Bounded depth quantum circuits

Olivia Di Matteo, CS 867/QIC 890, 17 July 2014

1 Motivation and background

1.1 Why should I care about making small circuits?

Qubits don't last forever - in fact, for the most part they don't stick around for very long at all. Qubit *decoherence* makes it highly desirable to come up with quantum circuits that are very time efficient. In response to this problem, classes of quantum circuits called small depth, or bounded depth circuits, have emerged over the last 15 or so years. Based loosely on their classical counterparts, these circuits achieve small depth by taking advantage of the high degree of parallelization offered by the addition of a quantum fanout gate, and of ancilla qubits - in a sense, they trade circuit depth for circuit breadth. Bounded depth circuits can replace their 'normal' counterparts in many interesting, and also commonly encountered situations, leading to circuits that can both scale and perform better.

These notes will give a brief summary of classical and quantum complexity classes for bounded depth circuits, as well as show some significant results which distinguish the two. We will go through a number of examples of how circuits can be parallelized to reduce their depth; the quantum fanout gate will be a major component of these. In particular, we will work through a key proof of a containment result that led to the conclusion that bounded depth quantum circuits are significantly more powerful than their classical counterparts. Finally, we will review some other interesting results and open problems.

1.2 Parallelization of circuits

A quantum circuit can be envisioned as the product of unitaries $L_1, \ldots, L_d$ called 'layers', as shown in Figure 1. The *depth* of a circuit is the number of layers, d.

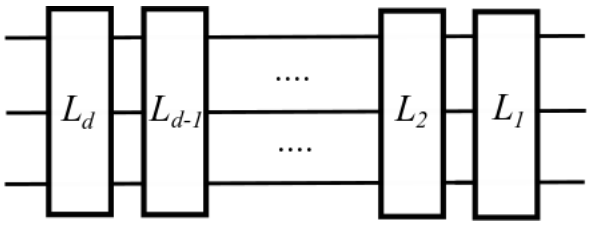

Figure 1: *A circuit U of depth d can be represented as $U = L_1 L_2 \cdots L_d$ where all the L_i are tensor products of unitaries acting on one or more qubits simultaneously. We note that the order of the L_i is reversed for circuits and unitaries, since unitaries are applied from right to left, while circuits work from left to right.*

There are two notions of parallelization which can be taken advantage of in quantum circuits [1]. The first is simple: *gates acting on different qubits can be applied in parallel.* Consider the example circuit in Figure 2. The depth of the circuit looks significantly higher when the gates are applied one at a time, but when we parallelize this, we can see that the depth is halved.

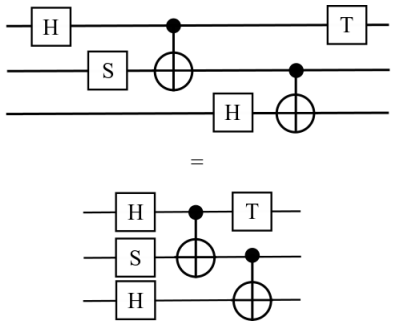

Figure 2: *The first type of parallelization: gates applied to different qubits can be applied at the same time. Here we see that a circuit having apparent depth 6 can be condensed into depth 3.*

The second type of parallelization is that *commuting gates acting on the same qubit can be applied in parallel.* We show an example of this in Figure 3. We will focus primarily on this nontrivial type of parallelization, because it is a special tool indispensable for making small depth circuits. We'll return to Figure 3 in Section 4.2.3.

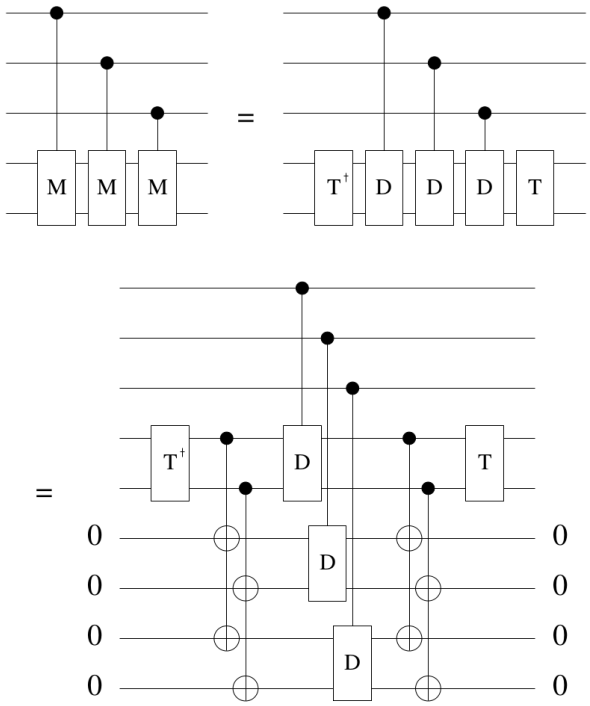

Figure 3: *Commuting gates (here it's the same gate) can be applied simultaneously to the same qubit. First, the gate M is diagonalized as $M = TDT^\dagger$. $T^\dagger$ is applied to the target qubit (recall that circuits perform the layers in reverse), which is then fanned out (entangled) to multiple ancilla qubits. The diagonal gate is applied simultaneously to each ancilla, then everything is unentangled and T is applied again. Image from [4].*

2 Fanout, parity and modulo gates

Before we dive in to the complexity theory, let us define a few important gates.

The most basic of the Boolean gates are AND, OR and NOT. Their truth tables are shown below:

AND

Input 1	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

OR

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

NOT

Input	Output
0	1
1	0

A classical fanout gate takes one input bit, and 'fans' out as many copies of it as desired. It can be imagined physically as soldering additional wires to the input in order to 'split' it. A classical parity gate, on the other hand, is like a fan-in gate. It takes an arbitrary number of bits as input, and computes their parity, i.e. the sum of the input bits modulo 2. Both gates are displayed graphically in Figure 4.

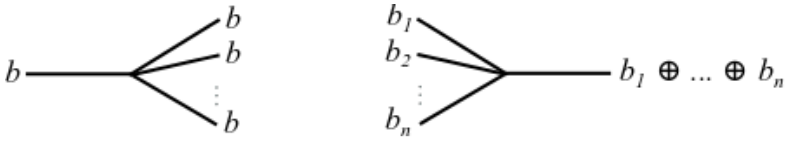

Figure 4: *Circuit diagrams for classical fanout (left) and parity (right).*

We also define the classical Mod_q gate, where

$$\text{Mod}_q(b_1, \ldots, b_n) = \begin{cases} 0 & \text{if } b_1 + \ldots + b_n \equiv 0 \mod q \\ 1 & \text{otherwise.} \end{cases} \tag{1}$$

Notice that the parity gate is just a Mod_2 gate.

The standard gates we'll consider for quantum circuits are the Hadamard, CNOT, and Toffoli:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad \text{TOF} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2}$$

The Toffoli here is pictured for three qubits - it flips the target qubit only if the first two qubits are both in state $|1\rangle$. We'll also consider an unbounded Toffoli gate, which for n input qubits flips a target qubit only if all n inputs are in state $|1\rangle$.

Quantum fanout and parity are defined somewhat differently than in the classical sense. Consider some n qubit state $|x_1, \ldots, x_n\rangle$ where $x_i \in \{0, 1\}$. Mathematically, let us define fanout F_2

and parity MOD_2 as having the action

$$F_2|x_1,\ldots,x_n,b\rangle = |x_1\oplus b,\ldots,x_n\oplus b,b\rangle, \quad (3)$$

$$\mathrm{MOD}_2|x_1,\ldots,x_n,b\rangle = |x_1,\ldots,x_n,b\oplus(x_1\oplus\cdots\oplus x_n)\rangle. \quad (4)$$

Quantum fanout and parity are shown graphically in Figure 5.

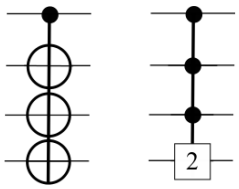

Figure 5: *Circuit diagrams for quantum fanout (left), and parity (right). Note that the quantum fanout gate is commonly drawn 'upside-down' when compared to its definition, with the control bit shown here as the first qubit rather than the last.*

In addition, we define the MOD_q gate, the quantum analog of the classical Mod_q, which has the action

$$\mathrm{MOD}_q|x_1,\ldots,x_n,b\rangle = |x_1,\ldots,x_n,b\oplus\mathrm{Mod}_q(x_1,\ldots,x_n)\rangle. \quad (5)$$

There is an important thing to note here. Quantum fanout means something subtly different than classical fanout. In classical fanout, we are making arbitrarily many copies of the state simply by attaching additional wires. This clearly cannot be the case in the quantum version, due to the no-cloning theorem. The quantum fanout is a means of copying a qubits state to various ancilla by entangling it with them, using a cascade of simultaneous CNOTs with the 'copied' qubit as control. This copying, however, is of the classical bit values rather than the 'true' quantum state.

3 Complexity theory

3.1 Circuit complexity classes

When we speak of small depth circuits, we mean a depth polylogarithmic, or even constant, in the number of input qubits. There are a handful of important complexity classes to which we will frequently refer. We begin with a summary of the classical classes [4]:

Definition (**NC**): *The class* $\mathbf{NC}^i$ *is the set of Boolean circuits with n inputs consisting of NOT, and bounded (i.e. two input) fan-in* AND *and* OR*, which can be executed in depth polylogarithmic in the input size,* $O(\log^i n)$. $\mathbf{NC} = \cup_i \mathbf{NC}^i$.

Definition (**AC**): *The class* $\mathbf{AC}^i$ *is similar to* $\mathbf{NC}^i$*, except that fan-in* AND *and* OR *are allowed to be unbounded, meaning they can have more than two input bits.* $\mathbf{AC} = \cup_i \mathbf{AC}^i$.

Definition (**ACC**): *The class* $\mathbf{ACC}^i[q]$ *is the same as* $\mathbf{AC}^i$ *but with the addition of unbounded* Mod_q *gates.* $\mathbf{ACC}^i = \cup_q \mathbf{ACC}^i[q]$*, and* $\mathbf{ACC} = \cup_i \mathbf{ACC}^i$.

The first quantum class, **QNC**, was defined in 1998 by Moore and Nilsson [2]. **QAC** and **QACC** were defined initially in 1999 by Moore [3], and were elaborated on in a 2001 paper [4] where they proved some of the important containment results which we will be summarizing here. The classes are defined as follows [2, 3, 4]:

Definition (**QNC**): *Let F_n be a family of operators on n qubits. F_n is in $\mathbf{QNC}^i$ if it can be written as a product of $O(\log^i n)$ operators, with a number of ancilla polynomial in n.* $\mathbf{QNC} = \cup_i \mathbf{QNC}^i$.

Definition (**QAC**): *A family of operators F_n is in $\mathbf{QAC}^i$ if it can be written as a product of $O(\log^i n)$ gates and number of ancilla polynomial in n, where the circuit layers consist of tensor products of a finite set of single qubit gates, (unbounded) Toffoli gates, or controlled-not layers.* $\mathbf{QAC} = \cup_i \mathbf{QAC}^i$.

Definition (**QACC**): *The class $\mathbf{QACC}^i[q]$ is the same as $\mathbf{QAC}^i$ but with the addition of MOD_q gates.* $\mathbf{QACC}^i = \cup_q \mathbf{QACC}^i[q]$, *and* $\mathbf{QACC} = \cup_i \mathbf{QACC}^i$.

Please see Appendix A for an alternate definition of **QNC** and a brief discussion.

The classes are shown side by side with their classical counterparts in Table 1. We can also define variants of the classes including quantum fanout gate - these are denoted by subscript 'wf'.

Classical	Quantum
$\mathbf{NC}^i$	$\mathbf{QNC}^i$, $\mathbf{QNC}^i_{wf}$
$\mathbf{AC}^i$	$\mathbf{QAC}^i$, $\mathbf{QAC}^i_{wf}$
$\mathbf{ACC}^i$	$\mathbf{QACC}^i$, $\mathbf{QACC}^i_{wf}$

Table 1: *Quantum analogs of classical complexity classes. The subscripted 'wf' denotes the class with the addition of quantum fanout.*

In what follows, we will work for the most part with $i = 0$, meaning constant depth circuits.

3.2 Containment results

There are a number of interesting results for quantum classes, however we will concern ourselves with the following two, due to their stark contrast with the classical case:

1. $\mathbf{QACC}^0[q] = \mathbf{QACC}^0[p]$ for all q and p not equal to 1 [3], as opposed to $\mathbf{ACC}^0[q] \neq \mathbf{ACC}^0[p]$ for any relatively prime q, p [5].
2. $\mathbf{QAC}^0_{wf} = \mathbf{QACC}^0[2] = \mathbf{QACC}^0$ [4], whereas $\mathbf{AC}^0 \subset \mathbf{ACC}^0[2] \subset \mathbf{ACC}^0$ [5, 6].

4 Parallelization of quantum circuits

4.1 Equivalence of fanout and parity

There is a very close, and useful relationship between quantum fanout and parity (MOD_2). Specifically, they are related by conjugation of each qubit with a Hadamard gate [3, 4], as is presented in Figure 6.

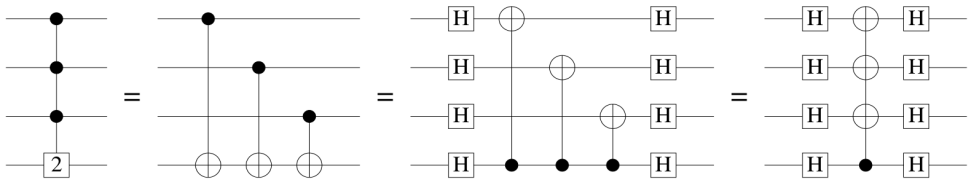

Figure 6: *A parity gate is just a fanout gate conjugated by Hadamards. This is useful because if we have a means of implementing one of these gates in constant depth, we immediately have a means of implementing the other. Image from* [4].

Showing this to be true has been relegated to Appendix B for the perusal of the mathematically inclined. Alternatively, consider it an exercise (it's not that bad). For now, let us carry on with the knowledge that this equivalence holds true.

4.2 Quantum fanout is powerful!

Having a fanout gate immediately leads to the ability to parallelize quantum circuits with specific, but frequently encountered traits. The basic premise of parallelization of circuits to small depth is as follows: *fanout, compute, fanout*. If we are able to parallelize a sequence of gates, first we fanout the inputs to entangle them with ancillae. Next, we apply the gates simultaneously to each ancilla copy. Then, we fanout again to unentangle and return the ancillae to their original states.

We will not prove any of the following propositions rigourously, but instead provide a circuit for each to see them in action. One thing to note is that the original versions of these propositions in [2] assume that fanout can be done in depth $O(\log n)$ using layers of CNOTs, whereas we will show in Section 4.3 that they can in fact be performed in constant depth. Furthermore, these diagrams are all shown for small numbers of qubits - the true advantage of these types of parallelizations is seen with increasing number of qubits.

4.2.1 Same control qubit, multiple target qubits

Proposition *A series of n controlled gates coupling the same input to n target qubits can be parallelized to $O(\log n)$ depth with $O(n)$ ancillae* [2].

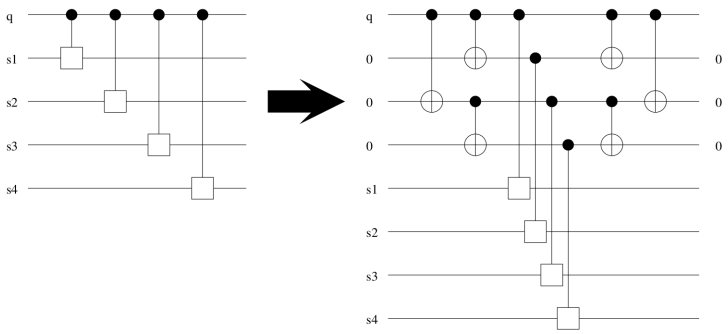

Figure 7: *Multiple gates with the same control bit can be applied simultaneously to multiple target qubits by copying the control bit down to ancillae using a fanout gate (shown here as a layer of* CNOT*s). Gates are applied, and we fanout again to unentangle the ancillae and return them to their original state. Image from* [2].

4.2.2 Diagonal gates on the same qubit

Proposition *A series of n diagonal gates on the same qubits can be parallelized to $O(\log n)$ depth with $O(n)$ ancillae* [2].

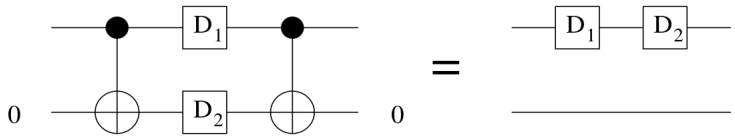

Figure 8: *A sequence of diagonal gates acting on the same qubits can be parallelized by fanning out to ancillae, applying the gates simultaneously, and fanning out again to unentangle the ancillae. Image from* [2].

4.2.3 Commuting gates with multiple control qubits, same target qubit

Proposition *A series of n controlled-U gates acting on the same target qubit(s) where the U's mutually commute can be parallelized to $O(\log n)$ depth with $O(n)$ ancillae* [2].

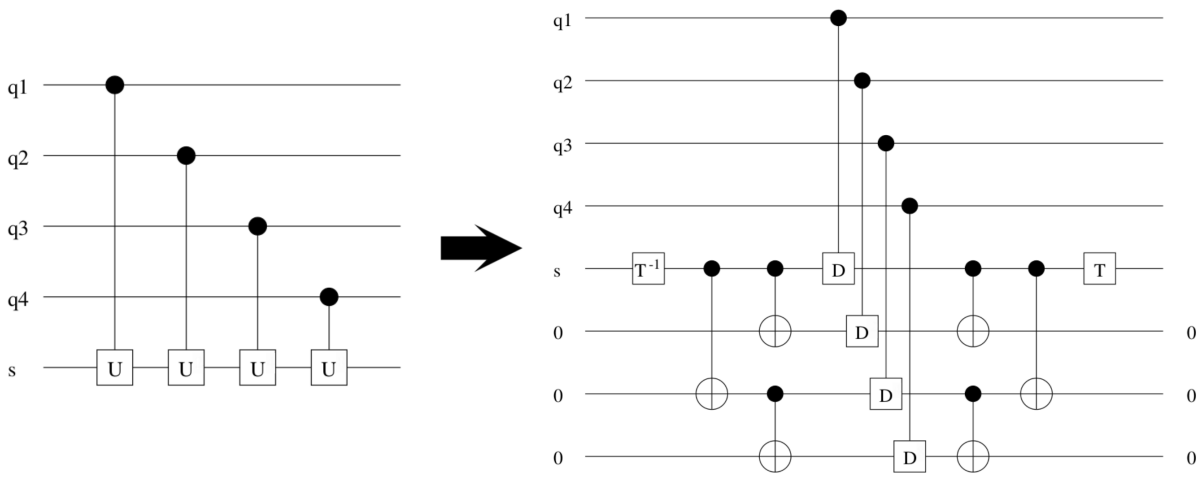

Figure 9: *Gates which commute (or are identical) applied to the same qubit(s) can be parallelized to $O(\log n)$ depth with ancillae by simultaneously diagonalizing them, fanning out, and applying all the diagonal gates simultaneously before fanning out again. In this case, $U = TDT^\dagger$. Image modified from* [2].

4.3 Fanout and parity in constant depth

As previously mentioned, circuits in Section 4.2 show the implementation of a fanout gate on n qubits as taking $O(\log n)$ steps, using a 'divide and conquer'-style layer of CNOTs. However, there is a very nice proposition from [3, 4] that intertwines quantum fanout, parity, and cat states, allowing us to implement each of these in *constant* depth:

Proposition *In any class of quantum circuits that includes Hadamard and controlled-not gates, the following are equivalent:*

1. *It is possible to map* $\alpha|0\rangle + \beta|1\rangle$ *and* $n-1$ *ancillae in the state* $|0\rangle$ *onto an* n*-qubit cat state* $\alpha|0\cdots0\rangle + \beta|1\cdots1\rangle$ *in constant depth.*

2. *The* n*-ary fanout gate can be implemented in constant depth with at most* $n-1$ *additional ancillae.*

3. *An* n*-ary parity or* MOD_2 *gate can be implemented in constant depth with at most* $n-1$ *additional ancillae.*

In other words, if we can make a cat state in constant depth, we can do fanout and parity in constant depth.

We can immediately see that 2 and 3 are equivalent, since fanout and parity are related by a conjugation of Hadamards as we showed in Section 4.1. If we find a constant depth implementation for one of these, we can clearly do the other in that same depth with the addition of just two layers of Hadamards.

We can reason $1 \Rightarrow 3$ using the circuit below in Figure 10. Gates denoted by π are the controlled phase gate with a phase of π,

$$\mathrm{CZ} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, \tag{6}$$

which is often denoted simply as a controlled-Z gate.

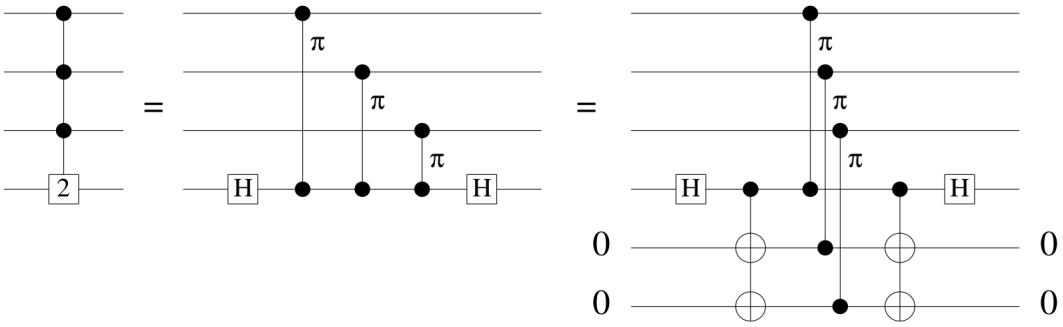

Figure 10: *If we have a means of producing a cat state in constant depth, such as a fanout gate in this case, we can produce a parity gate in constant depth by noticing that the parity gate is a series of controlled-Z gates (denoted by the* π*) with the target conjugated by Hadamards. Image from* [4].

In the 3rd panel of Figure 10, we can see that the parallelization style of Section 4.2.3 was used, which is what allows for parity to be implemented in constant depth. We can also see that fanout is used here as a means of producing a cat state in constant depth. This leads to somewhat circular reasoning, but also shows how $2 \Rightarrow 1$ is somewhat obvious: since the result of a fanout gate is simply a cat state provided we start with the state $|+\rangle$ and all ancillae are in $|0\rangle$, if we can do fanout in constant depth this is equivalent to making a cat state in constant depth. Furthermore, if we didn't use a fanout as a means of making the cat state, we could simply conjugate this

entire circuit by H gates to produce fanout from parity, and the depth would remain constant, completing the proof.

With the ability to do fanout and parity in constant depth, we can finally prove some exceptional results for small quantum circuit classes.

5 $\mathbf{QAC}^0_{wf} = \mathbf{QACC}^0[2] = \mathbf{QACC}^0$

The equivalence

$$\mathbf{QAC}^0_{wf} = \mathbf{QACC}^0[2] = \mathbf{QACC}^0 \tag{7}$$

is the main point we will prove through this lecture. This is an important result, and one of the first of the area. It is in great contrast to the classical case:

$$\mathbf{AC}^0 \subset \mathbf{ACC}^0[2] \subset \mathbf{ACC}^0. \tag{8}$$

It has been proved classically that the parity gate, which is in $\mathbf{ACC}^0[2]$, cannot be simulated in $\mathbf{AC}^0$, leading to the first inclusion [6]. The second comes from the fact that for two integers p and q which are relatively prime,

$$\mathbf{ACC}^0[p] \neq \mathbf{ACC}^0[q], \tag{9}$$

which essentially means you cannot simulate Mod_p gates using Mod_q gates in constant depth (in fact, the depth required would be exponential [5]). However, as we will soon see, this is indeed possible for MOD_q gates in quantum circuits, and we can even use any MOD_q gate to simulate another in constant depth. Since parity is a MOD_2 gate, as long as we have parity we can simulate the rest of $\mathbf{QACC}^0$, making this quantum analog of $\mathbf{ACC}^0$ significantly more powerful.

First, we will see how $\mathbf{QAC}^0_{wf} = \mathbf{QACC}[2]$. Then, we will see a complete proof of how $\mathbf{QACC}[q] \subseteq \mathbf{QACC}[2]$, meaning that using parity gates, we can simulate any MOD_q gate. Showing the converse, that any MOD_q gate can simulate a parity gate, $\mathbf{QACC}^0[2] \subseteq \mathbf{QACC}^0[q]$, is slightly more cumbersome, so a general sketch will be presented, with the details left to [4].

5.1 $\mathbf{QAC}^0_{wf} = \mathbf{QACC}^0[2]$

This equality can be reasoned fairly simply. If we add the quantum fanout gate to $\mathbf{QAC}^0$, this is equivalent to adding the parity gate as well since we can just conjugate the fanout with Hadamards. Similarly, $\mathbf{QACC}^0[2]$ contains the parity gate and is defined as being identical to $\mathbf{QAC}^0$ with only the addition of the parity gate; this means it also contains the fanout gate. Thus, it should be clear that $\mathbf{QAC}^0_{wf} = \mathbf{QACC}^0[2]$.

5.2 $\mathbf{QACC}^0[q] \subseteq \mathbf{QACC}^0[2]$

Formally, we would like to prove the following proposition from [4]:

Proposition *In any circuit class containing n-ary parity gates and one-qubit gates, we can construct an n-ary* MOD_q *gate, with* $O(n \log q)$ *work bits, in depth depending only on* q.

Less formally, we can use parity to simulate any MOD_q gate, using some additional work bits, in depth that depends *only on* q, and not on the size of the inputs.

The proof is as follows, and is for the most part shown graphically with relevant circuits. Let $k = \lceil \log_2 q \rceil$. Consider a quantum state $|x\rangle = |x_{k-1} \cdots x_0\rangle$ on k qubits where $x \in 0, 1, \ldots, k-1$ and the $x_i \in \{0, 1\}$ comprise the binary representation of x.

Let M be a Boolean permutation matrix on k qubits with the following action:

$$M|x\rangle = \begin{cases} |(x+1) \mod q\rangle & \text{if } x < q \\ |x\rangle & \text{if } x \geq q. \end{cases} \tag{10}$$

A matrix of this type means that if we apply M q times, the $|0\rangle$ state will end up back at $|0\rangle$. This cyclic nature is essential to showing why the circuit works.

We can simulate any MOD_q gate with the circuit in Figure 11:

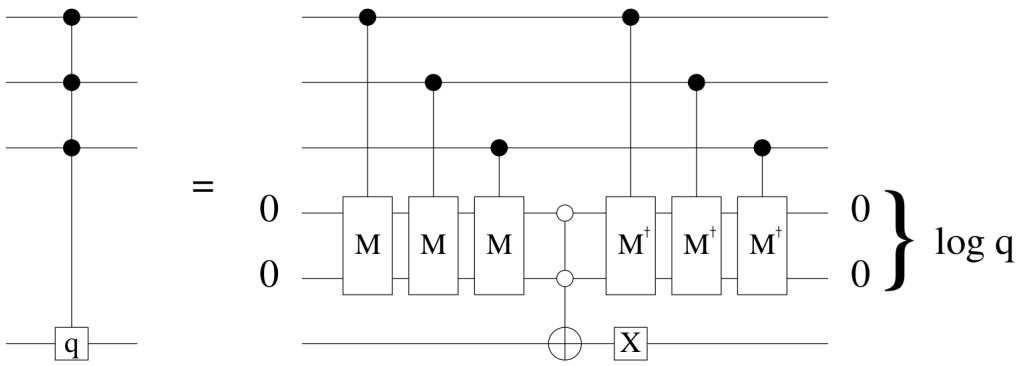

Figure 11: *Using only a permutation matrix M, we can simulate a* MOD_q *gate with the addition of* $\lceil \log_2 q \rceil$ *extra ancillae. The suspicious looking gate between conjugation with M is a generalized Toffoli gate that has had all its inputs conjugated by X. Image from* [4].

The Toffoli gate is written with blank circles instead of filled-in circles for its control bits to denote conjugation by X.

We can do a simple example to show that this circuit works.

Suppose we would like to perform a MOD_5 gate. Then we will need $k = \lceil \log_2 5 \rceil = 3$ ancillae, which will start in the state $|000\rangle$. Suppose our input state is $|0110101\rangle$.

Our matrix M takes the following form, permuting only the first 5 basis states of the 3 qubit ancilla block (the remaining ones are untouched):

$$M = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{11}$$

For each bit of the input state, from left to right, we will apply a controlled-M gate to the ancillae. The state will undergo the following transformation with each input bit:

Input bit ($\|0110101\rangle$)	Ancillae
-	$\|000\rangle$
0	$\|000\rangle$
1	$\|001\rangle$
1	$\|010\rangle$
0	$\|010\rangle$
1	$\|011\rangle$
0	$\|011\rangle$
1	$\|100\rangle$

With the resultant state $|100\rangle$, we conjugate by X, giving $|011\rangle$. Now these 3 qubits become the input for a Toffoli gate on the output. Since not all the bits are 1, there is no effect on the target bit. Assuming the target bit started in state $|0\rangle$, it subsequently gets flipped by X and outputs '1', as it should, since the sum of the input bits in 0110101 is 4, which is *not* 0 mod 5. After the output is produced, we conjugate the ancillae again by X and apply the inverse of M in order to unentangle all the ancillae.

From this example we can see why M must have the structure it does - the only way for the circuit to output 0, meaning the input was 0 mod q, is if the output bit was 1 before the X gate was applied. This means that all the input ancillae had to have been 1, meaning that they must have all been 0 before they too had the X gate applied. The only way for all the ancillae to have been 0 before the Toffoli is if all the n input qubits were in state 0, in which case either no M gates were ever applied to the ancillae, or a multiple of q M gates were applied (since M is cyclic, after q applications of M we'll be back at 0). A number of Ms not a multiple of q would have shifted the state away from all 0s.

The circuit in Figure 11 works just fine, but by taking advantage of the fact that we are applying identical controlled gates to the same target qubit, we can parallelize this circuit down to depth depending only on q by diagonalizing the gate M as $TDT^\dagger$. This is shown in Figure 12.

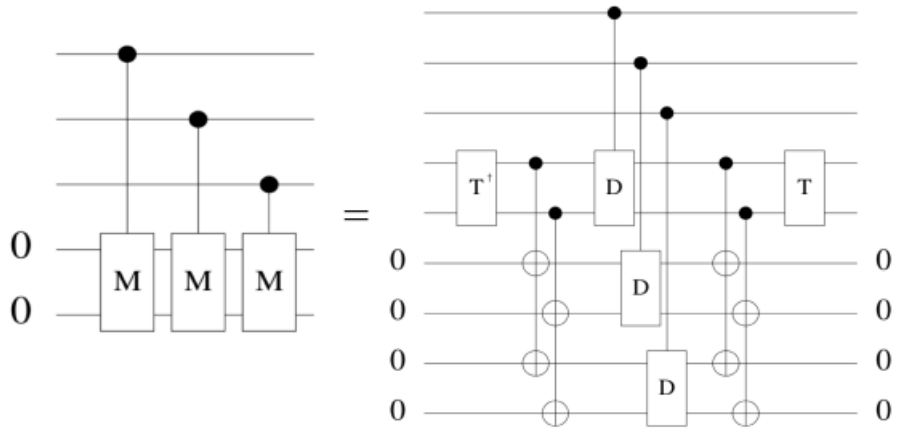

Figure 12: *We can parallelize our circuit for MOD_q using the proposition in Section 4.2.3, where we diagonalize M as $TDT^\dagger$. This circuit represents only the first half of Figure 11. The second half is analogous, but we diagonalize $M^\dagger$ instead. Image from* [4].

The fanout gate can be performed in constant depth, as per the result of Section 4.3. Fanout is equivalent to parity, so we can just as well imagine that we are using parity gates here. The dependence of depth on q arises from the fact that both T and D will have depths dependent on

k, and thus on q. A gate on k qubits can be performed with $O(k^3 4^k)$ two qubit gates without any ancillae [7], so T and D can be done in depth $O(q^2 \log^3 q)$. The number of ancillae required is $(n-1)k$, which does depend on n, however, the depth of the circuit itself clearly no longer depends on the number of inputs, which leads to the ability to implement MOD_q circuits in constant depth using only MOD_2.

5.3 $\mathbf{QACC}^0[2] \subseteq \mathbf{QACC}^0[q]$

This containment was conjectured by Moore in the final remarks of [3], but was suspected not to be true. Two years later in [4], a proof had emerged. We will not cover the complete proof here, but rather sketch the general process by which it can be achieved.

To prove this direction, we must first define the notion of $\mathbf{QAC}^0$ equivalence between two families of operators.

Definition *Let $\{F_n\}$ and $\{G_n\}$, $G_n, F_n \in U(2^n)$ be families of operators. We say $\{F_n\}$ is reducible to $\{G_n\}$ if there is a family $\{R_n\}, R_n \in U(2^{n+p(n)})$ of* $\mathbf{QAC}^0$ *operators augmented with operators from $\{G_n\}$ such that for all $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$, there is a setting of $z_1, \ldots, z_{p(n)} \in \{0,1\}$ for which $\langle \mathbf{y}|F_n|\mathbf{x}\rangle = \langle \mathbf{y}, \mathbf{z}|R_n|\mathbf{x}, \mathbf{z}\rangle$. Operator families are* $\mathbf{QAC}^0$ equivalent *if they are* $\mathbf{QAC}^0$ *reducible to each other.*

In other words, given two families of operators, they are $\mathbf{QAC}^0$ equivalent if we can simulate the action of the first family using only $\mathbf{QAC}^0$ operators and those of the second family with some ancillae, and vice versa.

The authors define the generalized gates

$$M_q|x_1, \ldots, x_n, b\rangle = |x_1, \ldots, x_n, (b + x_1 + \ldots + x_n) \bmod q\rangle, \tag{12}$$

$$F_q|x_1, \ldots, x_n, b\rangle = |(x_1 + b) \bmod q, \ldots, (x_n + b) \bmod q,\ b\rangle, \tag{13}$$

where the x_i are no longer limited to the 0 and 1 of qubits, but instead are qudits. Of course, for the case where $q = 2$, F_2 is the standard qubit fanout gate as previously defined, and M_2 is the qubit parity gate.

The steps they take are as follows:

1. Show that F_q and M_q are $\mathbf{QAC}^0$ equivalent. This is done by showing

$$M_q = (H_q^{\otimes(n+1)})^{-1}(F_q)(H_q^{\otimes(n+1)}), \tag{14}$$

   where H_q is the quantum Fourier transform, or the generalized Hadamard gate for qudits. This fact is analogous to how F_2 is equal to parity conjugated by Hadamard gates, and the math can be done in a similar way as that in Appendix B.

2. Show that MOD_q and M_q are $\mathbf{QAC}^0$ equivalent. This is the cumbersome step, and is done primarily using a series of circuits.

3. As a result of steps 1 and 2, it follows that F_q and MOD_q are $\mathbf{QAC}^0$ equivalent.

4. From the proof of the other direction, we know that MOD_q is $\mathbf{QAC}^0$ equivalent to F_2 because we can use F_2 conjugated by Hadamards to get MOD_2, which we now know can be used to make any MOD_q gate.

5. Finally, this means that F_q must be $\mathbf{QAC}^0$ equivalent to F_2, meaning that we can use $\mathbf{QAC}^0$ operations augmented with F_2 and some ancillae to obtain F_q, and vice versa. This means we can make F_q given only some F_2, and M_q given only some MOD_2, and finally MOD_q with only the MOD_2 as desired.

It is thus clear how $\mathbf{QAC}^0_{wf} = \mathbf{QACC}^0[2] = \mathbf{QACC}^0$. Similarly, we can also see how $\mathbf{QACC}^0[q] = \mathbf{QACC}^0[p]$ - if MOD_q gates can be used to make parity gates, then we can use those parity gates to make any MOD_p gate, so MOD_q gates effectively simulate any other MOD_p gate!

6 Other interesting results and open questions

Many results have emerged from the study of small depth quantum circuits. Here are a few interesting ones:

- Sorting, arithmetic, phase estimation, and the quantum Fourier transform can all be approximated in constant depth with polynomially small error [1].
- Stabilizer codes are in $\mathbf{QNC}^1$, which means that encoding and decoding using n qubit code words can be done in depth $O(\log n)$ with $O(n^2)$ ancillae [2].
- There is another class which we did not mention, $\mathbf{QTC}^0$, which is like $\mathbf{QAC}^0$ but with the addition of a threshold gate (i.e. the circuit outputs 1 if the Hamming weight of the input state is greater than some threshold). Recently, it was proved that $\mathbf{QNC}^0_{wf} = \mathbf{QAC}^0_{wf} = \mathbf{QTC}^0_{wf}$ [10]. This is a very powerful result! It greatly contrasts the classical case $\mathbf{NC}^0 \subset \mathbf{AC}^0 \subset \mathbf{TC}^0$. [8]

There are also a number of open questions in the field, with one in particular remaining elusive: does $\mathbf{QAC}^0 = \mathbf{QAC}^0_{wf}$? There is some preliminary evidence to refute this though it has yet to be deemed conclusive. This evidence hinges on the fact that parity requires a minimum of $\log n$ gates, when not implicitly assumed to be doable in constant depth, leading to the fact that parity and consequently fanout could not be in $\mathbf{QAC}^0$ [9]. As of the writing of [10] in 2012, this problem and a similar one, asking whether $\mathbf{QTC}^0 = \mathbf{QTC}^0_{wf}$, have yet to be solved, and no more recent papers could be found on the topic.

Another very important problem is the number of ancillae required for these circuits. We mention in the introduction that bounded depth circuits trade depth for breadth by the addition of polynomially many ancillae - surely, too many ancillae might counteract any benefits we obtain by shrinking the depth of our circuit. We may reduce the number of errors that occur due to decoherence, but more ancillae means more gates and thus increases the chance for gate errors [2]. The tradeoff between depth and the number of ancillae is explored in [9]. More recently, however, researchers have found different techniques using measurement-based quantum computing to eliminate the ancillae while attempting to keep the depth low [11].

Either way, we have seen how small and constant depth circuits can be quite useful. They have already been shown to be more powerful than their classical counterparts. Also, with or without ancillae, they will likely be indispensable as quantum computers take shape. Efficiency in time is key when dealing with systems that decohere so quickly, so the ability to parallelize as much as possible will be an asset.

References

[1] P. Høyer and R. Špalek. "Quantum Fan-out is Powerful" arXiv:quant-ph/0208043v4

[2] C. Moore and M. Nilsson. "Parallel Quantum Computation and Quantum Codes" arXiv:quant-ph/9808027v1

[3] C. Moore. "Quantum Circuits: Fanout, Parity and Counting" arXiv:quant-ph/9903046v3

[4] F. Green, S. Homer, C. Moore and C. Pollett. "Counting, Fanout, and the Complexity of Quantum ACC" arXiv:quant-ph/0106017v1

[5] R. Smolensky. "Algebraic methods in the theory of lower bounds for Boolean circuit complexity." *Proc. 19th Annual ACM Symposium on Theory of Computing* (1987) 77-82

[6] A. A. Razborov. "Lower bounds on the size of bounded depth circuits over a complete basis with logical addition." *Math. notes of the Academy of Science of the USSR* (1987) **41** (4) 333-338.

[7] A. Barenco et. al. "Elementary gates for quantum computation" (1995) Phys. Rev. A **52** 3457-3467

[8] `https://complexityzoo.uwaterloo.ca/Complexity_Zoo` ($\mathbf{TC}^0$, $\mathbf{AC}^0$ articles)

[9] M. Fang, S. Fenner, F. Green, S. Homer, and Y. Zhang. "Quantum Lower Bounds for Fanout" arXiv:quant-ph/0312208v1

[10] Y. Takahashi and S. Tani. "Collapse of the Hierarchy of Constant-Depth Exact Quantum Circuits" arXiv:quant-ph/11126063v2

[11] R. Dias da Silva, E. Pius and E. Kashefi. "Global Quantum Circuit Optimization" arXiv:quant-ph/13010341v1

[12] S. Fenner, F. Green, S. Homer and Y. Zhang. "Bounds on the Power of Constant-Depth Quantum Circuits" (2005) *Fundamentals of Computation Theory*, volume 3623 of *Lecture Notes in Computer Science*, pp. 44-55

The Complexity Zoo at `https://complexityzoo.uwaterloo.ca/Complexity_Zoo` has overall been a very useful resource and contains many additional details for both classical and quantum bounded-depth circuits.

A guest column in the SIGACT News Complexity Theory Column 55, "Small Depth Quantum Circuits" by D. Bera, F. Green and S. Homer (2007) provides a concise summary of the field and is a good starting point.

Appendix A: QNC vs. QAC

Recall the definitions of **NC** and **AC**.

Definition (**NC**): *The class* $\mathbf{NC}^i$ *is the set of Boolean circuits with n inputs consisting of NOT, and bounded (i.e. two input) fan-in* AND *and* OR*, which can be executed in depth polylogarithmic in the input size,* $O(\log^i n)$*.* $\mathbf{NC} = \cup_i \mathbf{NC}^i$.

Definition (**AC**): *The class* $\mathbf{AC}^i$ *is similar to* $\mathbf{NC}^i$*, except that fan-in* AND *and* OR *are allowed to be unbounded, meaning they can have more than two input bits.* $\mathbf{AC} = \cup_i \mathbf{AC}^i$.

For these classical classes, it would appear that for a given i, $\mathbf{NC}^i$ is weaker than $\mathbf{AC}^i$ due to the bounds place on its AND and OR gates. It is quite clear that $\mathbf{NC}^i \subset \mathbf{AC}^i$.

The quantum classes, on the other hand, at first glance appear to have a very different relationship.

Definition (**QNC**): *Let* F_n *be a family of operators on n qubits.* F_n *is in* $\mathbf{QNC}^i$ *if it can be written as a product of* $O(\log^i n)$ *operators, with a number of ancilla polynomial in n.* $\mathbf{QNC} = \cup_i \mathbf{QNC}^i$.

Definition (**QAC**): *A family of operators* F_n *is in* $\mathbf{QAC}^i$ *if it can be written as a product of* $O(\log^i n)$ *gates and number of ancilla polynomial in n, where the circuit layers consist of tensor products of a finite set of single qubit gates, (unbounded) Toffoli gates, or controlled-not layers.* $\mathbf{QAC} = \cup_i \mathbf{QAC}^i$.

Here, **QNC** looks like the 'broader' class, since there is no immediate stipulation on which gates may be used - so long as we can implement that circuit in polylogarithmic depth, we can use whatever (presumably universal) gate set we'd like. **QAC** appears to be limited in the sense that we're restricted to single qubit gates, CNOTs, and Toffolis (which are unbounded, in analogy with the unbounded AND gates of **QAC**).

Let us consider instead for **QNC** a different definition, from [12]:

Definition $\mathbf{QNC}^k$ *is the class of quantum circuit families* $\{C_n\}_{n\geq 0}$ *for which there exists a polynomial p such that each* C_n *contains n input qubits and at most p(n) ancillae. Each* C_n *has depth* $O(\log^k n)$ *and uses only single-qubit gates and* CNOT *gates. The single-qubit gates must be from a fixed finite set.*

This alternate definition arose from the need of these additional constraints in [12], namely for "technical reasons", however it provides perhaps a more useful working definition than the original does. It also more closely mirrors the classical case, as **QNC** now appears to be less powerful than **QAC** because it is lacking the unbounded Toffoli gate.

Appendix B: Mathematical equivalence of fanout and parity

As per Figure 6, we will show that fanout and parity are equivalent by conjugation of Hadamards.

Let us begin with $n+1$ qubits organized into in any n-qubit input state $|\mathbf{x}\rangle = |x_1\rangle|x_2\rangle\cdots|x_n\rangle$, $x_i \in \{0,1\}$, and some bit b to hold the output:

$$|\phi\rangle = |\mathbf{x}\rangle|b\rangle. \tag{15}$$

Application of the Hadamard to state $|\phi\rangle$ produces

$$(H^{\otimes(n+1)})|\mathbf{x}\rangle|0\rangle = \frac{1}{\sqrt{2}^{n+1}}\left(\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\right)(|0\rangle+|1\rangle), \tag{16}$$

where $\mathbf{x}\cdot\mathbf{y} = \sum_{i=1}^n x_i y_i$ is the dot product modulo 2.

For the next step, it is useful to expand this expression like so:

$$(H^{\otimes(n+1)})|\mathbf{x}\rangle|0\rangle = \frac{1}{\sqrt{2}^{n+1}}\left(\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\cdot|0\rangle + \sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\cdot|1\rangle\right). \tag{17}$$

Now, we must apply a series of CNOT gates (i.e. a fanout, F_2) from the output qubit back to every qubit in the initial state. In the term of Eq. (17) attached to the $|0\rangle$ ket, nothing will happen. In the other term, every $|\mathbf{y}\rangle$ will have all its bits y_i flipped:

$$\begin{aligned}(F_2)(H^{\otimes(n+1)})|\mathbf{x}\rangle|0\rangle &= \frac{1}{\sqrt{2}^{n+1}}\left(\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\cdot|0\rangle + \sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|y_1\oplus 1\rangle\cdots|y_n\oplus 1\rangle\cdot|1\rangle\right)\\ &= \frac{1}{\sqrt{2}^{n+1}}\left(\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\cdot|0\rangle + \sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\sum_{i=1}^n x_i y_i}|y_1\oplus 1\rangle\cdots|y_n\oplus 1\rangle\cdot|1\rangle\right).\end{aligned}$$

Let us make a change of variables, letting $z_i = y_i \oplus 1$. Then we can write this as

$$(F_2)(H^{\otimes(n+1)})|\mathbf{x}\rangle|0\rangle = \frac{1}{\sqrt{2}^{n+1}}\left(\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\cdot|0\rangle + \sum_{\mathbf{z}\in\{0,1\}^n}(-1)^{\sum_{i=1}^n x_i(z_i\oplus 1)}|z_1\rangle\cdots|z_n\rangle\cdot|1\rangle\right).$$

We can now expand the exponent in the second term, pull part of it out of the sum, and relabel our variables:

$$\begin{aligned}(F_2)(H^{\otimes(n+1)})|\mathbf{x}\rangle|0\rangle &= \frac{1}{\sqrt{2}^{n+1}}\left(\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\cdot|0\rangle + (-1)^{\sum_{i=1}^n x_i}\sum_{\mathbf{z}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{z}}|\mathbf{z}\rangle\cdot|1\rangle\right)\\ &= \frac{1}{\sqrt{2}^{n+1}}\left(\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\cdot|0\rangle + (-1)^{\sum_{i=1}^n x_i}\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\cdot|1\rangle\right)\\ &= \frac{1}{\sqrt{2}^{n+1}}\left(\sum_{\mathbf{y}\in\{0,1\}^n}(-1)^{\mathbf{x}\cdot\mathbf{y}}|\mathbf{y}\rangle\right)\cdot\left(|0\rangle + (-1)^{\mathrm{Mod}_2(\mathbf{x})}|1\rangle\right)\end{aligned} \tag{18}$$

where the term $\sum_{i=1}^{n} x_i$ in the exponent is simply the classical parity of $\mathbf{x}$, $\mathrm{Mod}_2(\mathbf{x})$.

This product is exactly the result of applying a Hadamard matrix to the state $|\mathbf{x}\rangle|\mathrm{Mod}_2(\mathbf{x})\rangle$. Since the Hadamard is its own inverse, applying it to the state resulting from Eq. (18) will produce:

$$\begin{aligned}(H^{\otimes(n+1)})(F_2)(H^{\otimes(n+1)})|\mathbf{x}\rangle|0\rangle &= |\mathbf{x}\rangle|\mathrm{Mod}_2(\mathbf{x})\rangle. &(19)\\ &= \mathrm{MOD}_2\left(|\mathbf{x}\rangle|0\rangle\right) &(20)\end{aligned}$$

Therefore, conjugating a fanout gate with Hadamards is exactly the same as a parity gate. A similar process can be used to show the converse, that conjugating a parity gate with Hadamards will product a fanout.