logic as the Clifford+T basis plays in binary logic.

Arguably, a natural choice to implement Shor's algorithm on a ternary quantum computer is to translate the entire arithmetic into a ternary form. In this paper we analyze a different encoding, however, namely the emulation of the standard binary version of the algorithm by encoding each qubit in a three level system. Despite the absence of a native Clifford CNOT gate in ternary logic, we find that binary integer arithmetic can be emulated efficiently on both generic ternary and metaplectic ternary quantum computers. In fact, the emulation is more efficient in terms of magic state consumption than the native binary integer arithmetic. In the case of a metaplectic ternary computer, relatively expensive magic state distillation is replaced by more streamlined ternary magic state preparation.

What is more, we demonstrate that Shor's period finding algorithm can be emulated with a constant cost reduction factor of roughly 6/7 in terms of clean magic state consumption compared to native binary implementations. We also show that on a metaplectic ternary computer the magic state preparation coprocessor is asymptotically smaller than a magic state distillation coprocessor, such as the one developed in [11] for the generic ternary quantum computer.

The cost benefits of using exotic non-Abelian anyonic frameworks for integer factorization has been previously noted, for example in [2], who estimated the benefits of using a hypothetical Fibonacci anyon framework over Ising anyons. It is worthwhile noting that neither binary nor ternary logic is irreducibly native to Fibonacci anyons, so the NOT, CNOT or Toffoli gates are much harder to emulate there than on a hypothetical metaplectic anyon computer. Another benefit of the MTQC is the ability to get rid of magic states (and hence of the magic state coprocessor) altogether by approximating all non-Clifford reflections directly to the required fidelity. This increases the depth of the emulated Shor's period finding circuit by a logarithmic factor, which is tolerable for the majority of instances.

The paper is organized as follows. In Section II we state the definitions and equations pertaining to the generic quantum computer, metaplectic topological quantum computer, and Shor's period finding function. In Section III we perform a detailed analysis of reversible classical circuits for modular exponentiation. One of the main goals is to show that the ternary non-Cliffordness of the CNOT introduces only a small overhead in ternary emulations of the classical circuits. The emulation cost is dominated by the cost of emulating the binary Toffoli gate. In Section IV we propose an efficient design for emulating the Toffoli gate either by a state injection circuit of constant depth or by a metaplectic circuit of logarithmic depth; the section concludes with important resource count comparisons.

We show that ternary emulation reduces the magic state consumption through Toffoli gates by a factor of roughly 6/7 compared to Clifford+T solution. When a depth $O(n^3)$ circuit based on ripple carry adders is acceptable, ternary emulation reduces the width of the modular exponentiation to a barebone minimum of $n + 3$ qutrits. However logarithmic depth circuits require additional ancillas to maintain the 6/7 emulation depth advantage factor due to the need to emulate several Toffoli gates in parallel; for example the use of carry lookahead adder requires up to n additional ancillary qutrit. We also compare the size of the magic state preparation coprocessor between the platforms and highlight the huge advantage of the metaplectic topological computer in this respect since the magic

state preparation is done at width linear in $\log(n)$ on an MTQC, where it appears to require width in $O(\log^3(n))$ on a generic ternary quantum computer and width in $O(\log(n)^{\log_3(15)}$ in binary case.

II. BACKGROUND AND NOTATION

A. Ternary Clifford group

Let $\{|0\rangle, |1\rangle, |2\rangle\}$ be the standard computational basis for a qutrit. Let $\omega_3 = e^{2\pi\, i/3}$ be the third primitive root of unity. The ternary *Pauli* group is generated by the *increment* gate

$$\mathrm{INC} = |1\rangle\langle 0| + |2\rangle\langle 1| + |0\rangle\langle 2| \tag{2}$$

and the ternary Z gate

$$Z = |0\rangle\langle 0| + \omega_3|1\rangle\langle 1| + \omega_3^2|2\rangle\langle 2|. \tag{3}$$

The ternary *Clifford* group stabilizes the Pauli group and is generated by INC, the ternary Hadamard gate H,

$$H = \frac{1}{\sqrt{3}} \sum \omega_3^{j\,k} |j\rangle\langle k|, \tag{4}$$

the Q gate

$$Q = |0\rangle\langle 0| + |1\rangle\langle 1| + \omega_3|2\rangle\langle 2|, \tag{5}$$

and the two-qutrit SUM gate,

$$\mathrm{SUM}|j,k\rangle = |j, j+k \bmod 3\rangle, j,k \in \{0,1,2\}. \tag{6}$$

Compared to the binary Clifford group, H is the ternary counterpart of the binary Hadamard gate, Q is the counterpart of the phase gate S, and SUM is an analog of the CNOT (although, intuitively it is a "weaker" entangler than CNOT, as described below).

For any n, ternary Clifford gates and their various tensor products generate a finite subgroup of $U(3^n)$; therefore they are not sufficient for universal quantum computation. We consider and compare two methods of building up quantum universality: by implementing the P_9 gate (1) and by expanding into the metaplectic basis (subsection II D). Given enough ancillae, these two bases are effectively and efficiently equivalent in principle (see Appendix A), and the costs in ancillae create practical tradeoffs depending on the given application.

B. Binary and ternary control

Given an n-qutrit unitary operator U there are different ways of expanding it into an $(n+1)$-qutrit unitary using the additional qutrit as "control". Let $|c\rangle$ be a state of the control qutrit and $|t\rangle$ be a state of the n-qutrit register. We now define

$$C_\ell(U)|c\rangle|t\rangle = |c\rangle \otimes (U^{\delta_{c,\ell}})|t\rangle,\ \ell \in \{0,1,2\},$$

wherein δ denotes the Kronecker delta symbol. We refer to this operator as *binary-controlled* unitary U and denote it in circuit diagrams as

For better readability we omit the label ℓ when $\ell = 1$. We also define the *ternary-controlled* extension of U by

$$\Lambda(U)|c\rangle|t\rangle = |c\rangle \otimes (U^c\,|t\rangle)$$

and denote it in circuit diagrams as

It is paramount to keep in mind that

$$\mathrm{SUM} = \Lambda(\mathrm{INC})$$

(see equations (2) and (6)). Another useful observation is that for any unitary U we have that $\Lambda(U) = C_1(U)\,(C_2(U))^2$.

The reader is encouraged to peruse our paper [7] for more detail on these concepts and notations.

C. The P_9 gate and its corresponding magic state

It is easy to see that the P_9 gate (1) is not a Clifford gate, e.g., it does not stabilize the ternary Pauli group. However, it can be realized by a certain deterministic measurement-assisted circuit given a copy of the *magic* state

$$\mu = \omega_9^{-1}|0\rangle + |1\rangle + \omega_9|2\rangle,\ \omega_9 = e^{2\pi\, i/9}. \tag{7}$$

An appropriate deterministic magic state injection circuit, as proposed in Ref. [11], is shown in Figure 1. For

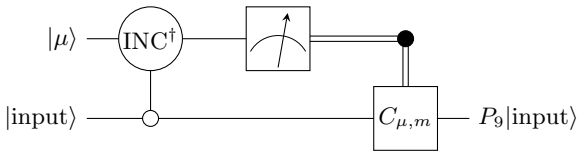

Figure 1: Exact representation of the P_9 gate by state injection. $C_{\mu,m}$ stands for a certain precompiled ternary Clifford gate, classically predicated by the measurement result m.

completeness, $C_{\mu,m} = (P_9\,\mathrm{INC}\,P_9^\dagger)^{-m}\,\mathrm{INC}^m$. Note that $P_9\,\mathrm{INC}\,P_9^\dagger$ is a Clifford gate, since P_9 is at level 2 of the ternary Clifford hierarchy (cf. [7]).

This magic state naturally exists in any multi-qudit framework with qudits of prime dimension [11]. When the framework supports the full multi-qudit Clifford group, projective measurements and classical control, then it also supports stabilizer protocols for magic state distillation based on generalized Reed-Muller codes. In particular, a multi-qutrit framework supports a distillation protocol that requires $O(\log^3(1/\delta))$ raw magic states of low fixed fidelity in order to distill a copy of the magic state μ at fidelity $1-\delta$. The distillation protocol is iterative and converges to that fidelity in $O(\log(\log(1/\delta)))$ iterations. The protocol performance is analogous to (albeit asymptotically somewhat less advanced than) the magic state distillation protocol for the T gate in the Clifford+T framework [9].

The magic state technique splits the actual computation into "online" and "offline" components where the main part of quantum processor runs the target quantum circuit whereas the (potentially rather large) "offline" coprocessor distills copies of a magic state that are subsequently injected into the main circuit by a deterministic widget of constant depth. Discussing the details of the distillation protocol for the magic state μ is beyond the scope of this paper and we refer the reader to [11].

D. Metaplectic quantum basis

The ternary *metaplectic* quantum basis is obtained by adding the *single qutrit axial reflection* gate

$$R_{|2\rangle} = |0\rangle\langle 0| + |1\rangle\langle 1| - |2\rangle\langle 2| \tag{8}$$

to the ternary Clifford group. It is easy to see that $R_{|2\rangle}$ is a non-Clifford gate and that Clifford+$R_{|2\rangle}$ framework is universal for quantum computation.

In Ref. [14] this framework has been realized with certain weakly integral non-Abelian anyons called *metaplectic anyons* which explains our use of the "metaplectic" epithet in the name of this universal basis. In Ref. [14], $R_{|2\rangle}$ is produced by injection of the magic state

$$|\psi\rangle = |0\rangle - |1\rangle + |2\rangle. \tag{9}$$

Unlike more familiar cases, the injection circuit is coherent probabilistic, succeeds in three iterations on average and consumes three copies of the magic state $|\psi\rangle$ on average.

For completeness we present the logic of the injection circuit on Figure 2. Each directed arrow in the circuit is labeled with the result of standard measurement of the first qutrit in the state $\mathrm{SUM}_{2,1}\,(|\psi\rangle \otimes |\mathit{input}\rangle)$. On $m = 0$ the sign of the third component of the input is flipped; on $m = 1, 2$ the sign of the first or second component respectively is flipped.

In the original anyonic framework the $|\psi\rangle$ state is produced by a relatively inexpensive protocol that uses topological measurement and consequent intra-qutrit projection (see [14], Lemma 5). This protocol requires only three qutrits and produces an exact copy of $|\psi\rangle$ in 9/4 trials on average. This is much better than any state distillation method, especially because it produces a copy of $|\psi\rangle$ with fidelity 1.

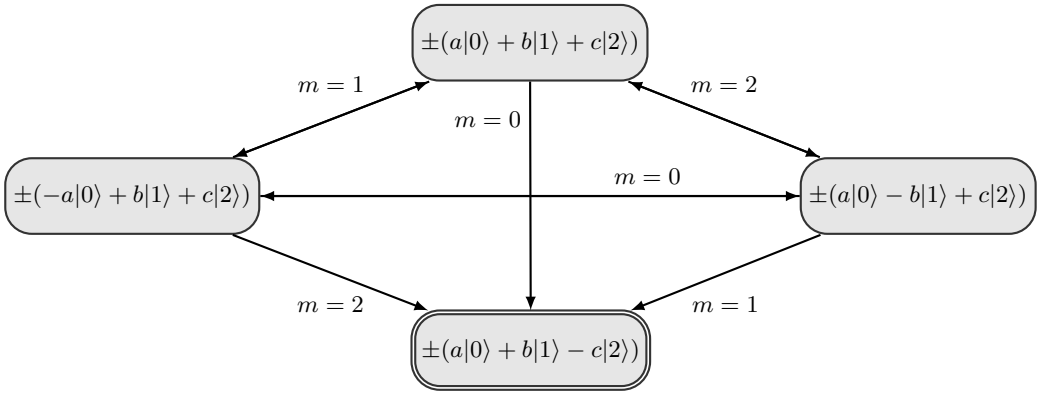

Figure 2: Markov chain for repeat-until-success implementation of the injection of the $R_{|2\rangle}$ gate [14]. Starting point is a general input $a|0\rangle + b|1\rangle + c|2\rangle$, where $a, b, c \in \mathbb{C}$. Arrows indicate transitions between single-qutrit states. Each arrow represent a single trial including measurement and consumption of the resource state $|\psi\rangle$, where each of the transitions is labeled with the measurement result. The absorbing state corresponds to successful implementation of the $R_{|2\rangle}$ gate and is denoted by double borders.

In [6] we have developed effective compilation methods to compile efficient circuits in the metaplectic basis Clifford+$R_{|2\rangle}$. In particular, given an arbitrary two-level Householder reflection r and a desired target precision ε, then r is effectively approximated by a metaplectic circuit of R-count at most $8\log_3(1/\varepsilon) + O(\log(\log(1/\varepsilon)))$, where R-count is the number of occurrences of non-Clifford axial reflections in the circuit. This allows us to approximate the CNOT and Toffoli gates very tightly and at low cost over the metaplectic basis (see Section IV B). Moreover if we wanted constant-depth high-fidelity widgets for CNOT and Toffoli we can do so by emulating, rather than distilling the magic state $|\mu\rangle$ of (7) by a metaplectic circuit and thus obtaining a high fidelity emulation of the P_9 gate at constant online depth (see Section IV A).

As we show in Appendix A, the converse also works. With available ancillas and enough reversible classical gates we can prepare the requisite magic state $|\psi\rangle$ exactly on a generic ternary computer. The particular method in the appendix is probabilistic circuit for the magic state $|\psi\rangle$ of (9) using the classical non-Clifford gate $C_2(\mathrm{INC})$. Our current method for the latter gate is to implement it as a ancilla-free circuit with three P_9 gates.

E. Top-level view of Shor's integer factorization algorithm

The polynomial-time algorithm for integer factorization originally developed in [28] is a hybrid algorithm that combines a certain quantum circuit with classical preprocessing and post-processing. In general, the task of factoring an integer can be efficiently reduced classically to a set of hard cases. A hard case of factorization problem comprises factoring a large integer N that is odd, square-free and composite.

Let a be a randomly picked integer that is relatively prime with N. By Euler theorem, $a^{\varphi(N)} = 1 \bmod N$, where φ is the Euler's totient function, and thus the modular exponentiation function $e_a : x \mapsto a^x \bmod N$ is periodic with period $\varphi(N) < N$. Let now $0 < r < N$ be a period of the $e_a(x)$ function ($e_a(x + r) = e_a(x), \forall x$) and suppose, additionally that r is even and $a^{r/2} \neq -1 \bmod N$. Then the $\gcd(a^{r/2} - 1, N)$ must be a non-trivial divisor of N. The greatest common divisor is computed efficiently by classical means and it can be shown that the probability of satisfying the conditions $r = 0 \bmod 2$ and $a^{r/2} \neq -1 \bmod N$ is rather high when a is picked at random. Therefore in Shor's algorithm a quantum circuit is only used for finding the small period r of $e_a(x)$ once an appropriate a has been randomly picked.

The quantum circuit in question consists of three stages:

1. Prepare quantum state proportional to the following superposition:

$$\sum_{k=0}^{N^2} |k\rangle |a^k \bmod N\rangle. \tag{10}$$

2. Perform in-place quantum Fourier transform of the first register.

3. Measure the first register.

The classical integer state j obtained as the result of measurement in step 3) can be "useful", in that the value of a small period r can be recovered from it by efficient classical post-processing or it can be not useful. We iterate steps 1) – 3) until a useful j is found and hence a value of r is recovered.

Shor has shown in [28] that the probability of obtaining a useful measurement in one of the iterations is in $\Omega(1/\log(\log(N)))$. Therefore with probability 1 we will succeed in finding a desired small period r in $O(\log(\log(N)))$ trials.

Given the known efficiency of quantum Fourier transform, most of the quantum complexity of this solution falls onto the step 1) where (10) is prepared. Specific quantum circuits for preparing this superposition had been proposed by a number of researchers (cf. [3, 4, 12, 20, 21, 30, 31, 33, 35]).

In the context of this paper we need to distinguish between two types of period-finding circuits. One type, as in Ref. [3], is width-optimizing that tend to use Draper-style approximate arithmetics. As a matter of tradeoff, such circuits interleave multiple quantum Fourier transform and inverse Fourier transform blocks into modular arithmetic circuits, which in practice leads to significant depth overhead. We forego the analysis of circuits of this type for the lack of space leaving such analysis for future research.

The second type of period-finding circuits are framed as exact reversible arithmetic circuits and efficient ternary emulation of such amounts to efficient emulation of CNOT and Toffoli gates, possibly after some peephole optimization of the circuit itself. We discuss two typical circuits of this kind in detail in Section III, and briefly touch upon a number of alternatives in Appendix C.

It is important to note that, with a couple of exceptions the multi-qubit designs for Shor state preparation have been assuming ideal CNOT and ideal Toffoli gates. Discussing the fidelity of the binary CNOT gate is outside the scope of this paper. However, we must note that, in Clifford+T framework, for one, the Toffoli gate is only as ideal as the T gate, which is to say - it is not. The question then is around the required fidelity of CNOT and Toffoli gates for the quantum period funding loop to work.

A fairly routine unitary argument proves that if the superposition (10) is prepared imperfectly, with fidelity $1-\varepsilon$ for some ε in $o(1/\sqrt{\log(\log(N))})$, then the probability of obtaining one of the "useful" measurements is going to be asymptotically the same as with the ideal superposition, i.e. in $\Omega(1/\log(\log(N)))$. (For completeness, we spell out the argument in Appendix B.) Therefore, if d is the depth of the corresponding quantum circuit preparing the state, then the bound on the required precision of the individual gates in the circuit can be in $o(1/(d\sqrt{\log(\log(N))}))$ in the context of Shor's algorithm.

In the rest of the paper we explore ternary emulations of certain binary period finding circuits and compare them to natively binary (and, more specifically, Clifford+T) implementations of these circuits. We will demonstrate that the fidelity of the emulated circuits is almost entirely defined by the fidelity of emulated Toffoli gates, and that the latter emulation can be done efficiently in ternary frameworks.

III. EMULATION OF MULTI-QUBIT ARITHMETIC ON GENERIC TERNARY QUANTUM COMPUTER

As follows from Remarks 4 and 6 in section IV A below, ternary emulation of binary Toffoli gate is reduced to ternary emulation of binary CNOT by ternary Clifford adjustment. Clearly the Toffoli gate does not allow a ternary Clifford emulation, and so neither does the CNOT. This puts ternary processing of binary data into apparent disadvantage compared to binary processing where CNOT is usually assumed to be relatively cheap. Our *thesis* here is that, in the context of Shor's factorization algorithm the apparent disadvantage does not amount to much.

To give this a mathematical form, let us agree to take into account only non-Clifford gates in either binary decomposition or ternary emulation of Shor's period-finding functions and let us agree to count a stack of non-Clifford gates performed in parallel in one time slice as a single *unit of non-Clifford depth*. Let us call the number of units of non-Clifford depth in a circuit the *non-Clifford depth* of the circuit.

Thesis 1. Given n-bit numbers a and N (where $n > 1$) and a reversible classical purely binary circuit of non-Clifford depth d that represents exactly the function $|x\rangle \mapsto |a^x \bmod N\rangle$, one can effectively produce an exact ternary emulation of an equivalent circuit with non-Clifford depth not exceeding $d\,(1 + O(1/\log(n)))$.

We are not going to offer a rigorous general proof of this thesis in this paper, but we will write specific proofs for two selected circuits representing the period-finding routine. In fact for these two circuits the overhead factor does not exceed $1 + 3/(4\,\log_2(n))$.

Throughout the rest of the paper we use the following

Definition 2. *For integer $n > 0$ let $|j\rangle$, $|k\rangle$ be two different standard basis vectors in the n-qudit Hilbert space. We*

call the classical gate

$$\tau_{|j\rangle,|k\rangle} = I^{\otimes n} - |j\rangle\langle j| - |k\rangle\langle k| + |j\rangle\langle k| + |k\rangle\langle j| \tag{11}$$

a two-level axial reflection *in n qudits.*

As a motivation for this term, note that $\tau_{|j\rangle,|k\rangle}$ can be rewritten as the two-level Householder reflection

$$I^{\otimes n} - 2\,|u\rangle\langle u|, |u\rangle = (|j\rangle - |k\rangle)/\sqrt{2}.$$

Clearly, in binary, the CNOT, the Toffoli and any variably controlled Toffoli gate is a two-level axial reflection in the corresponding number of dimensions.

A. Ternary circuit for binary ripple carry additive shift

Here we discuss emulating an additive shift circuit based on quantum ripple carry adder proposed in [13]. That adder has simple structure and requires only two ancillas. In the context of Shor's algorithm it would be very practical to use the adder when factorizing medium-size numbers. However, since for n-bit numbers the depth of the circuit is $O(n)$, it is asymptotically inferior to lookahead additive shift described in the next subsection and may become unusable in practice when factoring very large numbers. Either way, the simpler ripple carry adder gives an excellent opportunity to introduce the emulation concepts.

Let a be a classically known n-bit integer and b be a quantumly-stored n-qubit basis state. We are looking for a quantum implementation of the function $|b\rangle \mapsto |a+b\rangle$. More specifically, we are looking for a pre-compiled quantum circuit C_a parameterized by a which is known at compilation time. Consider the original quantum ripple carry adder from [13] (in particular, the circuit illustrated on Figure 4 for $n = 6$ there that is copied, for completeness into our Fig. 3).

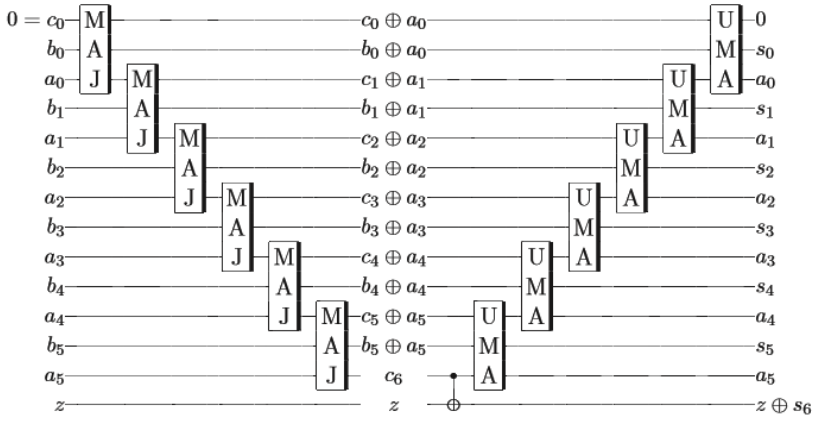

Figure 3: A simple ripple carry adder from [13] for $n = 6$.

The adder uses $2n + 2$ qubits. It performs a ladder of n MAJ gates to compute all the carry bits, including the top one. The top carry bit is copied unto the last qubit and the ladder of n UMA gates is performed. Each UMA gate uncomputes the corresponding MAJ function and performs the three-way $\mathbb{Z}_2$ addition $a_i \oplus b_i \oplus c_i$.

It is somewhat hard to fold in the classically known a in the multi-qubit framework using this design. Note however, that a solution along these line is offered in [30]. However it is easy to fold in a in ternary emulation using the third basis state of the qutrit. We are going to show that it takes exactly $n+2$ qutrits to emulate the binary shift $|b\rangle \mapsto |a+b\rangle$ following the above design. It is helpful for our purposes that the mod 2 additive shifts $b_i\oplus$ and $a_i\oplus$ can be taken out of the UMA blocks, pushed into the end of the circuit and all the additive shifts can be done in parallel in depth at most two there.

Consider $n+2$ qutrits where the top and the bottom ones are prepared in $|0\rangle$ state and the remaining n encode $|b\rangle$. Consider the two two-qutrit gates parameterized by $\mathbb{Z}_2$: $Y_0 = \tau_{|01\rangle,|20\rangle}$ and $Y_1 = \tau_{|10\rangle,|21\rangle}$. Assuming a, b and all the carries are encoded as binary numbers, it is easily verified by direct computation that $Y_{a_j}\,|c_j, b_j\rangle = |c'_j, c_{j+1}\rangle$, where

c_{j+1} is the correct carry bit for the $a_j + b_j + c_j$ addition and c'_j is a trit that can assume the value of 2 in two out of the eight cases.

Since all the bits of a are known we can precompile a ladder of Y gates that correctly computes the top carry bit c_n and puts the modified carry trit c'_j on each b_j wire. Having copied c_n onto the last qutrit, we sequentially undo the Y gates (note that each Y gate is self-inverse) in lockstep with computing partial $\mathbb{Z}_2$-sums $b_j \oplus c_j$ on all the b_j wires using CNOT gates. It remains to perform the $a_j\oplus$ additions. Again the effect of these can be precompiled. We need to put the NOT gate on the b_j wire if and only if $a_j = 1$. All these NOT gates are performed in parallel in one time slice.

Recall that CNOT gate cannot be exactly represented by a ternary Clifford circuit. Therefore, comparing our ternary solution to the original binary template in [13] we see that we have acquired n additional non-Clifford gates, compared to the template. This separation all but disappears however when we consider controlled versions of the additive shifts.

In order to control the shift in either of the two layouts it suffices to control only the bitwise addition gates. In both binary and ternary solutions adding one level of control produces n additional Toffoli gates and $\mathrm{w}(a)$ additional CNOT gates (that can be all stacked in parallel in one time slice). Thus the ternary emulation overhead caused by the cost of the CNOT reduces to $\mathrm{w}(a)$ non-Clifford gates all performed in parallel. Here $\mathrm{w}(a)$ is the Hamming weight of a.

In our terminology, in terms of *non-Clifford depth* the overhead factor due to ternary emulation does not exceed $(1+1/(3\,n))$. Since $(1+1/(3\,n)) < (1+3/(4\,\log_2(n)))$ we have illustrated thesis 1 in the context of this circuit design.

When two or more levels of control are added to the shift circuit the number of non-Clifford gates in purely binary framework is exactly equal to the number of non-Clifford gates used by the ternary emulation circuit described above. We also note that the width of the ternary emulation circuit is equal to $n + 2$ qutrits, whereas the original purely binary design appears to require $2\,n + 2$ qubits.

B. Circuit for binary carry lookahead additive shift

In this subsection we review and optimize a circuit for the carry lookahead additive shift based on the in-place binary carry lookahead adder (in place BCLAA) of the [16]. BCLAA for two n-bit numbers is a classical reversible circuit with the depth $O(\log(n))$ and $O(n)$ clean ancillary qubits. It computes all the carry bits in $O(\log(n))$ steps and performs all the bitwise summations in parallel. In order to do that, BCLAA computes a binary tree of "carry status indicators" (CSIs) that are stored on some of the ancillary qubits. The notion and formal definition of CSI are given in section 3 of [16] and specifically on Figures 2 and 3 there. An example of in-place BCLAA for $n = 10$ is presented on Figure 5 which we copy here for completeness (our Fig. 4).

As easily seen, $O(\log(n))$ steps in that circuit are spent computing the CSIs, while only $O(1)$ steps compute (and, occasionally, uncompute) the bitwise sums. For us it is important that the $O(\log(n))$ CSI time steps are populated with the (non-Clifford) Toffoli gates and only $O(1)$ time steps are populated with gates that are Clifford in binary framework but may be non-Clifford in ternary framework. From this it is already clear that the non-Clifford inefficiency factor due to ternary processing is upper-bounded by $(1 + O(1/\log(n)))$. Let us make sure that this bound can be practically specified and, in fact, improved on when folding in the classically known shift value a resulting in the desired circuit for $|b\rangle \mapsto |a + b\rangle$.

The purpose of constant folding is to remove all the wires (qubits) that contain the encoded values of $a_0, \ldots, a_{n-1}$ from the circuit by pre-compiling all the gates where these wires serve as control wires. Thus a Toffoli gate on $\{A_i, B_i, \mathrm{ancilla}\}$ resolves into identity when $a_i = 0$ or into CNOT on $\{B_i, \mathrm{ancilla}\}$ when $a_i = 1$; and CNOT on $\{A_i, B_i\}$ resolves into identity when $a_i = 0$ or into NOT on B_i otherwise. Since none of the Toffoli gates in the $P^{\pm1}, G^{\pm1}, C$ rounds are controlled by any of the A_is, then these Toffoli gates are unaffected by the constant folding; the Toffoli gates that can fold into CNOT occur only in the opening step (that computes $C_{i,i+1}$) and its mirror counterpart; on the other hand the constant folding immediately converts CNOT gates in three distinct time steps into identities or NOTs. Thus the number of time steps where CNOT gates can occur has decreased by 1 and stands at 3 so far.

For further practical benefit let us transform the circuit we have obtained so far somewhat by removing uncontrolled NOT gates from the second half (the ancilla clean-up part) of the folded circuit. To this end we will allow some of the Toffoli gates proper to morph into variably controlled Toffoli gates. If $\gamma_1, \gamma_2 \in \mathbb{Z}_2$, let $\mathrm{Tof}^{[\gamma_1,\gamma_2]} : |c_1, c_2, t\rangle \mapsto |c_1, c_2, t \oplus (\delta_{c_1,\gamma_1}\delta_{c_2,\gamma_2})\rangle$. As we already noted such variably controlled Toffoli gate is Clifford-equivalent to the regular Toffoli gate in both binary and ternary frameworks.

Obviously $(\mathrm{NOT}\otimes I\otimes I)\mathrm{Tof}^{[\gamma_1,\gamma_2]} = \mathrm{Tof}^{[\sim\gamma_1,\gamma_2]}(\mathrm{NOT}\otimes I\otimes I)$ and $(I\otimes\mathrm{NOT}\otimes I)\mathrm{Tof}^{[\gamma_1,\gamma_2]} = \mathrm{Tof}^{[\gamma_1,\sim\gamma_2]}(I\otimes\mathrm{NOT}\otimes I)$ Thus all the uncontrolled NOT gates in the second half of the folded circuit can be commuted through the control sites of the Toffoli gates possibly morphing the Toffoli gates into variably controlled Toffoli gates. In order to illustrate

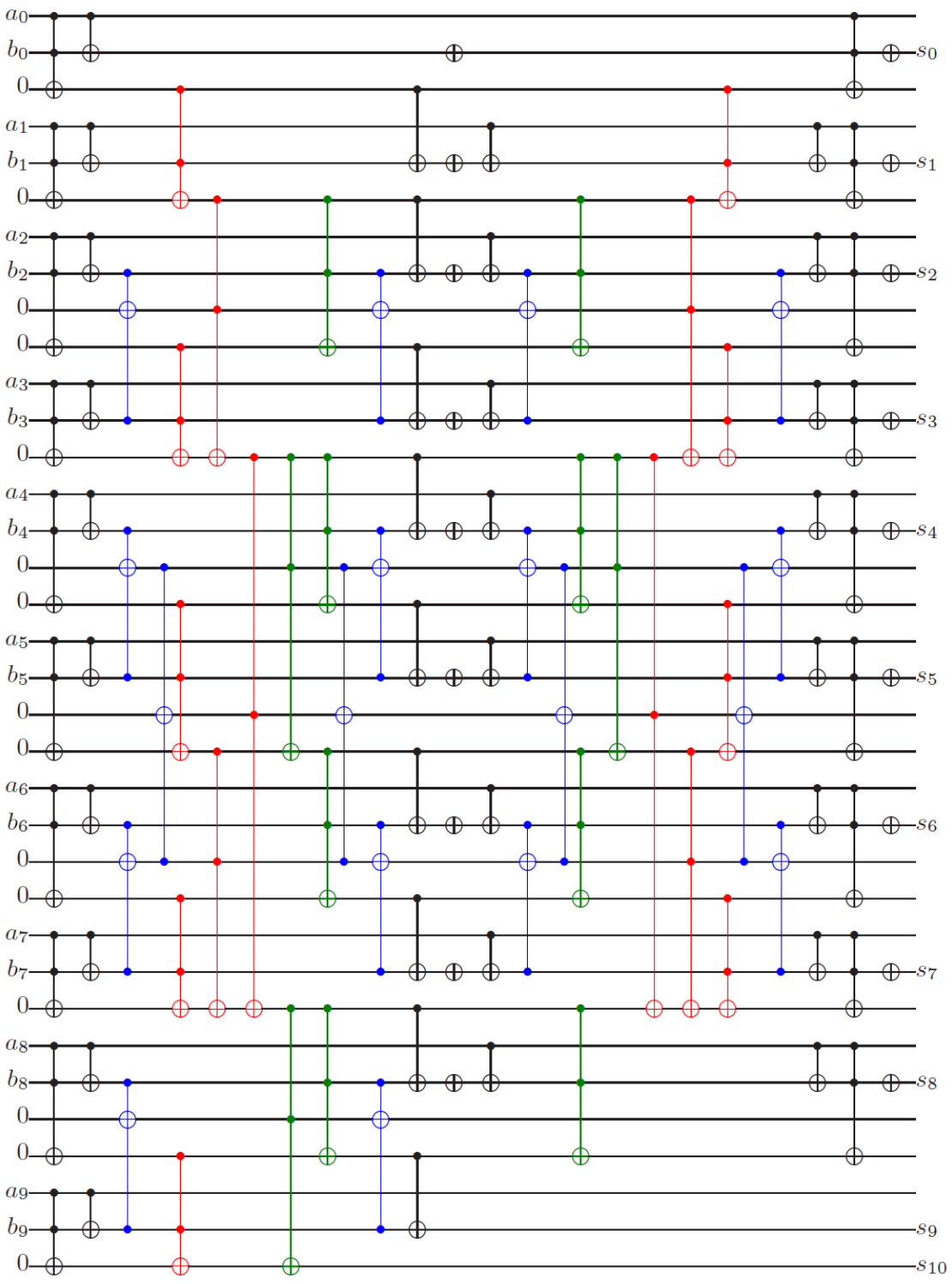

Figure 4: In-place QCLA adder for 10 bits from [16]. The carry status indicators are computed and uncomputed by the P, G (resp. P^{-1}, G^{-1}) rounds shown in blue and red. The concluding C rounds are shown in green.

this analysis, in Figure 5 we show a folded precompiled version of the circuit from Fig. 4 for specific constant $a_0 \dots a_9 = 1010010011$.

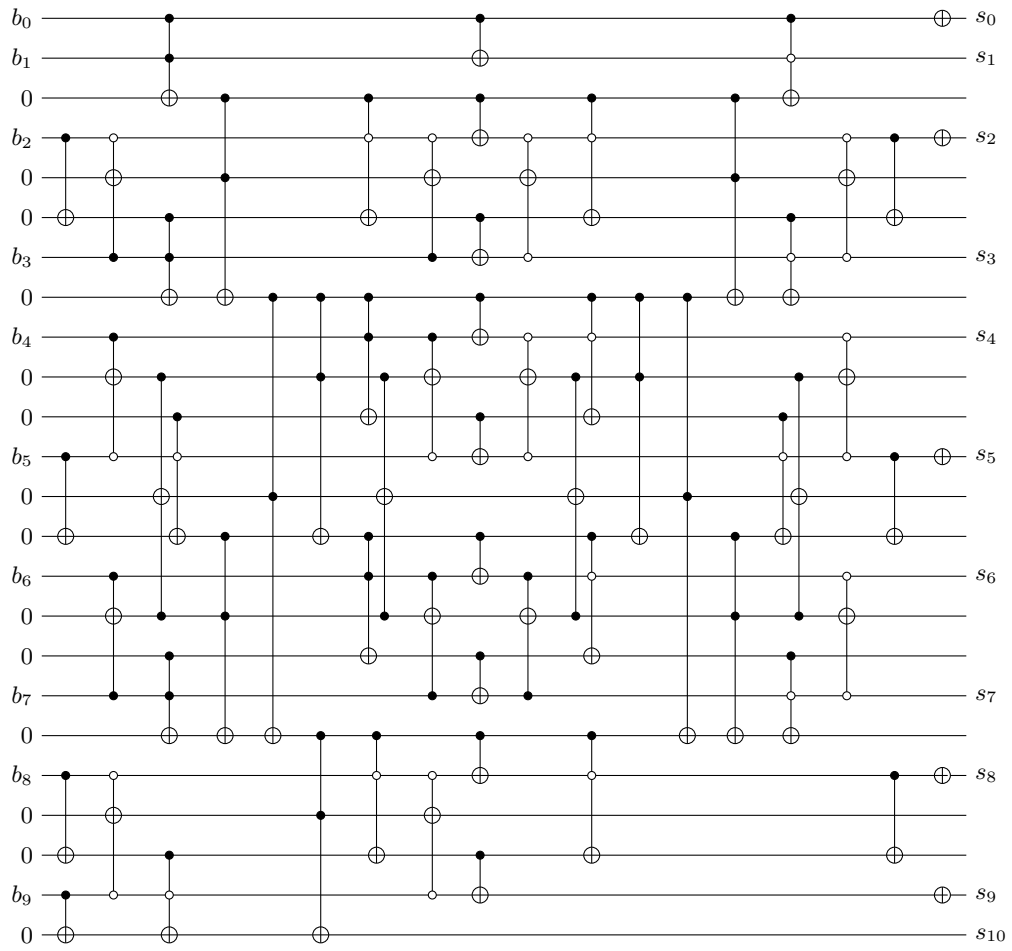

Figure 5: In-place QCLA additive shift, precompiled for the specific 10-bit shift value $a_0 \dots a_9 = 1010010011$.

The $a_j\oplus$ shift necessary for producing the bits of the sum of $a+b$ can be precompiled as a sequence of $w(a)$ NOT gates, where $w(a)$ is the Hemming weight of a, that can be done in parallel as the last step of the circuit. The remaining CNOTs can be corralled into at most three time slices. As per [16], section 4.2 the non-Clifford depth of the circuit exceeds $4\log_2(n)$, therefore CNOT layers which amount to at most $3/(4\log_2(n))$ fraction of the circuit depth. But it is important to tally the CNOT layers in the binary controlled version of the circuit.

For optimal size we need to control only those parts of the circuit that compute the bits of the $a+b$. Those are limited to one layer of CNOT gates and the final layer of NOT gates. We observe that adding a layer of control in this fashion, the CNOT gates turn into Toffoli gates and the NOT gates turn into CNOT gates. Thus the number of layers populated solely by CNOTs does not change. In summary the ternary emulation of either uncontrolled or controlled version of the additive shift circuit inflates the non-Clifford depth by a factor not exceeding $(1+3/(4\log_2(n)))$

C. Circuits for comparison to a classical threshold

Let a be a classically known integer and b a quantumly encoded integer. Assume, for simplicity that both are n-bit integers. We want to implement a Boolean function that returns 1 when $b<a$ and returns 0 otherwise. We observe that for $b'=2^n-1-b$ the result of such comparison function is equal to the top carry bit for the sum $b'+a$. Therefore the implementation is achieved by modifying the classical shift circuit. Before doing anything further, the comparison circuit applies NOT gates in parallel to all the B_i wires (correspondingly, at the very end of the circuit those NOTs are undone, also in parallel.)

The circuit proceeds to compute all the carries, including, most importantly, the top one. The top carry is copied onto the result ancilla. Then we uncompute all the carries, but skip the actual bitwise summation on the B_i wires. This way we end up with the A and B wires and all the clean carries restored to the original state. And the result

ancilla in the desired result state. The details differ somewhat between the ripple carry additive shift and the carry lookahead shift templates. In the former it simply suffices to literally skip the bitwise summation gates. This actually eliminates uncontrolled CNOT gates in the singly-controlled version of the circuit. Thus singly controlled comparison circuit ends up being populated with Toffoli gates only.

In the carry lookahead template, we must be cognizant that the $a_i \oplus b_i$ part of the bitwise addition is dual purpose and cannot be dropped; it must be undone before the end of the circuit (which, again, is executed in parallel in one time slice). But the time slice performing all the $\oplus c_i$ bitwise additions can be just dropped. This reduces the non-Clifford depth of the comparison circuit by 1 compared to the carry lookahead additive shift. It follows that in ternary emulation the non-Clifford depth of the circuit would get inflated by a factor not exceeding $(1 + 2/(4\ \log_2(n))) < (1 + 3/(4\ \log_2(n)))$.

D. Circuits for modular additive shifts

In this subsection we review a layout for modular additive shift and binary controlled additive shift and observe that, in the latter case the non-Clifford overhead guarantees derived in the previous subsections still hold. Let $N >> 0$ and $a < N$ be classically known integers. The commonly used scheme to compute the quantum modular additive shift $|b\rangle \mapsto |(a+b) \bmod N\rangle$ is to compute $|a+b\rangle$, figure out whether $a+b < N$ and, if not, then subtract N. In order to do it coherently without measurement we need to

1. Speculatively compute the $|(a-N)+b\rangle$ shift; structure it so that the top carry bit c_{n+1} is 1 iff $(a-N)+b < 0$.

2. Copy c_{n+1} to a clean ancilla x,

3. Apply the shift by $+N$ controlled by the ancilla x,

4. Clean up the ancilla x.

Surprisingly, the last step is less than trivial. We need to compare the encoded integer $|y\rangle$ after step 3) to a. Then $y \geq a$ if and only if $c_{n+1} = 1$. Therefore we must flip the ancilla if and only if $y \geq a$. We do this by taking the comparison circuit as described in the previous subsection and wiring the NOT x into it in place of the top carry qubit. It is easy to see that performing the comparison circuit has the exactly the desired effect on the ancilla x. A top level layout of the modular additive shift is shown in Figure 6. We note that the three-stage layout shown in the Figure is not entirely new. It is very similar to designs proposed in [31] and [30]. Clearly the non-Clifford depth of this scheme is roughly triple the non-Clifford depth of the additive shift circuit in either binary or ternary framework.

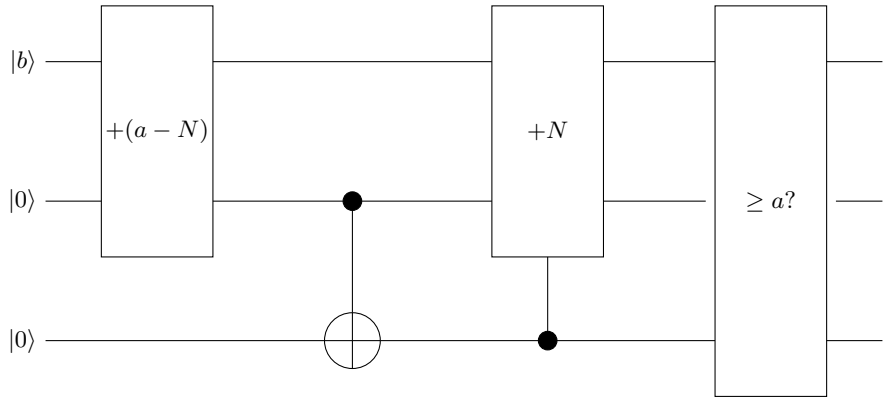

Figure 6: Top-level layout of modular additive shift.

In ternary framework we propose that each of the three components of the modular shift circuit described above emulates the corresponding binary component as described in the preceding sections. Given this design, the non-Clifford depth overhead factor due to ternary emulation (compared to binary solution) is asymptotically insignificant when $n \to \infty$ and in practice does not exceed $(1 + 3/(4\ \log_2(n)))$. The same applies to the controlled versions of the modular shift circuit.

E. Circuits for modular exponentiation

For modular exponentiation $|k\rangle|1\rangle \mapsto |k\rangle|a^k \bmod N\rangle$ we follow the known implementation scheme proposed in the first half of [35]. Our designs are also motivated in part by [12].

Suppose that a, N are classically known integers $a < N$. Suppose $|k\rangle$ is quantumly encoded, $k = \sum_{j=0}^{2\,n-1} k_j\, 2^j$ is binary expansion of k, where k_j are the corresponding qubit states. First, we observe that

$$a^k \bmod N = \prod_{j=0}^{2\,n-1} (a^{2^j} \bmod N)^{k_j} \bmod N. \tag{12}$$

Note that $(a^{2^j} \bmod N)$ are $2\,n$ are classical values that are known and easily pre-computable at compilation time. Thus $|a^k \bmod N\rangle$ is computed as a sequence of modular multiplicative shifts, each quantumly controlled by the $|k_j\rangle$.

Suppose we have computed the partial product

$$p_{k,m} = \prod_{j=0}^{m} (a^{2^j} \bmod N)^{k_j} \bmod N,$$

and let

$$p_{k,m} = \sum_{\ell=0}^{n-1} p_{k,m,\ell} 2^\ell$$

be the binary expansion of $p_{k,m}$. Then

$$p_{k,m+1} = \sum_{\ell=0}^{n-1} p_{k,m,\ell} (2^\ell a^{2^{m+1}} \bmod N)^{k_{m+1}} \bmod N.$$

Observe, again, that $\{(2^\ell a^{2^{m+1}} \bmod N)\}$ is the set of fewer than $2\,n^2$ pre-computable classical values known a priori. Therefore, promoting $p_{k,m}$ to $p_{k,m+1}$ is performed as a sequence of modular additive shifts, doubly-controlled by $p_{k,m,\ell}$ and k_{m+1}. Thus it takes $2\,n^2$ doubly-controlled modular additive shifts to perform the modular exponentiation $|k\rangle|1\rangle \mapsto |k\rangle|a^k\rangle$.

Our approach here is different from Zalka [35]. Instead of looking for high-width optimizations of modular multiplication, for larger n we simply use the carry lookahead circuits for the additive shifts and threshold comparisons. The non-Clifford depth of these circuits is in $O(\log(n))$ and, in practice, is roughly $4\,\log_2(n)$. This yields the non-Clifford depth of roughly $12\,\log_2(n)$ for a modular additive shift and overall non-Clifford depth of roughly $24\,n^2\,\log_2(n)$ for the modular exponentiation. This is asymptotically more costly than the fastest of the solutions in [35], but allows for keeping the width of the circuit well in check. Overall, the circuit uses roughly $4\,n$ qubits.

We perform ternary emulation of the circuit by doing component-wise ternary emulations for all the additive shifts and threshold comparisons as discussed above. Consequently, the overall non-Clifford depth overhead in ternary emulation is bounded by the overhead factor not exceeding $(1 + 3/(4\,\log_2(n))$. Note that for medium size values of n we can afford using ripple carry circuits for additive shifts and threshold comparison. This leaves the non-Clifford depth of binary solution at approximately $12\,n^3$ while allowing for the circuit layout with approximately n fewer qutrits. As shown in subsection III A, the overhead factor due to ternary emulation does not exceed $(1 + 1/(3\,n))$.

F. Circuits for quantum Fourier transform

In the solutions for period finding functions discussed so far, the quantum cost is dominated by the cost of modular exponentiation represented by an appropriate reversible classical circuit. In this context just a fraction of the cost falls onto the quantum Fourier transform. Nevertheless, for the sake of completeness we discuss some designs for emulating binary quantum Fourier transform on ternary computers.

A familiar binary circuit for approximate Fourier transform in dimension 2^n with precision δ consists of $\Theta(n\,\log(1/\delta))$ controlled phases and n binary Hadamard gates (see [26], Section 5). In known fault-tolerant binary frameworks, the phases $e^{\pi\,i/2^k}, k \in \mathbb{Z}$ occurring in the Fourier transform have to be treated just like generic phases. Of all the possible ways to emulate a controlled phase gate we will focus on just one with minimal parametric cost. This is the one with

one clean ancilla, two Toffoli gates and one uncontrolled phase gate. (It is not clear when exactly this design has been invented, but c.f. [34], section 2 for a more recent discussion.)

Given the control qubit $|c\rangle$ and target qubit $|t\rangle$ the controlled phase gate $C(P(\varphi)), |\varphi| = 1$ is emulated by applying Toffoli$(I \otimes I \otimes P(\varphi))$ Toffoli to the state $|c\rangle|t\rangle|0\rangle$. Ternary emulation of Toffoli gate is discussed in detail in section IV. Somewhat surprisingly, ternary emulation of uncontrolled phase gates in practice incurs larger overhead than emulation of classical gates. Also the binary Hadamard gate is a Clifford gate in the binary framework, but cannot be emulated by a ternary Clifford circuit. This introduces additional overhead factor of $(1 + \Theta(1/\log(1/\delta)))$. We discuss this in more detail for each the two platforms in our focus.

1. *Emulation of phase gate in Clifford+$R_{|2\rangle}$ basis*

In ternary framework a $P(\varphi) = |0\rangle\langle 0| + \varphi\,|1\rangle\langle 1|, |\varphi| = 1$ can be emulated exactly by the balanced two-level gate $P'(\varphi) = |0\rangle\langle 0| + \varphi\,|1\rangle\langle 1| + \varphi^{-1}\,|2\rangle\langle 2|$ which is a composition of the Clifford reflection H^2 and the non-Clifford reflection $P''(\varphi) = |0\rangle\langle 0| + \varphi\,|1\rangle\langle 2| + \varphi^{-1}\,|2\rangle\langle 1|$. Also, the binary Hadamard gate $h = (|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1|)/\sqrt{2}$ is a two-level Householder reflection as it is. As per [6], both $P''(\varphi)$ and h can be effectively approximated to precision δ by Clifford+$R_{|2\rangle}$ circuits with R-counts in $8\,\log_3(1/\delta) + O(\log(\log(1/\delta)))$. For reference, in Clifford+T framework the T-count of δ-approximation of a generic phase gate is in $3\,\log_2(1/\delta) + O(\log(\log(1/\delta)))$.

Although the formal depth expressions suggest a cost overhead factor of roughly $8/(3\,\log_2(3)) \sim 1.68$ for Clifford+$R_{|2\rangle}$, we need to keep in mind that the $R_{|2\rangle}$ is overwhelmingly less expensive than the T gate. Thus the cost of quantum Fourier transform is never an issue on metaplectic computer.

2. *Emulation of phase gate in Clifford+P_9 basis*

At the time of this writing emulation of the binary QFT on a generic ternary computer is not entirely straightforward.

First of all, we currently do not know an efficient direct circuit synthesis method for Householder reflections in the Clifford+P_9 basis. If follows from [8] that any ternary unitary gate can be also approximated to precision δ by an ancilla-free Clifford+P_9 circuit of depth in $O(\log(1/\delta))$; but we do not have a good effective procedure for finding ancilla-free circuits of this sort, neither do we have a clear idea of the practical constant hidden in the $O(\log(1/\delta))$.

As a bridge solution, we show in Appendix A, that the requisite magic state $|\psi\rangle$ (see eq. (9)) for the gate $R_{|2\rangle}$ gate can be emulated exactly and coherently by a set of effective repeat-until-success circuits with four ancillary qutrits and expected average P_9-count of $27/4$. Thus we can approximate a required uncontrolled phase gate with an efficient Clifford+$R_{|2\rangle}$ circuit and then transcribe the latter into a corresponding ancilla-assisted probabilistic circuit over Clifford+P_9 basis. In order to have a good synchronization with the Clifford+$R_{|2\rangle}$ circuit execution it would suffice to have the magic state preparation coprocessor of width somewhat greater than 27. Since the controlled phase gates and hence the approximating Clifford+$R_{|2\rangle}$ circuits are performed sequentially in the context of the QFT, this coprocessor is shared across the QFT circuit and thus the width overhead is bound by a constant.

On the balance, we conclude that, while ternary emulation of the QFT on binary data is likely to be more expensive in terms of required non-Clifford units, the non-Clifford depth overhead factor is upper bounded by an $(\alpha + \Theta(1/\log(1/\delta)))$ where α is a small constant. Such overhead becomes practically valid, however, when hosting period-finding solutions that make heavy use of Fourier transform, such as for example the Beauregard circuit [3] (see Appendix C for a further brief discussion).

G. Conclusion on the comparative cost of ternary emulation

In the context of designs described in preceding subsections, the overhead for ternary emulation measured as overhead in non-Clifford depth (informally, "Toffoli depth") is insignificant. Therefore the practical cost comparison between the binary and emulated ternary solutions, *reduces to comparative cost of implementing three-qubit Toffoli gate* on each of the two frameworks. Comparing the cost of Toffoli gate implementation is the sole purpose of the next section.

IV. IMPLEMENTING REFLECTIONS ON GENERIC TERNARY AND METAPLECTIC TOPOLOGICAL QUANTUM COMPUTERS

A state of the art implementation of the three-qubit binary Toffoli gate assumes the availability of the Clifford+T basis [26]. It has been known for quite some time cf. [1] that a Toffoli gate can be implemented ancilla-free using a network of CNOTs and 7 $T^{\pm 1}$ gates. It has been shown in [17] that this is the minimal T-count for ancilla-free implementation of the Toffoli gate.

In subsection IV A we develop certain emulations of classical two-level reflections (which generalize Toffoli and Toffoli-like gates) on generic ternary computer endowed with the Clifford+P_9 basis as described in section II C. In particular we point out that any n-qutrit classical two-level reflection with $n > 1$ on binary data can be implemented by a network of ternary Clifford gates and 6 P_9 gates using $n-2$ clean ancillas. This implies of course an emulation of the three-qubit Toffoli gate with 6 P_9 gates and one clean ancilla. We currently do not have a proof that this emulation is optimal given the number of ancillas they use.

Assuming, speculatively, that the "hardness" of performing the binary T gate and the ternary P_9 gate fault-tolerantly is about the same, we find that, compared to the binary Clifford+T basis, the emulation of the Toffoli gate on a generic ternary computer is somewhat less expensive when one ancilla is allowed. In subsection IV B we reevaluate the emulation cost assuming a *metaplectic topological quantum computer* (MTQC) with Clifford+$R_{|2\rangle}$ basis as described in section II D. In that setup we get two different options for implementing non-Clifford classical two-way transpositions (including the Toffoli gate).

One is direct approximation using Clifford+$R_{|2\rangle}$ circuits. The other is based on the P_9 gate but it uses *magic state preparation* in the Clifford+$R_{|2\rangle}$ basis instead of magic state distillation. This is explained in detail in subsection IV B. The first option might be ideal for smaller quantum computers. It allows circuits of fixed widths but creates implementation circuits for Toffoli gates with the R-count of approximately $8\log_3(1/\delta)$ when $1-\delta$ is the desired fidelity of the Toffoli gate. The second option supports separation of the cost of the P_9 gate into the "online" and "offline" components (similar to the Clifford+T framework) with the "online" component depth in $O(1)$ and the "offline" cost offloaded to a state preparation part of the computer, which has the width of roughly $12\log_3(1/\delta)$ but does not need to stay always coherent.

A. Implementing classical reflections in Clifford+P_9 basis

The synthesis described here is a generic ternary counterpart of the exact, constant T-count representation of the three-qubit Toffoli gate in the Clifford+T framework. A baseline constant-depth implementation of n-qutrit classical reflection using $n-2$ ancillas is based on the ancilla-free implementation of two-qutrit reflection described in our paper [7]. A new result in this subsection is emulation of n-qutrit classical reflection on *binary data* using 6 P_9 gates. We describe the corresponding circuit designs below.

One distinction of the ternary framework from the binary one is that not all two-qutrit classical gates are Clifford gates. In particular the $\tau_{|10\rangle,|11\rangle}$ reflection which is a strict emulation of the binary CNOT is not a Clifford gate and neither is the $\tau_{|10\rangle,|01\rangle}$ which which is a strict emulation of the binary SWAP. However, while binary SWAP can be emulated simply as a restriction of the (Clifford) ternary swap on binary subspace, the CNOT cannot be so emulated.

A particularly important two-qutrit building block is the following non-Clifford gate

$$C_1(\mathrm{INC})|j\rangle|k\rangle = |j\rangle|(k+\delta_{j,1}) \bmod 3\rangle.$$

A peculiar phenomenon in multi-qudit computation (in dimension greater than two) is that a two-qudit classical non-Clifford gate (such as $C_1(\mathrm{INC})$) along with the INC gate is universal for the ancilla-assisted reversible classical computation , cf. [10], whereas a three-qubit gate, such as Toffoli is needed for the purpose in multi-qubit case.

Following is a slight variation of a circuit from [7]:

$$\tau_{|02\rangle,|2,0\rangle} = \mathrm{TSWAP}\, C_1(\mathrm{INC})_{2,1}\, C_1(\mathrm{INC})_{1,2}\, C_1(\mathrm{INC})_{2,1}\, C_1(\mathrm{INC})_{1,2}\, C_1(\mathrm{INC})_{2,1}, \tag{13}$$

where TSWAP is the ternary (Clifford) swap gate. This would seem to suggest using 5 copies of $C_1(\mathrm{INC})$ gate for implementing a two-level two-qutrit reflection. Turns out, this is inefficient when we only need to process binary data.

By direct computation, the following classical circuit is an exact emulation of the binary CNOT gate on the binary data:

$$\mathrm{SUM}_{2,1}(\tau_{|1\rangle,|2\rangle}\otimes\tau_{|1\rangle,|2\rangle})\,\mathrm{TSWAP}\, C_1(\mathrm{INC})_{2,1}\, C_1(\mathrm{INC}^\dagger)_{1,2}\,(\tau_{|1\rangle,|2\rangle}\otimes\tau_{|1\rangle,|2\rangle})\mathrm{SUM}^\dagger_{2,1} \tag{14}$$

The two non-Clifford gates in this circuit are the $C_1(\mathrm{INC})$ and $C_1(\mathrm{INC}^\dagger)$. (To avoid confusion, let us note that the gate (14) is no longer an axial reflection on the ternary data.) In its turn the $C_1(\mathrm{INC})$ is Clifford-equivalent to the

$C_1(Z) = diag(1,1,1,1,\omega_3,\omega_3^2,1,1,1)$ gate ($\omega_3 = e^{2\pi\, i/3}$), and the latter gate is represented exactly by the network shown in Figure 7 (up to a couple of local $\tau_{|0\rangle|1\rangle}$ gates and a local Q gate).

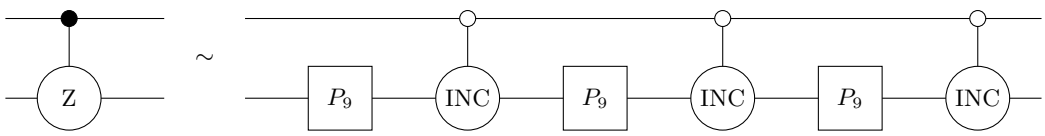

Figure 7: Exact representation of $C_1(Z)$ in terms of P_9 gates.

Plugging in corresponding representations of $C_1(\text{INC})$ and $C_1(\text{INC}^\dagger)$ into the circuit (14) we obtain an exact emulation of CNOT that uses 6 instances of the $P_9^{\pm 1}$ gate.

Remark 3. By using an available clean ancilla, we can exactly represent the $C_1(Z)$ in P_9-depth one. The corresponding circuit is equivalent to one shown in Figure 8. Thus the CNOT gate can be emulated on binary data using a clean ancilla in P_9-depth two.

Thus when depth is the optimization goal, a clean ancilla can be traded for triple compression in non-Clifford depth of ternary emulation of the CNOT. (This rewrite is similar in nature to the one employed in [18] for the binary Margolus-Toffoli gate.)

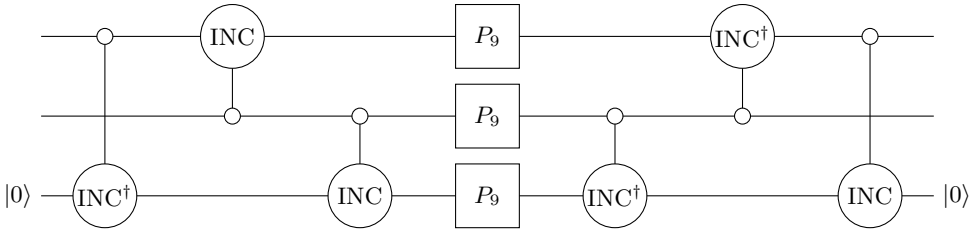

Figure 8: Exact representation of $C_0(Z)$ in P_9-depth one.

We can now emulate the binary Toffoli gate on binary data using one copy of CNOT, one clean ancillary qutrit and no additional non-Clifford gates. It follows from the following useful more general remark:

Remark 4. Let U be a ternary unitary operator and let $C^k(U), k > 0$ emulate the multi-controlled version of U with k levels of binary control. Then the emulation of $C^{k+1}(U)$ can be performed with one additional clean ancilla and no additional non-Clifford gates.

Indeed, let us represent $C^k(U)$ as $C(C^{k-1}(U))$, or, more specifically, $C(C^{k-1}(U))|c_k,\vec{c}\rangle|x\rangle = (C^{k-1}(U))^{c_k}|\vec{c}\rangle|x\rangle$ Let c_{k+1} be the new control qubit that we want to add and $|a\rangle$ be an ancilla qutrit intialized with $|0\rangle$. Now the following circuit emulates $C^{k+1}(U)$ assuming only binary controls:

$$\text{SUM}_{c_{k+1},a}\,\text{SUM}_{c_k,a}\,(I_{c_{k+1}} \otimes I_{c_k} \otimes (C_a(C^{k-1}(U)))\,\text{SUM}^\dagger_{c_k,a}\,\text{SUM}^\dagger_{c_{k+1},a}.$$

Indeed the ancilla $|a\rangle$ acquires the $|1\rangle$ state if and only if $c_k = c_{k+1} = 1$. By design of the circuit the ancilla is uncomputed and unentangled at its end. The circuit described in the Remark 4 is presented as a diagram on Figure 9.

In summary, we have established the following

Proposition 5. *There is a four-qutrit circuit (including one ancillary qutrit) that emulates the three-qubit Toffoli gates on binary data at the non-Clifford cost of* 6 P_9 *gates.*

There is a five-qutrit circuit (including two ancillary qutrits) that emulates the three-qubit Toffoli gates on binary data in P_9*-depth of* 2 *and* P_9*-count of* 6*.*

This follows directly from the circuits and remarks we have developed above.

Given the above building blocks it is *also* possible to emulate the three-qubit Toffoli gate on three qutrits ancilla-free, albeit at additional cost.

Remark 6. Any n-qubit axial reflection (multi-controlled Toffoli for $n > 3$) can be emulated on binary data by a circuit that is Clifford-equivalent to the two-qutrit axial reflection (13).

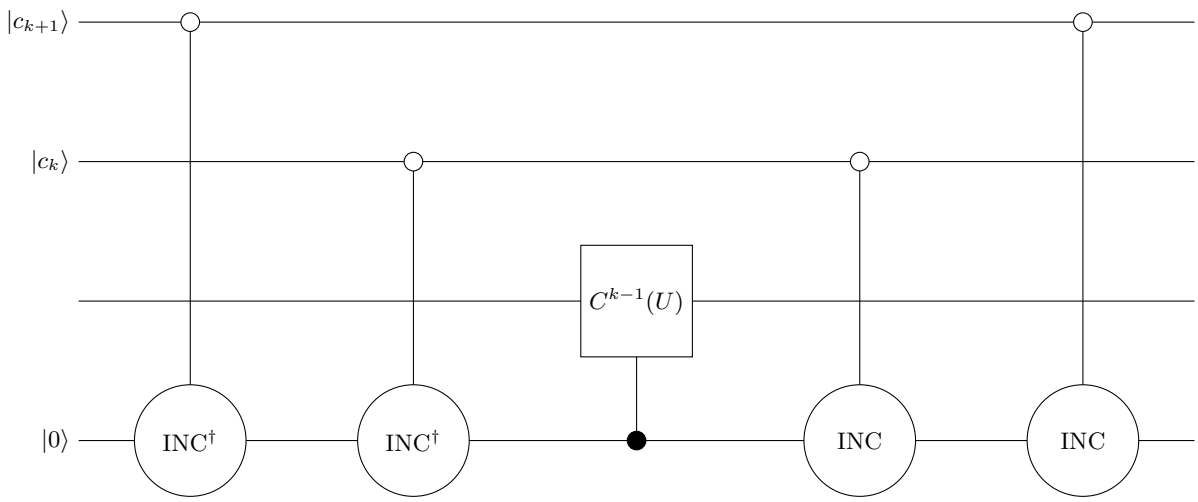

Figure 9: Adding a level of binary control in ternary network.

We spell out a specific circuit for the three-qubit Toffoli (it is easy to extend the pattern for $n > 3$):

$$\text{Toffoli} = (\text{SUM}^\dagger \otimes I)(I \otimes \text{SUM}^\dagger_{3,2})(I \otimes I \otimes \tau_{|1\rangle,|2\rangle})(I \otimes \tau_{|02\rangle,|2,0\rangle})(I \otimes I \otimes \tau_{|1\rangle,|2\rangle})(I \otimes \text{SUM}_{3,2})(\text{SUM} \otimes I)$$

(again, to avoid confusion: this circuit is not an axial reflection on ternary data).

The best known circuit for the $\tau_{|02\rangle,|2,0\rangle}$ is (13) and it explicitly requires five $C_1(\text{INC})$ gates, hence the non-Clifford cost of the above ancilla-free emulation of the three-qubit Toffoli is 15 P_9 gates. On the flip side: this non-Clifford part of the cost does not change as we add arbitrarily many levels of binary control to the gate.

B. Implementing classical reflections in metaplectic Clifford+$R_{|2\rangle}$ basis

It has been shown in [6] that, given a small enough $\delta > 0$ any n-qutrit two-level Householder reflection can be approximated effectively and efficiently to precision δ by a Clifford+$R_{|2\rangle}$ circuit containing at most $8\log_3(1/\delta) + O(\log(\log(1/\delta))) + O((2+\sqrt{5})^n)$ instances of the $R_{|2\rangle}$ gate. In particular, when $n = 1$ the asymptotic term $O((2+\sqrt{5})^n)$ resolves to exactly 1 and when $n = 2$ it resolves to exactly 4. In both cases it is safe to merge this term with the $O(\log(\log(1/\delta)))$ asymptotics.

It is easily seen that the single-qutrit P_9 gate is the composition of the ternary Clifford gate $\tau_{|0\rangle,|2\rangle}$ and the Householder reflection $\omega_9 |0\rangle\langle 2| + |1\rangle\langle 1| + \omega_9^{-1} |2\rangle\langle 0|$. The two-qutrit gate CNOT $= \tau_{|10\rangle,|11\rangle}$ is by itself a two-level Householder reflection $R_{(|10\rangle-|11\rangle)/\sqrt{2}}$. Similarly, Toffoli $= \tau_{|110\rangle,|111\rangle} = R_{(|110\rangle-|111\rangle)/\sqrt{2}}$. Therefore, our results apply and we have efficient strict emulations of P_9, CNOT and Toffoli gates at depths that are logarithmic in $1/\delta$ and, in practice are roughly $8\log_3(1/\delta)$.

A sufficient per-gate precision δ may be found in $O(1/(d\log(n)))$ where d is the depth of the modular exponentiation circuit expressed in non-Clifford reflections. Therefore, injecting metaplectic circuits in place of reflections creates an overhead factor in $\Theta(\log(d)\log(\log(n)))$. While being asymptotically moderate, such overhead could be a deterrent when factoring very large numbers. This motivates us to explore constant depth approximations of classical reflections which we do in the next section.

C. Constant-depth implementation of CNOT on ternary quantum computers.

This is a key section intended to demonstrate that integer arithmetic on a ternary quantum computer can be truly efficient. We are building upon subsection IV A that describes exact emulation of CNOT with 6 instances of the P_9 gate. One of the core results in [11] implies that the P_9 gate can be executed exactly by a deterministic state injection circuit using one ancilla, one measurement and classical feedback, *provided* availability of the "magic" ancillary state

$$\mu = \omega_9^{-1} |0\rangle + |1\rangle + \omega_9 |2\rangle.$$

	Clean magic states	Raw resources
Binary	7	$7(2\log_2(1/\delta))^{2.5}$
Generic ternary, no ancilla	15	$15\log_2^3(1/\delta)$
Generic ternary, one ancilla	6	$6\log_2^3(1/\delta)$
Metaplectic, one ancilla	6	$48\log_3(1/\delta)$

Table I: Resource count factors for three-qubit Toffoli gates.

A diagram for the state injection circuit is given in Figure 1.

Assuming, hypothetically, that the magic state μ can be prepared outside the main quantum circuit, we get, conceptually, a separation of the quantum complexity into "online" and "offline" components - similar to one employed in the binary Clifford+T network. Now the "online" parts of the P_9 gate, hence CNOT and Toffoli emulations are constant depth. The "offline" part for generating instances of the magic state can evolve on a separate quantum processor that does not even have to stay coherent for extended periods of time. The main bottleneck of this solution is synchronization between the "online" and "offline" components. The throughput of the "offline" component must be sufficiently high in order to yield the required instances of the magic state on a quantum clock schedule.

In the context of the binary Clifford+T network, assuming the required fidelity of the T gate is $1-\delta, \delta > 0$, there is a choice of magic state distillation solutions stemming from the original one in [9]. At top level a solution from this class can be described as a quantum processor of depth in $O(\log(\log(1/\delta)))$ and width of approximately $O(\log^{2.5}(1/\delta))$ (more precisely: $O(\log^{\log_3(15)}(1/\delta))$).

In comparison, the magic state distillation for a generic ternary quantum computer, described in [11] maps onto quantum processor of depth in $O(\log(\log(1/\delta)))$ and width of approximately $O(\log^3(1/\delta))$. Therefore the "offline" component for magic state distillation on a generic ternary quantum computer is asymptotically somewhat larger than the one for Clifford+T computer assuming the same initial fidelity of raw magic states. (As a concrete example, it can be about 5 times larger when the target δ is around 10^{-21} and fidelity of raw magic states is 90% in both cases.)

We observe that the offline width can be reduced quite significantly when the target ternary computer is a metaplectic topological one (MTQC) rather than a generic ternary one. Since the MTQC implements the Clifford+$R_{|2\rangle}$ basis that is universal even without the magic state distillation protocol, the instances of the magic state μ can be prepared on much smaller scale.

Observation 7. *(see [6]) An instance of magic state μ can be prepared offline at fidelity $1-\delta$ by Clifford+$R_{|2\rangle}$ circuit of non-Clifford depth in $r(\delta) = 8\log_3(1/\delta) + O(\log(\log(1/\delta)))$.*

In order to have a perfect syncronization with the P_9 gates in the main circuit we should simply pipeline $r(\delta)$ instances of the magic state preparation circuit, so we always have a magic state at fidelity $1-\delta$ ready to be injected into the P_9 protocol. In particular an n-bit quantum carry lookahead adder that would require performing up to n state injections in parallel, might need an offline magic state preparation processor of width approximately $8\,n\log_3(1/\delta)$. This processor is asymptotically smaller than one of width $n\log_2^3(1/\delta)$ (required in the case of generic ternary quantum computer with full magic state distillation protocol) and asymptotically smaller than one of width $n\log_2^{2.5}(1/\delta)$ required for binary Clifford+T solution with full magic state distillation protocol for the T gate.

D. Resource count comparison on the logical level

Here we conclude the cost comparison between the ternary Clifford+P_9 emulation and binary Clifford+T implementation on logical level in the context of reversible classical circuits implementing the modular exponentiation. Suppose, t is the Toffoli count of one of the original binary circuits we have reviewed in section III. As per our main thesis, for n bit numbers the cost the corresponding ternary emulation is equivalent to emulating a circuits with $(1+O(1/\log(n)))\,t$ Toffoli gates and no CNOTs.

The cost metrics for a Toffoli gate, expressed in magic state count is presented in Table I. The first column tallies the number of "clean" magic states at fidelity $1-O(\delta)$ needed for executing a Toffoli gate at fidelity $1-\delta$. The second column counts the raw magic states (at fidelity 7/8) consumed by the magic state distillation coprocessor, except for the bottom-most cell that counts the number of concurrent magic state preparation processes on a natively metaplectic quantum computer.

As per our convention, the non-Clifford reflection cost of a Shor's period-finding circuit is *proportional* to the Toffoli count of the original circuit up to the "ternary overhead" factor of $(1+O(1/\log(n)))$. Thus the platform-dependent resource costs expressed in magic state counts are directly proportional to the numbers given in Table I.

Circuits	Online width	Offline width
Takahashi [30]	$2\,n+1$ (qubits)	$7(6\,\log_2(n))^{2.5}$
Section III A, generic ternary	$n+3$ (qutrits)	$6(3\,\log_2(n))^3$
Section III A, metaplectic	$n+3$ (qutrits)	$48\times 3\,\log_3(n)$

Table II: Widths comparison for low-widths modular exponentiation circuits.

Circuits	Online width	Offline width
Original QCLA, section III B	$3\,n-w(n)$ (qubits)	$7\,n\,(6\,\log_2(n))^{2.5}$
Generic emulation, 15 P_9/Toffoli	$3\,n-w(n)$ (qutrits)	$15\,n\,(3\,\log_2(n))^3$
Generic emulation, 6 P_9/Toffoli	$4\,n-w(n)$ (qutrits)	$6\,n\,(3\,\log_2(n))^3$
Metaplectic emulation, 15 P_9/Toffoli	$3\,n-w(n)$ (qutrits)	$120\times 3\,n\,\log_3(n)$
Metaplectic emulation, 6 P_9/Toffoli	$4\,n-w(n)$ (qutrits)	$48\times 3\,n\,\log_3(n)$

Table III: Widths comparison for reduced-depth modular exponentiation circuits. ($w(n)$ is the Hamming weight of n)

Example 8. *Consider the task of factoring an n-bit integer with $n = 4096$. This requires the T or, respectively, P_9, gate precision δ of roughly $\delta = 10^{-15}$. For perfect synchronization a magic state at the corresponding fidelity requires approximately $740,000$ concurrent raw magic states on a generic ternary quantum computer, approximately $590,000$ concurrent raw magic states in the Clifford+T framework and approximately 1500 concurrent metaplectic magic state preparation circuits. (In both magic state distillation scenarios, the raw state fidelity of $7/8$ was assumed.)*

This example illustrates significant reduction in the logical size of magic state preparation coprocessor on a ternary computer with available metaplectic basis.

Another metrics to compare would be the widths of online and offline parts of the end-to-end computation expressed in the qubit counts or, respectively, qutrit counts. Suppose we are factoring an n-bit number. When tallying the widths, we are going to disregard the size of the first register of the state (10). In a more conventional circuit layout that register has the fixed size of $2\,n$ bits. Under certain assumptions on measurements and QFT compilation, there is however an available "one control qubit trick" (cf. [27] or [3], section 2.4), and that register can be reduced to one qubit (and, respectively, emulated with one qutrit). In either case, this component of the width is fixed and can be taken out of the comparison tables, which only need to contain the width comparisons for the modular exponentiation circuits.

We present the width comparisons in two tables: Tables II and III. The first one tallies the counts for low-width circuits - namely the Takahashi circuit [30] and our circuit from section III A. We observe that the ancillary qutrit needed for lower depth emulation of the Toffoli gate is shared across all the Toffoli gates. The second table compares widths for the binary circuit based on the carry lookahead adder (our section III B) and its ternary emulations. We note that when using lower depth circuits we use up to n clean magic states in parallel per a non-Clifford time slice: this affects all the "offline" width bounds in the rightmost column of Table III.

Here we assume that $\varepsilon = 1/\log(n)$ is a sufficient end-to-end precision of the period-finding circuit. Then the atomic precision δ per individual gate, or, rather per individual clean magic state within the circuit depends on its size. The circuits under comparison differ only in depth but not in size asymptotics which is in $O(n^3)$ (disregarding the slower $O(\log(n))$ terms). We observe that $\log(1/\delta)$ is roughly $3\,\log(n)$ for the required δ.

One observes that using the metaplectic emulation would be our best chance of making the reduced-depth circuit based on carry lookahead practically sustainable on the "offline" width. Indeed the need to have n clean magic states to be available synchronously inflates the size of the "offline" coprocessor by the factor of n. Given that that coprocessor was large to begin with in both the native binary and generic ternary cases might undermine the feasibility of the reduced-depth circuit in those cases.

V. CONCLUSION

We have investigated the use of reversible ternary logic for emulating the binary Shor's period finding function ([28]). We have demonstrated that the efficiency of the emulated circuit is defined almost exclusively by the efficiency of emulation of the binary Toffoli gate. Keeping in mind that in practice the Toffoli gate is never perfect in either binary or ternary framework, we have chosen to measure the actual resource efficiency in terms of the magic state counts. Ternary emulation proposed here has the practical depth advantage factor of $6/7$ in terms of clean magic state count. We have also demonstrated that, when the target ternary computer is the metaplectic topological quantum computer (MTQC), then the expensive ("offline") magic state distillation can be replaced by much more streamlined

magic state preparation.

This solution brings down the *logical* size of the magic state preparation coprocessor to be linear in $\log(\log(N))$, instead of nearly cubic in $\log(\log(N))$. Additional benefit of this solution is the fact that MTQC has all operations, including the magic state preparation, topologically protected from local decoherence, and therefore we expect the physical size of the overall computer to be another couple of orders of magnitude smaller, compared to computers that require extensive quantum error correction. Alternative option availed by the MTQC is getting rid of magic states (and hence of the magic state coprocessor) altogether by approximating all the occurring non-Clifford reflections directly to the required fidelity. This increases the depth of the period finding by a factor in $\Theta(\log(\log(N)))$ which can be well tolerated unless N is extremely large.

Even though solutions offered here appear to be asymptotically optimal given the computer size constraints, we make no representation as to their practical optimality. Investigating the latter is one direction of future research. Another open research question is whether using the arithmetic data in ternary (instead of binary) encoding carries any additional cost benefits. So far our designs in this area appeared to be less cost effective than the emulation of binary-encoded arithmetic in ternary logic, but we leave a more detailed comparison for future work.

ACKNOWLEGDMENTS

We would like to thank Tom Draper and Sandy Kutin for providing $\langle q|pic\rangle$ [15] which we used for typesetting most of the figures in this paper.

[1] M. Amy, D. Maslov, M. Mosca, and M. 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(6)** (2013)
[2] M. Baraban, N. E. Bonesteel, and S. H. Simon, "Resources required for topological quantum factoring," Phys. Rev. A **81(062317)** (2010)
[3] S. Beauregard, "Circuit for Shor's algorithm using 2n+3 qubits," Quantum Information and Computation **3(2** (2003)
[4] D. Beckman, A. N. Chari, S. Devabhaktuni, and J. Preskill, "Efficient networks for quantum factoring," Phys. Rev. A **54(1034-1063)** (1996)
[5] E. Bernstein, and U. V. Vazirani, "Quantum complexity theory," SIAM J. Comput. **26(5)** (1997)
[6] A. Bocharov, S. X. Cui, V. Kliuchnikov, and Z. Wang, "Efficient topological compilation for a weakly integral anyonic model," Phys. Rev. A **93(012313)** (2016)
[7] A. Bocharov, S. X. Cui, M. Roetteler, and K. M. Svore, "Improved quantum ternary arithmetics," 2015, 1512.03824
[8] J. Bourgain, and A. Gamburd, "Spectral gaps in $SU(d)$," Comptes Rendus Mathematique **348(11,12)** (2010)
[9] S. Bravyi, and A. Kitaev, "Universal quantum computation with ideal Clifford gates and noisy ancillas," Phys. Rev. A **32(6)** (2005)
[10] G. K. Brennen, S. S. Bullock, and D. P. O'Leary, "Efficient circuits for exact-universal computation with qudits," QIC **6(4,5)** (2006)
[11] E. T. Campbell, H. Anwar, and D. E. Browne, "Magic state distillation in all prime dimensions using quantum Reed-Muller codes," Phys. Rev. X **2(041021)** (2012)
[12] R. Cleve, and J. Watrous, "Fast parallel circuits for the quantum Fourier transform," FOCS '00 Proceedings of the 41st Annual Symposium on Foundations of Computer Science **526** (2000)
[13] S. A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton, "A new quantum ripple-carry addition circuit," 2004, quant-ph/0410184
[14] S. X. Cui, and Z. Wang, "Universal quantum computation with metaplectic anyons," J. Math. Phys. **56(032202)** (2015)
[15] A. Draper and S. Kutin "$\langle q|pic\rangle$: Creating quantum circuit diagrams in TikZ," 2016, Available from https://github.com/qpic/qpic
[16] T. G. Draper, S. A. Kutin, E. M. Rains, and K. M. Svore, "A logarithmic-depth quantum carry-lookahead adder," Quantum Information and Computation **6(4–5)** (2006)
[17] D. Gosset, V. Kliuchnikov, M. Mosca, and V. Russo, "An algorithm for the T-count," QIC **14(15,16)** (2014)
[18] C. Jones, "Novel constructions for the fault-tolerant Toffoli gate," Phys. Rev. A **87(022328)** (2013)
[19] M. Malik, M. Erhard, M. Huber, H. Sosa-Martinez, M. Krenn, R. Fickler, and A. Zeilinger, "Multi-photon entanglement in high dimensions," Nature Photonics **10(248–252)** (2016)
[20] I. L. Markov, and M. Saeedi, "Constant-optimized quantum circuits for modular multiplication and exponentiation," Quantum Information and Computation **12(5,6)** (2012)
[21] I. L. Markov, and M. Saeedi, "Faster quantum number factoring via circuit synthesis," Phys. Rev. A **87(012310)** (2013)
[22] M. Morisue, K. Oochi, and M. Nishizawa, "A novel ternary logic circuit using Josephson junction," IEEE Trans. Magn. **25(2)** (1989)

[23] M. Morisue, J. Endo, T. Morooka, and N. Shimzu, "A Josephson ternary memory circuit," Multiple-Valued Logic, 1998. Proceedings. 1998 28th IEEE International Symposium on (1998)
[24] A. Muthukrishnan, and C. Stroud Jr., "Multivalued logic gates for quantum computation," Phys. Rev. A **62(051309)** (2000)
[25] C. Nayak, S. H. Simon, M. Freedman, and S. D. Sarma, "Non-abelian anyons and topological quantum computation," Rev. Mod. Phys. **80(1083)** (2008)
[26] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, UK, 2000)
[27] S. Parker, and M. B. Plenio, "Efficient factorization with a single pure qubit and log N mixed qubits," Phys. Rev. Lett., 85 **85(3049–3052)** (2000)
[28] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," SIAM J. Comput. **26 (1484–1509)** (1997)
[29] A. Smith, B. E. Anderson, H. Sosa-Martinez, I. H. Deutsch, C. A. Riofrio, and P. S. Jessen, "Quantum control in the Cs $6S_{1/2}$ ground manifold using rf and μw magnetic fields.," Phys. Rev. Lett. **111(170502)**(2013)
[30] Y. Takahashi, and N. Kunihiro, "A quantum circuit for Shor's factoring algorithm using 2n+2 qubits," QIC **6(2)** (2006)
[31] R. Van Meter, and K. M. Itoh, "Fast quantum modular exponentiation," Phys. Rev. A **71(052320)** (2005)
[32] U. V. Vazirani, "On the power of quantum computation," Phil. Trans. R. Soc. Lond. A **356(1759–1768)** (1998)
[33] V. Vedral, A. Barenco, and A. Ekert, "Quantum networks for elementary arithmetic operations," Phys. Rev. A **54(147)** (1995)
[34] J. Welch, A. Bocharov, and K. M. Svore, "Efficient approximation of diagonal unitaries over the Clifford+T basis ," QIC **16(1,2)** (2016)
[35] C. Zalka, "Fast versions of Shor's quantum factoring algorithm," 1998, quant-ph/9806084

Appendix A: Exact emulation of the $R_{|2\rangle}$ gate in the Clifford+P_9 basis.

At this time we lack a good effective classical compilation procedure for approximating non-classical unitaries by efficient ancilla-free circuits in the Clifford+P_9 basis.

We show here, however, that the magic state $|\psi\rangle$ of (9) that produces the $R_{|2\rangle}$ gate by state injection can be prepared by certain probabilistic measurement-assisted circuits over the Clifford+P_9 basis. Therefore the compilation into the Clifford+P_9 basis can be reduced to a compilation into the Clifford+$R_{|2\rangle}$ basis, while incurring a certain "offline" cost. This solution, however inelegant, is sufficient, for example, in the context of Shor's integer factorization.

We have seen in section IV A that the classical C_1(INC) and, hence, the classical C_2(INC) gates can be represented exactly and ancilla-free using three P_9 gates. We use the availability of these gates to prove the key lemma below.

Recall that $\omega_3 = e^{2\pi\, i/3}$ is a Clifford phase.

Lemma 9. *Each of the ternary resource states*

$$(|0\rangle + \omega_3|1\rangle)/\sqrt{2}, \quad \textit{and} \quad (|0\rangle + \omega_3^2|1\rangle)/\sqrt{2}$$

can be represented exactly by a repeat-until-success (RUS) circuit over Clifford+P_9 with one ancillary qutrit and expected average number of trials equal to $3/2$.

Proof. Let us give a proof for the second resource state. (The proof is symmetrical for the first one.)

We initialize a two-qutrit register in the state $|20\rangle$ and compute

$$C_2(\mathrm{INC})(H \otimes I)|20\rangle = (|00\rangle + \omega_3^2|10\rangle + \omega_3|21\rangle)/\sqrt{3}.$$

If we measure 0 on the second qutrit, then the first qutrit is in the desired state. Overwise, we discard the register and start over. Since the probability of measuring 0 is $2/3$, the Lemma follows. □

Corollary 10. *A copy of the two-qutrit resource state*

$$|\eta\rangle = (|0\rangle + \omega_3|1\rangle) \otimes (|0\rangle + \omega_3^2|1\rangle)/2 \tag{A1}$$

can be represented exactly by a repeat-until-success circuit over Clifford+P_9 with two ancillary qutrits and expected average number of trials equal to $9/4$.

To effectively build a circuit for the Corollary, we stack together the two RUS circuits described in Lemma 9.

Lemma 11. *There exists a measurement-assisted circuit that, given a copy of resource state $|\eta\rangle$ as in (A1), produces a copy of the resource state*

$$|\psi\rangle = (|0\rangle - |1\rangle + |2\rangle)/\sqrt{3} \tag{A2}$$

with probability 1.

Proof. Measure the first qutrit in the state $(H^\dagger \otimes I)\mathrm{SUM}|\eta\rangle$.

Here is the list of reduced second qutrit states given the measurement outcome m:

$$m = 0 \mapsto (|0\rangle - |1\rangle + |2\rangle)/\sqrt{3},$$

$$m = 1 \mapsto (|0\rangle - \omega_3|1\rangle + \omega_3^2|2\rangle)/\sqrt{3},$$

$$m = 2 \mapsto (|0\rangle - |1\rangle + \omega_3|2\rangle)/\sqrt{3}.$$

While the first state on this list is the desired $|\psi\rangle$, each of the other two states can be turned into $|\psi\rangle$ by classically-controlled Clifford correction.

□

As shown in [14],Lemma 5, the resource state $|\psi\rangle$ as in (A2) can be injected into a coherent repeat-until-success circuit of expected average depth 3 to execute the $R_{|2\rangle}$ gate on a coherent state. See our Figure 2 in section II D.

Recall that the $C_2(\mathrm{INC})$ gate appearing in the lemma 9 construction has the non-Clifford cost of three P_9 gates. Thus, to summarize the procedure: we can effectively and exactly prepare the magic state $|\psi\rangle$ using four-qutrit register at the expected average P_9-count of 27/4.

To have a good synchronization of the magic state preparation with the $R_{|2\rangle}$ gate injection is would suffice to have a magic state preparation coprocessor of width greater than 27 (to compensate for the variances in repeat-until-success circuits).

Appendix B: Circuit fidelity requirements for Shor's period finding function

To recap the discussion in section II E, the quantum period finding function consists of preparing a unitary state $|u\rangle$ proportional to the superposition

$$\sum_{k=0}^{N^2} |k\rangle|a^k \bmod N\rangle \tag{B1}$$

followed by quantum Fourier transform, followed by measurement, followed by classical postprocessing.

As we know, the measurement result j can be useful for recovering a period r or it can be useless. It has been shown in [28] that the probability p_{useful} of getting a useful measurement is in $\Omega(1/(\log(\log(N))))$.

Speaking in more general terms, let $\mathcal{H}$ be the Hilbert space where the $\mathrm{QFT}|u\rangle$ is to be found after the quantum Fourier transform step, let $\mathcal{G} \subset \mathcal{H}$ be the subspace spanned by all possible state reductions after all possible useful measurements and let $\mathcal{G}^\perp$ be its orthogonal complement in $\mathcal{H}$.

Let $\mathrm{QFT}|u\rangle = |u_1\rangle + |u_2\rangle, |u_1\rangle \in \mathcal{G}, |u_2\rangle \in \mathcal{G}^\perp$ be the orthogonal decomposition of $\mathrm{QFT}|u\rangle$. Then $p_{\mathrm{useful}} = ||u_1\rangle|^2$.

Let now $|v\rangle$ be an imperfect unitary copy of $\mathrm{QFT}|u\rangle$ at Hilbert distance ε. What is the probability of obtaining *some* useful measurement on measuring $|v\rangle$?

By definition, it is the probability of $|v\rangle$ being projected to $\mathcal{G}$ upon measurement.

Proposition 12. *In the above context the probability of $|v\rangle$ being projected to $\mathcal{G}$ upon measurement is greater than*

$$p_{useful} - 2\sqrt{p_{useful}}\,\varepsilon$$

Proof. Let $|v\rangle = |v_1\rangle + |v_2\rangle, |v_1\rangle \in \mathcal{G}, |v_2\rangle \in \mathcal{G}^\perp$ be the orthogonal decomposition of the state $|v\rangle$.

Clearly $||u_1\rangle - |v_1\rangle| < \varepsilon$ and, by triangle inequality, $||v_1\rangle| \geq ||u_1\rangle| - ||u_1\rangle - |v_1\rangle| > ||u_1\rangle| - \varepsilon$.

Hence $||v_1\rangle|^2 > (||u_1\rangle| - \varepsilon)^2 > ||u_1\rangle|^2 - 2\,||u_1\rangle|\,\varepsilon = p_{\mathrm{useful}} - 2\sqrt{p_{\mathrm{useful}}}\,\varepsilon$ as claimed. □

Corollary 13. *In the above context, if* $\varepsilon < \gamma \sqrt{p_{useful}}$ *where* $0 < \gamma < 1/2$*, then the probability of obtaining* some *useful measurement on measuring* $|v\rangle$ *is greater than* $(1 - 2\,\gamma)\, p_{useful}$*.*

In particular if $\varepsilon < \sqrt{p_{\text{useful}}}/4$, we are at least half as likely to obtain a useful measurement from the proxy state $|v\rangle$ as from the ideal state $\text{QFT}|u\rangle$.

In summary, there is a useful precision threshold ε in $O(1/(\sqrt{\log(\log(N))}))$ that allows to use an imprecisely prepared state at precision ε in place of the ideal state in the measurement and classical post-processing part of Shor's period finding function.

This translates into per-gate tolerance in the preparation circuit in a usual way. If d is the unitary depth of the state preparation circuit, then it suffices to represent each of the consecutive unitary gates to fidelity $1-\varepsilon/d$ or better.

For completeness, let us also make this part explicit.

Let $||U||$ denote the spectral norm of a unitary operator U.

Proposition 14. *Assume that an ideal quantum computation* $U = \prod_{k=1}^{d} U_k$ *is specified using* d *perfect unitary gates* U_k *and we actually implementing it using* d *imperfect unitary gates* V_k *where for all* $k = 1, \ldots, d$ *it holds that* $||U_k - V_k|| \leq \delta$*. Then for the actually implemented unitary transformation* $V = \prod_{k=1}^{d} V_k$ *it holds that* $||U - V|| \leq d\,\delta$*.*

Proof. (See also [5], [32]). For a more readable proof we will do induction in d. When $d = 1$ there is nothing to prove.

Assume the inequality has been proven for a product of length $d - 1$.

We have $||U - V|| = ||\prod_{k=1}^{d} U_k - (\prod_{k=1}^{d-1} U_k)\, V_d + (\prod_{k=1}^{d-1} U_k)\, V_d - \prod_{k=1}^{d} V_k|| \leq ||\prod_{k=1}^{d} U_k - (\prod_{k=1}^{d-1} U_k)\, V_d|| + ||(\prod_{k=1}^{d-1} U_k)\, V_d - \prod_{k=1}^{d} V_k|| = ||\prod_{k=1}^{d-1} U_k||\, ||U_d - V_d|| + ||(\prod_{k=1}^{d-1} U_k) - \prod_{k=1}^{d-1} V_k||\, ||V_d|| \leq \delta + (d-1)\delta = d\,\delta$,
where in the second step we used the triangle inequality, in the third step the multiplicativity of the norm, i.e. $||U\,V|| = ||U||\,||V||$ for all unitaries U,V, and that $||U|| = 1$ for all unitary U. In the last step we used the inductive hypothesis.

□

Appendix C: Alternative circuits for modular exponentiation

1. Width optimizing circuits

One way for the modular exponentiation to use qubits (resp., qutrits) sparingly is to perform computation in phase space.

First consistent solution of this kind has been proposed in [3]. A peculiar feature of the proposed solution is that the modular additive shift block for $|b\rangle \mapsto |(a + b) \bmod N\rangle$ has four interleaved quantum Fourier transforms (two direct and two inverse, see Figure 5 in [3]), the sole purpose of which is establishing and then forgetting the sign of $a + b - N$. It is unlikely that any of these transforms can be made redundant without significant redesign of the circuit. As we have pointed out in section III F, ternary quantum computers are comparatively inefficient in practice when emulating non-classical modular exponentiation circuits such as Beauregard [3].

Fortunately the Beauregard circuit has been superseded by Takahashi circuit [30] which is classical. That circuit sidesteps the need for clean ancillas by a clever use of the "idle" qubits. Instead of emulating Takahashi circuit directly, we point out that our ternary modular exponentiation circuit based on ripple-carry adder (see section III A), maintains smaller width in qutrits by much simpler means - systematic use of ternary basis state $|2\rangle$ which, for our purposes is always "idle".

2. Depth-optimizing circuits

Reversible classical circuits that stand apart from relatively simple layouts we have analyzed, are hinted at in a hidden-gem paragraph in section 5 of [12].

Let us revisit equation (12) in section III E

$$a^k \bmod N = \prod_{j=0}^{2\,n-1} (a^{2^j} \bmod N)^{k_j} \bmod N \tag{C1}$$

It is pointed out in [12] that, instead of accumulating the partial modular products sequentially, one can accumulate pairwise modular products at nodes of a binary tree, whose original leaves are classically pre-computed values $a^{2^j} \bmod N$. This prepares the entire product in depth $O(\log(n))$ (instead of $2\,n$) multiplications and size $O(n^2)$.

Furthermore, each of the pairwise multiplications can be set up as a binary tree of modular additions and so performed in depth $O(\log(n))$ and size $O(n)$.

Thus the entire modular exponentiation is done in depth $O(\log(n)^2)$ and size $O(n^3)$.

This proposal still uses modular addition as the core building block, and thus we can plug in an emulation of the the modular addition circuit built out of carry-lookahead adders and comparators as in the section III E. This fits well into the polylogarithmic depth promise of [12]. Here we are still in the domain of the purely classical reversible circuits, where the cost of emulation of the Toffoli gate is the dominating cost factor (see our thesis 1).