A Rich Type System for Quantum Programs

Aarthi Sundaram[1], Robert Rand[2], Kartik Singhal[2], and Brad Lackey[1]

[1]Microsoft Quantum, Redmond, WA
[2]University of Chicago, Chicago, IL

We show that Gottesman's semantics (GROUP22, 1998) for Clifford circuits based on the Heisenberg representation can be treated as a type system that can efficiently characterize a common subset of quantum programs. Our applications include (i) certifying whether auxiliary qubits can be safely disposed of, (ii) determining if a system is separable across a given bi-partition, (iii) checking the transversality of a gate with respect to a given stabilizer code, and (iv) typing post-measurement states for computational basis measurements. Further, this type system is extended to accommodate universal quantum computing by deriving types for the T-gate, multiply-controlled unitaries such as the Toffoli gate, and some gate injection circuits that use associated magic states. These types allow us to prove a lower bound on the number of T gates necessary to perform a multiply-controlled Z gate.

1 Introduction

Type systems have long been a central feature of quantum programming languages. Simple type systems allow us to determine before running a program that a given variable refers to an integer or a character, preventing us from attempting to perform invalid operations like negating a character. More powerful type systems allow us to guarantee that a variable is not aliased [30] or copied [35], or that it represents a list with a certain fixed length [36]. In this vein, the quantum lambda calculus [28] introduced linear types for guaranteeing the no-cloning theorem of quantum mechanics and QWIRE [21, 26] and Proto-Quipper [10] added dependent types for precisely specifying the size and structure of circuits. However, the basic type in all of these systems is the *qubit* from which we can build pairs or more complex data structures of qubits. This work asks whether we can provide richer types at the qubit level to describe whether (say) a qubit is in a given basis state or separable from the broader system.

To answer this question, we present a type system inspired by the stabilizer formalism used to efficiently simulate the action of Clifford circuits. We extend this system to handle universal quantum gate sets, both by explicitly adding the T gate and by showing how to handle arbitrary unitary gates. We also expand the system to handle measurement on stabilizer circuits and a restricted set of Clifford+T circuits. In general, type checking (proving that a program has the desired type) should be linear in the number of commands

Aarthi Sundaram: aarthi.sundaram@microsoft.com
Robert Rand: rand@uchicago.edu
Kartik Singhal: ks@cs.uchicago.edu
Brad Lackey: brad.lackey@microsoft.com

in a circuit: here, this is true for unitary Clifford circuits. However, given the expressiveness of our type system and the power of quantum computation, efficiency will not always be guaranteed, particularly for circuits with a large number of non-Clifford gates.

The starting point to understanding our system is the Heisenberg interpretation of quantum mechanics. This interpretation treats quantum operators as functions on operators, rather than on quantum states. For instance, given an arbitrary quantum state $|\phi\rangle$, the Hadamard operator H satisfies

$$HZ\,|\phi\rangle = XH\,|\phi\rangle\,. \tag{1}$$

In other words, the operator H can be viewed as a function that takes Z to X and similarly takes X to Z. Gottesman [12] used this representation to present the rules for how the Clifford set (H, S and $CNOT$) operates on Pauli X and Z matrices. Thus, H is given the following description based on its action above:

$$H : \mathbf{X} \to \mathbf{Z} \quad H : \mathbf{Z} \to \mathbf{X} \tag{2}$$

where $\mathbf{X}, \mathbf{Z}$ are used to denote the types corresponding to the Pauli X and Z matrices. Note that it suffices to just specify H on $\mathbf{X}$ and $\mathbf{Z}$ as we can derive the action of H on $\mathbf{Y}$ by treating the operator Y as iXZ (since $\sigma_y = i\sigma_x\sigma_z$). More specifically,

$$\begin{aligned} HY\,|\phi\rangle &= H(iXZ)\,|\phi\rangle \\ &= i(HX)Z\,|\psi\rangle \\ &= i(ZH)Z\,|\psi\rangle \\ &= iZ(HZ)\,|\psi\rangle \\ &= iZXH\,|\psi\rangle \\ &= -YH\,|\psi\rangle \end{aligned}$$

Throughout this work we develop a logic motivated on this semantic interpretation of types. Formally the syntax of the logic, as found in Figure 4 and Figure 5, is axiomatic with the semantics as described above being a sound interpretation. For example we can represent the general form of this last deduction by the following *typing rule*:

$$\frac{U : \mathbf{X} \to \mathbf{A} \qquad U : \mathbf{Z} \to \mathbf{B}}{U : \mathbf{Y} \to \mathbf{i}(\mathbf{AB})}\ \mathrm{Y}$$

Here $\mathbf{A}$ and $\mathbf{B}$ are assumed to be Paulis, so the product of $\mathbf{A}$ and $\mathbf{B}$ is simply the third Pauli, possibly negated or multiplied by i. This is indicative (and a special case) of the kinds of typing judgments we will use throughout the paper.

In Gottesman's paper, the end goal was to fully describe quantum programs and prove the *Gottesman-Knill theorem*, which shows that any Clifford circuit can be classically simulated efficiently. In our case, we observe that the transformations in eq. (2) look like typing judgments and build our system from there (§2). Furthermore, we move beyond Clifford circuits and expand the typing judgments to characterize some magic states, the T gate, and other gates in the Clifford hierarchy (§7). A key feature of our system is that the base types correspond to unitary Hermitian operators: when restricted to stabilizer quantum computing, these are (tensor products of) Pauli matrices, and for universal quantum computing, they are general unitary Hermitian matrices. Notationally, we use uppercase letters $U, V, \ldots$ to denote unitary gates or matrices and the boldface $\mathbf{U}, \mathbf{V}, \ldots$ to denote the corresponding types.

The semantics of our types In our system, a *typing judgment* of the form $|\psi\rangle : \mathbf{P}$ admits a straightforward interpretation: $|\psi\rangle$ is a $+1$ eigenstate of $\mathbf{P}$. In the context of a function, or circuit, $U : \mathbf{A} \to \mathbf{B}$ means that U maps a $+1$ eigenstate of A to a $+1$ eigenstate of $\mathbf{B}$. This closely mirrors the stabilizer formalism used for error correcting codes [11]. It works well as long as we restrict to Clifford circuits and are fine with very coarse judgements in the face of measurements. However, for more accurate typing judgements when measurements are performed and to work with more general gates, we will associate $|\psi\rangle : \mathbf{P}$ with the fact that $|\psi\rangle$ lies in the image of the projection $\Pi_P^+ := \frac{1}{2}(I+P)$ i.e., $\frac{1}{2}(I+P)|\psi\rangle = |\psi\rangle$.

We use the tensor operand $\otimes$ to represent multi-qubit types. Using our first interpretation, $|\psi\rangle : \mathbf{A} \otimes \mathbf{B}$ if $|\psi\rangle$ is a $+1$-eigenstate of $A \otimes B$. Observe that this does not restrict $|\psi\rangle$ to be a product state $|\phi_1\rangle \otimes |\phi_2\rangle$ with $|\phi_1\rangle : \mathbf{A}$ and $|\phi_2\rangle : \mathbf{B}$. In fact, $|\psi'\rangle : \mathbf{A} \otimes \mathbf{B}$ even when $|\psi'\rangle = |\phi_1'\rangle \otimes |\phi_2'\rangle$ such that $|\phi_1'\rangle : -\mathbf{A}$ (i.e., $|\phi_1'\rangle$ is a -1-eigenstate of A) and $|\phi_2'\rangle : -\mathbf{B}$. Moreover, arbitrary superpositions of $|\psi\rangle$ and $|\psi'\rangle$ also have type $\mathbf{A} \otimes \mathbf{B}$. This interpretation lifts easily to arrow types. Under our projection-based semantics, $|\psi\rangle : \mathbf{A} \otimes \mathbf{B}$ if $\frac{1}{2}(I \otimes I + A \otimes B)|\psi\rangle = |\psi\rangle$.

We borrow the notion of *intersection types* from programming language theory, and particularly the study of subtyping. As is common in such systems, $|\psi\rangle : \mathbf{P} \cap \mathbf{Q}$ means $|\psi\rangle : \mathbf{P}$ and $|\psi\rangle : \mathbf{Q}$ implying that $|\psi\rangle$ is *simultaneously* a $+1$-eigenstate of P and Q. In this case, P and Q must commute because Pauli operators that do not commute, instead anticommute and, have no common eigenvectors. In our projection semantics, $|\psi\rangle : \mathbf{P} \cap \mathbf{Q}$ is simultaneously in the image of the two projections Π_P^+ and Π_Q^+.

Finally we use the notion of *union types* to represent post-measurement states when the outcome is probabilistic. In this case, $|\psi\rangle : \mathbf{A} \cup \mathbf{B}$ denotes that the system is either a $+1$-eigenstate A or a $+1$-eigenstate of B. In the measurement context, it means that one outcome results in the system having type $\mathbf{A}$ and the other outcome will result in type $\mathbf{B}$. In our projection semantics, it implies that the system is either in the image of Π_A^+ or in the image of Π_B^+.

Applications Our syntax and typing rules for Clifford circuits and stabilizer states are methodically developed in §2. The full list of our rules are in Figures 4 and 5. The most straightforward use of our system is in characterizing properties of Clifford circuits, particularly entanglement and separability. For the textbook case of Deutsch's algorithm, we are easily able to verify three key properties: (i) the first qubit is $|0\rangle$ whenever the function is constant, (ii) the first qubit is $|1\rangle$ whenever the function is balanced, and (iii) that the two qubits are never entangled, and therefore the second can be safely discarded. These are three common and broadly useful properties to check.

A key property of type systems that allows us to statically verify program properties is determining the equivalence or equality of types. This can be done if we have a canonical representation for our types. Inspired by the *row echelon form* of a matrix, we describe in §3 an efficient algorithm to generate a canonical representation for our intersection types. This allows us to use the type system to *efficiently* track whether a given sub-system is separable from the rest of the system in §4. In §4.3, we generate and then disentangle a GHZ state $\frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)$ to show how the type system is capable of tracking both the creation and destruction of entanglement.

A crucial method used by quantum circuits to extract or output classical information is measurement (usually in the computational basis). It is challenging to tune our type system to accommodate measurement in light of the fact that it requires managing the operation on all the basis states, unlike the evolution of a single Pauli operator. However,

measurement on stabilizer states is well understood and this allows us to construct a procedure to generate a measurement outcome and post-measurement as discussed in §5. In particular, when the measurement outcome is random, we use union types to capture the fact that the system could have one of the many possible types depending on the outcome. As a simple example, applying a z-basis measurement on an $\mathbf{X}$ type to get a random 0 or 1 outcome is represented as

$$\frac{}{\mathsf{Meas} : \mathbf{X} \to \mathbf{Z} \cup -\mathbf{Z}}$$

Using all the elements described above, in §6, we demonstrate how our type-system can be used to verify the working of a stabilizer error correcting code (the Steane code on 7 qubits [32]). Specifically, we (i) derive the type for a logical qubit in the Steane code; (ii) verify that the encoding circuit constructs the appropriate logical qubit state; and (iii) show the transversality of the H and ZS gates as well as the non-transversality of the T gate for the Steane code.

Additive types and their applications All of the ideas discussed up to this point deal with the realm of stabilizer states and Clifford circuits which are not universal for quantum computing. For instance, while we can add the axiom $T : \mathbf{Z} \to \mathbf{Z}$ to our system, the stabilizer formalism is incapable of expressing the action of the T gate on $\mathbf{X}$. We address this shortcoming by developing *additive types* in §7, which are expressed as linear combinations of our basic types. This allows us to express the complete type for the T gate as $T : (\mathbf{X} \to \frac{1}{\sqrt{2}}(\mathbf{X} + \mathbf{Y})) \cap (\mathbf{Z} \to \mathbf{Z})$.

Since the T-gate is not the only way to achieve universal quantum computation, we produce a straightforward algorithm for adding new gates to the system (such as the Toffoli) by fully deriving their types. A particularly nice application has to do with multiply-controlled Z gates, which have very succinct types. In fact, comparing their types to that of the T gate, we can easily show that synthesizing an n-controlled Z-gate requires at least $(2k - 2)$ T-gates.

In §8, we discuss how to compute the post-measurement type following a single qubit computational basis measurement on single and two qubit additive types. Putting these pieces together in §8.3, we derive the type for gate injection circuits that use associated magic states to implement non-Clifford circuits. We focus on single-qubit unitaries that correspond to a rotation about the Z axis, i.e., that rotate types in the $\mathbf{X}/\mathbf{Y}$-plane by some angle θ.

Typechecking complexity. We conclude with a discussion on the complexity of type-checking in §9. Unsurprisingly, fully characterizing a circuit with high T-depth proves to be intractable in the general case. However, proving interesting properties of circuits with a few T gates is often quite possible. Moreover, Clifford circuits can be efficiently characterized to any degree of precision, allowing us to flexibly analyze a broad range of quantum programs. Note that efficiency here means that the procedure scales linearly with the number of gates in the operation and polynomially in the size of the system.

We place our work in context of related work in §10 and discuss possible future applications and extensions to this system in §11.

2 Our Type System and its Semantics

Here we present the internal syntax of our type system and several semantic interpretations. We will extend this to more general types in §7 below. Our atomic types are denoted $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$. We denote basic operators (or *gates*) by H, S, $CNOT$ and later T.

2.1 Basic Types

Our core interpretation for $\mathbf{X}, \mathbf{Y}$ and $\mathbf{Z}$ is that each of these types is inhabited by a single qubit state, the $+1$-eigenstate of the associated Pauli operator $\sigma_x, \sigma_y, \sigma_z$ respectively. Using the standard quantum computing notation for these states, $|+\rangle, |i\rangle, |0\rangle$ respectively, we have the three axioms.

$$\frac{}{|+\rangle : \mathbf{X}} \qquad \frac{}{|i\rangle : \mathbf{Y}} \qquad \frac{}{|0\rangle : \mathbf{Z}}$$

As stated we could view these axioms as definitions of the symbols $|+\rangle, |i\rangle, |-\rangle$, with this notation selected to emphasize that the interpretation that these states are the $+1$-eigenstates of the associated Pauli operator is sound.

Unlike $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$, the type $\mathbf{I}$ corresponding to the identity matrix is inhabited by every qubit:

$$\frac{}{|\psi\rangle : \mathbf{I}}$$

where at this stage $|\psi\rangle$ refers to any of the states defined above. Of course, in terms of its semantic interpretation (to follow) $|\psi\rangle$ could be any 1-qubit state.

2.2 Arrow types and Clifford operators

To define arrow types, we turn to the characterization of H, S and $CNOT$ in Gottesman [12]:

Proposition 1. *Given a unitary $U : A \to B$ in the Heisenberg interpretation, U takes every eigenstate of A to an eigenstate of B with the same eigenvalue.*

Proof. From eq. [1] in Gottesman [12], given a state $|\psi\rangle$ and an operator U,

$$UN|\psi\rangle = UNU^\dagger U|\psi\rangle .$$

In the Heisenberg interpretation this can be denoted as: $U : N \to UNU^\dagger$. Suppose that $|\psi\rangle$ is an eigenstate of N with eigenvalue λ and let $|\phi\rangle$ denote the state after U acts on $|\psi\rangle$. Then,

$$\lambda|\phi\rangle = U(\lambda|\psi\rangle) = UN|\psi\rangle = UNU^\dagger U|\psi\rangle = (UNU^\dagger)|\phi\rangle .$$

Hence, $|\phi\rangle$ is an eigenstate of the modified operator $UNU^\dagger$ with eigenvalue λ. □

Therefore in our interpretation, $U : \mathbf{A} \to \mathbf{B}$ will mean that U takes a $+1$ eigenstate of A to a $+1$ eigenstate of B.

As a result, our arrow types will be precisely those in Gottesman, though we delay the introduction of $CNOT$ until the next section:

$$\frac{}{H : \mathbf{X} \to \mathbf{Z}} \qquad \frac{}{H : \mathbf{Z} \to \mathbf{X}} \qquad \frac{}{S : \mathbf{X} \to \mathbf{Y}} \qquad \frac{}{S : \mathbf{Z} \to \mathbf{Z}}$$

So, for instance, H takes $|+\rangle$ to $|0\rangle$. We can add an application rule to reflect this:

$$\frac{|\psi\rangle : \mathbf{A} \qquad p : \mathbf{A} \to \mathbf{B}}{p|\psi\rangle : \mathbf{B}}\ \text{APP} \tag{3}$$

Note that in this work we focus on providing types for *quantum programs* (or circuits), not quantum states, so we will not use the application rule. However, it is useful for proving the correctness of circuits that generate resource states – for example, magic state distillation protocols.

Note that every qubit is an $+1$-eigenstate of I and similarly every quantum state is an $+1$-eigenstate of I^k (our notation for $I^{\otimes k}$ where k is the number of qubits in the system) so we have the following rule for any single qubit unitary U:

$$\frac{}{U : \mathbf{I} \to \mathbf{I}}\ \mathbf{I}$$

We combine our arrow types using the standard composition rule from programming languages, equivalent to cut-elimination in many deductive systems:

$$\frac{p_1 : \mathbf{A} \to \mathbf{B} \qquad p_2 : \mathbf{B} \to \mathbf{C}}{p_1; p_2 : \mathbf{A} \to \mathbf{C}}\ \text{SEQ} \tag{4}$$

For instance, here is our derivation of the type for $Z = S; S$ on $\mathbf{Z}$:

$$\frac{\dfrac{}{S : \mathbf{Z} \to \mathbf{Z}} \qquad \dfrac{}{S : \mathbf{Z} \to \mathbf{Z}}}{S; S : \mathbf{Z} \to \mathbf{Z}}\ \text{SEQ}$$

Since S has type $\mathbf{X} \to \mathbf{Y}$, and we only derive the types for unitaries acting upon $\mathbf{Y}$ from the types of the same unitaries acting upon $\mathbf{X}$ and $\mathbf{Z}$, we will need to introduce rules for coefficients and multiplication that generalize the $\mathbf{Y}$ rule presented in the introduction:

$$\frac{p : \mathbf{A} \to \mathbf{B}}{p : \mathbf{cA} \to \mathbf{cB}}\ \text{SCALE} \qquad \frac{p : \mathbf{A} \to \mathbf{B} \qquad p : \mathbf{C} \to \mathbf{D}}{p : \mathbf{AC} \to \mathbf{BD}}\ \text{MUL} \tag{5}$$

In SCALE, $\mathbf{c}$ can be any complex number, although in any derivation that stays within the Clifford group, a well-typed circuit will only use $c \in \{-1, i, -i\}$. We should note that there is no matrix multiplication happening when we apply the MUL rule: There are only 16 possible combinations of two Paulis, each of which produces a Pauli, so we can efficiently simplify these symbolically. The same is true for $c \in \{1, -1, i, -i\}$.

We can now derive the type for Z on $\mathbf{X}$:

$$\frac{\dfrac{}{S : \mathbf{X} \to \mathbf{Y}} \qquad \dfrac{\dfrac{\dfrac{}{S : \mathbf{X} \to \mathbf{Y}} \qquad \dfrac{}{S : \mathbf{Z} \to \mathbf{Z}}}{S : \mathbf{XZ} \to \mathbf{YZ}}\ \text{MUL}}{S : \mathbf{Y} \to \mathbf{iYZ}}\ \text{SCALE}}{S; S : \mathbf{X} \to -\mathbf{X}}\ \text{SEQ}$$

In this deduction, $\mathbf{XZ}$ is simply a notation for $\mathbf{iY}$ included for readability. Likewise, $\mathbf{YZ}$ is simply $\mathbf{iX}$.

We can similarly show that $X := H; Z; H$ has the types $\mathbf{X} \to \mathbf{X}$ and $\mathbf{Z} \to -\mathbf{Z}$ and $Y := S; X; Z; S$ has the types $\mathbf{X} \to -\mathbf{X}$ and $\mathbf{Z} \to -\mathbf{Z}$.

2.3 Tensors and multi-qubit types

In order to do anything interesting, we're going to need to consider multi-qubit systems. The type associated with an n qubit system is $\mathbf{P}_1 \otimes \mathbf{P}_2 \otimes \cdots \otimes \mathbf{P}_n$ for Pauli types $\mathbf{P}$.

We use $\mathbf{T}[i]$ to refer to $\mathbf{P}_i$ from that tensor product and $U\ i$ to apply U to the ith qubit in a quantum state. We can therefore introduce the following typing rule for applying a single-qubit operator to a multi-qubit state:

$$\frac{\mathbf{T}[i] = \mathbf{A} \qquad U : \mathbf{A} \to \mathbf{B}}{U\ i : \mathbf{T} \to \mathbf{T}\{i \mapsto \mathbf{B}\}} \otimes_1 \tag{6}$$

Here $\mathbf{T}\{i \mapsto \mathbf{B}\}$ replaces the ith type in the tensor product with $\mathbf{B}$.

We can now introduce Gottesman's axioms for $CNOT$:

$$\frac{}{CNOT : \mathbf{X} \otimes \mathbf{I} \to \mathbf{X} \otimes \mathbf{X}} \qquad \frac{}{CNOT : \mathbf{I} \otimes \mathbf{X} \to \mathbf{I} \otimes \mathbf{X}}$$

$$\frac{}{CNOT : \mathbf{Z} \otimes \mathbf{I} \to \mathbf{Z} \otimes \mathbf{I}} \qquad \frac{}{CNOT : \mathbf{I} \otimes \mathbf{Z} \to \mathbf{Z} \otimes \mathbf{Z}}$$

To apply $CNOT$ to multi-qubit states, we'll need a new rule:

$$\frac{\mathbf{T}[i] = \mathbf{A} \qquad \mathbf{T}[j] = \mathbf{B} \qquad U : \mathbf{A} \otimes \mathbf{B} \to \mathbf{C} \otimes \mathbf{D}}{U\ i\ j : \mathbf{T} \to \mathbf{T}\{i \mapsto \mathbf{C}; j \mapsto \mathbf{D}\}} \otimes_2 \tag{7}$$

Note that we'll often need to use this in conjunction with the MUL rule, where multiplication distributes over addition. Consider this simple derivation:

$$\frac{\dfrac{\dfrac{(\mathbf{Z} \otimes \mathbf{Y} \otimes \mathbf{X})[1] = \mathbf{Z} \qquad (\mathbf{Z} \otimes \mathbf{Y} \otimes \mathbf{X})[3] = \mathbf{X}}{CNOT : \mathbf{Z} \otimes \mathbf{I} \to \mathbf{Z} \otimes \mathbf{I} \qquad CNOT : \mathbf{I} \otimes \mathbf{X} \to \mathbf{I} \otimes \mathbf{X}}\ \text{MUL}}{CNOT : \mathbf{Z} \otimes \mathbf{X} \to \mathbf{Z} \otimes \mathbf{X}}}{CNOT\ 1\ 3 : \mathbf{Z} \otimes \mathbf{Y} \otimes \mathbf{X} \to \mathbf{Z} \otimes \mathbf{Y} \otimes \mathbf{X}} \otimes_2$$

Noting that $cA \otimes B = c(A \otimes B) = A \otimes cB$ we add two distributive rules for tensors:

$$\frac{U : \mathbf{S} \to \mathbf{T} \qquad \mathbf{T}[i] = c\mathbf{A}}{U : \mathbf{S} \to c\mathbf{T}\{i \mapsto \mathbf{A}\}} \otimes_{c1} \qquad \frac{U : \mathbf{S} \to \mathbf{cT} \qquad \mathbf{T}[i] = \mathbf{A}}{U : \mathbf{S} \to \mathbf{T}\{i \mapsto c\mathbf{A}\}} \otimes_{c2} \tag{8}$$

This rule would also be sound for manipulating the left-hand side of the arrow, but this isn't necessary in practice.

Note that the identity rule also applies to the $CNOT$ gate:

$$\frac{}{CNOT : \mathbf{I} \otimes \mathbf{I} \to \mathbf{I} \otimes \mathbf{I}}\ \mathbf{I}_2$$

On this basis, it's easy to show that $\mathbf{I}^k$ is a universal type for all quantum programs, where $\mathbf{I}^k$ corresponds to $I^{\otimes k}$ and k is greater than or equal to the number of qubits in our program. We further note that this typing judgment subsumes the dependent types of many quantum programming languages (such as $\mathcal{Q}$WIRE's [21] sized tensor type).

2.4 Negation and Complements

Our types admit a unary negation operation i.e.,

> If $\mathbf{P}$ is a type, then $-\mathbf{P}$ is a type.

The core interpretation extends naturally to these types. For a type $\mathbf{P}$, we write $|\psi\rangle : \mathbf{P}$ when $|\psi\rangle$ is a $+1$-eigenstate of the negation of the corresponding matrix. This is equivalent to $|\psi\rangle$ being a -1 eigenstate of the matrix itself. In case $\mathbf{P}$ is one of $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$, then there is only a single state that inhabits this type. We can expand our axioms to include

$$\frac{}{|-\rangle : -\mathbf{X}} \qquad \frac{}{|-i\rangle : -\mathbf{Y}} \qquad \frac{}{|1\rangle : -\mathbf{Z}} \tag{9}$$

using the obvious notation for these states.

2.5 Intersection types

If we want to fully describe an operator's behavior, we need to add an intersection type. These use the standard typing rules for intersection:

$$\frac{g : \mathbf{A} \qquad g : \mathbf{B}}{g : \mathbf{A} \cap \mathbf{B}} \cap\text{-I} \qquad \frac{g : \mathbf{S} \cap \mathbf{T}}{g : \mathbf{S}} \cap\text{-E-L} \qquad \frac{g : \mathbf{S} \cap \mathbf{T}}{g : \mathbf{T}} \cap\text{-E-R} \tag{10}$$

Using these rules we can, for instance, give the fully descriptive type for Hadamard of $H : (\mathbf{X} \to \mathbf{Z}) \cap (\mathbf{Z} \to \mathbf{X})$. However, this isn't enough: often we will want to distribute $\cap$ over the arrow to gain more precise types. We add the following rule, which generally is derivable in most type systems with $\cap$:

$$\frac{g : (\mathbf{A} \to \mathbf{A}') \cap (\mathbf{B} \to \mathbf{B}')}{g : (\mathbf{A} \cap \mathbf{B}) \to (\mathbf{A}' \cap \mathbf{B}')} \cap\text{-Arr-Dist} \tag{11}$$

Now we can show that a $CNOT$ given two 0 qubits returns two 0 qubits:

$$\frac{CNOT : (\mathbf{Z} \otimes \mathbf{I} \to \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{I} \otimes \mathbf{Z} \to \mathbf{Z} \otimes \mathbf{Z})}{CNOT : (\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{Z}) \to (\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{Z} \otimes \mathbf{Z})} \cap\text{-Arr-Dist}$$

Nonetheless, this rule alone may not lead the description of a type we want.[1] We introduce the two "rewrite" rules, which are specific to our system:

$$\cap\text{-MUL-L}\ \frac{g : \mathbf{A} \cap \mathbf{B} \to \mathbf{C}}{g : \mathbf{A} \cap \mathbf{AB} \to \mathbf{C}} \qquad \frac{g : \mathbf{A} \to \mathbf{B} \cap \mathbf{C}}{g : \mathbf{A} \to \mathbf{B} \cap \mathbf{BC}} \cap\text{-MUL-R} \tag{12}$$

Why our semantics is still sound under these rules slightly subtle: if $|\psi\rangle : \mathbf{A} \cap \mathbf{B}$ then in our semantics $|\psi\rangle$ is a $+1$-eigenstate of both (multi-qubit) Pauli operators A and B. But then $|\psi\rangle$ is also a $+1$-eigenstate of AB. The converse is also true (as $A^2 = I$), and so semantically $\mathbf{A} \cap \mathbf{B}$ and $\mathbf{A} \cap \mathbf{AB}$ refer to the same set of states.

We can now add one more line to the derivation above:

$$\frac{CNOT : (\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{Z}) \to (\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{Z} \otimes \mathbf{Z})}{CNOT : (\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{Z}) \to (\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{Z})} \cap\text{-MUL-R}$$

As we will see later (§4), $\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{Z}$ has only the single eigenvalue $|00\rangle$, so this is an identity on $|00\rangle$.

Note that these last rules eqs. (11) and (12) are neither syntax-directed (that is, we don't know where to apply them from the syntax of the preceding judgment alone) nor consistently useful. Consider the following (valid!) typing derivation:

$$\frac{\overline{H : (\mathbf{X} \to \mathbf{Z}) \cap (\mathbf{Z} \to \mathbf{X})}}{H : (\mathbf{Z} \cap \mathbf{X}) \to (\mathbf{X} \cap \mathbf{Z})} \cap\text{-ARR-DIST}$$

This takes a useful typing judgment (a complete characterization of H) and deduces a vacuously true judgment since X and Z have no joint $+1$ eigenstates. §3 will discuss how to put these rules to good use.

[1] In particular, many of the results in the following sections will concern recognizing properties of quantum states or operations based on their types.

2.6 Example: Deutsch's Algorithm

A complete list of our rules and grammar for Gottesman types is given in Figures 4 and 5. Here, we show an example of how we can apply these rules to make non-trivial judgments about quantum programs.

Many quantum circuits introduce ancillary qubits that perform some classical computation and are then discarded in a basis state. Several efforts have been made to verify this behavior: The Quipper [13] and Q# [33] languages allow us to *assert* that ancilla are separable and can be safely discarded, while $\mathcal{Q}$WIRE allows us to manually verify this [27]. More recently, Silq [3] allows us to define "qfree" functions that never put qubits into a superposition. We can use our type system to avoid this restriction and automatically guarantee ancilla correctness by showing that the ancillae are discarded with the type $\mathbf{Z}$ and is separable from the rest of the system.

A simple example to demonstrate this ability to safely discard auxiliary qubits is Deutsch's algorithm [8]. Given a function $f : \{0,1\} \to \{0,1\}$, the algorithm uses oracle access to f and a single auxiliary qubit to determine if f has a constant value or is balanced.

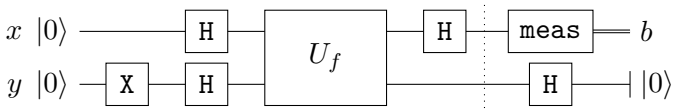

Figure 1: Deutsch's algorithm to check if $f : \{0,1\} \to \{0,1\}$ is constant or balanced

We want to show that the qubit y is never entangled with qubit x despite the application of the oracle $U_f : |x\rangle |y\rangle \to |x\rangle |y \oplus f(x)\rangle$. In this case, it would be safe to discard qubit y just after the dotted line in Figure 1 (i.e., even before measurement destroys any hypothetical entanglement).

Before analyzing the circuit, consider the possible behaviours for f. Acting on a single bit, one can conclude that $f(x) \in \{0, 1, x, (1-x)\}$. Doing a case-by-case analysis, it is easy to derive the oracle application as:

$$\texttt{U}_\texttt{f}\ \texttt{1 2} = \begin{cases} \texttt{I 2} & \text{if } f(x) = 0 \\ \texttt{X 2} & \text{if } f(x) = 1 \\ \texttt{CNOT 1 2} & \text{if } f(x) = x \\ \texttt{X 1; CNOT 1 2; X 1} & \text{if } f(x) = 1 - x. \end{cases} \tag{13}$$

Clearly, the first two cases are not entangling gates. The last case is a 0-controlled *CNOT* which is equivalent to the *CNOT* gate for our purposes. Hence, we analyze the circuit for the case where $\texttt{U}_\texttt{f}\ \texttt{1 2} \equiv \texttt{CNOT 1 2}$. The input type for this circuit is two qubits initialized in the computational basis, or equivalently $\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{Z}$. Instead of building a full proof tree for `deutsch`, for the sake of readability we will simply write the initial type to the right of `INIT` and put the intermediate types to the right of every operation (with comments on the far right). Note that the ∩-ARR-DIST rule implicitly allows us to map sequencing across intersections, which we will do for convenience.

```
Definition deutsch :=
  INIT ;      I ⊗  Z ∩ Z ⊗ I   (* input type *)
  X 2  ;      I ⊗ -Z ∩ Z ⊗ I   (* y set to 1 *)
  H 1;        I ⊗ -Z ∩ X ⊗ I
  H 2;        I ⊗ -X ∩ X ⊗ I
```

```
Uf 1 2;     I ⊗ -X ∩ X ⊗ X    (* Uf = CNOT *)
H 1;        I ⊗ -X ∩ Z ⊗ X    (* output type *)
```

Since the output types aren't very readable, we apply the ∩-MUL and $\otimes_c$ rules, to obtain `deutsch` : $\mathbf{I} \otimes \mathbf{Z} \cap \mathbf{Z} \otimes \mathbf{I} \to \mathbf{I} \otimes -\mathbf{X} \cap -\mathbf{Z} \otimes \mathbf{I}$. This is precisely what Deutsch's algorithm is supposed to produce – two separable qubits (implied by $\mathbf{A} \otimes \mathbf{I}$, see §4), the first of which has the type $-\mathbf{Z}$, corresponding to a $|1\rangle$ qubit. Note that we could return the second qubit to having type $\mathbf{Z}$ by applying a Hadamard. However, as we statically verified that the ancillary y qubit is unentangled with x, we may freely discard it and optimize away the final `H 2`.

This derivation could also be extended to the more generic Deutsch-Jozsa algorithm [9] in a similar fashion. This, of course, would require extending both the language and the type system to deal with recursion. We leave this challenge for future work.

3 Normal Forms

Our intersection types have the property that there exists a canonical form with which to describe them. This allows us to verify the equality of intersection types by verifying the equality of their canonical forms. The canonical form we use is inspired by the *row echelon form* of a matrix in which every row has its first nonzero term before any subsequent row. We translate this into $\mathbf{I}$ being 0 and further impose that $\mathbf{X} \prec \mathbf{Y} \prec \mathbf{Z} \prec \mathbf{I}$ so $\mathbf{I} \otimes \mathbf{X}$ precedes $\mathbf{I} \otimes \mathbf{Z}$. Further, an intersection type involving commuting, independent terms can be viewed as a matrix with each term corresponding to a row and each column corresponding to a qubit. Then, in the canonical form, any column contains at most one $\mathbf{X}$, and any column without an $\mathbf{X}$ has at most one $\mathbf{Z}$. The unique $\mathbf{X}$ or $\mathbf{Z}$ in each column will be called the $\mathbf{X}$ or $\mathbf{Z}$ pivot.

The ∩-MUL rules will be useful to reduce the types to their canonical forms. Using ∩-MUL-L and ∩-MUL-R, the antecedent and consequent in an arrow type, respectively, can be updated as $\mathbf{A} \cap \mathbf{B} \leftarrow \mathbf{A} \cap \mathbf{AB}$. Given an n-qubit intersection type with m independent terms $\mathbf{A}_{(1)} \cap \ldots \cap \mathbf{A}_{(m)}$, do the following:

1. Let $\mathcal{P}$ be an ordered set of indices, initialized to $\emptyset$.
2. For each qubit $i = 1 \ldots n$:
   - For the first term $j \notin \mathcal{P}$ such that $\mathbf{A}_{(j)}[i] \in \{\mathbf{X}, \mathbf{Y}\}$
     - Update $\mathcal{P} \leftarrow \mathcal{P} \cup \{j\}$.
     - For terms $k \neq j$, if $\mathbf{A}_{(k)}[i] \in \{\mathbf{X}, \mathbf{Y}\}$, rewrite $\mathbf{A}_{(k)} \leftarrow \mathbf{A}_{(j)}\mathbf{A}_{(k)}$.
   - If there is no term with $\mathbf{X}$ or $\mathbf{Y}$ on qubit i , for the first term $j \notin \mathcal{P}$ such that $\mathbf{A}_{(j)}[i] = \mathbf{Z}$:
     - Update $\mathcal{P} \leftarrow \mathcal{P} \cup \{j\}$.
     - For terms $k \neq j$, if $A_{(k)}[i] = \mathbf{Z}$, rewrite $A_{(k)} \leftarrow A_{(j)}\mathbf{A}_{(k)}$.
   - If no term contains $\mathbf{X}$, $\mathbf{Y}$, or $\mathbf{Z}$ on qubit i proceed.
3. We order the terms as follows:
   - For the the terms in $\mathcal{P}$, place them in the order in which they appear in $\mathcal{P}$.
   - Order the remaining terms lexicographically.

Notice that each term can be added to $\mathcal{P}$ at most once and this, along with the ordering of terms makes the canonical form unique. Further, by viewing the ith term in $\mathcal{P}$ to be the $\mathbf{X}-, \mathbf{Y}-$ or $\mathbf{Z}-$pivot for qubit i, this procedure is functionally equivalent to row echelonization for matrices and can be computed using $O(n^3)$ operations.

Note that this normalization process is functionally similar to the reduction of a stabilizer code to a standard form, see for example [20, §10.5.7]. Given the standard form of a stabilizer code, there are efficient methods for generating its encoding circuit using only Clifford gates [6]. In particular, if we are given a state with a complete Gottesman type, we know that we can efficiently construct a Clifford circuit from the normal form of the type that prepares the given state.

Proposition 2. *Let $|\psi\rangle$ be an n-qubit state. Then $|\psi\rangle : \mathbf{P}_{(1)} \cap \cdots \cap \mathbf{P}_{(n)}$ if and only if $|\psi\rangle$ can be prepared from $|0 \ldots 0\rangle$ with a Clifford circuit. That is, for any set of commuting Pauli operators $P_{(1)}, \ldots, P_{(n)}$ there exists a Clifford operator $C : \mathbf{Z}_j \to \mathbf{P}_{(j)}$ for each $j = 1, \ldots, n$.*

Example 3. *Consider the following type:*

$$\mathbf{X} \otimes \mathbf{X} \otimes \mathbf{I} \cap \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I} \cap \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{Z}.$$

Conveniently, the first term contains an $\mathbf{X}$ on qubit 1. However, no subsequent terms have an $\mathbf{X}$ on this qubit, so we move on to qubit 2.

For the second qubit, no $\mathbf{X}$'s remain in pivot terms, so we take the $\mathbf{Z}$ in second term, $\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}$. The third term is now rewritten as:

$$\begin{aligned}(\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{Z})(\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}) \\ = \ \mathbf{ZZ} \otimes \mathbf{ZZ} \otimes \mathbf{ZI} \\ = \ \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Z}.\end{aligned}$$

For the last qubit, there is only one term with a $\mathbf{X}$ or $\mathbf{Z}$ in the third position, so we are done.

The entire procedure yields the normal form:

$$\mathbf{X} \otimes \mathbf{X} \otimes \mathbf{I} \cap \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Z}.$$

An essential property of the normal form is that it is oblivious to the original ordering of the terms. For instance Theorem 3, if we had first swapped the 2nd and 3rd terms then $\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{Z}$ would have been the pivot for the second qubit and we would replace the 3rd term with $\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Z}$. We would then use the third term as our pivot, replacing the second term ($\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{Z}$) with $\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}$. The entire procedure yields $\mathbf{X} \otimes \mathbf{X} \otimes \mathbf{I} \cap \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Z}$ just as before.

Since all of our normalization operations are justified by the $\cap$-MUL, associativity, and commutativity rules, the following typing rule is admissible:

$$\frac{g : (\mathbf{A} \to \mathbf{B})}{g : \mathbf{A} \to \mathrm{norm}(\mathbf{B})} \textsc{Norm}$$

where norm is our normalization procedure. This is the rule we will apply in practice before making separability judgments.

4 Separability

In this section, we present the first application of our type system – the ability to make judgments on whether a given sub-system is separable from the remainder of the system. We start with determining whether a single qubit is separable before moving to multi-qubit sub-systems.

4.1 Single qubit separability

Following the core semantics that a type refers to the +1-eigenstate of its semantic operator, we first prove a statement about the separable eigenstates of some operators. For notational simplicity we state the following proposition with a focus on the first qubit, however the result holds for any operator of the form $I^{k-1} \otimes U \otimes I^{n-k}$

Proposition 4. *For any 2×2 unitary, Hermitian matrix U, the eigenstates of $U \otimes I^{n-1}$ are all vectors of the form $|u\rangle \otimes |\psi\rangle$ where $|u\rangle$ is an eigenstate of U and $|\psi\rangle \in \mathbb{C}^{2^{n-1}}$ is an arbitrary state.*

Proof. Let $|\phi\rangle$ be the λ-eigenstate and $\left|\phi^{\perp}\right\rangle$ be the $(-\lambda)$-eigenstate of U where $\lambda \in \{1, -1\}$. Note that $\{|\phi\rangle, \left|\phi^{\perp}\right\rangle\}$ forms a single-qubit basis.

First, consider states of the form $|\gamma\rangle = |u\rangle \otimes |\psi\rangle$ where $|u\rangle \in \{|\phi\rangle, \left|\phi^{\perp}\right\rangle\}$ and $|\psi\rangle \in \mathbb{C}^{2^{n-1}}$. Clearly,

$$(U \otimes I^{n-1}) |\gamma\rangle = (U \otimes I^{n-1})(|u\rangle \otimes |\psi\rangle) = (U |u\rangle) \otimes |\psi\rangle = \lambda_u |u\rangle \otimes |\psi\rangle .$$

Hence, every state of the form of $|\gamma\rangle$ is an eigenstate of $U \otimes I^{n-1}$. Additionally, note that by similar reasoning, for every separable state $|\gamma\rangle = |v\rangle \otimes |\psi\rangle$, where $|v\rangle \notin \{|\phi\rangle, \left|\phi^{\perp}\right\rangle\}$, is not an eigenstate of $U \otimes I^{n-1}$.

Now we show that any state not in this separable form cannot be an eigenstate of $U \otimes I^{n-1}$. By way of contradiction assume that $|\delta\rangle$ is an eigenstate of $U \otimes I^{n-1}$ with $(U \otimes I^{n-1}) |\delta\rangle = \mu |\delta\rangle$. Expand

$$|\delta\rangle = \alpha |\phi\rangle \otimes |\psi_1\rangle + \beta \left|\phi^{\perp}\right\rangle \otimes |\psi_2\rangle$$

where $|\psi_1\rangle, |\psi_2\rangle \in \mathbb{C}^{2^{n-1}}$. Then we compute

$$\begin{aligned} (U \otimes I^{n-1}) |\delta\rangle &= \alpha(U |\phi\rangle) \otimes |\psi_1\rangle + \beta(U \left|\phi^{\perp}\right\rangle) \otimes |\psi_2\rangle \\ &= \lambda\alpha |\phi\rangle \otimes |\psi_1\rangle - \lambda\beta \left|\phi^{\perp}\right\rangle \otimes |\psi_2\rangle \\ &= \mu\alpha |\phi\rangle \otimes |\psi_1\rangle + \mu\beta \left|\phi^{\perp}\right\rangle \otimes |\psi_2\rangle \end{aligned}$$

where we have used that $|\phi\rangle$ and $\left|\phi^{\perp}\right\rangle$ are the $+\lambda$ and $-\lambda$ eigenvalues of U respectively. As the components of the expansion are orthogonal to each other, μ must satisfy:

$$\mu\alpha = \lambda\alpha \text{ and } \mu\beta = -\lambda\beta.$$

Since $U \otimes I^{n-1}$ is unitary, $\lambda \neq 0$ and we either have (i) $\alpha = 0$, $\mu = -\lambda$, and $|\delta\rangle = \left|\phi^{\perp}\right\rangle \otimes |\psi_2\rangle$ or (ii) $\beta = 0$, $\mu = +\lambda$, and $|\delta\rangle = |\phi\rangle \otimes |\psi_1\rangle$. In either case $|\delta\rangle$ has a separable form as claimed. □

As every Pauli matrix is both Hermitian and unitary, combining Theorems 1 and 4, we immediately obtain the following corollary:

Corollary 5. *Every term of type $\mathbf{I}^{i-1} \otimes \mathbf{U} \otimes \mathbf{I}^{n-i}$ is separable, for any $U \in \{\pm X, \pm Y, \pm Z\}$. That is, the ith factor has type $\mathbf{U}$ and is not entangled with the rest of the system.*

Following Gottesman's notation, let $\mathbf{U}_k$ be the n-qubit type where the kth factor has the single-qubit type $\mathbf{U}$ and is separable from the rest of the system. For example, the type $\mathbf{X}_1 \equiv \mathbf{X} \otimes \mathbf{I}$ describes the set of two separable qubits where the first qubit is in the $\mathbf{X}$ eigenstate[2]. The two-qubit product state $|0\rangle \otimes |+\rangle$ can be given the intersection type $\mathbf{Z}_1 \cap \mathbf{X}_2$ to signify that each qubit is separable from the other. Theorem 5 then justifies the following separability rules:

$$\text{SEP1-R}\ \frac{g : \mathbf{A} \to \mathbf{I}^{k-1} \otimes \mathbf{B} \otimes \mathbf{I}^{n-k}}{g : \mathbf{A} \to \mathbf{B}_k} \qquad \frac{g : \mathbf{I}^{k-1} \otimes \mathbf{A} \otimes \mathbf{I}^{n-k} \to \mathbf{B}}{g : \mathbf{A}_k \to \mathbf{B}}\ \text{SEP1-L}$$

Since A being separable in a larger system B implies that the rest of B is separable from A, we can add the following rules for distributing separability judgments across intersections:

$$\cap\text{-SEP1-R}\ \frac{g : \mathbf{A} \to \mathbf{B}_k \cap \mathbf{T} \qquad \mathbf{T}[k] \in \{\mathbf{B}, \mathbf{I}\}}{g : \mathbf{A} \to \mathbf{B}_k \cap \mathbf{T}_{[n]\setminus\{k\}}} \qquad \frac{g : \mathbf{A}_k \cap \mathbf{T} \to \mathbf{B} \qquad \mathbf{T}[k] \in \{\mathbf{A}, \mathbf{I}\}}{g : \mathbf{A} \cap \mathbf{T}_{[n]\setminus\{k\}} \to \mathbf{B}_k}\ \cap\text{-SEP1-L}$$

Using these rules, we can re-write $\mathbf{X}_1 \cap (\mathbf{X} \otimes \mathbf{Z} \otimes \mathbf{Z})$ as $\mathbf{X}_1 \cap (\mathbf{Z} \otimes \mathbf{Z})_{2,3}$.

4.2 Multi-qubit separability

While Theorem 5 can be used to identify if a single qubit is separable from the rest of the system, we would also like to make judgments about a multi-qubit subsystem $S \subset \{1, \ldots, n\}$ being separable from $\{1, \ldots, n\} \setminus S$. Generalizing Theorem 4 will help us in this regard. However, we only generalize it for the case when the unitaries are Pauli matrices (rather than generic Hermitian matrices). The following fact about Pauli matrices, adapted from Nielsen and Chuang [20, Prop. 10.5] by setting $n \leftarrow k, k \leftarrow 0$, will be useful for the proof.

Fact 6. *For k-qubit Pauli matrices $V \in \{\pm I, \pm X, \pm Y, \pm Z\}^k$ such that $V \neq I^k$, the eigenvalue $\lambda \in \{-1, 1\}$ has an eigenspace of dimension 2^{k-1}. For k independent, commuting k-qubit Pauli matrices $U_{(1)}, \ldots U_{(k)}$, the joint eigenspace for an eigenvalue tuple $(\lambda_1, \ldots, \lambda_k)$ has dimension 1.*

This fact can be intuitively argued from the observation that each Pauli matrix divides the total, 2^k-dimensional Hilbert space into two sub-spaces of the same dimension, each corresponding to the $+1$ or -1 eigenvalues. The k-tuple then identifies a 1-dimensional subspace at the intersection of the corresponding eigenspaces for $U_{(1)}, \ldots, U_{(k)}$.

Theorem 6 requires the k-qubit Pauli matrices to be independent and pairwise commuting. It is straightforward to check independence by ensuring that multiplying any combination of the k matrices together does not yield the $\mathbf{I}^k$ term. Pairwise commutativity can also be directly determined using the following fact.

Fact 7. *For each pair of matrices $A = A_1 \otimes \cdots \otimes A_k$, $B = B_1 \otimes \cdots \otimes B_k$, where the A_is and B_is are Pauli matrices, A and B commute if and only if*

$$\bigoplus_i [A_i, B_i \neq I \ \&\ A_i \neq B_i] = 0.$$

That is, they commute if and only if there are an even number of positions from $\{1, \ldots, k\}$ where X, Z and Y do not correspond in both matrices.

[2]For precision, we should say $\mathbf{X}_{1 \in [2]}$ to indicate the size of the system, but this will always be clear from the context.

We can now state the conditions under which a set of Pauli operators could correspond to a separable sub-system.

Proposition 8. *For independent, commutative, non-identity k-qubit matrices $U_{(1)}, \dots U_{(k)} \in \{\pm I, \pm X, \pm Y, \pm Z\}^k$ such that $U_{(i)} \cap U_{(j)} \neq \emptyset$ for all $i \neq j$, the eigenstate of $(I^{n-k} \otimes U_{(1)}) \cap \dots \cap (I^{n-k} \otimes U_{(k)})$ are all vectors of the form $|u\rangle \otimes |\Psi\rangle$ where $|\Psi\rangle$ is an eigenstate of $U_{(1)}, \dots, U_{(k)}$.*

Proof. First, it is clear that any state of the form $|u\rangle \otimes |\Psi\rangle$ where $|\Psi\rangle$ is an eigenstate of $U_{(1)}, \dots, U_{(k)}$ is an eigenstate of $I^{n-k} \otimes U_{(1)}, \dots, I^{n-k} \otimes U_{(k)}$. This implies that it is also an eigenstate of $(I^{n-k} \otimes U_{(1)}) \cap \dots \cap (I^{n-k} \otimes U_{(k)})$.

To prove the inverse direction, assume by way of contradiction that there exists an entangled n-qubit state $|\delta\rangle$ that is an eigenstate of $(I \otimes U_{(i)}$ with eigenvalue $\lambda_i \in \{-1, 1\}$ for each $i \in \{1, \dots, k\}$. Let the n-qubit state $|\delta\rangle$ be written in terms of its *Schmidt* (singular value) decomposition across the $(n-k, k)$ qubit bipartition as

$$|\delta\rangle = \sum_{j=1}^{K} \alpha_j |\phi_j\rangle \otimes |\gamma_j\rangle$$

where $\{|\phi_i\rangle\}_i$ and $\{|\gamma_i\rangle\}_i$ are orthonormal vectors in each of their respective subsystems.

$$\begin{aligned}
\forall i \in \{1, \dots, k\} \quad (I \otimes U_{(i)}) |\delta\rangle &= \sum_j \alpha_j (I |\phi_j\rangle) \otimes (U_{(i)} |\gamma_j\rangle) \\
&= \lambda_i \sum_j \alpha_j |\phi_j\rangle \otimes |\gamma_j\rangle \\
&= \sum_j \alpha_j |\phi_j\rangle \otimes (\lambda_i |\gamma_j\rangle) \\
&\Rightarrow \forall i, j, \quad U_{(i)} |\gamma_j\rangle = \lambda_i |\gamma_j\rangle . \\
&\Rightarrow \forall i, j, \quad \lambda_i U_{(i)} |\gamma_j\rangle = |\gamma_j\rangle \quad \text{Since, } \lambda_i \in \{-1, 1\}. \qquad (14)
\end{aligned}$$

As $\{|\gamma_i\rangle\}_i$ forms a set of orthonormal vectors, the span of these vectors is contained in the eigenspace for the eigenvalue tuple $(+1, +1, \dots, +1)$ corresponding to $\lambda_1 U_{(1)}, \dots, \lambda_k U_{(k)}$ respectively. Additionally, when $U_{(i)}$ is a k-qubit Pauli matrix, $\lambda_i U_{(i)}$ is also in $\{\pm I, \pm X, \pm Y, \pm Z\}^k$. Then, from Theorem 6, the joint eigenspace for the all-1s tuple has dimension 1. Specifically, there exists only a single $|\gamma\rangle$ that satisfies Equation (14). Hence, $K = 1$ contradicting the assumption that $|\delta\rangle$ is entangled across the $(n-k, k)$ qubit bi-partition. □

Extending the $\mathbf{U}_i$ notation to the multi-qubit setting where $K \subset \{1, \dots, n\}$ and $0 < |K| < n$, let $(\mathbf{U})_K$ be the type such that qubits in K are separable from the $\{1, \dots, n\} \setminus K$ sub-system. Formally, we define $\mathbf{U}_K := \left(\cap_{j=1}^{|K|} \mathbf{U}_{(j)}\right)_K$ where each $\mathbf{U}_{(j)}$ is a non-trivial $|K|$-qubit non-identity Pauli string. For example, consider a 2-qubit type $(\mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z})$ whose joint eigenspace is spanned by the two maximally entangled Bell states $\{|\Phi^+\rangle, |\Psi^-\rangle\}$. Also, consider an n-qubit state with this type on the first and third qubits. Being maximally entangled, these qubits should be disjoint from the rest of the system, and hence, their type is $(\mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z})_{1,3}$. If the second and fourth qubits are similarly entangled, the system has type $(\mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z})_{1,3} \cap (\mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z})_{2,4}$. This idea to gather the nontrivial factors within a subsystem is not unique to our work and has been previously employed by Honda [15] to determine the entangled components in his type system.

Combining this representation with Theorems 1 and 8, we obtain the following corollary:

Corollary 9. *Let $K \subset \{1,\ldots,n\}$ with $|K| = k$ and $\overline{K} := \{1,\ldots,n\}\backslash K$. Every intersection type that contains the term $\bigcap_{j=1}^{k}\left(\mathbf{U}_{(j)} \otimes \mathbf{I}^{n-k}\right)$ where each of the $\mathbf{U}_{(j)}$s acts on K, is pairwise commuting and independent as a sub-term is separable across the bi-partition $(K, \overline{K})$. That is, the factors in K are separable from the $\overline{K}$ subsystem.*

Given a canonical n-qubit intersection type with m independent terms $\mathbf{A}_{(1)} \cap \ldots \cap \mathbf{A}_{(m)}$, finding if a subsystem of qubits $K \subset \{1,\ldots,n\}$ with $|K| = k < m$, is separable from the remaining system can be determined in a straightforward way. We first verify that every qubit in K has a pivot, otherwise, some qubit in K has type $\mathbf{I}$ in all terms, and we can conclude that K is not separable from the remaining system. If every qubit in K has a pivot, we run the following procedure:

- Let $\mathbf{A}_{(j_1)},\ldots,\mathbf{A}_{(j_k)}$ be the k terms which have the pivots for qubits in K.
- For each $i = 2\ldots k$, check that $\mathbf{A}_{(j_i)}$ commutes with $\mathbf{A}_{(j_1)}$ using Theorem 7.[3]
- For each $i = 1\ldots k$, check that every qubit $\ell \in \overline{K}$ has type $\mathbf{I}$ in the term $\mathbf{A}_{(j_i)}$.

Theorem 9 justifies our multi-qubit separability rules

$$\frac{g : \mathbf{A} \to \mathbf{B} \cap \mathbf{T}_{(1)} \cap \ldots \cap \mathbf{T}_{(k)} \qquad \forall_{j\in[k]}\mathbf{T}_{(j)}[S] = \mathbf{C}_{(j)} \qquad \forall_{j\in[k]}\mathbf{T}_{(j)}[\overline{S}] = \mathbf{I}^{n-k} \qquad \mathbf{B}[S] = \mathbf{I}^{k}}{g : \mathbf{A} \to \mathbf{B}_{\overline{S}} \cap \left(\mathbf{C}_{(1)} \cap \ldots \cap \mathbf{C}_{(k)}\right)_{S}} \text{ SEP-R}$$

$$\frac{g : \mathbf{A} \cap \mathbf{T}_{(1)} \cap \ldots \cap \mathbf{T}_{(k)} \to \mathbf{B} \qquad \forall_{j\in[k]}\mathbf{T}_{(j)}[S] = \mathbf{C}_{(j)} \qquad \forall_{j\in[k]}\mathbf{T}_{(j)}[\overline{S}] = \mathbf{I}^{n-k} \qquad \mathbf{A}[S] = \mathbf{I}^{k}}{g : \mathbf{A}_{\overline{S}} \cap \left(\mathbf{C}_{(1)} \cap \ldots \cap \mathbf{C}_{(k)}\right)_{S} \to B} \text{ SEP-L}$$

4.2.1 Example: Multi-qubit separability

Continuing the example from Theorem 3, consider the type

$$\mathbf{X} \otimes \mathbf{X} \otimes \mathbf{I} \cap \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Z}.$$

As $(\mathbf{X}\otimes\mathbf{X})$ and $(\mathbf{Z}\otimes\mathbf{Z})$ are two independent and commuting operators, the first two terms with $\mathbf{I}$ on the third qubit ensure that we can apply Theorem 9 to determine that the first two qubits are separable from the third. We can write this as:

$$(\mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z})_{1,2} \cap \mathbf{Z}_3.$$

4.3 Example: GHZ state, Entanglement Creation and Disentanglement

To demonstrate how we can track the possibly entangling and disentangling properties of the *CNOT* gate, we can look at the example of creating the GHZ state $\frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)$ starting from $|000\rangle$ and then disentangling it. A similar example was considered by Honda [15] to demonstrate how his system can track when *CNOT* displays either its entangling or disentangling behavior. One crucial difference is that Honda uses the denotational semantics of density matrices which, in practice, would scale poorly with the size of the program being type-checked. Our approach is closer to that of Perdrix [23, 22] in terms of design and scalability but capable of showing separability where the prior systems could not.

We will consider the following GHZ program acting on the initial state $\mathbf{Z}_1 \cap \mathbf{Z}_2 \cap \mathbf{Z}_3$. We first follow the derivation for $\mathbf{Z}_1$ (which we immediately rewrite to $\mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{I}$):

[3] This will ensure that the type of qubits in K is $\mathbf{I}$ in all terms where the $\overline{K}$ qubits are pivots.

```
Definition GHZ :=
  INIT;       Z ⊗ I ⊗ I   (* initial state *)
  H 1;        X ⊗ I ⊗ I
  CNOT 1 2;   X ⊗ X ⊗ I   (* Bell Pair *)
  CNOT 2 3:   X ⊗ X ⊗ X   (* GHZ State created *)
```

Repeating the derivation for $\mathbf{Z}_2$ and $\mathbf{Z}_3$, we obtain the following type:

$$\texttt{GHZ} : (\mathbf{Z}_1 \to \mathbf{X} \otimes \mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{Z}_2 \to \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{Z}_3 \to \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z})$$

If we now apply `CNOT 3 1`, we get the following type:

$$\texttt{GHZ; CNOT 3 1} : (\mathbf{Z}_1 \to \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{Z}_2 \to \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{Z}) \cap (\mathbf{Z}_3 \to \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z})$$

This is not immediately meaningful, so we normalize the output (the first and second row serving as the first and second pivots):

$$\mathbf{I} \otimes \mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{Z} \cap \mathbf{IZ} \otimes \mathbf{ZZ} \otimes \mathbf{ZZ} = \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{Z} \cap \mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{I}.$$

Recognizing that the first qubit can now be separated from the other two, we obtain $\mathbf{Z}_1 \cap (\mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z})_{2,3}$, that is, a $\mathbf{Z}$ qubit and a Bell pair.

If we then apply `CNOT 3 2`, we get

$$\texttt{GHZ; CNOT 3 1; CNOT 3 2} : (\mathbf{Z}_1 \to \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X}) \cap (\mathbf{Z}_2 \to \mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{Z}_3 \to \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I})$$

to which we can apply distributivity and separability judgments to obtain

$$\mathbf{Z}_1 \cap \mathbf{Z}_2 \cap \mathbf{Z}_3 \to \mathbf{Z}_1 \cap \mathbf{Z}_2 \cap \mathbf{X}_3$$

showing that the whole procedure moves the $\mathbf{X}$ generated by the initial Hadamard gate to the third position.

5 Measurement

It is challenging to turn Gottesman's semantics for measurement into a type system because it looks at its operation on all the basis states rather than simply the evolution of a single Pauli operator. Namely, it adds significant computational complexity, while type checking should be linear. Nonetheless, our normalization in §3 parallels that in the stabilizer formalism, and the action of measurement on stabilizer groups is well-understood [12]. This produces a method for type checking that is quadratic in the number of qubits in the worst case [1].

5.1 Union types

Before discussing how we type check measurement, it helps to consider how we can represent post-measurement states. Unlike unitary gate application, which is deterministic, implying that each input type has a specified output type, not all measurements have deterministic outcomes. While we don't want to use our type system to verify the probabilities of measurement outcomes, it would be useful to be able to compute the possible post-measurement states for the system. With this, we could still track how the system evolves with subsequent operations depending on the measurement results.

We use disjoint union types, $\mathbf{A} \cup \mathbf{B}$, to denote that the system either has type $\mathbf{A}$ or type $\mathbf{B}$. We show how to use this in the context of measurement with this simple example.

Example 10 (Measuring $|+\rangle$)**.** *Consider the single qubit in the* $|+\rangle$ *state on which a computational basis measurement is performed. The outcome has equal probability to be* 0 *or* 1 *which we cast as qubits in states* $|0\rangle : \mathbf{Z}$ *and* $|1\rangle : -\mathbf{Z}$ *respectively. We represent this in our type system as*

$$\frac{}{\mathit{Meas} : \mathbf{X} \to \mathbf{Z_1} \cup -\mathbf{Z_1}}$$

Applying a gate to a union type distributes across the union and each term in the union evolves separately. This gives the following rules for unions:

$$\frac{g : \mathbf{A}}{g : \mathbf{A} \cup \mathbf{B}} \cup\text{-I} \qquad \frac{g : \mathbf{A} \cup \mathbf{A}}{g : \mathbf{A}} \cup\text{-E} \qquad \frac{g : (\mathbf{A} \to \mathbf{A}') \cup (\mathbf{B} \to \mathbf{B}')}{g : (\mathbf{A} \cup \mathbf{B}) \to (\mathbf{A}' \cup \mathbf{B}')} \cup\text{-Arr-Dist}$$

As with intersections, the ordering of the terms doesn't matter with commutativity and associativity holding for unions too:

$$\frac{g : \mathbf{A} \to \mathbf{B} \cup \mathbf{C}}{g : \mathbf{A} \to \mathbf{C} \cup \mathbf{B}} \cup\text{-comm-r} \qquad \frac{g : \mathbf{A} \to \mathbf{B} \cup (\mathbf{C} \cup \mathbf{D})}{g : \mathbf{A} \to (\mathbf{B} \cup \mathbf{C}) \cup \mathbf{D}} \cup\text{-assoc-r}$$

5.2 Types of post-measurement states

For ease of exposition, we will assume that we are performing a z-basis measurement on the jth qubit of an n qubit system. In §3 we introduced a normalization procedure for intersection types. There, we constructed the normal form by examining each qubit in turn $i = 1, \ldots, n$, and looked for an intersection term whose i^{th} factor is $\mathbf{X}$, $\mathbf{Y}$, or $\mathbf{Z}$. As there, let us write $\mathbf{A}_{(1)} \cap \cdots \cap \mathbf{A}_{(m)}$ for the pre-measurement type. Now however, we begin by searching for an i, such its jth factor $\mathbf{A}_{(i)}[j] \in \{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$.

1. If there exists an i such that $\mathbf{A}_{(i)}[j] = \mathbf{X}$ or $\mathbf{A}_{(i)}[j] = \mathbf{Y}$, then the measurement outcome is uniformly random:
   (a) Replace $\mathbf{A}_{(k)} \leftarrow \mathbf{A}_{(i)} A_{(k)}$ for all $k \neq i$ with $\mathbf{A}_{(k)}[j] \in \{\mathbf{X}, \mathbf{Y}\}$.
   (b) Let $\mathbf{U}' = \mathbf{A}_{(1)} \cap \cdots \cap \mathbf{A}_{(i-1)} \cap \mathbf{A}_{(i+1)} \cap \cdots \cap \mathbf{A}_{(m)}$.
   (c) The post-measurement state is of type $(\mathbf{Z}_j \cap \mathbf{U}') \cup (-\mathbf{Z}_j \cap \mathbf{U}')$.
   (d) Normalize each branch of the union separately to get the normalized post-measurement state.
2. If no i has $\mathbf{A}_{(i)}[j] \in \{\mathbf{X}, \mathbf{Y}\}$ find an i such that $\mathbf{A}_{(i)}[j] = \mathbf{Z}$. When this is the case the outcome is deterministic as some combination of the intersection terms is $\mathbf{Z}_j$ or $-\mathbf{Z}_j$ [1]:[4]
   (a) Use the $\cap$-mul-r rule to obtain $\pm\mathbf{Z}_j$ as an intersection term (our normalization procedure ensures this can be done efficiently). Let the rest of the intersection be $\mathbf{U}$.
   (b) Normalize the term $(\pm\mathbf{Z}_j \cap \mathbf{U})$ to obtain the normalized post-measurement state.

[4] We refer the interested reader to the discussion following Proposition 3 in Aaronson and Gottesman [1] for details on why this fact holds.

3. If all $A_{(i)}[j] = \mathbf{I}$ then the post-measurement state will have type

$$(\mathbf{Z}_1 \cap \mathbf{A}_{(1)} \cap \cdots \cap \mathbf{A}_{(m)}) \cup (-\mathbf{Z}_1 \cap \mathbf{A}_{(1)} \cap \cdots \cap \mathbf{A}_{(m)}).$$

   Normalize each branch of the union separately to get the normalized post-measurement type.

Observe that case (3) can occur only when an $m < n$–that is the type is under-determined. This will commonly be the case while dealing with the physical qubit types for stabilizer-based error-correcting codes. Finally, by construction, the measured qubit has the type $\mathbf{Z}$ or $-\mathbf{Z}$ and is separable from the rest of the system.

Example 11. *As an example of our normalization and measurement rules, consider measuring the first qubit in the z-basis a state of type* $\mathbf{X} \otimes \mathbf{X}$*. According to our rules above, we remove this term when considering the post-measurement type; that is, our we know nothing of the resulting type except the consequence of the measurement. In particular, rule (1) above states our post-measurement state is of type* $\mathbf{Z}_1 \cup -\mathbf{Z}_1$*. To valid this in semantics, we note* $|\psi\rangle : \mathbf{X} \otimes \mathbf{X}$ *if and only if*

$$|\psi\rangle = \alpha\,|{+}{+}\rangle + \beta\,|{-}{-}\rangle = \tfrac{1}{\sqrt{2}}\,|0\rangle \otimes (\alpha\,|+\rangle + \beta\,|-\rangle) + \tfrac{1}{\sqrt{2}}\,|1\rangle \otimes (\alpha\,|+\rangle - \beta\,|-\rangle).$$

Regardless of measuring 0 *or* 1*, the resulting state in the other qubit is arbitrary. Hence output state is indeed of type* $(\mathbf{Z} \otimes \mathbf{I}) \cup (-\mathbf{Z} \otimes \mathbf{I}) = \mathbf{Z}_1 \cup -\mathbf{Z}_1$*.*

5.3 Example: Measuring a GHZ state

Continuing our analysis of the GHZ state from §4.3, the circuit `GHZ` had the output type

$$(\mathbf{X} \otimes \mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z}).$$

To compute the type of `GHZ; MEAS 1` we enact the above program. Fortunately our intersection already has the requisite form, with the first term being the only one with an $\mathbf{X}$ in the initial position. We remove the term $(\mathbf{X} \otimes \mathbf{X} \otimes \mathbf{X})$ and replace the intersection with

$$(\mathbf{Z}_1 \cap (\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z})) \cup (-\mathbf{Z}_1 \cap (\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z}))$$

Using the $\cap$-MUL rewrite rules, normalize the second term to obtain

$$(\mathbf{Z}_1 \cap \mathbf{Z}_2 \cap (\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z})) \cup (-\mathbf{Z}_1 \cap -\mathbf{Z}_2 \cap (\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z})).$$

Finally, the last term can also be simplified to give $(\mathbf{Z}_1 \cap \mathbf{Z}_2 \cap \mathbf{Z}_3) \cup (-\mathbf{Z}_1 \cap -\mathbf{Z}_2 \cap -\mathbf{Z}_3)$. □

6 Example: Error-correcting Codes

We can also use our type system to analyze error-correcting codes. In this section, we consider the 7-qubit Steane [31] code. Recall that the Steane code encodes a single qubit into 7 qubits and has the ability to detect errors on 2 qubits and correct all single-qubit errors. The stabilizers and logical operators for the Steane code are generated by:

$$\begin{array}{llll} g_1 = IIIXXXX & g_2 = IXXIIXX & & \overline{X} = XXXXXXX \\ g_3 = XIXIXIX & g_4 = IIIZZZZ & & \overline{Z} = ZZZZZZZ \\ g_5 = IZZIIZZ & g_6 = ZIZIZIZ & & \end{array}$$

We realize $|0\rangle$ in this setup through the logical state $|0_L\rangle$ defined by projecting the all 0s state using the stabilizer generators of the code:

$$|0_L\rangle \propto \frac{1}{2^6}\Pi_{i=1}^{6}(I + g_i)\,|0000000\rangle$$

By virtue of being the logical 0 state, it should also be stabilized by the logical-σ_Z equivalent $\overline{Z}$. In other words, $|0_L\rangle$ is uniquely stabilized by $g_1, \ldots, g_6$ and $\overline{Z}$. In our type system, this means that $|0_L\rangle : \mathbf{Z}_L$ where $\mathbf{Z}_L$ is the 7-term intersection type

$$\begin{aligned}\mathbf{Z}_L &:= \mathbf{g_1} \cap \ldots \cap \mathbf{g_6} \cap \overline{\mathbf{Z}} \\ &= \mathbf{I}\otimes\mathbf{I}\otimes\mathbf{I}\otimes\mathbf{X}\otimes\mathbf{X}\otimes\mathbf{X}\otimes\mathbf{X} \\ &\cap \ldots \cap \mathbf{Z}\otimes\mathbf{I}\otimes\mathbf{Z}\otimes\mathbf{I}\otimes\mathbf{Z}\otimes\mathbf{I}\otimes\mathbf{Z} \\ &\cap \mathbf{Z}\otimes\mathbf{Z}\otimes\mathbf{Z}\otimes\mathbf{Z}\otimes\mathbf{Z}\otimes\mathbf{Z}\otimes\mathbf{Z}\end{aligned} \tag{15}$$

By a similar argument, $|+_L\rangle : \mathbf{X}_L$ where $\mathbf{X}_L = \mathbf{g_1} \cap \ldots \cap \mathbf{g_6} \cap \overline{\mathbf{X}}$. All states in the Steane code space are stabilized by $g_1, \ldots, g_6$. Then, we can associate the following type to the logical Steane code space as

$$\mathbf{St_7} := \mathbf{g_1} \cap \ldots \cap \mathbf{g_6} \qquad \text{and} \qquad \mathbf{Z}_L = \mathbf{St_7} \cap \overline{\mathbf{Z}}; \quad \mathbf{X}_L = \mathbf{St_7} \cap \overline{\mathbf{X}}.$$

Being consistent with the equation above, we can conclude that $|+i_L\rangle : \mathbf{Y}_L$ where

$$\mathbf{Y}_L := \mathbf{g_1} \cap \ldots \cap \mathbf{g_6} \cap \overline{\mathbf{Y}} \qquad \text{where} \qquad \overline{Y} = i\overline{XZ}.$$

Further, we use $\mathbf{Y}_L = i\mathbf{X}_L\mathbf{Z}_L$ as syntactic sugar to derive the action of any gate on $\mathbf{Y}_L$. In this scenario, we can manipulate the types at the logical level i.e., $\{\mathbf{X}_L, \mathbf{Y}_L, \mathbf{Z}_L\}$ as if they share the same algebraic relations as their corresponding Pauli counterparts, $\{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$.

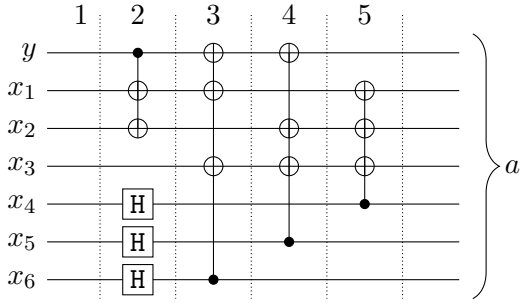

Figure 2: Encoding circuit for the Steane $[[7, 1, 3]]$ code

Consider the Steane code unitary encoding circuit `Enc−St` given in Figure 2 where a data qubit y is converted into a logical qubit a. By construction, it takes $|a\rangle \otimes |000000\rangle \rightarrow |a_L\rangle$ for $a \in \{0, 1, +, -\}$. Consider $a = 0$ for instance. Then we can derive an arrow type for `Enc−St` as follows:

1. start with the input type $\mathbf{Z}_y \cap \mathbf{Z}_{x_1} \cap \ldots \cap \mathbf{Z}_{x_6}$;
2. apply each gate from Figure 2 using the axioms for H and $CNOT$;
3. normalize the output.

A straightforward computation (an exercise left to the reader) will show that we indeed obtain $\text{norm}(\mathbf{Z}_L)$ as the output type. Extending this argument, we can characterize $\texttt{Enc} - \texttt{St}$ as:

$$\begin{aligned}\texttt{Enc}-\texttt{St} \ : (\mathbf{Z}_y \cap \mathbf{Z}_{x_1} \cap \ldots \cap \mathbf{Z}_{x_6}) &\to \text{norm}(\mathbf{Z}_L) \\ \cap\, (\mathbf{X}_y \cap \mathbf{Z}_{x_1} \cap \ldots \cap \mathbf{Z}_{x_6}) &\to \text{norm}(\mathbf{X}_L).\end{aligned}$$

Another application of our type system is verifying the transversality of a gate with respect to a code. For instance, it is straightforward to verify that $\texttt{H}_\texttt{L} := \texttt{H y}; \texttt{H x}_1; \texttt{H x}_2; \texttt{Hx}_3; \texttt{H x}_4; \texttt{H x}_5; \texttt{H x}_6$ is transversal for the Steane code i.e.,

$$H_L \ : \ (\mathbf{X}_L \to \mathbf{Z}_L) \cap (\mathbf{Z}_L \to \mathbf{X}_L) \tag{16}$$

Clearly, $H_L : (\mathbf{g_1} \cap \mathbf{g_2} \cap \mathbf{g_3}) \to (\mathbf{g_4} \cap \mathbf{g_5} \cap \mathbf{g_6})$ and vica-versa. Hence, $H_L : \mathbf{St_7} \to \mathbf{St_7}$. Further, $H_L : \overline{\mathbf{X}} \to \overline{\mathbf{Z}}$ and vica-versa. Therefore, H_L takes $\mathbf{Z}_L = \mathbf{St_7} \cap \overline{\mathbf{Z}} \to \mathbf{St_7} \cap \overline{\mathbf{X}} = \mathbf{X}_L$ and vica-versa.

In a similar vein, we can prove that the operation $U = \texttt{S y};\ \texttt{S x}_1;\ \texttt{S x}_2;\ \texttt{S x}_3;\ \texttt{S x}_4;\ \texttt{S x}_5;\ \texttt{S x}_6$ is not the logical-S gate S_L. Firstly, the type for the logical-S should satisfy

$$S_L : (\mathbf{Z}_L \to \mathbf{Z}_L) \cap (\mathbf{X}_L \to \mathbf{Y}_L)$$

Now, $U : (\mathbf{g_4} \cap \mathbf{g_5} \cap \mathbf{g_6} \cap \overline{\mathbf{Z}}) \to (\mathbf{g_4} \cap \mathbf{g_5} \cap \mathbf{g_6} \cap \overline{\mathbf{Z}})$ as the S acts only on $\mathbf{Z}$ or $\mathbf{I}$. In the case of $\{\mathbf{g_1}, \mathbf{g_2}, \mathbf{g_3}, \overline{\mathbf{X}}\}$, the $\mathbf{X}$s are converted to $\mathbf{Y}$s such that the types are changed on output. Clearly, $U : \overline{\mathbf{X}} \to \mathbf{Y}^7$ but $\mathbf{Y}^7 = -\mathbf{i}\overline{\mathbf{X}}\overline{\mathbf{Z}} = -\overline{\mathbf{Y}}$. Let's take $\mathbf{g_1} \cap \mathbf{g_4}$ to see how the remaining stabilizers would evolve:

$$\dfrac{\dfrac{U : \mathbf{g_1} \cap \mathbf{g_4} \to (\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Y} \otimes \mathbf{Y} \otimes \mathbf{Y} \otimes \mathbf{Y}) \cap \mathbf{g_4}}{U : \mathbf{g_1} \cap \mathbf{g_4} \to (\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Y} \otimes \mathbf{Y} \otimes \mathbf{Y} \otimes \mathbf{Y})\mathbf{g_4} \cap \mathbf{g_4}}\ \cap\text{-MUL-R}}{U : \mathbf{g_1} \cap \mathbf{g_4} \to \mathbf{g_1} \cap \mathbf{g_4}}$$

Extending this reasoning to $(\mathbf{g_2} \cap \mathbf{g_5})$ and $(\mathbf{g_3} \cap \mathbf{g_6})$, $U : \mathbf{St_7} \to \mathbf{St_7}$.. Putting the pieces together, $U : \mathbf{Z}_L \to \mathbf{Z}_L$ but U takes $\mathbf{X}_L$ to $\mathbf{St_7} \cap -\overline{\mathbf{Y}} = -\mathbf{Y}_L$. By contrast, defining

$$\texttt{S}_\texttt{L} := \texttt{Z y};\ \texttt{S y};\ \texttt{Z x}_1; \texttt{S x}_1\ \texttt{Z x}_2;\ \texttt{S x}_2;\ \texttt{Z x}_3;\ \texttt{S x}_3;\ \texttt{Z x}_4;\ \texttt{S x}_4;\ \texttt{Z x}_5;\ \texttt{S x}_5;\ \texttt{Z x}_6;\ \texttt{S x}_6$$

gives us the desired behaviour.

We would also like to show that the T-gate is not transversal for the Steane code. However, with T not being a Clifford gate, we find that the Gottesman types are insufficient to fully describe it. For this, we consider the additive extension to our type system in subsequent sections and demonstrate this in Theorem 15.

6.1 Logical Multi-qubit types

Extending the discussion on logical qubits and quantum error correcting codes to multi-qubit logical states requires us to add some additional rules to our system.

Separable states Describing states where each qubit is separable will be the most straightforward of these. A simple example is with the state $|01\rangle : \mathbf{X}_1 \cap \mathbf{Z}_2$. Correspondingly the state $|0_L 1_L\rangle : (\mathbf{X}_L)_{\underline{1}} \cap (\mathbf{Z}_L)_{\underline{2}}$ where $\underline{1}, \underline{2}$ represent the logical qubits. From the point of the physical qubits $\underline{1}, \underline{2}$ denote the sets of physical qubits that encode each logical qubit. For instance, for the 7-qubit Steane code, $\underline{1} := (y, x_1, \ldots, x_6)$ and $\underline{2} := (y', x'_1, \ldots, x'_6)$. Formally, using $\cap$-I, we get,

$$(\mathbf{X}_L)_{\underline{1}} \cap (\mathbf{Z}_L)_{\underline{2}} = (\mathbf{g_1} \cap \ldots \mathbf{g_6} \cap \overline{\mathbf{X}})_{\underline{1}} \cap (\mathbf{g_1} \cap \ldots \mathbf{g_6} \cap \overline{\mathbf{Z}})_{\underline{2}}$$

Entangled multi-qubit states To express the type of two logical qubits as, say $\mathbf{X}_L \otimes_L \mathbf{Z}_L$, and effectively tracking their evolution requires more advanced notions such as a logical tensor product $\otimes_L$ between logical types and further rules on how tensor products behave with intersection types. For the sake of this example, we consider a logical tensor operation $\otimes_L$ that acts as follows:

- Consider two basic logical types $\mathbf{A}_L, \mathbf{B}_L$ for $A, B \in \{X, Z\}$
- Order their intersection terms as $\mathbf{A}_L = \mathbf{g_1} \cap \ldots \cap \mathbf{g_6} \cap \overline{\mathbf{A}}$ and $\mathbf{B}_L = \mathbf{g_1} \cap \ldots \cap \mathbf{g_6} \cap \overline{\mathbf{B}}$
- Define $\mathbf{A}_L \otimes_L \mathbf{B}_L := (\mathbf{St_7} \otimes \mathbf{I}^7) \cap (\mathbf{I}^7 \otimes \mathbf{St_7}) \cap (\overline{\mathbf{A}} \otimes \overline{\mathbf{B}})$.

Not that this definition is not arbitrary but can, in fact, be derived from existing rules in our system along with the assumption that the tensor distributes across intersections when the terms involved commute i.e.,

$$\frac{g : \mathbf{T} \to (\mathbf{A} \cap \mathbf{B}) \otimes \mathbf{C}}{g : \mathbf{T} \to (\mathbf{A} \otimes \mathbf{I}) \cap (\mathbf{B} \otimes \mathbf{I}) \cap (\mathbf{I} \otimes \mathbf{C})} \cap\text{-}\otimes\text{-DIST-R} \tag{17}$$

Recalling that $\mathbf{St_7} = \mathbf{g_1} \cap \ldots \cap \mathbf{g_6}$, we can fully expand the $\otimes_L$ expression as

$$\mathbf{A}_L \otimes_L \mathbf{B}_L = (\mathbf{g_1} \otimes \mathbf{I}^7) \cap \ldots \cap (\mathbf{g_6} \otimes \mathbf{I}^7) \cap (\mathbf{I}^7 \otimes \mathbf{g_1}) \cap \ldots \cap (\mathbf{I}^7 \otimes \mathbf{g_6}) \cap (\overline{\mathbf{A}} \otimes \overline{\mathbf{B}})$$

Now, the question becomes: can we show the transversality of $CNOT$ with respect to the Steane code? We can begin by defining:

$$\begin{aligned} \texttt{CNOT}_\texttt{L}\ \underline{1}\ \underline{2} := \texttt{CNOT y y}'\texttt{; CNOT x}_1\ \texttt{x}'_1\texttt{; CNOTx}_2\ \texttt{x}'_2\texttt{; CNOT x}_3\ \texttt{x}'_3\texttt{;} \\ \texttt{CNOT x}_4\ \texttt{x}'_4\texttt{; CNOT x}_5\ \texttt{x}'_5\texttt{; CNOT x}_6\ \texttt{x}'_6\texttt{;}. \end{aligned}$$

Using the behavior of $CNOT$ from Table 2, we need to show that

$$CNOT_L : (\mathbf{X}_L \otimes \mathbf{I}_L \to \mathbf{X}_L \otimes \mathbf{X}_L) \cap (\mathbf{I}_L \otimes \mathbf{X}_L \to \mathbf{I}_L \otimes \mathbf{X}_L) \tag{18}$$

$$\cap\, (\mathbf{I}_L \otimes \mathbf{Z}_L \to \mathbf{Z}_L \otimes \mathbf{Z}_L) \cap (\mathbf{Z}_L \otimes \mathbf{I}_L \to \mathbf{Z}_L \otimes \mathbf{I}_L) \tag{19}$$

Here, by $\mathbf{I}_L$ we mean any state that lies in the codespace of the Steane code and so,

$$\mathbf{I}_L := \mathbf{g_1} \cap \ldots \cap \mathbf{g_6}.$$

It will be easier to derive the action of $CNOT_L$ by understanding it's actions on each of the stabilizers and logical operators for the Steane code as all logical types use these as the building blocks. Applying $CNOT_L$ to each of the operators gate-wise, we get:

- For the X-terms, i.e., for $\mathbf{A} \in \{\mathbf{g_1}, \mathbf{g_2}, \mathbf{g_3}, \overline{\mathbf{X}}\}$

$$CNOT_L : (\mathbf{A} \otimes \mathbf{I}^7 \to \mathbf{A} \otimes \mathbf{A}) \cap (\mathbf{I}^7 \otimes \mathbf{A} \to \mathbf{I}^7 \otimes \mathbf{A}) \tag{20}$$

- For the Z-term, i.e., for $\mathbf{A} \in \{\mathbf{g_4}, \mathbf{g_5}, \mathbf{g_6}, \overline{\mathbf{Z}}\}$

$$CNOT_L : (\mathbf{A} \otimes \mathbf{I}^7 \to \mathbf{A} \otimes \mathbf{I}^7) \cap (\mathbf{I}^7 \otimes \mathbf{A} \to \mathbf{A} \otimes \mathbf{A}) \tag{21}$$

As the term $(\mathbf{St_7} \otimes \mathbf{I^7}) \cap (\mathbf{I^7} \otimes \mathbf{St_7})$ appears in all entangled types, we first derive the action of $CNOT_L$ on it.

$$\dfrac{\dfrac{CNOT_L : (\mathbf{St_7} \otimes \mathbf{I^7}) \cap (\mathbf{I^7} \otimes \mathbf{St_7}) \to \bigcap_{\mathbf{i=1}}^{\mathbf{6}} (\mathbf{g_i} \otimes \mathbf{g_i}) \bigcap_{\mathbf{i=4}}^{\mathbf{6}} (\mathbf{g_i} \otimes \mathbf{I^7}) \bigcap_{\mathbf{i=1}}^{\mathbf{3}} (\mathbf{I^7} \otimes \mathbf{g_i})}{CNOT_L : (\mathbf{St_7} \otimes \mathbf{I^7}) \cap (\mathbf{I^7} \otimes \mathbf{St_7}) \to \bigcap_{\mathbf{i=1}}^{\mathbf{6}} (\mathbf{g_i} \otimes \mathbf{I^7}) \bigcap_{\mathbf{i=1}}^{\mathbf{6}} (\mathbf{I^7} \otimes \mathbf{g_i})} \cap\text{-MUL-R}}{CNOT_L : (\mathbf{St_7} \otimes \mathbf{I^7}) \cap (\mathbf{I^7} \otimes \mathbf{St_7}) \to (\mathbf{St_7} \otimes \mathbf{I^7}) \cap (\mathbf{I^7} \otimes \mathbf{St_7})} \tag{22}$$

Now, using eqs. (20) to (22), we can derive the action of $CNOT_L$ on the remaining logical types. Taking $\mathbf{X}_L \otimes \mathbf{I}_L = (\mathbf{St_7} \otimes \mathbf{I^7}) \cap (\mathbf{I^7} \otimes \mathbf{St_7}) \cap (\overline{\mathbf{X}} \otimes \mathbf{I^7})$ as an example,

$$\dfrac{CNOT_L : \mathbf{X_L} \otimes \mathbf{I_L} \to (\mathbf{St_7} \otimes \mathbf{I^7}) \cap (\mathbf{I^7} \otimes \mathbf{St_7}) \cap (\overline{\mathbf{X}} \otimes \overline{\mathbf{X}})}{CNOT_L : \mathbf{X_L} \otimes \mathbf{I_L} \to \mathbf{X_L} \otimes \mathbf{X_L}} \otimes_L\text{-I}$$

Similarly for the other three cases, we can directly get

$$CNOT_L : \mathbf{Z}_L \otimes \mathbf{I}_L \to \mathbf{Z}_L \otimes \mathbf{I}_L$$
$$CNOT_L : \mathbf{I}_L \otimes \mathbf{X}_L \to \mathbf{I}_L \otimes \mathbf{X}_L$$
$$CNOT_L : \mathbf{I}_L \otimes \mathbf{Z}_L \to \mathbf{Z}_L \otimes \mathbf{Z}_L$$

thereby satisfying Equation (18) and confirming the transversality of $CNOT$ for the Steane code.

7 Additive Types

7.1 The Clifford + T set

Up to this point, our type system has been sound, but the underlying language of the Clifford set is not universal for quantum computation. The easiest path from the Clifford set to a universal set is adding the T operator to our language. Appealing to the original Gottesman semantics

$$T\sigma_z T^\dagger = \sigma_z$$

so we can give T the type $\mathbf{Z} \to \mathbf{Z}$. Unfortunately,

$$T\sigma_x T^\dagger = \tfrac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1-i \\ 1+i & 0 \end{pmatrix}$$

is not in the Pauli group and hence not expressible in our type system.

However, $\frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1-i \\ 1+i & 0 \end{pmatrix}$ can be rewritten as the weighted sum of Pauli matrices $\frac{1}{\sqrt{2}}(\sigma_x + \sigma_y)$, so if our typing judgments distribute over addition, we can expand it to deal with *additive types* $\mathbf{A}+\mathbf{B}$. Indeed $U(A+B)U^\dagger = UAU^\dagger + UBU^\dagger$. To incorporate terms involving added types, we extend our grammar to include words of the form $\mathbf{G} + \mathbf{G}$, where $\mathbf{G}$ is the language of Gottesman types. Throughout we will use the shorthand $\mathbf{A} - \mathbf{B} = \mathbf{A} + (-\mathbf{B})$. We extend our judgements with the rule

$$\dfrac{p : \mathbf{A} \to \mathbf{B} \qquad p : \mathbf{C} \to \mathbf{D}}{p : \mathbf{A} + \mathbf{B} \to \mathbf{C} + \mathbf{D}} \text{ ADD} \tag{23}$$

Note that this will so frequently be combined with our SCALE rule (which can now see a broader range of coefficients c) to deal with additive types that we will tend to apply them together.

Example 12 (Typing T on $\mathbf{Y}$)**.** *We prove the type of T on $\mathbf{Y}$:*

$$\frac{\overline{T : \mathbf{X} \to \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{X}+\mathbf{Y})} \qquad \overline{T : \mathbf{Z} \to \mathbf{Z}}}{T : \mathbf{Y} \to \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{Y}-\mathbf{X})}\ \textit{MUL+SCALE}$$

Example 13 (Typing S)**.** *We now prove that S has the same type as $T;T$. This is trivially true on $\mathbf{Z}$ since both S and T have the type $\mathbf{Z} \to \mathbf{Z}$. We prove the $T;T$ has the desired type on $\mathbf{X}$ explicitly:*

$$\frac{\overline{T : \mathbf{X} \to \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{X}+\mathbf{Y})} \qquad \frac{\overline{T : \mathbf{X} \to \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{X}+\mathbf{Y})} \qquad \overline{T : \mathbf{Y} \to \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{Y}-\mathbf{X})}}{T : \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{X}+\mathbf{Y}) \to \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{X}+\mathbf{Y}+\mathbf{Y}-\mathbf{X}))}\ \textit{ADD+SCALE}}{T;T : \mathbf{X} \to \mathbf{Y}}\ \textit{SEQ}$$

Example 14 (Typing $T^\dagger$)**.** *Finally, it's useful to know the type of $T^\dagger$, which we'll define as $Z;S;T$. This again trivially has the type $\mathbf{Z} \to \mathbf{Z}$, so we simply prove its action on $\mathbf{X}$ as follows:*

$$\frac{\frac{\overline{Z : \mathbf{X} \to -\mathbf{X}} \qquad \overline{S : -\mathbf{X} \to -\mathbf{Y}}}{Z;S : \mathbf{X} \to -\mathbf{Y}}\ \textit{SEQ} \qquad \overline{T : -\mathbf{Y} \to \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{X}-\mathbf{Y})}}{Z;S;T : \mathbf{X} \to \frac{\mathbf{1}}{\sqrt{\mathbf{2}}}(\mathbf{X}-\mathbf{Y})}\ \textit{SEQ}$$

Example 15 (Steane code non-transversality of T)**.** *Now that we have the type for T, the logical-T gate for the Steane code T_L should satisfy*

$$T_L : (\mathbf{Z}_L \to \mathbf{Z}_L) \cap \left(\mathbf{X}_L \to \frac{1}{\sqrt{2}}(\mathbf{X}_L + \mathbf{Y}_L)\right)$$

where we use the descriptions from §6 for $\mathbf{Z}_L, \mathbf{X}_L$ and $\mathbf{Y}_L$. However, we can easily show that the operation $U := \texttt{T y; T x}_1\texttt{; T x}_2\texttt{; T x}_3\texttt{; T x}_4\texttt{; T x}_5\texttt{; T x}_6$ does not satisfy this behaviour. In fact, U acting on $\mathbf{St_7}$ changes the output type as any stabilizer containing an $\mathbf{X}$ is converted into a non-trivial additive type. Then, T applied to $\mathbf{St_7}$ becomes a type containing states outside both the Steane code space as well as the larger stabilizer state space. For instance

$$U : \mathbf{g_1} \to \left(\mathbf{I}\otimes\mathbf{I}\otimes\mathbf{I}\otimes\frac{1}{\sqrt{2}}(\mathbf{X}+\mathbf{Y})\otimes\frac{1}{\sqrt{2}}(\mathbf{X}+\mathbf{Y})\otimes\frac{1}{\sqrt{2}}(\mathbf{X}+\mathbf{Y})\otimes\frac{1}{\sqrt{2}}(\mathbf{X}+\mathbf{Y})\right)$$

which is clearly not a simple tensor product of Paulis as a stabilizer is expected to be.

Proposition 16. *Let $|\psi\rangle$ be an n-qubit state. If $|\psi\rangle : \frac{1}{\sqrt{2}}(\mathbf{P}_0 + \mathbf{P}_1) \cap \mathbf{P}_2 \cdots \cap \mathbf{P}_n$ with P_0, P_1 anticommuting then $|\psi\rangle$ can be prepared from $|0\ldots0\rangle$ with a Clifford plus one T-gate circuit.*

Proof. As P_0, P_1 are anticommuting, by Theorem 41 there exists a Clifford circuit C such that $C : \mathbf{P}_0 \to -\mathbf{X}\otimes\mathbf{I}\otimes\cdots\mathbf{I}$ and $C : \mathbf{P}_1 \to \mathbf{Y}\otimes\mathbf{I}\otimes\cdots\mathbf{I}$. Hence

$$C\,|\psi\rangle : \left(\tfrac{1}{\sqrt{2}}(\mathbf{X}+\mathbf{Y})\otimes\mathbf{I}^{\otimes(n-1)}\right) \cap \mathbf{P}'_2 \cdots \cap \mathbf{P}'_n.$$

Note that each P'_j $(j = 2,\ldots,n)$ must commute with $\frac{1}{\sqrt{2}}(X+Y)\otimes I^{\otimes(n-1)}$ and hence $P_j = I\otimes Q_j$ or $P_j = \sigma_z \otimes Q_j$. In either case, apply a $T^\dagger$-gate to the first qubit gives $T_1^\dagger : \mathbf{P}'_j \to \mathbf{P}'_j$. Consequently,

$$T_1^\dagger C\,|\psi\rangle : \left(\mathbf{X}\otimes\mathbf{I}^{\otimes(n-1)}\right) \cap \mathbf{P}'_2 \cdots \cap \mathbf{P}'_n.$$

Now, we can apply Theorem 2 and obtain a Clifford C' such that $C'T_1^\dagger C\,|\psi\rangle : \mathbf{Z}_1 \cap \cdots \cap \mathbf{Z}_n$. Therefore $(C'T_1^\dagger C)^\dagger$ is our desired circuit. □

Note that the converse of this results is also true, after a fashion. One can always squander the single T-gate, by say applying it directly to $|0\ldots0\rangle$. But presuming this is not the case, and we prepare a state on which T acts nontrivially, then additional Clifford gates can only produce states $|\psi\rangle : \frac{1}{\sqrt{2}}(\mathbf{P}_0 + \mathbf{P}_1) \cap \mathbf{P}_2 \cdots \cap \mathbf{P}_n$ with P_0, P_1 anticommuting.

7.2 Example: Typing Toffoli

Now that we have a type for T, we can use it to derive a type for Toffoli, via the latter's standard decomposition into T, H and $CNOT$ gates:

```
Definition TOFFOLI a b c :=
  H c; CNOT b c; T† c; CNOT a c; T c; CNOT b c; T† c;
  CNOT a c; T b; T c; H c; CNOT a b; T a; T† b; CNOT a b.
```

Showing that $\texttt{TOFFOLI} : \mathbf{Z}_1 \to \mathbf{Z}_1 \cap \mathbf{Z}_2 \to \mathbf{Z}_2$ proves remarkably straightforward:

```
Definition TOFFOLI a b c :=
  INIT                 Z ⊗ I ⊗ I ∩ I ⊗ Z ⊗ I
  H c;                 Z ⊗ I ⊗ I ∩ I ⊗ Z ⊗ I
  CNOT b c; T† c;      Z ⊗ I ⊗ I ∩ I ⊗ Z ⊗ I
  CNOT a c; T c;       Z ⊗ I ⊗ I ∩ I ⊗ Z ⊗ I
  CNOT b c; T† c;      Z ⊗ I ⊗ I ∩ I ⊗ Z ⊗ I
  CNOT a c; T b; T c;  Z ⊗ I ⊗ I ∩ I ⊗ Z ⊗ I
  H c;                 Z ⊗ I ⊗ I ∩ I ⊗ Z ⊗ I
  CNOT a b; T a; T† b; Z ⊗ I ⊗ I ∩ Z ⊗ Z ⊗ I
  CNOT a b.            Z ⊗ I ⊗ I ∩ Z ⊗ Z ⊗ I
```

Noticeably, the derivation that $\texttt{TOFFOLI} : \mathbf{X}_3 \to \mathbf{X}_3$ also proves trivial (since `H c` immediately converts the $\mathbf{X}$ to a $\mathbf{Z}$), showing that a $|+\rangle$ in the third position isn't entangled by a Toffoli gate. By contrast, Toffoli's action on $\mathbf{Z}_3$ does get a bit messy (we leave off the coefficients for readability's sake):

```
Definition TOFFOLI a b c :=
  INIT       I ⊗ I ⊗ Z
  H c;       I ⊗ I ⊗ X
  CNOT b c;  I ⊗ I ⊗ X
  T† c;      I ⊗ I ⊗ X + I ⊗ I ⊗ -Y
  CNOT a c;  I ⊗ I ⊗ X + Z ⊗ I ⊗ -Y
  T c;       I ⊗ I ⊗ X + I ⊗ I ⊗ Y + Z ⊗ I ⊗ -Y + Z ⊗ I ⊗ X
  CNOT b c;  I ⊗ I ⊗ X + I ⊗ Z ⊗ Y + Z ⊗ Z ⊗ -Y + Z ⊗ I ⊗ X
  T† c;      I ⊗ I ⊗ X + I ⊗ I ⊗ -Y + I ⊗ Z ⊗ X + I ⊗ Z ⊗ Y +
             Z ⊗ Z ⊗ -X + Z ⊗ Z ⊗ -Y + Z ⊗ I ⊗ X + Z ⊗ I ⊗ -Y
  CNOT a c;  I ⊗ I ⊗ X + Z ⊗ I ⊗ -Y + I ⊗ Z ⊗ X + Z ⊗ Z ⊗ Y +
             Z ⊗ Z ⊗ -X + I ⊗ Z ⊗ -Y + Z ⊗ I ⊗ X + I ⊗ I ⊗ -Y
  T b;       I ⊗ I ⊗ X + Z ⊗ I ⊗ -Y + I ⊗ Z ⊗ X + Z ⊗ Z ⊗ Y +
             Z ⊗ Z ⊗ -X + I ⊗ Z ⊗ -Y + Z ⊗ I ⊗ X + I ⊗ I ⊗ -Y +
  T c;       I ⊗ I ⊗ X + I ⊗ I ⊗ Y + Z ⊗ I ⊗ -Y + Z ⊗ I ⊗ X +
             I ⊗ Z ⊗ X + I ⊗ Z ⊗ Y + Z ⊗ Z ⊗ Y + Z ⊗ Z ⊗ -X +
             Z ⊗ Z ⊗ -X + Z ⊗ Z ⊗ -Y + I ⊗ Z ⊗ -Y + I ⊗ Z ⊗ X +
             Z ⊗ I ⊗ X + Z ⊗ I ⊗ Y + I ⊗ I ⊗ -Y + I ⊗ I ⊗ X
  H c;       I ⊗ I ⊗ Z + I ⊗ I ⊗ -Y + Z ⊗ I ⊗ Y + Z ⊗ I ⊗ Z +
             I ⊗ Z ⊗ Z + I ⊗ Z ⊗ -Y + Z ⊗ Z ⊗ -Y + Z ⊗ Z ⊗ -Z +
             Z ⊗ Z ⊗ -Z + Z ⊗ Z ⊗ Y + I ⊗ Z ⊗ Y + I ⊗ Z ⊗ Z +
```

```
           Z ⊗ I ⊗ Z + Z ⊗ I ⊗ -Y + I ⊗ I ⊗ Y + I ⊗ I ⊗ Z
CNOT a b;  I ⊗ I ⊗ Z + I ⊗ I ⊗ -Y + Z ⊗ I ⊗ Y + Z ⊗ I ⊗ Z +
           Z ⊗ Z ⊗ Z + Z ⊗ Z ⊗ -Y + I ⊗ Z ⊗ -Y + I ⊗ Z ⊗ -Z +
           I ⊗ Z ⊗ -Z + I ⊗ Z ⊗ Y + Z ⊗ Z ⊗ Y + Z ⊗ Z ⊗ Z +
           Z ⊗ I ⊗ Z + Z ⊗ I ⊗ -Y + I ⊗ I ⊗ Y + I ⊗ I ⊗ Z
T a; T† b; I ⊗ I ⊗ Z + I ⊗ I ⊗ -Y + Z ⊗ I ⊗ Y + Z ⊗ I ⊗ Z +
           Z ⊗ Z ⊗ Z + Z ⊗ Z ⊗ -Y + I ⊗ Z ⊗ -Y + I ⊗ Z ⊗ -Z +
           I ⊗ Z ⊗ -Z + I ⊗ Z ⊗ Y + Z ⊗ Z ⊗ Y + Z ⊗ Z ⊗ Z +
           Z ⊗ I ⊗ Z + Z ⊗ I ⊗ -Y + I ⊗ I ⊗ Y + I ⊗ I ⊗ Z
CNOT a b.  I ⊗ I ⊗ Z + I ⊗ I ⊗ -Y + Z ⊗ I ⊗ Y + Z ⊗ I ⊗ Z +
           I ⊗ Z ⊗ Z + I ⊗ Z ⊗ -Y + Z ⊗ Z ⊗ -Y + Z ⊗ Z ⊗ -Z +
           Z ⊗ Z ⊗ -Z + Z ⊗ Z ⊗ Y + I ⊗ Z ⊗ Y + I ⊗ Z ⊗ Z +
           Z ⊗ I ⊗ Z + Z ⊗ I ⊗ -Y + I ⊗ I ⊗ Y + I ⊗ I ⊗ Z
CANCEL     I ⊗ I ⊗ Z + Z ⊗ I ⊗ Z + I ⊗ Z ⊗ Z + Z ⊗ Z ⊗ -Z +
           Z ⊗ Z ⊗ -Z + I ⊗ Z ⊗ Z + Z ⊗ I ⊗ Z + I ⊗ I ⊗ Z
COMBINE    I ⊗ I ⊗ Z + Z ⊗ I ⊗ Z + I ⊗ Z ⊗ Z + Z ⊗ Z ⊗ -Z
```

Note that, despite the presence of seven T or $T^\dagger$-gates (and hence a potential of 128 summands appearing in the additive type), only four of them enlarged the term – all T or $T^\dagger$ gates applied to either of the first two qubits failed to change the type. The $\mathbf{X}$ types all left the picture once we applied the second Hadamard, and the $\mathbf{Y}$ types all canceled out, leaving only $\mathbf{Z}$ and $\mathbf{I}$. In fact, every summand has a $\mathbf{Z}$ in the last position, and the first type leaves us no way to introduce a negative on the $\mathbf{Z}$.

Hence, we can conclude that `Toffoli` has type $\mathbf{Z}_1 \cap \mathbf{Z}_2 \cap \mathbf{Z}_3 \to \mathbf{Z}_1 \cap \mathbf{Z}_2 \cap \mathbf{Z}_3$.

7.3 General additive types

Clifford+T, as studied in the previous section, is universal in that all unitaries can be approximated as a composition of such gates. However, one often wishes to do an exact analysis or simply use a different universal gate set. Recall that our core semantics for the typing statement $|\psi\rangle : \mathbf{P}$ is that $|\psi\rangle$ is a $+1$-eigenstate of the (multi-qubit) Pauli operator P. The critical feature of Pauli operators is they are both unitary and Hermitian. Thus operators that are both unitary and Hermitian forms a natural basis to extend Gottesman types. Note that the Pauli operators form a linear basis of the vector space of all linear operators, and so for any unitary and Hermitian operator M, we can write $M = \sum_j c_j P_j$ for some (real) coefficients c_j and Pauli operators P_j.

Definition 17. *An* additive type *is an expression of the form* $\mathbf{M} = \sum_j c_j \mathbf{P}_j$ *where* $c_j \in \mathbb{R}$ *and* $\mathbf{P}_j$ *are Gottesman types, such that semantic operator* $M = \sum_j c_j P_j$ *is both unitary and Hermitian. We say a state* $|\psi\rangle$ *inhabits this type* $|\psi\rangle : \mathbf{M}$ *if* $M|\psi\rangle = |\psi\rangle$.

Lemma 18. *Any one-qubit state* $|\psi\rangle : \mathbf{M}$ *implies* $\mathbf{M} = a\mathbf{X} + b\mathbf{Y} + c\mathbf{Z}$ *with* $a^2 + b^2 + c^2 = 1$.

Proof. Any one-qubit operator may written $M = tI + a\sigma_x + b\sigma_y + c\sigma_z$. As M is Hermitian $t, a, b, c \in \mathbb{R}$. But M is also unitary so

$$I = M^2 = (t^2 + a^2 + b^2 + c^2)I + 2ta\sigma_x + 2tb\sigma_y + 2tc\sigma_z.$$

Therefore we see $t = 0$ and $a^2 + b^2 + c^2 = 1$ as desired. □

This lemma shows that 1-qubit additive types are particularly simple in that they form a representation of the familiar Bloch sphere.

Proposition 4 on separability was stated at a level of generality that supports additive types as follows.

Corollary 19. *Let* $\mathbf{I}^{(k-1)} \otimes \mathbf{M} \otimes \mathbf{I}^{(n-k)}$ *be an additive type, and suppose* $|\psi\rangle : \mathbf{I}^{(\mathbf{k-1})} \otimes \mathbf{M} \otimes \mathbf{I}^{(\mathbf{n-k})}$. *Then the* k^{th} *qubit of* $|\psi\rangle$ *is unentangled from the rest of the system.*

7.4 Arrow types of general unitary maps

The Heisenberg semantics of Clifford operators carries over to arrow types of general unitaries. With Cliffords, we claimed that a complete description of an n-qubit operator's type is as the intersection type of the operators evaluation on $\mathbf{X}_j$ and $\mathbf{Z}_j$ as $j = 1, \ldots, n$. In the case of 1-qubit unitaries consider a generic typing statement $U : \mathbf{M} \to \mathbf{N}$. Semantically this implies that $U|m\rangle\langle m|U^\dagger = |n\rangle\langle n|$, which in turn implies $UMU^\dagger = N$. If $\mathbf{M} = a\mathbf{X} + b\mathbf{Y} + c\mathbf{Z}$ then also $M = a\sigma_x + b\sigma_y + c\sigma_z$, and thus

$$N = U(a\sigma_x + b\sigma_y + c\sigma_z)U^\dagger = aU\sigma_xU^\dagger + bU\sigma_yU^\dagger + cU\sigma_zU^\dagger.$$

That is, if we knew merely $U : \mathbf{X} \to \mathbf{N}_x$ and $U : \mathbf{Z} \to \mathbf{N}_z$ then we can prove $U : \mathbf{Y} \to \mathbf{N}_y$, leading to an additive type with sound Heisenberg semantics,

$$N_y = U\sigma_yU^\dagger = iU\sigma_xU^\dagger U\sigma_zU^\dagger = iN_xN_z,$$

and so deduce $N = aN_x + bN_y + cN_z$. In other words, knowing the arrow type of U on domain $\mathbf{X}$ and $\mathbf{Z}$ suffices to know the arrow type of U on any additive type.

This extends to multi-qubit unitaries as expected. If we know type of a n-qubit unitary U on $\mathbf{X}_j$ and $\mathbf{Z}_j$ for any $j = 1, \ldots, n$, then we can compute $UPU^\dagger$ for any n-qubit Pauli operator. From this we can then compute the type of U on any additive type $\mathbf{M}$ from $UMU^\dagger = \sum_j c_j UP_jU^\dagger$.

Theorem 20. *Let* U *be a unitary circuit on* n *qubits composed of* t *number of* T*-gates and an arbitrary number of Clifford gates, and write its arrow types as* $U : \mathbf{X}_j \to \mathbf{M}_j$ *and* $U : \mathbf{Z}_j \to \mathbf{N}_j$, *for* $j = 1, \ldots, n$. *Then every coefficient of* $\mathbf{M}_j$ *and* $\mathbf{N}_j$ *is of the form* $\frac{c}{2^{s/2}}$ *where* $c, s \in \mathbb{Z}$ *and* $s \leq t$.

Proof. Inductively, if $t = 0$ then U is a Clifford operator and so each $\mathbf{M}_j$ and $\mathbf{N}_j$ is a Gottesman type, and hence as additive types all their coefficients are in $\{-1, 0, 1\}$ as desired.

Suppose the statement is true for all unitary circuits containing at most $t-1$ number of T-gates, and suppose U is a unitary circuit with t number of T-gates. Suppose $U = C \circ U'$ with C a Clifford operator. We claim U' has the same assumptions and requirements as U: clearly U' also contains t number of T-gates, and if we write $U' : \mathbf{X}_j \to \mathbf{M}'_j$ and $U' : \mathbf{Z}_j \to \mathbf{N}'_j$ then we must have $C : \mathbf{M}'_j \to \mathbf{M}_j$ and $C : \mathbf{N}'_j \to \mathbf{N}_j$; since C is a Clifford operator the coefficients of $\mathbf{M}_j$ (respectively $\mathbf{N}_j$) are the same as those of $\mathbf{M}'_j$ (respectively $\mathbf{N}'_j$) up to sign changes and reordering.

Therefore we may assume $U = T_k \circ U'$, where T_k represents a T-gate operating on the k-th qubit. For notational convenience let us assume $k = 1$ as the general case will follow identically. As above write $U' : \mathbf{X}_j \to \mathbf{M}'_j$ and $U' : \mathbf{Z}_j \to \mathbf{N}'_j$, and let us write

$$M'_j = I \otimes \left(\sum_J c_{J,0}P_{J,0}\right) + \sigma_x \otimes \left(\sum_J c_{J,1}P_{J,1}\right) + \sigma_y \otimes \left(\sum_J c_{J,2}P_{J,2}\right) + \sigma_z \otimes \left(\sum_J c_{J,0}P_{J,0}\right)$$

where each $P_{J,l}$ is a $(n-1)$-qubit Pauli operator. Inductively each coefficient satisfies

$2^{(t-1)/2}c_{J,l} \in \mathbb{Z}$. Then $T_1 : \mathbf{M}'_j \to \mathbf{M}_j$ and so

$$\begin{aligned} M_j &= I \otimes \left(\sum_J c_{J,0}P_{J,0}\right) + \tfrac{1}{\sqrt{2}}(\sigma_x + \sigma_y) \otimes \left(\sum_J c_{J,1}P_{J,1}\right) \\ &\quad + \tfrac{1}{\sqrt{2}}(-\sigma_x + \sigma_y) \otimes \left(\sum_J c_{J,2}P_{J,2}\right) + \sigma_z \otimes \left(\sum_J c_{J,0}P_{J,0}\right) \\ &= I \otimes \left(\sum_J c_{J,0}P_{J,0}\right) + \sigma_x \otimes \left(\sum_J \frac{c_{J,1} - c_{J,2}}{\sqrt{2}}P_{J,1}\right) \\ &\quad + \sigma_y \otimes \left(\sum_J \frac{c_{J,1} + c_{J,2}}{\sqrt{2}}P_{J,2}\right) + \sigma_z \otimes \left(\sum_J c_{J,0}P_{J,0}\right). \end{aligned}$$

Finally, $2^{t/2} \cdot \frac{c_{J,1} \pm c_{J,2}}{\sqrt{2}} = 2^{(t-1)/2}(c_{J,1} \pm c_{J,2}) \in \mathbb{Z}$. The same argument works for the $\mathbf{N}_j$. □

Note that certain unitaries, those that are Hermitian, also define an additive type. This was clear for Pauli operators in the context of Gottesman types: $\mathbf{X}$ is a type and $\sigma_x : (\mathbf{X} \to \mathbf{X}) \cap (\mathbf{Z} \to -\mathbf{Z})$. It is straightforward to check when a unitary is also Hermitian (up to a global phase) using just its arrow type. A Hermitian unitary has $U = U^\dagger = U^{-1}$ and so $U^2 = I$. Thus one just checks if $U^2 : \mathbf{X}_j \to \mathbf{X}_j$ and $U^2 : \mathbf{Z}_j \to \mathbf{Z}_j$ for all $j = 1, \ldots, n$.

Example 21. *Consider for example the Hadamard gate $H : (\mathbf{X} \to \mathbf{Z}) \cap (\mathbf{Z} \to \mathbf{X})$, which is also Hermitian, and so defines an additive type $\mathbf{H}$. It is straightforward to verify $\mathbf{H} = \frac{1}{\sqrt{2}}(\mathbf{X} + \mathbf{Z})$ by writing out H and $\frac{1}{\sqrt{2}}(\sigma_x + \sigma_z)$ in the computational basis and comparing the resulting matrices. However, we can deduce this expression (again up to a global sign change) from the arrow type of H as follows. From the lemma above we know $H = a\sigma_x + b\sigma_y + c\sigma_z$; just using the Pauli relations*

$$\begin{aligned} H\sigma_x H = (a^2 - b^2 - c^2)\sigma_x + 2ab\sigma_y + 2ac\sigma_z &= \sigma_z \\ H\sigma_z H = 2ac\sigma_x + 2bc\sigma_y + (c^2 - a^2 - b^2)\sigma_z &= \sigma_x. \end{aligned}$$

And so we obtain the quadratic system

$$\begin{aligned} 0 &= a^2 - b^2 - c^2 = ab = bc \\ 1 &= 2ac = a^2 + b^2 + c^2. \end{aligned}$$

This is easy to solve by noting $1 = (a^2 + b^2 + c^2) + (a^2 - b^2 - c^2) = 2a^2$ and so $a = c = \pm\frac{1}{\sqrt{2}}$ and $b = 0$.

While a general unitary U is not Hermitian, we can construct additive types associated to U by adding an ancillary qubit. This is based on the real and imaginary parts of U as defined as follows.

Definition 22. *Given any operator U define its* real part *as* $\mathrm{Re}(U) = \frac{1}{2}(U + U^\dagger)$ *and* imaginary part *as* $\mathrm{Im}(U) = \frac{1}{2i}(U - U^\dagger)$.

Clearly both $\mathrm{Re}(U)$ and $\mathrm{Im}(U)$ are Hermitian, however neither is generally unitary. Nonetheless, we claim they do satisfy $\mathrm{Re}(U)^2 + \mathrm{Im}(U)^2 = I$ and $\mathrm{Re}(U) \cdot \mathrm{Im}(U) = \mathrm{Im}(U) \cdot \mathrm{Re}(U)$, and so look like the blocks in a 2×2 block unitary. Hence we could extend them to a unitary with an additional qubit.

Lemma 23. *Let U be unitary and $\mathrm{Re}(U), \mathrm{Im}(U)$ be as above. Let P and Q be any anticommuting Pauli operators (on any number of qubits). Then $P \otimes \mathrm{Re}(U) + Q \otimes \mathrm{Im}(U)$ is both unitary and Hermitian.*

Proof. As $P, Q, \mathrm{Re}(U), \mathrm{Im}(U)$ are all Hermitian so is $P \otimes \mathrm{Re}(U) + Q \otimes \mathrm{Im}(U)$. Now compute

$$(P \otimes \mathrm{Re}(U) + Q \otimes \mathrm{Im}(U))^2 = I \otimes (\mathrm{Re}(U)^2 + \mathrm{Im}(U)^2) + PQ \otimes \mathrm{Re}(U)\mathrm{Im}(U) + QP \otimes \mathrm{Im}(U)\mathrm{Re}(U).$$

So, to finish, we merely complete our claims from above:

$$\mathrm{Re}(U)^2 + \mathrm{Im}(U)^2 = \frac{1}{4}(U^2 + 2I + (U^\dagger)^2) - \frac{1}{4}(U^2 - 2I + (U^\dagger)^2) = I$$

and

$$\mathrm{Re}(U)\mathrm{Im}(U) = \frac{1}{4i}(U^2 - (U^\dagger)^2) = \mathrm{Im}(U)\mathrm{Re}(U).$$

□

For example if $P = \sigma_x$ and $Q = \sigma_z$ then

$$P \otimes \mathrm{Re}(U) + Q \otimes \mathrm{Im}(U) = \begin{pmatrix} \mathrm{Im}(U) & \mathrm{Re}(U) \\ \mathrm{Re}(U) & -\mathrm{Im}(U) \end{pmatrix}.$$

Definition 24. *Let U be a n-qubit unitary, and $P, Q \in \{\sigma_x, \sigma_y, \sigma_z\}$ be distinct. Then the additive type of U relative to P, Q is the $(n+1)$-qubit additive type corresponding to $P \otimes \mathrm{Re}(U) + Q \otimes \mathrm{Im}(U)$. We denote this type as $\mathbf{P} \otimes \mathbf{Re}(U) + \mathbf{Q} \otimes \mathbf{Im}(U)$ (despite that $\mathbf{Re}(U)$ and $\mathbf{Im}(U)$ do not refer to types themselves).*

7.5 Example: Types for controlled unitaries

Consider the arrow types associated to the controlled-phase gate:

$$\texttt{control-}\sigma_z : \mathbf{I} \otimes \mathbf{X} \to \mathbf{Z} \otimes \mathbf{X}, \quad \texttt{control-}\sigma_z : \mathbf{X} \otimes \mathbf{I} \to \mathbf{X} \otimes \mathbf{Z},$$
$$\texttt{control-}\sigma_z : \mathbf{I} \otimes \mathbf{Z} \to \mathbf{I} \otimes \mathbf{Z}, \quad \texttt{control-}\sigma_z : \mathbf{Z} \otimes \mathbf{I} \to \mathbf{Z} \otimes \mathbf{I}.$$

The arrow types for σ_z itself are $\sigma_z : \mathbf{X} \to -\mathbf{X}$ and $\sigma_z : \mathbf{Z} \to \mathbf{Z}$. One is naturally led to the question: could we have deduced the arrow types for `control-`σ_z from those of σ_z? More generally, if we are given a unitary U can the arrow types of `control-`U be deduced directly from those of U. Unfortunately, the answer must be no in general as this would imply "control-" is a functor, which is not the case. Nonetheless, we can construct the arrow type of control-U from the arrow types of U *and* its additive type $\mathbf{X} \otimes \mathbf{Re}(U) + \mathbf{Y} \otimes \mathbf{Im}(U)$.

To discover a sound rule for deducing the arrow type of `control-`U, we move to its Heisenberg semantics and decompose our Hilbert space along the control bit $\mathfrak{H} = \mathfrak{H}_0 \oplus \mathfrak{H}_1$. Our controlled unitary control-U becomes the matrix operator

$$\begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix}.$$

That is, the component of our state where the control bit is $|0\rangle$ lives in $\mathfrak{H}_0$ where `control-`U is trivial, but the component of the state where the control bit is $|1\rangle$ lives in $\mathfrak{H}_1$ where `control-`U act as U. We use this semantics to assert typing judgments involving controlled operations in the following lemma.

Lemma 25. *Let U be any n-qubit unitary. Then define the types of `control-`U as*

1. `control-`$U : \mathbf{Z} \otimes \mathbf{I} \to \mathbf{Z} \otimes \mathbf{I}$, *and*

2. `control-`$U : \mathbf{X} \otimes \mathbf{I} \to \mathbf{X} \otimes \mathbf{Re}(U) + \mathbf{Y} \otimes \mathbf{Im}(U)$.

If P is a n-qubit Pauli, and $U : \mathbf{P} \to \mathbf{V}$ then

(3) `control-`$U : \mathbf{I} \otimes \mathbf{P} \to \mathbf{I} \otimes \frac{1}{2}(\mathbf{P} + \mathbf{V}) + \mathbf{Z} \otimes \frac{1}{2}(\mathbf{P} - \mathbf{V})$.

Then Heisenberg semantics is sound with respect to these axioms.

Proof. For (1), we simply note:

$$\begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & -I \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & U^\dagger \end{pmatrix} = \begin{pmatrix} I & 0 \\ 0 & -I \end{pmatrix}.$$

For (2), again we compute

$$\begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix} \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & U^\dagger \end{pmatrix} = \begin{pmatrix} 0 & U^\dagger \\ U & 0 \end{pmatrix}.$$

But now, similar to above, we compute

$$\begin{aligned} \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix} \begin{pmatrix} 0 & U^\dagger \\ U & 0 \end{pmatrix} &= \begin{pmatrix} U & 0 \\ 0 & U^\dagger \end{pmatrix} \\ &= \left(\tfrac{1}{2}(I + \sigma_z) \otimes U + \tfrac{1}{2}(I - \sigma_z) \otimes U^\dagger\right). \end{aligned}$$

Therefore

$$\begin{aligned} \begin{pmatrix} 0 & U^\dagger \\ U & 0 \end{pmatrix} &= (\sigma_x \otimes I)\left(\tfrac{1}{2}(I + \sigma_z) \otimes U + \tfrac{1}{2}(I - \sigma_z) \otimes U^\dagger\right) \\ &= (\sigma_x \otimes I)\left(I \otimes \tfrac{1}{2}(U + U^\dagger) + \sigma_z \otimes \tfrac{1}{2}(U - U^\dagger)\right) \\ &= \sigma_x \otimes \mathrm{Re}(U) + \sigma_y \otimes \mathrm{Im}(U). \end{aligned}$$

Finally for (3), we compute

$$\begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix} \begin{pmatrix} P & 0 \\ 0 & P \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & U^\dagger \end{pmatrix} = \begin{pmatrix} P & 0 \\ 0 & UPU^\dagger \end{pmatrix} = \begin{pmatrix} P & 0 \\ 0 & V \end{pmatrix}.$$

In the computational basis $\frac{1}{2}(I + \sigma_z) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ and $\frac{1}{2}(I - \sigma_z) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$. Therefore

$$\begin{aligned} \begin{pmatrix} P & 0 \\ 0 & V \end{pmatrix} &= \begin{pmatrix} P & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & V \end{pmatrix} \\ &= \tfrac{1}{2}(I + \sigma_z) \otimes P + \tfrac{1}{2}(I - \sigma_z) \otimes V \\ &= I \otimes \tfrac{1}{2}(P + V) + \sigma_z \otimes \tfrac{1}{2}(P - V). \end{aligned}$$

□

Example 26. *Recall $S : \mathbf{Z} \to \mathbf{Z}$ and $S : \mathbf{X} \to \mathbf{Y}$. It is straightforward to verify that* $\mathrm{Re}(S) = \frac{1}{2}(I + \sigma_z)$ *and* $\mathrm{Im}(S) = \frac{1}{2}(I - \sigma_z)$*. Thus we have*

$$\begin{aligned}
&\texttt{control-}S : \mathbf{Z} \otimes \mathbf{I} \to \mathbf{Z} \otimes \mathbf{I} \\
&\texttt{control-}S : \mathbf{X} \otimes \mathbf{I} \to \Big(\mathbf{X} \otimes \tfrac{1}{2}(\mathbf{I} + \mathbf{Z})\Big) + \Big(\mathbf{Y} \otimes \tfrac{1}{2}(\mathbf{I} - \mathbf{Z})\Big) \\
&\qquad\qquad\qquad\qquad = \Big(\tfrac{1}{2}(\mathbf{X} + \mathbf{Y}) \otimes \mathbf{I}\Big) + \Big(\tfrac{1}{2}(\mathbf{X} - \mathbf{Y}) \otimes \mathbf{Z}\Big) \\
&\texttt{control-}S : \mathbf{I} \otimes \mathbf{Z} \to \mathbf{I} \otimes \mathbf{Z} \\
&\texttt{control-}S : \mathbf{I} \otimes \mathbf{X} \to \Big(\mathbf{I} \otimes \tfrac{1}{2}(\mathbf{X} + \mathbf{Y})\Big) + \Big(\mathbf{Z} \otimes \tfrac{1}{2}(\mathbf{X} - \mathbf{Y})\Big).
\end{aligned}$$

Note that by Theorem 20, any unitary Clifford+T circuit that synthesizes `control-`S *requires at least 2 T-gates.*

Theorem 27. *Let U be a n-qubit Hermitian unitary with associated additive type $\mathbf{U}$. Then for each $k \geq 0$ we have* `control`^{k}`-`U *is also a Hermitian unitary and its associated additive type is given by*

$$\mathbf{C}^k\mathbf{U} = \mathbf{I}^{(k+n)} - \tfrac{1}{2^k}(\mathbf{I} - \mathbf{Z})^k \otimes (\mathbf{I}^n - \mathbf{U}).$$

Proof. As above, we write the operator relation

$$\begin{aligned}
\begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix} &= \tfrac{1}{2}(I + \sigma_z) \otimes I + \tfrac{1}{2}(I - \sigma_z) \otimes U \\
&= I \otimes \tfrac{1}{2}(I + U) + \sigma_z \otimes \tfrac{1}{2}(I - U).
\end{aligned}$$

Now we can prove the theorem by induction. Clearly the $k = 0$ case holds:

$$\mathbf{U} = \mathbf{C}^0\mathbf{U} = \mathbf{I}^n - (\mathbf{I}^n - \mathbf{U}).$$

Inductively suppose $\mathbf{C}^k\mathbf{U} = \mathbf{I}^{(k+n)} - \frac{1}{2^k}(\mathbf{I} - \mathbf{Z})^k \otimes (\mathbf{I}^n - \mathbf{U})$ then using the relation above

$$\begin{aligned}
\mathbf{C}^{k+1}\mathbf{U} &= \mathbf{I} \otimes \tfrac{1}{2}(\mathbf{I}^{(k+n)} + \mathbf{C}^k\mathbf{U}) + \mathbf{Z} \otimes \tfrac{1}{2}(\mathbf{I}^{(k+n)} - \mathbf{C}^k\mathbf{U}) \\
&= \mathbf{I} \otimes \tfrac{1}{2}(\mathbf{I}^{(k+n)} + \mathbf{I}^{(k+n)} - \tfrac{1}{2^k}(\mathbf{I} - \mathbf{Z})^k \otimes (\mathbf{I}^n - \mathbf{U})) \\
&\quad + \mathbf{Z} \otimes \tfrac{1}{2}(\mathbf{I}^{(k+n)} - \mathbf{I}^{(k+n)} + \tfrac{1}{2^k}(\mathbf{I} - \mathbf{Z})^k \otimes (\mathbf{I}^n - \mathbf{U})) \\
&= \mathbf{I}^{(k+1+n)} - \tfrac{1}{2^{k+1}}\mathbf{I} \otimes (\mathbf{I} - \mathbf{Z})^k \otimes (\mathbf{I}^n - \mathbf{U}) \\
&\quad + \tfrac{1}{2^{k+1}}\mathbf{Z} \otimes (\mathbf{I} - \mathbf{Z})^k \otimes (\mathbf{I}^n - \mathbf{U}) \\
&= \mathbf{I}^{(k+1+n)} - \tfrac{1}{2^{k+1}}(\mathbf{I} - \mathbf{Z})^{(k+1)} \otimes (\mathbf{I}^n - \mathbf{U})).
\end{aligned}$$

□

Corollary 28. $\mathbf{C}^{k-1}\mathbf{Z} = \mathbf{I}^k - \frac{1}{2^{k-1}}(\mathbf{I} - \mathbf{Z})^k$.

As a simple example of the utility of the above formulation, we can easily derive the arrow type of an arbitrarily multiply controlled Z operator as follows.

Theorem 29. *We have*

$$\begin{aligned}
&\texttt{control}^k\texttt{-}\sigma_z : \mathbf{Z}_j \to \mathbf{Z}_j \\
&\texttt{control}^k\texttt{-}\sigma_z : \mathbf{X}_j \to \mathbf{X}_j - \frac{1}{2^{k-1}}(\mathbf{I} - \mathbf{Z})^{j-1} \otimes \mathbf{X} \otimes (\mathbf{I} - \mathbf{Z})^{(k+1-j)}.
\end{aligned}$$

Proof. As $\texttt{control}^k\text{-}\sigma_z$ is symmetric, it suffices to prove these statements for $j = 1$. For the first we already have

$$\texttt{control}\text{-}(\texttt{control}^{k-1}\text{-}\sigma_z) : \mathbf{Z} \otimes \mathbf{I}^k \to \mathbf{Z} \otimes \mathbf{I}^k$$

from (1) of Lemma 25. Now from Lemma 25 part (2), and that $\texttt{control}^{k-1}\text{-}\sigma_z$ is Hermitian, we have

$$\texttt{control}\text{-}(\texttt{control}^{k-1}\text{-}\sigma_z) : \mathbf{X} \otimes \mathbf{I}^k \to \mathbf{X} \otimes \mathbf{C}^{k-1}\mathbf{Z}.$$

Then the result follows from the previous corollary. □

Corollary 30. *Any unitary Clifford+T circuit that synthesizes* $\texttt{control}^k\text{-}\sigma_z$ *contains at least* $(2k-2)$ *T-gates.*

Lemma 31. *For any unitary* U *we have*

1. $\mathrm{Re}(\texttt{control}^k\text{-}U) = \texttt{control}^k\text{-}(\mathrm{Re}(U))$, *and*
2. $\mathrm{Im}(\texttt{control}^k\text{-}U) = \frac{1}{2^k}(I - \sigma_z)^k \otimes \mathrm{Im}(U)$.

Proof. Note that

$$\frac{1}{2}\left[\begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix} + \begin{pmatrix} I & 0 \\ 0 & U^\dagger \end{pmatrix}\right] = \begin{pmatrix} I & 0 \\ 0 & \frac{1}{2}(U + U^\dagger) \end{pmatrix}$$

and so $\mathrm{Re}(\texttt{control-}U) = \texttt{control-}(\mathrm{Re}(U))$. Then (1) follows from straightforward recursion.

Similarly,

$$\frac{1}{2i}\left[\begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix} - \begin{pmatrix} I & 0 \\ 0 & U^\dagger \end{pmatrix}\right] = \begin{pmatrix} 0 & 0 \\ 0 & \frac{1}{2i}(U - U^\dagger) \end{pmatrix} = \frac{1}{2}(I - \sigma_z) \otimes \mathrm{Im}(U).$$

Therefore (2) also follows from recursion. □

Theorem 32. *Let* U *be any n-qubit unitary, and* $k > 0$. *Then for* $j = 1, \ldots, k$, *define the typing rules*

1. $\texttt{control}^k\text{-}U : \mathbf{Z}_j \to \mathbf{Z}_j$, *and*
2. $$\begin{aligned}\texttt{control}^k\text{-}U : \mathbf{X}_j \to \mathbf{X}_j &- \frac{1}{2^{k-1}}(\mathbf{I} - \mathbf{Z})^{j-1} \otimes \mathbf{X} \otimes (\mathbf{I} - \mathbf{Z})^{(k-j)} \otimes \mathbf{I}^n \\ &+ \frac{1}{2^{k-1}}(\mathbf{I} - \mathbf{Z})^{j-1} \otimes \mathbf{X} \otimes (\mathbf{I} - \mathbf{Z})^{(k-j)} \otimes \mathbf{Re}(U) \\ &+ \frac{1}{2^{k-1}}(\mathbf{I} - \mathbf{Z})^{j-1} \otimes \mathbf{Y} \otimes (\mathbf{I} - \mathbf{Z})^{(k-j)} \otimes \mathbf{Im}(U).\end{aligned}$$

If P *is a n-qubit Pauli, and* $U : \mathbf{P} \to \mathbf{V}$ *then*

(3) $\texttt{control}^k\text{-}U : \mathbf{I}^k \otimes \mathbf{P} \to \mathbf{I}^k \otimes \mathbf{P} - \frac{1}{2^k}(\mathbf{I} - \mathbf{Z})^k \otimes (\mathbf{P} - \mathbf{V})$.

Then Heisenberg semantics is sound with respect to these axioms.

Proof. Clearly (1) follows immediately from Lemma 25 part (1).

For (2), we will assume $j = 1$ for clarity as the general case follows identically. From Lemma 25 part (2) we have

$$\texttt{control-(control}^{k-1}\texttt{-}U) : \mathbf{X}\otimes\mathbf{I}^{(k+n-1)} \to \mathbf{X}\otimes\mathbf{Re}(\texttt{control}^{k-1}\texttt{-}U)+\mathbf{Y}\otimes\mathbf{Im}(\texttt{control}^{k-1}\texttt{-}U).$$

From part (1) of the previous lemma we have $\mathrm{Re}(\texttt{control}^{k-1}\texttt{-}U) = \texttt{control}^{k-1}\texttt{-}(\mathrm{Re}(U))$, and so from Theorem 27

$$\mathrm{Re}(\texttt{control}^{k-1}\texttt{-}U) = I^{k+n-1} - \frac{1}{2^{k-1}}(I-\sigma_z)^{(k-1)}(I^n - \mathrm{Re}(U)).$$

Part (2) of the that lemma simply gives $\mathrm{Im}(\texttt{control}^{k-1}\texttt{-}U) = \frac{1}{2^{k-1}}(I-\sigma_z)^{(k-1)} \otimes \mathrm{Im}(U)$. Substituting these into the formula above gives the desired results.

We prove (3) inductively. For $k = 1$, Lemma 25 part (3) gives

$$\begin{aligned}\texttt{control-}U : \mathbf{I}\otimes\mathbf{P} \to \mathbf{I}\otimes\tfrac{1}{2}(\mathbf{P}+\mathbf{V}) + \mathbf{Z}\otimes\tfrac{1}{2}(\mathbf{P}-\mathbf{V}) \\ = \tfrac{1}{2}\mathbf{I}\otimes\mathbf{P} + \tfrac{1}{2}\mathbf{I}\otimes\mathbf{V} + \tfrac{1}{2}\mathbf{Z}\otimes\mathbf{P} - \tfrac{1}{2}\mathbf{Z}\otimes\mathbf{V} \\ = \mathbf{I}\otimes\mathbf{P} - \tfrac{1}{2}(\mathbf{I}-\mathbf{Z})\otimes(\mathbf{P}-\mathbf{V}).\end{aligned}$$

Now suppose the formula above holds for $k-1$. Then from the typing statement we just derived,

$$\begin{aligned}&\texttt{control}^{k-1}\texttt{-(control-}U) : \mathbf{I}^{(k-1)}\otimes(\mathbf{I}\otimes\mathbf{P}) \to \\ &\quad \mathbf{I}^{(k-1)}\otimes(\mathbf{I}\otimes\mathbf{P}) - \tfrac{1}{2^{k-1}}(\mathbf{I}-\mathbf{Z})^{(k-1)}\otimes\Big[(\mathbf{I}\otimes\mathbf{P}) - \Big((\mathbf{I}\otimes\mathbf{P}) - \tfrac{1}{2}(\mathbf{I}-\mathbf{Z})\otimes(\mathbf{P}-\mathbf{V})\Big)\Big] \\ &\quad = \mathbf{I}^k\otimes\mathbf{P} - \tfrac{1}{2^k}(\mathbf{I}-\mathbf{Z})^k\otimes(\mathbf{P}-\mathbf{V}).\end{aligned}$$

□

8 Measurement for additive types

One missing component of our additive type system is normal forms. As a consequence, a full formalism for measurement on additive types is incomplete. Nonetheless, we can go some distance in characterizing post-measurement states of additive types using the projection semantics introduced in the introduction.

8.1 Projection Semantics

Recall that our core semantic interpretation $\mathbf{X}$ corresponds to the Pauli operator σ_x in the sense that $|+\rangle$ is the $+1$-eigenstate of σ_x. However to derive post-measurement types of general additive types, we need a second semantic interpretation where we associate each type to a projection operator, and a state inhabits the type precisely when it is in the image of associated projection operator. In this semantics,

$$[\![\mathbf{X}]\!] = |+\rangle\langle+|\,,\ [\![\mathbf{Y}]\!] = |i\rangle\langle i|\,,\ \text{and}\ [\![\mathbf{Z}]\!] = |0\rangle\langle 0|\,.$$

This clarifies the behaviour of $-$ as, for instance,

$$[\![-\mathbf{Z}]\!] = I - [\![\mathbf{Z}]\!] = I - |0\rangle\langle 0| = |0\rangle\langle 0|^{\perp} = |1\rangle\langle 1|\,.$$

Indeed, negation of types should behave like the orthogonal complement on the lattice of projections.

For any (multi-qubit) Pauli operator P, the projection onto its $+1$-eigenspace is precisely $\Pi_P^+ = \frac{1}{2}(I+P)$. Similarly, the projection onto its -1-eigenspace is $\Pi_P^- = \frac{1}{2}(I-P)$, illustrating the relationship between operator negation in one semantics versus orthogonal complement in the other.

8.2 Computing post-measurement states

When studying Gottesman types, we were able to exploit standard methods from the stabilizer formalism for treating measurements in Pauli bases. However, for general additive types, these techniques no longer apply. Hence we need to revisit measurement from the first principles.

First, consider the problem of measuring a single qubit in the z-basis. Post-measurement we know the state is of type $\mathbf{Z} \cup -\mathbf{Z}$ where the factor in this union depends on the measurement outcome. Although it is outside our logical formalism, we see the probability of these outcomes directly in the input type of the qubit.

Lemma 33. *Let* $\mathbf{M} = a\mathbf{X} + b\mathbf{Y} + c\mathbf{Z}$ *be an additive type and* $|\psi\rangle : \mathbf{M}$*. Then in the z-basis,*

$$\Pr_\psi\{\mathit{meas} = +1\} = \frac{1+c}{2}, \text{ and } \Pr_\psi\{\mathit{meas} = -1\} = \frac{1-c}{2}.$$

Proof. In our semantics $|\psi\rangle : \mathbf{M}$ when $|\psi\rangle$ is the $+1$-eigenvector of the unitary Hermitian unitary operator $M = a\sigma_x + b\sigma_y + c\sigma_z$. As $|\psi\rangle\langle\psi|$ is the projector onto the $+1$-eigenspace of M, and $I - |\psi\rangle\langle\psi|$ is the projector onto the -1-eigenspace, we must have

$$M = |\psi\rangle\langle\psi| - (I - |\psi\rangle\langle\psi|) = 2\,|\psi\rangle\langle\psi| - I.$$

Let Π^Z be the projector onto the $+1$-eigenspace of Z (that is $\Pi^Z = |0\rangle\langle 0|$). Born's rule has

$$\begin{aligned}\Pr_\psi\{\text{meas} = +1\} &= \operatorname{tr}\Big(\Pi^Z\,|\psi\rangle\langle\psi|\Big) = \tfrac{1}{2}\operatorname{tr}\Big(\Pi^Z(I+M)\Big)\\ &= \frac{1}{2}\Big(1 + a\operatorname{tr}\Big(\Pi^Z X\Big) + b\operatorname{tr}\Big(\Pi^Z Y\Big) + c\operatorname{tr}\Big(\Pi^Z Z\Big)\Big) = \frac{1+c}{2}.\end{aligned}$$

Similarly, $\Pr_\psi\{\text{meas} = -1\} = \frac{1-c}{2}$ follows. □

A similar fact holds for multi-qubit additive types, however it is significantly more challenging to derive. Let us illustrate the key ideas on 2 qubits. Suppose $|\psi\rangle : \mathbf{M}_{(1)} \cap \mathbf{M}_{(2)}$ where $\mathbf{M}_{(1)}, \mathbf{M}_{(2)}$ are 2-qubit additive types. Let M_1 and M_2 be the unitary Hermitian operators associated to these types. This typing statement means $|\psi\rangle$ is the joint $+1$-eigenvector of M_1 and M_2, which by its very existence implies $M_1M_2 = M_2M_1$. As above we have $|\psi\rangle\langle\psi|$ is the projector onto this space and thus

$$|\psi\rangle\langle\psi| = \tfrac{1}{4}(I + M_1)(I + M_2) = \tfrac{1}{4}(I + M_1 + M_2 + M_1M_2).$$

For convenience, let us write $M_0 = I$ and $M_3 = M_1M_2$. Suppose we measure the first qubit in the z-basis. As in the lemma above the measurement projector is $\Pi^Z \otimes I$ and Born's rule reads

$$\Pr_\psi\{\text{meas} = +1\} = \operatorname{tr}\Big((\Pi^Z \otimes I)\,|\psi\rangle\langle\psi|\Big) = \frac{1}{4}\sum_{j=0}^{3}\operatorname{tr}\Big((\Pi^Z \otimes I)M_j\Big).$$

To compute these traces we write

$$M_j = I \otimes N_{j0} + X \otimes N_{j1} + Y \otimes N_{j2} + Z \otimes N_{j3},$$

and so

$$\begin{aligned}\operatorname{tr}\Big((\Pi^Z \otimes I)M_j\Big) &= \operatorname{tr}\Big(\Pi^Z \otimes N_{j0}\Big) + \operatorname{tr}\Big(\Pi^Z X \otimes N_{j1}\Big) + \operatorname{tr}\Big(\Pi^Z Y \otimes N_{j2}\Big) + \operatorname{tr}\Big(\Pi^Z Z \otimes N_{j3}\Big)\\ &= \operatorname{tr}(N_{j0}) + \operatorname{tr}(N_{j3}).\end{aligned}$$

Now, $M_0 = I$, and for $j > 0$ we have M_j is trace zero. Thus we may write

$$N_{00} = I \text{ and } N_{01} = N_{02} = N_{03} = 0,$$

and for $j > 0$:

$$\begin{aligned} N_{j0} = &\quad \tilde{x}_j X + \tilde{y}_j Y + \tilde{z}_j Z \\ N_{j3} = c_j I &+ x_j X + y_j Y + z_j Z. \end{aligned} \tag{24}$$

So,

$$\begin{aligned} \Pr_\psi\{\text{meas} = +1\} &= \tfrac{1}{4}(2 + \operatorname{tr}(N_{13}) + \operatorname{tr}(N_{23}) + \operatorname{tr}(N_{33})) \\ &= \frac{1 + c_1 + c_2 + c_3}{2} \end{aligned}$$

where we extract each c_j as:

$$M_j = c_j Z \otimes I + \text{other terms.} \tag{25}$$

For c_1 and c_2 this is by direct examination of $\mathbf{M}_{(1)}$ and $\mathbf{M}_{(2)}$. However c_3 can only be obtained by computing $M_1 M_2$. A similar computation holds for the probability of measuring -1, and so we have proven the following result.

Proposition 34. *Suppose $|\psi\rangle : \mathbf{M}_{(1)} \cap \mathbf{M}_{(2)}$ where $\mathbf{M}_{(1)}, \mathbf{M}_{(2)}$ are 2-qubit additive types, and suppose we measure the first qubit in the z-basis. As above write M_1 and M_2 for the operators associated to these types and $M_0 = I$ and $M_3 = M_1 M_2$. For $j = 0, 1, 2, 3$ define*

$$M_j = I \otimes N_{j0} + X \otimes N_{j1} + Y \otimes N_{j2} + Z \otimes N_{j3}.$$

Then

$$\Pr_\psi\{\mathit{meas} = +1\} = p_+ = \frac{1 + c_1 + c_2 + c_3}{2}, \text{ and } \Pr_\psi\{\mathit{meas} = -1\} = p_- = \frac{1 - c_1 - c_2 - c_3}{2}$$

where the c_j are given in (25).

From this we can bootstrap the post-measurement type for a general 2-qubit state as follows.

Theorem 35. *On 2-qubit states, measurement in the z-basis of the first qubit is of type*

$$\mathit{meas}_1 : \mathbf{M}_{(1)} \cap \mathbf{M}_{(2)} \to (\mathbf{Z}_1 \cap \mathbf{M}_+) \cup ((-\mathbf{Z})_1 \cap \mathbf{M}_-)$$

where

$$\mathbf{M}_+ = \tfrac{1}{2p_+} \sum_{j=1}^{3} ((\tilde{x}_j + x_j)\mathbf{X} + (\tilde{y}_j + y_j)\mathbf{Y} + (\tilde{z}_j + z_j)\mathbf{Z}) \tag{26}$$

$$\mathbf{M}_- = \tfrac{1}{2p_-} \sum_{j=1}^{3} ((\tilde{x}_j - x_j)\mathbf{X} + (\tilde{y}_j - y_j)\mathbf{Y} + (\tilde{z}_j - z_j)\mathbf{Z}), \tag{27}$$

where $p_\pm$ are given in the proposition above, and the coefficients of $\mathbf{M}_\pm$ are in (24).

Proof. As above, write $p_+ = \frac{1+c_1+c_2+c_3}{2}$ for seeing outcome $+1$, then the post-measurement state given outcome $+1$ is

$$\begin{aligned}
\tfrac{1}{p_+}(\Pi^Z \otimes I)\left|\psi\right\rangle\left\langle\psi\right|(\Pi^Z \otimes I) &= \frac{1}{4p_+}\sum_{j=0}^{3}\Big((\Pi^Z \otimes I)M_j(\Pi^Z \otimes I)\Big) \\
&= \frac{1}{4p_+}\sum_{j=0}^{3}\Big(\Pi^Z \otimes N_{j0} + (\Pi^Z X \Pi^Z) \otimes N_{j1} + (\Pi^Z Y \Pi^Z) \otimes N_{j2} + \Pi^Z \otimes N_{j3}\Big) \\
&= \Pi^Z \otimes \frac{1}{4p_+}\left(I + \sum_{j=1}^{3}(N_{j0} + N_{j3})\right) \\
&= \Pi^Z \otimes \left(\tfrac{1}{2}I + \tfrac{1}{4p_+}\sum_{j=1}^{3}((\tilde{x}_j + x_j)X + (\tilde{y}_j + y_j)Y + (\tilde{z}_j + z_j)Z)\right).
\end{aligned}$$

While we wrote this as a density operator it is a pure state

$$\tfrac{1}{p_+}(\Pi^Z \otimes I)\left|\psi\right\rangle\left\langle\psi\right|(\Pi^Z \otimes I) = \left|0,\psi'\right\rangle\!\left\langle 0,\psi'\right|.$$

As above $|\psi'\rangle\langle\psi'| = \frac{1}{2}(I + M_+)$ so by examination the post-measurement state has type $\mathbf{Z}_1 \cap (\mathbf{M}_+)_2$.

For seeing outcome -1, which has probability $p_- = \frac{1-c_1-c_2-c_3}{2}$, the computation is similar:

$$\begin{aligned}
\tfrac{1}{p_-}((I-\Pi^Z) \otimes I)\left|\psi\right\rangle\left\langle\psi\right|((I-\Pi^Z) \otimes I) &= (I-\Pi^Z) \otimes \frac{1}{4p_-}\left(I + \sum_{j=1}^{3}(N_{j0} - N_{j3})\right) \\
&= \Pi^Z \otimes \left(\tfrac{1}{2}I + \tfrac{1}{4p_-}\sum_{j=1}^{3}((\tilde{x}_j - x_j)X + (\tilde{y}_j - y_j)Y + (\tilde{z}_j - z_j)Z)\right).
\end{aligned}$$

So the post-measurement state has type $(-\mathbf{Z})_1 \cap (\mathbf{M}_-)_2$ □

Example 36. *Note that in the case that $\mathbf{M}_{(1)}$ and $\mathbf{M}_{(2)}$ are Gottesman types, the Proposition above recovers our measurement rules from earlier. Each additive type only contains one Pauli term. From (25) we see that at most one c_j can be nonzero, as otherwise two of M_1, M_2, and M_3 would equal $Z \otimes I$ contradicting independence of $\mathbf{M}_{(1)}$ and $\mathbf{M}_{(2)}$. In the case where one $c_j = \pm 1$ the measurement is deterministic (with outcome equal to this c_j) and the input state is separable. So suppose this is not the case, and the measurement is uniformly random. One of M_1, M_2, and M_3 must be of the form $\pm X \otimes P$ for some Pauli, as otherwise one would be $Z \otimes I$ since they are independent and pairwise commuting. Without loss of generality suppose $M_1 = s_1 X \otimes P$, where $s_1 \in \{-1, +1\}$. As $M_3 = M_1 M_2$ one of M_2 or M_3 has of the form*

1. *$\pm I \otimes Q$ where Q commutes with P, or*
2. *$\pm Z \otimes Q$ where Q anti-commutes with P.*

Again without loss of generality we can assume M_2 take one of these forms. Therefore either:

1. *$M_2 = s_2 I \otimes Q$ and $M_3 = s_1 s_2 X \otimes Q$, and so in (24) the only nonvanishing coefficient is one of $\tilde{x}_2$, $\tilde{y}_2$, or $\tilde{z}_2$ (according to Q) and we obtain output type $\mathbf{M}_+ = \mathbf{M}_- = s_2\mathbf{Q}$; or,*

2. *$M_2 = s_2 Z \otimes Q$ and $M_3 = -s_1 s_2 Y \otimes iPQ$, and so in (24) the only nonvanishing coefficient is one of x_j, y_j, or z_k (again according to Q) and we have output types $\mathbf{M}_+ = s_2\mathbf{Q}$ and $\mathbf{M}_- = -s_2\mathbf{Q}$.*

Example 37. *If we have a single T-gate, what other sort of gates can we synthesize using it, Clifford gates, and measurement in the computational basis? We will focus only synthesizing another one-qubit gate using a single ancillary qubit that will be measured, and so this example parallels gate injection, which we study in the next section. By Theorem 16, prior to measurement we can assume we have a state*

$$|\psi\rangle : \tfrac{1}{\sqrt{2}}(\mathbf{P}_{(0)} + \mathbf{P}_{(1)}) \cap \mathbf{P}_{(2)}$$

where the 2-qubit Pauli operators P_0 and P_1 anticommute, and P_2 commutes with both P_0 and P_1. Without loss of generality we can assume the first qubit is then measured in the z-basis. Using the notation above $M_1 = \frac{1}{\sqrt{2}}(P_0+P_1)$, $M_2 = P_2$, and $M_3 = \frac{1}{\sqrt{2}}(P_0P_2+P_1P_2)$. As in the previous example, we focus on cases involving $\mathbf{Z} \otimes \mathbf{I}$.

Case 1: $\mathbf{P}_{(2)} = \pm\mathbf{Z} \otimes \mathbf{I}$ *In the notation above, $c_1 = c_3 = 0$ while $c_2 = \pm 1$, and hence the probability of measuring $Z = +1$ is 0 or 1 depending on the sign in $\mathbf{P}_{(2)}$. Specializing (24) to this case we must have*

$$M_1 = \tfrac{1}{\sqrt{2}}\mathbf{I} \otimes (\tilde{x}\mathbf{X} + \tilde{y}\mathbf{Y} + \tilde{z}\mathbf{Z}) + \tfrac{1}{\sqrt{2}}\mathbf{Z} \otimes (x\mathbf{X} + y\mathbf{Y} + z\mathbf{Z})$$
$$M_3 = \pm(\tfrac{1}{\sqrt{2}}\mathbf{I} \otimes (x\mathbf{X} + y\mathbf{Y} + z\mathbf{Z}) + \tfrac{1}{\sqrt{2}}\mathbf{Z} \otimes (\tilde{x}\mathbf{X} + \tilde{y}\mathbf{Y} + \tilde{z}\mathbf{Z}))$$

Hence the post-measured type is

$$\mathbf{M}'_\pm = \pm\tfrac{1}{\sqrt{2}}((\tilde{x} \pm x)\mathbf{X} + (\tilde{y} \pm y)\mathbf{Y} + (\tilde{z} \pm z)\mathbf{Z}).$$

As P_0 and P_1 anticommute, precisely one of x, y, z is nonzero and precisely one of $\tilde{x}, \tilde{y}, \tilde{z}$ is nonzero, and these cannot both be $x, \tilde{x}$ or $y, \tilde{y}$ or $z, \tilde{z}$. So by Proposition 16 this circuit is equivalent to one that using Clifford plus one T-gate (without measurement).

Case 2: $\mathbf{P}_{(0)} = \pm\mathbf{Z} \otimes \mathbf{I}$ *Note this case also covers when P_1, P_0P_2, or P_1P_2 is $\pm\sigma_z \otimes I$, after relabeling terms as needed. As P_0 and P_1 anticommute we must have $P_1 = \pm\sigma_x \otimes Q$ or $P_1 = \pm\sigma_y \otimes Q$ for some Pauli operator Q (that may be I). Hence M_1 does not contribute to the post-measurement type. Yet, P_2 must commute with both P_0 and P_1, and hence $P_2 = I \otimes Q'$ where $Q' \in \{\sigma_x, \sigma_y, \sigma_z\}$. But then M_3 will not contribute to the post-measurement type either, and hence $\mathbf{M}'_\pm = \mathbf{P}_{(2)}$ and the circuit is equivalent to a Clifford gate.*

Case 3: None of $\mathbf{P}_{(0)}, \mathbf{P}_{(1)}, \mathbf{P}_{(2)}$ is $\pm\mathbf{Z} \otimes \mathbf{I}$ *This case is somewhat tedious, and so we let the reader verify the details. Regardless, the measurement has probability $\frac{1}{2}$ of obtaining $z = +1$ or $z = -1$. In the subcase where $P_2 = \sigma_x \otimes Q$ or $P_2 = \sigma_y \otimes Q$, then the result is similar to Case 1 above in that between M_1 and M_3 precisely two of $x, y, z, \tilde{x}, \tilde{y}, \tilde{z}$ contribute to the output type, and so the circuit is equivalent to a Clifford with one T-gate circuit (without measurement). In the subcase $P_2 = I \otimes Q$, then the results is similar to Case 2 above in that the state is separable and hence the post-measurement type is Q and the circuit is equivalent to a Clifford gate. Finally in the subcase $P_2 = \sigma_z \otimes Q$, we must have $M_1 = \sigma_x \otimes N_1 + \sigma_y \otimes N_2$ (as otherwise either M_1 or M_3 would have a $\sigma_z \otimes I$ term); then just as above neither M_1 or M_3 contribute to the post-measurement type, which is Q, and so the circuit is equivalent to a Clifford gate.*

The n-qubit analysis follows in a similar way as with two qubits, however we do not have such a concrete result. Suppose $|\psi\rangle : \mathbf{M}_{(1)} \cap \cdots \cap \mathbf{M}_{(n)}$. Continuing our notation from above, let M_j be the unitary Hermitian operator associated to $\mathbf{M}_{(j)}$. Then

$$|\psi\rangle\langle\psi| = \frac{1}{2^n}\prod_{j=1}^{n}(I + M_j) = \frac{1}{2^n}\sum_{J\subseteq\{1,\cdots,n\}} M_J$$

where the "multi-index" J selects a subset of $\{1, \cdots, n\}$ over which $M_J = \prod_{j\in J} M_j$. Here we adopt the convention $M_\emptyset = I$ similar to $M_0 = I$ in the 2-qubit case. Then Born's rule for measuring the first qubit reads

$$\Pr\{\text{meas} = +1\} = \frac{1}{2^n}\sum_{J\subseteq\{1,\ldots,n\}} \operatorname{tr}\left((\Pi^Z \otimes I^{(n-1)})M_J\right).$$

Again we write

$$M_J = I \otimes N_{J0} + X \otimes N_{J1} + Y \otimes N_{J2} + Z \otimes N_{J3}$$

and just as in the 2-qubit case have

$$\operatorname{tr}\left((\Pi^Z \otimes I^{(n-1)})M_J\right) = \operatorname{tr}(N_{J0} + N_{J3}).$$

Now, $N_{\emptyset 0} = I$ and $N_{\emptyset K} = 0$. For $J \neq \emptyset$, we expand

$$N_{J0} = \sum_{K\neq\mathbf{0}} q_{JK}P_K \text{ and } N_{J3} = c_J I^{(n-1)} + \sum_{K\neq\mathbf{0}} r_{JK}P_K,$$

where here $K \in \{0, 1, 2, 3\}^{n-1}$ and for $K = (k_1, \ldots, k_{n-1})$ we write $P_K = P_{k_1} \otimes \cdots \otimes P_{k_{n-1}}$. Then for measuring the first qubit to be state $+1$ we have

$$p_+ = \Pr\{\text{meas} = +1\} = \frac{1}{2}\left(1 + \frac{1}{2^{n-1}}\sum_{J\neq\emptyset} c_J\right)$$

and the post-measurement state will be

$$\begin{aligned}&\tfrac{1}{p_+}(\Pi^Z \otimes I^{(n-1)})\,|\psi\rangle\,\langle\psi|\,(\Pi^Z \otimes I^{(n-1)})\\ &= \Pi^Z \otimes \frac{1}{2^{n-1}}\left(I^{(n-1)} + \frac{1}{2p_+}\sum_{J\neq\emptyset}\sum_{K\neq\mathbf{0}}(q_{JK} + r_{JK})P_K\right).\end{aligned}$$

Now however we face a challenge. The post-measurement state is a pure state $|\psi'\rangle$ and

$$|\psi'\rangle\langle\psi'| = \frac{1}{2^{n-1}}\left(I^{(n-1)} + \frac{1}{2p_+}\sum_{J\neq\emptyset}\sum_{K\neq\mathbf{0}}(q_{JK} + r_{JK})P_K\right).$$

But to find its type, we need to find $(n-1)$-qubit additive types $\mathbf{M}'_1, \ldots, \mathbf{M}'_{n-1}$ such that the associated operators satisfy

$$\prod_{j=1}^{n-1}(I^{(n-1)} + M'_j) = I^{(n-1)} + \frac{1}{2p_+}\sum_{J\neq\emptyset}\sum_{K\neq\mathbf{0}}(q_{JK} + r_{JK})P_K.$$

While this does not seem immediately tractable, we can prove a lemma that shows that one feature of measurement from Gottesman types carries over to general additive types: if a term in the intersection involves only I and Z in the measured qubit, then it becomes a term in the post-measurement type (possibly with a different sign).

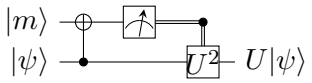

Figure 3: Gate injection circuit for U.

Lemma 38. *Suppose* $M = I \otimes N_0 + Z \otimes N_3$. *Then*

- $(\Pi^Z \otimes I^{(n-1)})M = (\Pi^Z \otimes (N_0 + N_3))(\Pi^Z \otimes I^{(n-1)})$, *and*
- $((I - \Pi^Z) \otimes I^{(n-1)})M = ((I - \Pi^Z) \otimes (N_0 - N_3))((I - \Pi^Z) \otimes I^{(n-1)})$.

Proof. Direct computation. □

To apply this lemma, without loss of generality suppose $|\psi\rangle : \mathbf{M}_{(1)} \cap \cdots \cap \mathbf{M}_{(n)}$ with $M_1 = I \otimes N_{1,0} + Z \otimes N_{1,3}$, and suppose the first qubit is measured (in the z-basis) with outcome $+1$. Then the post-measurement state is

$$\begin{aligned}
&\tfrac{1}{p_+}(\Pi^Z \otimes I^{(n-1)})\,|\psi\rangle\,\langle\psi|\,(\Pi^Z \otimes I^{(n-1)}) \\
&= \tfrac{1}{2^n p_+}(\Pi^Z \otimes I^{(n-1)}) \cdot \prod_{j=1}^{n}(I^n + M_j) \cdot (\Pi^Z \otimes I^{(n-1)}) \\
&= \tfrac{1}{2}(\Pi^Z \otimes (I^{(n-1)} + N_{1,0} + N_{1,3})) \cdot \tfrac{1}{2^{n-1}p_+}(\Pi^Z \otimes I^{(n-1)}) \cdot \prod_{j=2}^{n}(I^n + M_j) \cdot (\Pi^Z \otimes I^{(n-1)}).
\end{aligned}$$

Hence the output state $|0, \psi'\rangle : \mathbf{M}'_{(1)}$ where $M'_1 = N_{1,0} + N_{1,3}$ (up to a normalization term contained in p_+). The second conclusion in the lemma handles the case for outcome -1, where the post-measurement state $|1, \psi'\rangle : \mathbf{M}'_{(1)}$ where $M'_1 = N_{1,0} - N_{1,3}$ (again up to normalization).

8.3 Example: Gate Injection

A standard approach to fault-tolerant universal quantum computation is through implementing non-Clifford gates on codes through gate injection using associated "magic" states. While we can be explicit about the structure of the unitary gate we wish to inject, let us see what we can derive through simply appealing to typing statements. For concreteness, we focus on single-qubit unitaries and assume an axiom of the form

$$U : (\mathbf{X} \to \mathbf{M}) \cap (\mathbf{Z} \to \mathbf{Z}). \tag{28}$$

The additive type $\mathbf{M}$ cannot be arbitrary. In Heisenberg semantics these axioms would imply $U\sigma_z U^\dagger = \sigma_z$ and $U\sigma_x U^\dagger = M$. Since σ_x and σ_z anti-commute, so must M and σ_z and therefore $M = a\sigma_x + b\sigma_y$ where $a^2 + b^2 = 1$. We will parametrize $a = \cos\theta$ and $b = \sin\theta$. Naturally T fits this mold with $\theta = \frac{\pi}{4}$. It is straightforward to deduce

$$U : \mathbf{Y} \to i \cdot (\cos\theta \cdot \mathbf{X} + \sin\theta \cdot \mathbf{Y})\mathbf{Z} = -\sin\theta \cdot \mathbf{X} + \cos\theta \cdot \mathbf{Y},$$

and so we see U acts as a Bloch sphere rotation in the $\mathbf{X}/\mathbf{Y}$-plane by an angle θ.

We claim that we can synthesize U using the state $|m\rangle : \mathbf{M}$ in the circuit of fig. 3. That is we aim to show that this circuit has type

$$\mathbf{M}_1 \cap \mathbf{Z}_2 \to (\mathbf{Z}_1 \cup -\mathbf{Z}_1) \cap \mathbf{Z}_2 \text{ and } \mathbf{M}_1 \cap \mathbf{X}_2 \to (\mathbf{Z}_1 \cup -\mathbf{Z}_1) \cap \mathbf{M}_2$$

hence recovering eq. (28) in the separable second factor.

Beginning with $\mathbf{M}_1 \cap \mathbf{Z}_2 = (\mathbf{M} \otimes \mathbf{I}) \cap (\mathbf{I} \otimes \mathbf{Z})$ we evaluate the effect of the circuit on each term of the intersection:

$$(\cos\theta \cdot \mathbf{X} + \sin\theta \cdot \mathbf{Y}) \otimes \mathbf{I} \overset{NOTC}{\longrightarrow} \cos\theta \cdot \mathbf{X} \otimes \mathbf{I} + \sin\theta \cdot \mathbf{Y} \otimes \mathbf{Z}$$
$$\mathbf{I} \otimes \mathbf{Z} \overset{NOTC}{\longrightarrow} \mathbf{I} \otimes \mathbf{Z}.$$

Our post measurement type is then

$$((\mathbf{Z}_1 \cap \mathbf{I}_2) \cup (-\mathbf{Z}_1 \cap \mathbf{I}_2)) \cap ((\mathbf{Z}_1 \cap \mathbf{Z}_2) \cup (-\mathbf{Z}_1 \cap \mathbf{Z}_2)) = (\mathbf{Z}_1 \cap \mathbf{Z}_2) \cup (-\mathbf{Z}_1 \cap \mathbf{Z}_2),$$

as the second term on the left side is a subtype of the first.

Now turning to the case $\mathbf{M}_1 \cap \mathbf{X}_2 = \mathbf{M} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{X}$ we again evaluate the effect of the circuit:

$$(\cos\theta \cdot \mathbf{X} + \sin\theta \cdot \mathbf{Y}) \otimes \mathbf{I} \overset{NOTC}{\longrightarrow} \cos\theta \cdot \mathbf{X} \otimes \mathbf{I} + \sin\theta \cdot \mathbf{Y} \otimes \mathbf{Z}$$
$$\mathbf{I} \otimes \mathbf{X} \overset{NOTC}{\longrightarrow} \mathbf{X} \otimes \mathbf{X}.$$

Now however our input to the measurement

$$(\cos\theta \cdot \mathbf{X} \otimes \mathbf{I} + \sin\theta \cdot \mathbf{Y} \otimes \mathbf{Z}) \cap (\mathbf{X} \otimes \mathbf{X})$$

has too many terms with an $\mathbf{X}$ in the first factor. So we use the $\cap$-MUL-R rule to multiply the second term into the first yielding

$$(\cos\theta \cdot \mathbf{I} \otimes \mathbf{X} + \sin\theta \cdot \mathbf{Z} \otimes \mathbf{Y}) \cap (\mathbf{X} \otimes \mathbf{X}).$$

Now we apply the discussions from the previous section to write the post-measurement state as

$$(\mathbf{Z}_1 \cap (\cos\theta \cdot \mathbf{X} + \sin\theta \cdot \mathbf{Y})_2) \cup ((-\mathbf{Z})_1 \cap (\cos\theta \cdot \mathbf{X} - \sin\theta \cdot \mathbf{Y})_2).$$

So we see that upon measuring 0 the resulting state is of type $\mathbf{Z}_1 \cap \mathbf{M}_2$ as desired. But upon measuring 1 we have resulting type $(-\mathbf{Z})_1 \cap (\cos(-\theta)\mathbf{X} + \sin(-\theta)\mathbf{Y})_2$, and so have accomplished the rotation in the opposite direction. That is, we have implemented $U^\dagger$ and so doing a post-selected correction of U^2 as in Figure 3 produces the output type $(-\mathbf{Z})_1 \cap \mathbf{M}_2$ as desired.

9 Complexity of Type checking

We can now present the algorithm for type checking and making type inference on quantum circuits. These are noticeably different procedures: type checking verifies that a program has a given user-specified type while type inference attempts to derive a type for a program. Given that our type system is rich enough to give infinitely many types to any circuit (though many will be equivalent), we will not do full type inference on a circuit. Instead, we can ask the user to specify the input type and derive the output type through our inference rules. Alternatively, if the user has a specific output type in mind, we can do the same type inference procedure and normalize both the inferred and generated output (applying weakening rules as needed) and check that they are equivalent. Hence in this section, we will focus on type inference given a variety of programs and input types.

	$\mathbf{X}$	$\mathbf{Y}$	$\mathbf{Z}$
X	$\mathbf{X}$	$-\mathbf{Y}$	$-\mathbf{Z}$
Y	$-\mathbf{X}$	$\mathbf{Y}$	$-\mathbf{Z}$
Z	$-\mathbf{X}$	$-\mathbf{Y}$	$\mathbf{Z}$
H	$\mathbf{Z}$	$-\mathbf{Y}$	$\mathbf{X}$
S	$\mathbf{Y}$	$-\mathbf{X}$	$\mathbf{Z}$
T	$\mathbf{X}+\mathbf{Y}$	$\mathbf{Y}-\mathbf{X}$	$\mathbf{Z}$
$T^\dagger$	$\mathbf{X}-\mathbf{Y}$	$\mathbf{X}+\mathbf{Y}$	$\mathbf{Z}$

Table 1: Axiomatized and derived behavior of common one-qubit gates.

	$\mathbf{X}\otimes\mathbf{I}$	$\mathbf{I}\otimes\mathbf{X}$	$\mathbf{Y}\otimes\mathbf{I}$	$\mathbf{I}\otimes\mathbf{Y}$	$\mathbf{Z}\otimes\mathbf{I}$	$\mathbf{I}\otimes\mathbf{Z}$
$CNOT$	$\mathbf{X}\otimes\mathbf{X}$	$\mathbf{I}\otimes\mathbf{X}$	$\mathbf{Y}\otimes\mathbf{X}$	$\mathbf{Z}\otimes\mathbf{Y}$	$\mathbf{Z}\otimes\mathbf{I}$	$\mathbf{Z}\otimes\mathbf{Z}$
CZ	$\mathbf{X}\otimes\mathbf{Z}$	$\mathbf{Z}\otimes\mathbf{X}$	$\mathbf{Y}\otimes\mathbf{Z}$	$\mathbf{Z}\otimes\mathbf{Y}$	$\mathbf{Z}\otimes\mathbf{I}$	$\mathbf{I}\otimes\mathbf{Z}$

	$\mathbf{X}\otimes\mathbf{X}$	$\mathbf{X}\otimes\mathbf{Y}$	$\mathbf{X}\otimes\mathbf{Z}$	$\mathbf{Y}\otimes\mathbf{X}$	$\mathbf{Y}\otimes\mathbf{Y}$	$\mathbf{Y}\otimes\mathbf{Z}$	$\mathbf{Z}\otimes\mathbf{X}$	$\mathbf{Z}\otimes\mathbf{Y}$	$\mathbf{Z}\otimes\mathbf{Z}$
$CNOT$	$\mathbf{X}\otimes\mathbf{I}$	$\mathbf{Y}\otimes\mathbf{Z}$	$-\mathbf{Y}\otimes\mathbf{Y}$	$\mathbf{Y}\otimes\mathbf{I}$	$-\mathbf{X}\otimes\mathbf{Z}$	$\mathbf{X}\otimes\mathbf{Y}$	$\mathbf{Z}\otimes\mathbf{X}$	$\mathbf{I}\otimes\mathbf{Y}$	$\mathbf{I}\otimes\mathbf{Z}$
CZ	$\mathbf{Y}\otimes\mathbf{Y}$	$-\mathbf{Y}\otimes\mathbf{X}$	$\mathbf{X}\otimes\mathbf{I}$	$-\mathbf{X}\otimes\mathbf{Y}$	$\mathbf{X}\otimes\mathbf{X}$	$\mathbf{Y}\otimes\mathbf{I}$	$\mathbf{I}\otimes\mathbf{X}$	$\mathbf{I}\otimes\mathbf{Y}$	$\mathbf{Z}\otimes\mathbf{Z}$

Table 2: Behavior of common two-qubit gates over all Pauli pairs.

Checking simple tensor types Given a Clifford circuit and an input type $\mathbf{P}_1 \otimes \mathbf{P}_2 \otimes \cdots \otimes \mathbf{P}_n$ (consisting of no intersections or additive terms), we can type check the circuit in $O(m)$, where m is the number of gates. This follows from the fact that we can update the type on every gate application in constant time. In practice, this only takes a single lookup, since it proves convenient to add a number of derived typing judgments to the system (Tables 1 and 2). Note that we assume tensors are implemented by arrays, saving us the time of iterating through an n qubit list.

Checking intersection types Using Gottesman types, we can fully describe the semantics of a Clifford circuit (though we rarely will). Doing so requires determining the output type for $I^{k-1} \otimes \mathbf{X}_k \otimes I^{n-k}$ and $I^{k-1} \otimes \mathbf{Z}_k \otimes I^{n-k}$, for every $1 \leq k \leq n$ where n is the number of qubits. There are precisely $2n$ terms in this intersection, so the time to infer the fully descriptive output type is $O(mn)$.

Checking fully separable types When we get to separable types, we have to start doing normalization (§3). The normalization procedure iterates over each tensor in an intersection and then multiplies it by potentially all the remaining tensors. Since there are at most $2n$ elements in the intersection and each tensor is of length n, this winds up being an $O(n^3)$ operation. Once the normalization is done, applying the separability rules is straightforward. This gives us a complexity of $O(mn + n^3)$, which we can simplify to $O(mn)$ (our previous result) when $m \gg n^2$.

The complexity of measurement Measurement is where some complexity can start to appear especially with it being a non-unitary operation. In general, we want to use the *principle of deferred measurement* [20, §4.4] to push off measurements until the end of the circuit, allowing us to perform normalization only once. A single-qubit post-measurement type doubles in size when a random outcome is expected due to the use of unions. Then,

performing m measurements could potentially add a 2^m factor increase in number of terms. This is in line with the fact that there could be 2^m possible outcomes to track.

In practice, however, it is more common for us to post-select on certain measurement outcomes or perform subsequent operations conditioned on certain outcomes (e.g., error correction). In such cases, it will be possible to simplify the expression or focus only on a pre-determined set of outcomes to understand the measurement behaviour. The cost for computing the post-measurement state for a single Pauli measurement on an n-qubit system with ℓ terms in the union is $O(2\ell n^3)$.

The Clifford+T set and exponential blowup Naturally, universal quantum computing is the real test case for our type system. We know that our system is capable of fully describing arbitrary quantum computations, so unless quantum computing is efficiently simulable, we cannot efficiently typecheck arbitrary quantum circuits. This is clear in the case of Toffoli. As we saw in §7.2, despite having seven T gates, checking the Toffoli circuit only involves additive types with at most 4 terms (in the worst case). So in some sense, Toffoli only has an "effective" T-depth of 2 (which is essentially the content of Theorem 20).

Nevertheless, in the worst case, the running time of our typechecking algorithm is $O(2^t)$, where t is the number of T-gates, illustrating that our system cannot be efficiently applied to arbitrary circuits.

Measuring additive types At this time, we refrain from providing any asymptotic expressions for the complexity of measuring additive types as we only consider very restricted cases of measure single and 2-qubit systems. These are performed in an ad-hoc way by manipulating the underlying matrix corresponding to the given types. Hence, even generalizing the current process may not provide any non-trivial insights into asymptotic complexities beyond taking $O(2^n)$ time for an n-qubit system.

10 Related Work

We are not the first to consider lightweight static analysis of quantum programs, though similar works tend not to use type systems, but rather *abstract interpretation*. Abstract interpretation was developed by Cousot and Cousot [7] in order to show useful properties of programs at low cost. Abstract interpretation necessarily sacrifices fidelity (being able to perfectly describe a program) in favor of efficiency. Perdrix [22] was the first to apply abstract interpretation to quantum programs (expanding on earlier work [23] that did use types), but his system was quite limited: It could only precisely characterize a qubit as being in the z or x basis and *conservatively* tracked entanglement, meaning that it would err on the side of saying qubits were entangled if it couldn't rule that out.

Honda [15] presented a more powerful system based, like ours, on the stabilizer formalism. Rather than use types, it represented states using stabilizer arrays, which can be translated to our type system but are rather less useful as human-readable types. It dealt with non-stabilizer states simply by treating them as black boxes (literally represented as ■), which could propagate throughout the program. This could be useful in a few cases, such as where a non-stabilizer state was quickly discarded, but generally meant the system could not meaningfully speak about non-Clifford circuits.

More recently, Yu and Palsberg [38] developed an approach to quantum abstract interpretation based on *reduced density matrices*, specifically 4×4 partial traces of the full system. The expressivity of such an approach is not clear and it is mostly used to check

that qubits are in $|0\rangle$ or $|1\rangle$ states in practice. However, it does demonstrate remarkable performance in assertion checking and admits the possibility of using larger, more informative, reduced density matrices.

There are two approaches to quantum program verification that are adjacent to our own. Quantum assertions [16, 19, 18] allow one to embed assertions inside programs that will check that a given property holds. While prior work was limited to checking simple assertions, Li et al. [18] treats arbitrary projections as assertions. However, these systems can fail at runtime (eg. if the measured state has some probability of being in the desired state), and also require us to check a program's behavior on a quantum device or sufficiently powerful simulator. At the other end of the spectrum are sophisticated logical systems for quantum programs [37, 34] and powerful tools to formally verify quantum program behavior [4, 14]. However, even with an assist from automation, these tools tend to require substantial effort on the part of the programmer. We refer the reader to two recent surveys [5, 17] for an in-depth analysis of the advantages and disadvantages of these approaches.

11 Future work

There are still a variety of ways to further enrich our type system and these provide many promising avenues for us to explore.

Typing Quantum Channels Other than measurement, all the operations we type are unitary circuits. More general quantum operations are given by completely positive trace preserving maps i.e., quantum channels. Extending our type system to handle quantum channels could potentially allow us to perform static analysis on quantum cryptography and communication protocols. A starting point for this would be to use additive types and unions to characterize partial traces and post-selection.

Applications for error-correcting codes Implementing a fault-tolerant universal set of gates transversally will reduce the overall cost of error correction. However, as this cannot be achieved using just one code, a common method used switches between two sets of codes, each having a different set of transversal gates [2]. Extending our system to either typecheck or even infer the structure of the code-switching circuit given the type of two codes would prove to be fruitful. Similarly, inferring the encoding and decoding circuits for a code given its type could also be of value in verifying the implementation of error correcting codes.

Normal forms for additive types Finding a canonical representation for additive types is imperative to effectively check type equivalences. A big roadblock to it is that, unlike with Gottesman types, additive types (especially, multi-qubit ones) could have terms that neither commute nor anticommute. This makes it hard to find a normalization procedure for them similar to that in §3. Additionally, this also limits our ability to make multi-qubit separability judgements in the additive case.

General measurement for additive types Although we have outlined some cases in §8 where we can type the post-measurement states, this is limited to performing z-basis measurement on single and two qubit systems. In order to fully exploit the power of additive types, it is essential that we have a full characterization for post-measurement

states. An immediate consequence of this could be a deeper analysis of multi-qubit magic state types and applications associated with them.

Explicit use of subtyping In this paper, we used explicit typing rules like ∩-E, ∪-I, ∩-Arr-Dist to allow us to replace a typing judgement with a weaker one. These are all inspired by subtyping rules for classical programming languages. Our desire for our type system to be both small and syntax directed led us to make these rules explicit but as the system grows, this design choice begins to add complexity, rather than simplifying things. Future versions of this system may make use of explicit subtyping rules, allowing for the flexible manipulation of types.

A general purpose language with types Our ultimate goal is to find or develop a suitable programming language to augment with our type system. Such a language should have rich language features like first-class functions, recursion and while loops. However, since quantum programming languages (like Quipper [13] or Qiskit [25]) tend to be *circuit generation languages*, we would want our system to be dependently typed in order to accurately characterize parameterized circuit families (like QWIRE [21] or Proto-Quipper [10]). This will likely prove challenging. We will also run into the issue that not every circuit can be efficiently type-checked in our language. As a result, we will likely want to draw on the gradual types [29] literature to allow our type system to make guarantees about well-typed programs while allowing untyped programs. The fact that we can give any program the type $\mathbf{I}^n \to \mathbf{I}^n$, where n is the circuit width, could be the starting point for this work.

Acknowledgments

This material is based upon work supported by EPiQC, an NSF Expedition in Computing, under Grant No. 1730449 and the Air Force Office of Scientific Research under Grant No. FA95502110051.

References

[1] Scott Aaronson and Daniel Gottesman. "Improved Simulation of Stabilizer Circuits". In: *Physical Review A* 70.5 (2004), p. 052328. DOI: 10.1103/physreva.70.052328. arXiv: quant-ph/0406196.

[2] Jonas T. Anderson, Guillaume Duclos-Cianci, and David Poulin. "Fault-Tolerant Conversion between the Steane and Reed-Muller Quantum Codes". In: *Phys. Rev. Lett.* 113 (8 2014), p. 080501. DOI: 10.1103/PhysRevLett.113.080501.

[3] Benjamin Bichsel et al. "Silq: A High-Level Quantum Language with Safe Uncomputation and Intuitive Semantics". In: *Proc. PLDI '20*. New York, NY, USA: ACM, 2020, pp. 286–300. DOI: 10.1145/3385412.3386007. URL: https://files.sri.inf.ethz.ch/website/papers/pldi20-silq.pdf.

[4] Christophe Chareton et al. "An Automated Deductive Verification Framework for Circuit-Building Quantum Programs". In: *Programming Languages and Systems, ESOP 2021*. Ed. by Nobuko Yoshida. Vol. 12648. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, pp. 148–177. DOI: 10.1007/978-3-030-72019-3_6.

[5] Christophe Chareton et al. *Formal Methods for Quantum Programs: A Survey*. To appear as Chapter "Formal methods for Quantum Algorithms" in "Handbook of Formal Analysis and Verification in Cryptography", CRC. 2021. arXiv: 2109.06493.

[6] Richard Cleve and Daniel Gottesman. "Efficient Computations of Encodings for Quantum Error Correction". In: *Phys. Rev. A* 56 (1 1997), pp. 76–82. DOI: 10.1103/PhysRevA.56.76. arXiv: quant-ph/9607030.

[7] Patrick Cousot and Radhia Cousot. "Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints". In: *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages, Los Angeles, California, USA, January 1977*. New York, NY, USA: ACM, 1977, pp. 238–252. DOI: 10.1145/512950.512973. URL: https://courses.cs.washington.edu/courses/cse503/10wi/readings/p238-cousot.pdf.

[8] David Deutsch. "Quantum theory, the Church–Turing principle and the universal quantum computer". In: *Proc. R. Soc. Lond. A* 400.1818 (1985), pp. 97–117. DOI: 10.1098/rspa.1985.0070.

[9] David Deutsch and Richard Jozsa. "Rapid solution of problems by quantum computation". In: *Proc. R. Soc. Lond. A* 439.1907 (1992), pp. 553–558. DOI: 10.1098/rspa.1992.0167.

[10] Peng Fu, Kohei Kishida, and Peter Selinger. "Linear Dependent Type Theory for Quantum Programming Languages: Extended Abstract". In: *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '20. New York, NY, USA: ACM, 2020, pp. 440–453. DOI: 10.1145/3373718.3394765. arXiv: 2004.13472.

[11] Daniel Gottesman. "Class of quantum error-correcting codes saturating the quantum Hamming bound". In: *Phys. Rev. A* 54.3 (1996), pp. 1862–1868. DOI: 10.1103/physreva.54.1862. arXiv: quant-ph/9604038.

[12] Daniel Gottesman. "The Heisenberg Representation of Quantum Computers". In: *Group22: Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics*. Cambridge, MA: International Press, 1998, pp. 32–43. arXiv: quant-ph/9807006.

[13] Alexander S. Green et al. "Quipper: A Scalable Quantum Programming Language". In: *Proc. PLDI '13*. New York, NY, USA: ACM, 2013, pp. 333–342. DOI: 10.1145/2491956.2462177. arXiv: 1304.3390.

[14] Kesha Hietala et al. "Proving Quantum Programs Correct". In: *12th International Conference on Interactive Theorem Proving (ITP 2021)*. Ed. by Liron Cohen and Cezary Kaliszyk. Vol. 193. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, 21. DOI: 10.4230/LIPIcs.ITP.2021.21. URL: https://github.com/inQWIRE/SQIR.

[15] Kentaro Honda. "Analysis of Quantum Entanglement in Quantum Programs using Stabilizer Formalism". In: *Proc. QPL '15*. Vol. 195. Waterloo, NSW, Australia: Open Publishing Association, 2015, pp. 262–272. DOI: 10.4204/EPTCS.195.19.

[16] Yipeng Huang and Margaret Martonosi. "Statistical Assertions for Validating Patterns and Finding Bugs in Quantum Programs". In: *Proceedings of the 46th International Symposium on Computer Architecture*. ISCA '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 541–553. DOI: 10.1145/3307650.3322213.

[17] Marco Lewis, Sadegh Soudjani, and Paolo Zuliani. *Formal Verification of Quantum Programs: Theory, Tools and Challenges*. 2021. arXiv: 2110.01320.

[18] Gushu Li et al. “Projection-Based Runtime Assertions for Testing and Debugging Quantum Programs”. In: *Proceedings of the ACM on Programming Languages* 4.OOPSLA, 150 (2020), p. 150. DOI: 10.1145/3428218.

[19] Ji Liu, Gregory T. Byrd, and Huiyang Zhou. “Quantum Circuits for Dynamic Runtime Assertions in Quantum Computation”. In: *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. ASPLOS ’20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 1017–1030. DOI: 10.1145/3373376.3378488.

[20] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge: Cambridge University Press, 2010. DOI: 10.1017/CBO9780511976667.

[21] Jennifer Paykin, Robert Rand, and Steve Zdancewic. “QWIRE: A Core Language for Quantum Circuits”. In: *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL ’17)*. New York, NY: ACM, 2017, pp. 846–858. DOI: 10.1145/3009837.3009894.

[22] Simon Perdrix. “Quantum Entanglement Analysis Based on Abstract Interpretation”. In: *Static Analysis*. Berlin, Heidelberg: Springer, 2008, pp. 270–282. DOI: 10.1007/978-3-540-69166-2_18. arXiv: 0801.4230.

[23] Simon Perdrix. “Quantum Patterns and Types for Entanglement and Separability”. In: *Electron. Notes Theor. Comput. Sci.* 170 (2007). Proc. QPL ’05, pp. 125–138. DOI: 10.1016/j.entcs.2006.12.015.

[24] Benjamin C. Pierce. *Types and Programming Languages*. Cambridge, MA: MIT Press, 2002.

[25] Qiskit Community. *Qiskit: An Open-Source Framework for Quantum Computing*. 2017. DOI: 10.5281/zenodo.2562110. URL: https://github.com/Qiskit/qiskit.

[26] Robert Rand, Jennifer Paykin, and Steve Zdancewic. “QWIRE Practice: Formal Verification of Quantum Circuits in Coq”. In: *Proceedings 14th International Conference on Quantum Physics and Logic (QPL ’17)*. Waterloo, NSW, Australia: Open Publishing Association, 2017, pp. 119–132. DOI: 10.4204/EPTCS.266.8.

[27] Robert Rand et al. “ReQWIRE: Reasoning about Reversible Quantum Circuits”. In: *Proc. QPL ’18*. Waterloo, NSW, Australia: Open Publishing Association, 2018, pp. 299–312. DOI: 10.4204/EPTCS.287.17.

[28] Peter Selinger and Benoît Valiron. “A lambda calculus for quantum computation with classical control”. In: *Mathematical Structures in Computer Science* 16.3 (2006), pp. 527–552. DOI: 10.1017/S0960129506005238.

[29] Jeremy G. Siek and Walid Taha. “Gradual Typing for Functional Languages”. In: *Seventh Workshop on Scheme and Functional Programming*. ACM, 2006, pp. 81–92. URL: http://scheme2006.cs.uchicago.edu/13-siek.pdf.

[30] Frederick Smith, David Walker, and Greg Morrisett. “Alias Types”. In: *Programming Languages and Systems (ESOP ’00)*. Berlin, Heidelberg: Springer, 2000, pp. 366–381. DOI: 10.1007/3-540-46425-5_24.

[31] Andrew Steane. “Multiple-particle interference and quantum error correction”. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 452.1954 (1996), pp. 2551–2577. DOI: 10.1098/rspa.1996.0136.

[32] Andrew M. Steane. "Active Stabilization, Quantum Computation, and Quantum State Synthesis". In: *Phys. Rev. Lett.* 78 (11 1997), pp. 2252–2255. DOI: 10.1103/PhysRevLett.78.2252. arXiv: quant-ph/9611027.

[33] Krysta Svore et al. "Q#: Enabling Scalable Quantum Computing and Development with a High-level DSL". In: *Proc. Real World Domain Specific Languages Workshop (RWDSL) 2018*. New York, NY, USA: ACM, 2018, 7:1–7:10. DOI: 10.1145/3183895.3183901. arXiv: 1803.00652.

[34] Dominique Unruh. "Quantum Hoare Logic with Ghost Variables". In: *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '19. Los Alamitos, CA, USA: IEEE Computer Society, 2019, pp. 1–13. DOI: 10.1109/LICS.2019.8785779. arXiv: 1902.00325.

[35] Philip Wadler. "Linear Types can Change the World!" In: *Programming concepts and methods: Proceedings of the IFIP Working Group 2.2/2.3 Working Conference on Programming Concepts and Methods, Sea of Galilee, Israel, 2-5 April, 1990*. Ed. by Manfred Broy and Cliff B. Jones. Amsterdam: North-Holland, 1990, pp. 561–581. URL: https://homepages.inf.ed.ac.uk/wadler/topics/linear-logic.html#linear-types.

[36] Hongwei Xi and Frank Pfenning. "Eliminating Array Bound Checking through Dependent Types". In: *Proceedings of the ACM SIGPLAN 1998 Conference on Programming Language Design and Implementation*. PLDI '98. New York, NY, USA: ACM, 1998, pp. 249–257. DOI: 10.1145/277650.277732. URL: https://www.cs.cmu.edu/~fp/papers/pldi98dml.pdf.

[37] Mingsheng Ying. "Floyd–Hoare Logic for Quantum Programs". In: *ACM Transactions on Programming Languages and Systems* 33.6, 19 (2012). DOI: 10.1145/2049706.2049708.

[38] Nengkun Yu and Jens Palsberg. "Quantum Abstract Interpretation". In: *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*. PLDI '21. New York, NY, USA: Association for Computing Machinery, 2021, pp. 542–558. DOI: 10.1145/3453483.3454061. URL: http://web.cs.ucla.edu/~palsberg/paper/pldi21-quantum.pdf.

A Full Grammar and Rules

1. Grammar:

$$\mathbf{G} := \mathbf{I} \mid \mathbf{X} \mid \mathbf{Y} \mid \mathbf{Z} \mid \mathbf{cG} \mid \mathbf{G} \otimes \mathbf{G} \mid \mathbf{G} \to \mathbf{G} \mid \mathbf{G} \cap \mathbf{G} \mid \mathbf{G} \cup \mathbf{G} \mid \mathbf{G_S} \mid \mathbf{G} + \mathbf{G}$$

2. Tensor Rules:

$$\frac{\mathbf{T}[i] = \mathbf{A} \qquad U : \mathbf{A} \to \mathbf{B}}{U\ i : \mathbf{T} \to \mathbf{T}\{i \mapsto \mathbf{B}\}}\ \otimes_1 \qquad \frac{\mathbf{T}[i] = \mathbf{A} \qquad \mathbf{T}[j] = \mathbf{B} \qquad U : \mathbf{A} \otimes \mathbf{B} \to \mathbf{C} \otimes \mathbf{D}}{U\ i\ j : \mathbf{T} \to \mathbf{T}\{i \mapsto \mathbf{C}; j \mapsto \mathbf{D}\}}\ \otimes_2$$

$$\frac{g : \mathbf{S} \to \mathbf{T} \qquad \mathbf{T}[i] = c\mathbf{A}}{g : \mathbf{S} \to c\mathbf{T}\{i \mapsto \mathbf{A}\}}\ \otimes_{c1} \qquad \frac{g : \mathbf{S} \to \mathbf{cT} \qquad \mathbf{T}[i] = \mathbf{A}}{g : \mathbf{S} \to \mathbf{T}\{i \mapsto c\mathbf{A}\}}\ \otimes_{c2}$$

$$\frac{g : \mathbf{A} \otimes \mathbf{I} \to \mathbf{C} \otimes \mathbf{D} \qquad g : \mathbf{I} \otimes \mathbf{B} \to \mathbf{E} \otimes \mathbf{F}}{g : \mathbf{A} \otimes \mathbf{B} \to \mathbf{CE} \otimes \mathbf{DF}}\ \otimes\text{-MUL}$$

3. Arrow and Sequence Rules:

$$\frac{g : \mathbf{A} \to \mathbf{A'} \qquad g : \mathbf{B} \to \mathbf{B'}}{g : (\mathbf{AB}) \to (\mathbf{A'B'})}\ \text{MUL} \qquad \frac{g : \mathbf{A} \to \mathbf{A'}}{g : \mathbf{cA} \to \mathbf{cA'}}\ \text{SCALE}$$

$$\frac{g_1 : \mathbf{A} \to \mathbf{B} \qquad g_2 : \mathbf{B} \to \mathbf{C}}{g_1; g_2 : \mathbf{A} \to \mathbf{C}}\ \text{SEQ} \qquad \frac{g_1; (g_2; g_3) : \mathbf{A} \to \mathbf{A'}}{(g_1; g_2); g_3 : \mathbf{A} \to \mathbf{A'}}\ \text{SEQ-ASSOC}$$

4. Intersection Rules:

$$\frac{g : \mathbf{A} \qquad g : \mathbf{B}}{g : \mathbf{A} \cap \mathbf{B}}\ \cap\text{-I} \qquad \frac{g : \mathbf{A} \cap \mathbf{B}}{g : \mathbf{A}}\ \cap\text{-E} \qquad \frac{g : (\mathbf{A} \to \mathbf{A'}) \cap (\mathbf{B} \to \mathbf{B'})}{g : (\mathbf{A} \cap \mathbf{B}) \to (\mathbf{A'} \cap \mathbf{B'})}\ \cap\text{-Arr-Dist}$$

$$\frac{g : \mathbf{A} \to \mathbf{B} \cap \mathbf{C}}{g : \mathbf{A} \to \mathbf{C} \cap \mathbf{B}}\ \cap\text{-comm-r} \qquad \frac{g : \mathbf{A} \to \mathbf{B} \cap (\mathbf{C} \cap \mathbf{D})}{g : \mathbf{A} \to (\mathbf{B} \cap \mathbf{C}) \cap \mathbf{D}}\ \cap\text{-assoc-r}$$

5. Union Rules:

$$\frac{g : \mathbf{A}}{g : \mathbf{A} \cup \mathbf{B}}\ \cup\text{-I} \qquad \frac{g : \mathbf{A} \cup \mathbf{A}}{g : \mathbf{A}}\ \cup\text{-E} \qquad \frac{g : (\mathbf{A} \to \mathbf{A'}) \cap (\mathbf{B} \to \mathbf{B'})}{g : (\mathbf{A} \cup \mathbf{B}) \to (\mathbf{A'} \cup \mathbf{B'})}\ \cup\text{-Arr-Dist}$$

$$\frac{g : \mathbf{A} \to \mathbf{B} \cup \mathbf{C}}{g : \mathbf{A} \to \mathbf{C} \cup \mathbf{B}}\ \cup\text{-comm-r} \qquad \frac{g : \mathbf{A} \to \mathbf{B} \cup (\mathbf{C} \cup \mathbf{D})}{g : \mathbf{A} \to (\mathbf{B} \cup \mathbf{C}) \cup \mathbf{D}}\ \cup\text{-assoc-r}$$

6. Addition Rules for additive types:

$$\frac{g : \mathbf{A} \to \mathbf{B} + \mathbf{0C}}{g : \mathbf{A} \to \mathbf{B}}\ +\text{-E} \qquad \frac{g : \mathbf{A} \to \mathbf{c_1 B} + \mathbf{c_2 B}}{g : \mathbf{A} \to (\mathbf{c_1} + \mathbf{c_2})\mathbf{B}}\ c\text{-ADD}$$

$$\frac{g : \mathbf{A} \to \mathbf{B} \qquad g : \mathbf{C} \to \mathbf{D}}{g : \mathbf{A} + \mathbf{C} \to \mathbf{B} + \mathbf{D}}\ \text{ADD} \qquad \frac{U : \mathbf{A} \to \mathbf{B} + \mathbf{C} \qquad \mathbf{T}[i] = \mathbf{A}}{U\ i : \mathbf{T} \to \mathbf{T}\{i \mapsto \mathbf{B}\} + \mathbf{T}\{i \mapsto \mathbf{C}\}}\ \text{ADD}_2$$

Figure 4: The basic grammar and typing rules for Gottesman and additive types and their connectives. The grammar allows us to describe ill-formed types, such as $\mathbf{X} \cap (\mathbf{I} \otimes \mathbf{Z})$, but these don't type any states or circuits. The intersection and arrow typing rules are derived from standard subtyping rules [24, Chapter 15].

7. Normalization rules for Gottesman types only:

$$\frac{g : \mathbf{A} \to \mathbf{B} \cap \mathbf{C}}{g : \mathbf{A} \to \mathbf{B} \cap \mathbf{BC}} \cap\text{-MUL-R} \qquad \frac{g : \mathbf{A} \cap \mathbf{B} \to \mathbf{C}}{g : \mathbf{A} \cap \mathbf{AB} \to \mathbf{C}} \cap\text{-MUL-L}$$

8. Single-qubit Separability Rules:

$$\frac{g : \mathbf{A} \to \mathbf{I}^{k-1} \otimes \mathbf{B} \otimes \mathbf{I}^{n-k}}{g : \mathbf{A} \to \mathbf{B}_k} \text{SEP1-R} \qquad \frac{g : \mathbf{I}^{k-1} \otimes \mathbf{A} \otimes \mathbf{I}^{n-k} \to \mathbf{B}}{g : \mathbf{A}_k \to \mathbf{B}} \text{SEP1-L}$$

$$\frac{g : \mathbf{A} \to \mathbf{B}_k \cap \mathbf{T} \qquad \mathbf{T}[k] \in \{\mathbf{B}, \mathbf{I}\}}{g : \mathbf{A} \to \mathbf{B}_k \cap \mathbf{T}_{[n]\setminus\{k\}}} \cap\text{-SEP1-R} \quad \frac{g : \mathbf{A}_k \cap \mathbf{T} \to \mathbf{B} \qquad \mathbf{T}[k] \in \{\mathbf{A}, \mathbf{I}\}}{g : \mathbf{A} \cap \mathbf{T}_{[n]\setminus\{k\}} \to \mathbf{B}_k} \cap\text{-SEP1-L}$$

9. Multi-qubit separability rules for Gottesman types when $S = \{j_1, \ldots, j_k\} \subset [n]$:

$$\frac{g : \mathbf{A} \to \mathbf{B} \cap \mathbf{T}_{(1)} \cap \ldots \cap \mathbf{T}_{(k)} \qquad \forall_{j\in[k]}\ \mathbf{T}_{(j)}[S] = \mathbf{C}_{(j)} \qquad \forall_{j\in[k]}\ \mathbf{T}_{(j)}[\overline{S}] = \mathbf{I}^{n-k} \qquad \mathbf{B}[S] = \mathbf{I}^k}{g : \mathbf{A} \to \mathbf{B}_{\overline{S}} \cap \left(\mathbf{C}_{(1)} \cap \ldots \cap \mathbf{C}_{(k)}\right)_S} \text{SEP-R}$$

$$\frac{g : \mathbf{A} \cap \mathbf{T}_{(1)} \cap \ldots \cap \mathbf{T}_{(k)} \to \mathbf{B} \qquad \forall_{j\in[k]}\ \mathbf{T}_{(j)}[S] = \mathbf{C}_{(j)} \qquad \forall_{j\in[k]}\ \mathbf{T}_{(j)}[\overline{S}] = \mathbf{I}^{n-k} \qquad \mathbf{A}[S] = \mathbf{I}^k}{g : \mathbf{A}_{\overline{S}} \cap \left(\mathbf{C}_{(1)} \cap \ldots \cap \mathbf{C}_{(k)}\right)_S \to B} \text{SEP-L}$$

Figure 5: (Continued) Additional typing rules for Gottesman types. These cover our applications for normalization and separability judgements. Let $[n] = \{1, \ldots, n\}$ and $S \subset [n]$. The conditions that $\mathbf{C}_{(1)}, \ldots, \mathbf{C}_{(k)}$ need to satisfy to achieve multi-qubit separability are described in §4.2.

B Transitivity of Clifford groups

Recall that a group G acting on a set Ω is *transitive* if for any $x, y \in \Omega$ there exists a $g \in G$ with $g \cdot x = y$. Since Clifford operators act on Pauli operators by conjugation, the Clifford group can never be transitive as $C \cdot I = CIC^\dagger = I$. However, for nontrivial Paulis it is.

Proposition 39. *Let $P, Q \in \mathcal{P}_n \setminus \{\pm I\}$. Then there exists a $C \in C\ell_n$ such that $CPC^\dagger = Q$.*

More generally, a group is m-transitive if given tuples $(x_1, \ldots, x_m), (y_1, \ldots, y_m) \in \Omega^m$ with each $x_i \neq x_j$ and $y_i \neq y_j$, then there exists a $g \in G$ with $g \cdot x_i = y_i$ for $i = 1, \ldots, m$. Again, since the Clifford group acts by conjugation $C \cdot (-P) = -CPC^\dagger = -C \cdot P$ and so the Clifford group cannot be even 2-transitive. However we modify the definition to require our Pauli elements be distinct up to sign, then we do obtain a higher transitivity result in the one-qubit, which follows from simply counting the number of one-qubit Clifford operators.

Lemma 40. *Given $P_1, P_2, Q_1, Q_2 \in \mathcal{P}_1 \setminus \{\pm I\}$ with $P_1 \neq \pm P_2$ and $Q_1 \neq \pm Q_2$, then there exists a $C \in C\ell_1$ with $CP_1C^\dagger = Q_1$ and $CP_2C^\dagger = Q_2$.*

Note that from the conditions in the lemma above, we must have P_1 and P_2 (and respectively Q_1 and Q_2) anticommute. But for higher qubit Paulis, this is not the case: even if $P_1 \neq \pm P_2$ we could have P_1 and P_2 commute. Since conjugation preserves commutivity, again the Clifford group cannot be 2-transitive. However it is on pairs of commuting/anticommuting Paulis.

Theorem 41. *Given $P_1, P_2, Q_1, Q_2 \in \mathcal{P}_n \setminus \{\pm I\}$ with $P_1 \neq \pm P_2$ and $Q_1 \neq \pm Q_2$ and either both P_1, P_2 and Q_1, Q_2 commute or both anticommute. Then then there exists a $C \in C\ell_n$ with $CP_1C^\dagger = Q_1$ and $CP_2C^\dagger = Q_2$.*

The proof of this theorem follows from the 2-qubit case (much like building a general Clifford operator out of CNOT and one-qubit Cliffords). For two commuting 2-qubit Cliffords P, Q, using the lemma above (and CNOT if necessary) one can easily produce a C with $CPC^\dagger = \sigma_y \otimes \sigma_y$ and $CQC^\dagger = \sigma_z \otimes \sigma_z$. Similarly, for two anticommuting 2-qubit Cliffords P, Q, one gets a C with $CPC^\dagger = I \otimes \sigma_y$ and $CQC^\dagger = I \otimes \sigma_z$. Then the theorem follows from chaining each of P_1, Q_1 and P_2, Q_2 through the appropiate normal form.