Static Analysis of Quantum Programs via Gottesman Types

ROBERT RAND, University of Chicago, USA
AARTHI SUNDARAM, Microsoft Quantum, USA
KARTIK SINGHAL, University of Chicago, USA
BRAD LACKEY, Microsoft Quantum, USA and University of Maryland, USA

The Heisenberg representation of quantum operators provides a powerful technique for reasoning about quantum circuits, albeit those restricted to the common (non-universal) Clifford set H, S and $CNOT$. The Gottesman-Knill theorem showed that we can use this representation to efficiently simulate Clifford circuits. We show that Gottesman's semantics for quantum programs can be treated as a type system, allowing us to efficiently characterize a common subset of quantum programs. We apply this primarily towards tracking entanglement in programs, showing how superdense coding and GHZ circuits entangle and disentangle qubits and how to safely dispose of ancillae. We demonstrate the efficiency of our typechecking algorithm both for simple deductions and those involving entanglement and measurement.

1 INTRODUCTION

The *Heisenberg representation* of quantum mechanics treats quantum operators as functions on operators, rather than on the quantum state. For instance, for any quantum state $|\phi\rangle$,

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

so an H operator can be viewed as a function that takes Z to X (and similarly takes X to Z). Gottesman [1998] uses this representation to present the rules for how the Clifford set H, S and $CNOT$ operates on Pauli X and Z matrices, which is sufficient to fully describe the behavior of the Clifford operators. There, H is given the following description based on its action above:

$$H : \mathbf{Z} \rightarrow \mathbf{X} \qquad H : \mathbf{X} \rightarrow \mathbf{Z}$$

In Gottesman's paper, the end goal is to fully describe quantum programs and prove the *Gottesman-Knill theorem*, which shows that any Clifford circuit can be simulated efficiently. Here we observe that the judgments above look like typing judgments, and show that they can indeed be treated as such (§3). As such, they can be used to make coarse guarantees about programs, without fully describing the programs' behaviors. In particular, we show how this system can be used to track separability in §4. We apply this to Deutsch's algorithm to show its capability to verify if an auxiliary qubit has been safely discarded as it is not entangled with the rest of the system (§4.2). We further show that our type system can guarantee that the superdense coding algorithm successfully transmits two separable computational basis qubits (§4.4). In §5.2, we demonstrate, using the GHZ state $|000\rangle + |111\rangle$, how the type system is capable of tracking both the creation and destruction of entanglement. We extend the type system in §7 to deal with programs outside the Clifford group and use it to characterize the Toffoli gate. We discuss the possible future applications of this system in §9.

A previous version of this paper was presented at QPL 2020 [Rand et al. 2020]. This version includes an expanded treatment of multi-qubit separability judgments, measurement and normal forms. The system and examples in this paper (excluding normalization and measurement) are formalized in Coq at https://github.com/inQWIRE/GottesmanTypes.

Authors' addresses: Robert Rand, University of Chicago, USA, rand@uchicago.edu; Aarthi Sundaram, Microsoft Quantum, Seattle, USA, aarthi.sundaram@microsoft.com; Kartik Singhal, University of Chicago, USA, ks@cs.uchicago.edu; Brad Lackey, Microsoft Quantum, Seattle, USA, University of Maryland, USA, brad.lackey@microsoft.com.

2 THE HEISENBERG REPRESENTATION

The Heisenberg representation of H, S, and $CNOT$ are given by the following table.

$$\begin{array}{rlrl} H: & \mathbf{X} \rightarrow \mathbf{Z} & CNOT: & \mathbf{X} \otimes \mathbf{I} \rightarrow \mathbf{X} \otimes \mathbf{X} \\ H: & \mathbf{Z} \rightarrow \mathbf{X} & CNOT: & \mathbf{I} \otimes \mathbf{X} \rightarrow \mathbf{I} \otimes \mathbf{X} \\ S: & \mathbf{X} \rightarrow \mathbf{Y} & CNOT: & \mathbf{Z} \otimes \mathbf{I} \rightarrow \mathbf{Z} \otimes \mathbf{I} \\ S: & \mathbf{Z} \rightarrow \mathbf{Z} & CNOT: & \mathbf{I} \otimes \mathbf{Z} \rightarrow \mathbf{Z} \otimes \mathbf{Z} \end{array}$$

Note that these rules are intended to simply describe the action of each operator on the corresponding unitary matrices, as in equation 1. For our purposes, though, we will treat Clifford operators as terms in a programming language and the above descriptions as types, justifying that choice in §3.

We can combine our typing rules by simple multiplication, for instance combining the second and third rules for $CNOT$, we get

$$\begin{aligned} CNOT : (\mathbf{I} \otimes \mathbf{X})(\mathbf{Z} \otimes \mathbf{I}) &\rightarrow (\mathbf{I} \otimes \mathbf{X})(\mathbf{Z} \otimes \mathbf{I}) \\ = \mathbf{Z} \otimes \mathbf{X} &\rightarrow \mathbf{Z} \otimes \mathbf{X} \end{aligned}$$

In the rule for S, $\mathbf{Y}$ is equivalent to $i\mathbf{XZ}$ so we can reason about an S applied twice compositionally. We use $f \ \$ \ g$ for forward function composition (equivalent to $g \circ f$):

$$S; S : (\mathbf{X} \rightarrow i\mathbf{XZ} \ \$ \ i\mathbf{XZ} \rightarrow i\mathbf{YZ}) = \mathbf{X} \rightarrow -\mathbf{X}$$

Once we have a type system for H, S and $CNOT$ we can define additional gates in terms of these, and derive their types. For instance, the Pauli Z gate is simply $S; S$, for which we previously derived the type for $\mathbf{X}$ and trivially can derive the type for $\mathbf{Z}$:

$$Z : (\mathbf{Z} \rightarrow \mathbf{Z} \ \$ \ \mathbf{Z} \rightarrow \mathbf{Z}) = \mathbf{Z} \rightarrow \mathbf{Z}$$

Defining X as $H; Z; H$, we can derive the type for the Pauli X gate as:

$$X = H; Z; H : (\mathbf{X} \rightarrow \mathbf{Z} \ \$ \ \mathbf{Z} \rightarrow \mathbf{Z} \ \$ \ \mathbf{Z} \rightarrow \mathbf{X}) = \mathbf{X} \rightarrow \mathbf{X}$$
$$X = H; Z; H : (\mathbf{Z} \rightarrow \mathbf{X} \ \$ \ \mathbf{X} \rightarrow -\mathbf{X} \ \$ \ -\mathbf{X} \rightarrow -\mathbf{Z}) = \mathbf{Z} \rightarrow -\mathbf{Z}$$

Likewise, $Y = S; X; Z; S$ and so, the type for the Pauli Y gate would be:

$$Y = S; Z; X; S : (\mathbf{Z} \rightarrow \mathbf{Z} \ \$ \ \mathbf{Z} \rightarrow \mathbf{Z} \ \$ \ \mathbf{Z} \rightarrow -\mathbf{Z} \ \$ \ -\mathbf{Z} \rightarrow -\mathbf{Z}) = \mathbf{Z} \rightarrow -\mathbf{Z}$$
$$Y = S; Z; X; S : (\mathbf{X} \rightarrow \mathbf{Y} \ \$ \ \mathbf{Y} \rightarrow -\mathbf{Y} \ \$ \ -\mathbf{Y} \rightarrow \mathbf{Y} \ \$ \ \mathbf{Y} \rightarrow -\mathbf{X}) = \mathbf{X} \rightarrow -\mathbf{X}$$

We can also define more complicated gates like CZ and $SWAP$ as $H_2; CNOT; H_2$ (where H_2 is H applied to $CNOT$'s target qubit) and $CNOT; NOTC; CNOT$, (where $NOTC$ is a flipped $CNOT$) for which we can easily derive the following types:

$$\begin{array}{rlrl} CZ: & \mathbf{X} \otimes \mathbf{I} \rightarrow \mathbf{X} \otimes \mathbf{Z} & SWAP: & \mathbf{X} \otimes \mathbf{I} \rightarrow \mathbf{I} \otimes \mathbf{X} \\ CZ: & \mathbf{I} \otimes \mathbf{X} \rightarrow \mathbf{Z} \otimes \mathbf{X} & SWAP: & \mathbf{I} \otimes \mathbf{X} \rightarrow \mathbf{X} \otimes \mathbf{I} \\ CZ: & \mathbf{Z} \otimes \mathbf{I} \rightarrow \mathbf{Z} \otimes \mathbf{I} & SWAP: & \mathbf{Z} \otimes \mathbf{I} \rightarrow \mathbf{I} \otimes \mathbf{Z} \\ CZ: & \mathbf{I} \otimes \mathbf{Z} \rightarrow \mathbf{I} \otimes \mathbf{Z} & SWAP: & \mathbf{I} \otimes \mathbf{Z} \rightarrow \mathbf{Z} \otimes \mathbf{I} \end{array}$$

By combining the rules stated above, we can also derive the action of $SWAP$ on $\mathbf{X} \otimes \mathbf{Y}$ as:

$$\begin{aligned} SWAP : \ (\mathbf{X} \otimes \mathbf{I})(\mathbf{I} \otimes i\mathbf{X})(\mathbf{I} \otimes \mathbf{Z}) &\rightarrow (\mathbf{I} \otimes \mathbf{X})(i\mathbf{X} \otimes \mathbf{I})(\mathbf{Z} \otimes \mathbf{I}) \\ = \mathbf{X} \otimes \mathbf{Y} &\rightarrow \mathbf{Y} \otimes \mathbf{X} \end{aligned}$$

This gives us less information than the types for $SWAP$ above, but might be the type we intend for a given use of swapping. In particular, we might want to use a $SWAP$ to exchange qubits in the X and Y bases, as we will show.

We show the full rules for typing circuits in Figure 1, which we will reference throughout this paper. Unless otherwise specified, we will denote $AB := A * B$ and $A^m := A \otimes A \otimes \ldots$ (m times) for expressions A and B.

3 INTERPRETATION ON BASIS STATES

We can interpret the type $H : \mathbf{Z} \to \mathbf{X}$ as saying that H takes a qubit in the $\mathbf{Z}$ basis (that is, $|0\rangle$ or $|1\rangle$) to a qubit in the $\mathbf{X}$ basis ($|+\rangle$ and $|-\rangle$). This form of reasoning is present in Perdrix's [2008] work on abstract interpretation for quantum systems, which classifies qubits in an $\mathbf{X}$ or $\mathbf{Z}$ basis, for the purpose of tracking entanglement. Unfortunately, that system cannot leave the $\mathbf{X}$ and $\mathbf{Z}$ bases, and hence cannot derive that $Z : \mathbf{X} \to -\mathbf{X}$ due to the intermediate $\mathbf{Y}$. It also cannot handle potentially entangling gates like $CNOT$ applied to $\mathbf{X} \otimes \mathbf{Z}$, which Perdrix classifies as simply $\top$ (top) and marks as potentially entangled. By contrast, our system is closed under the application of Clifford operators, and $\mathbf{X} \otimes \mathbf{Z}$ has a broader interpretation: it classifies the eigenstates of $X \otimes Z$. Hence, Perdrix's system typically classifies most circuits as $\top$ after just a few gate applications, while ours can be used to characterize any Clifford circuit.

Proposition 3.1. *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 [1998], 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. □

This proposition admits two interpretations of the judgment $H : \mathbf{Z} \to \mathbf{X}$, depending on whether we care that the eigenvalue is preserved:

(1) H takes the +1 eigenstate of $\mathbf{Z}$ ($|0\rangle$ up to a global phase) to the +1 eigenstate of $\mathbf{X}$ ($|+\rangle$ up to a phase).
(2) H takes every eigenstate of $\mathbf{Z}$ (including $|0\rangle$ and $|1\rangle$) to an eigenstate of $\mathbf{X}$ ($|+\rangle$ and $|-\rangle$).

An illustration of the differences between these interpretations is in the types of Pauli operators, each of which take $\mathbf{X}$ to either $\mathbf{X}$ or $-\mathbf{X}$ (and similarly for $\mathbf{Z}$). According to the first interpretation $X : \mathbf{Z} \to -\mathbf{Z}$ means that the X operator negates the computational basis qubit. In the second interpretation, we can treat X (and Y and Z) as type-preserving operations since $\mathbf{Z}$ and $-\mathbf{Z}$ have the same set of eigenvectors. This allows us to ignore Pauli gates when type-checking circuits.

In this section, we carry around signs since the cost of doing so is low and it allows for a more precise interpretation of typing judgments. In fact, given the action of a circuit on every permutation of X and Z (or any spanning set) we can deduce the semantics of the program itself [Gottesman 1998]. The intuition behind this comes from the fact that this set of X and Z operators form the generators of the Pauli basis for quantum states. Hence, deducing the action of a unitary matrix on them provides an information theoretically complete picture of the action of the matrix on any input and suffices to deduce the semantics of the program. However, for rest of this paper we will follow the second interpretation and leave the first to future work, see §9.

(1) Grammar:

$$G := \mathbf{I} \mid \mathbf{X} \mid \mathbf{Z} \mid -G \mid iG \mid G * G \mid G \otimes G \mid G \rightarrow G \mid G \cap G \mid \boxdot$$

(2) Multiplication and Tensor Laws:

$$\mathbf{X} * \mathbf{X} = \mathbf{I} \qquad \mathbf{Z} * \mathbf{Z} = \mathbf{I} \qquad \mathbf{Z} * \mathbf{X} = -\mathbf{X} * \mathbf{Z} \qquad A * \mathbf{I} = A = \mathbf{I} * A$$

$$- - A = A \qquad i(iA) = -A \qquad i(-A) = -(iA) \qquad A * (B * C) = A * B * C$$

$$-A * B = -(A * B) = A * -B \qquad iA * B = i(A * B) = A * iB$$

$$A \otimes (B \otimes C) = (A \otimes B) \otimes C \qquad A \otimes B = (A \otimes \mathbf{I}) * (\mathbf{I} \otimes B)$$

$$iA \otimes B = i(A \otimes B) = A \otimes iB \qquad -A \otimes B = -(A \otimes B) = A \otimes -B$$

$$(A \otimes B) * (C \otimes D) = (A * C) \otimes (B * D) \text{ where } |A| = |C|$$

(3) Tensors Rules:

$$\frac{g\ 1 : A \rightarrow A' \qquad |E| = m - 1}{g\ m : E \otimes A \otimes E' \rightarrow E \otimes A' \otimes E'}\ \otimes_1 \qquad \frac{g\ 2\ 1 : A \otimes B \rightarrow A' \otimes B'}{g\ 1\ 2 : B \otimes A \rightarrow B' \otimes A'}\ \otimes\text{-REV}$$

$$\frac{g\ 1\ 2 : A \otimes B \rightarrow A' \otimes B' \qquad |E| = m - 1 \qquad |E'| = n - m - 1}{g\ m\ n : E \otimes A \otimes E' \otimes B \otimes E'' \rightarrow E \otimes A' \otimes E' \otimes B' \otimes E''}\ \otimes_2$$

(4) Arrow and Sequence Rules:

$$\frac{p : A \rightarrow A' \qquad p : B \rightarrow B'}{p : (A * B) \rightarrow (A' * B')}\ \text{MUL} \qquad \frac{p : A \rightarrow A'}{p : iA \rightarrow iA'}\ \text{IM}$$

$$\frac{p_1 : A \rightarrow B \qquad p_2 : B \rightarrow C}{p_1; p_2 : A \rightarrow C}\ \text{CUT} \qquad \frac{p : A \rightarrow A'}{p : -A \rightarrow -A'}\ \text{NEG}$$

$$p_1; (p_2; p_3) : A \equiv (p_1; p_2); p_3 : A$$

(5) Intersection Rules:

$$\mathbf{I}^n \cap A = A \qquad A \cap A = A \qquad A \cap B = B \cap A \qquad A \cap B \cap C = A \cap (B \cap C)$$

$$\frac{g : A \qquad g : B}{g : A \cap B}\ \cap\text{-I} \qquad \frac{g : A \cap B}{g : A}\ \cap\text{-E}$$

$$\frac{g : (A \rightarrow B) \cap (A \rightarrow C)}{g : A \rightarrow (B \cap C)}\ \cap\text{-Arr} \qquad \frac{g : (A \rightarrow A') \cap (B \rightarrow B')}{g : (A \cap B) \rightarrow (A' \cap B')}\ \cap\text{-Arr-Dist}$$

(6) Separability Rules (1-qubit case):

$$(\mathbf{I}^m \otimes A \otimes \mathbf{I}^n) = \boxdot^m \otimes A \otimes \boxdot^n \text{ where } A \in \{1, -1, i, -i\} * \{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$$

$$(\boxdot^{|B|} \otimes A \otimes \boxdot^{|C|}) \cap (B \otimes A \otimes C) = (\boxdot^{|B|} \otimes A \otimes \boxdot^{|C|}) \cap (B \otimes \boxdot^{|A|} \otimes C)$$

$$(\boxdot^{|B|} \otimes A \otimes \boxdot^{|C|}) \cap (B \otimes I^{|A|} \otimes C) = (\boxdot^{|B|} \otimes A \otimes \boxdot^{|C|}) \cap (B \otimes \boxdot^{|A|} \otimes C)$$

Fig. 1. The grammar and typing rules for Gottesman types. 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 circuits. The intersection and arrow typing rules are derived from standard subtyping rules [Pierce 2002, Chapter 15]. The arity of a type not containing $\{\times, \rightarrow\}$ is the longest sequence of atoms connected by tensors (which are implicit in superscripts). For instance, $|\mathbf{I} * \mathbf{Z} \otimes i\mathbf{X}| = |(\mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{Z} \otimes \mathbf{Z})| = \left|\mathbf{I}^2\right| = 2$. The general separability rules are described in §4.

Intersection Types. It may seem odd that we have given multiple types to H, S, $CNOT$ and the various derived operators. This isn't particularly rare in type systems with subtyping, which is an appropriate lens for viewing the types we have given above (for instance $\mathbf{X}$ is a subtype of $\mathbf{I}$). However, it is useful to have the most descriptive type for any term in our language. We can give these using intersection types. For instance, we have the following types for H and $CNOT$:

$$\begin{aligned} H &: (\mathbf{X} \rightarrow \mathbf{Z}) \cap (\mathbf{Z} \rightarrow \mathbf{X}) \\ CNOT &: (\mathbf{X} \otimes \mathbf{I} \rightarrow \mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{I} \otimes \mathbf{X} \rightarrow \mathbf{I} \otimes \mathbf{X}) \cap (\mathbf{Z} \otimes \mathbf{I} \rightarrow \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{I} \otimes \mathbf{Z} \rightarrow \mathbf{Z} \otimes \mathbf{Z}) \end{aligned}$$

From these we can derive any of the types given earlier, using the standard rules for intersections (Figure 1). Another advantage of using intersection types is that it is closely related to the stabilizer formalism that is used extensively in error-correction. This connection is further discussed in §9.

The role of I. $\mathbf{I}$ plays an interesting role in this type system. For $\mathbf{I}$ alone, we have the following two facts, the first drawn from the Heisenberg representation of quantum mechanics and the second from our interpretation of types as eigenstates:

$$\forall\, U, \qquad U : \mathbf{I}^n \rightarrow \mathbf{I}^n \qquad \text{where } U \text{ has dimensions } n \times n \tag{2}$$

$$\forall\, |\psi\rangle, \qquad |\psi\rangle : \mathbf{I}^n \qquad \text{where } |\psi\rangle \text{ has length } n \tag{3}$$

This would lead us to treat $\mathbf{I}$ as a kind of top type, where $A <: \mathbf{I}$ for any type A. However, this would be incompatible with $\otimes$. For example, the two qubit Bell pair $|\Phi^+\rangle$ has type $\mathbf{X} \otimes \mathbf{X}$ but not type $\mathbf{X} \otimes \mathbf{I}$. By contrast, $\mathbf{X} \otimes \mathbf{I}$ contains precisely the separable two qubit states where the first qubit is an eigenstate of $\mathbf{X}$. This second type, which is neither a subtype nor supertype of the first allows us to consider the important property of *separability* or non-entanglement of qubits.

4 SEPARABILITY

4.1 Single qubit separability

Proposition 4.1. *For any Pauli matrix $U \in \{-1, 1, -i, i\} * \{X, Y, Z\}$, the eigenstates of $U \otimes I^{n-1}$ are all the vectors $|u\rangle \otimes |\psi\rangle$ where $|u\rangle$ is an eigenstate of U and $|\psi\rangle$ is any state.*

Proof. Let $|\phi\rangle$ be the λ eigenstate and $|\phi^\perp\rangle$ be the $-\lambda$ eigenstate for $U \in \{1, -1, i, -i\} * \{X, Y, Z\}$ where $\lambda \in \{1, i\}$. Note that $\{|\phi\rangle, |\phi^\perp\rangle\}$ forms a single-qubit basis.

First, consider states of the form $|\gamma\rangle = |u\rangle \otimes |\psi\rangle$ where $|u\rangle \in \{|\phi\rangle, |\phi^\perp\rangle\}$ 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 $|v\rangle \notin \{|\phi\rangle, |\phi^\perp\rangle\}$, $|\gamma\rangle$ 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

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

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

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

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

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

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

Following Gottesman's notation, let $U_k^{\mathbf{I}} := I^{(k-1)} \otimes U \otimes I^{(n-k)}$ for $1 \leq k \leq n$ and a single qubit Pauli U. (Gottesman writes $\overline{U}_k$; we include the $\mathbf{I}$ to distinguish this from a $U_i^{\boxdot}$ notation to follow.) Combining Propositions 3.1 and 4.1, we obtain the following corollary:

Corollary 4.2. *Every term of type $U_i^{\mathbf{I}}$ is separable, for $U \in \{\pm X, \pm Y, \pm Z\}$. That is, the factor associated with U is unentangled with the rest of the system.*

This allows us to introduce a new type indicating separability, which we write $\boxdot$. The type $\mathbf{X} \otimes \boxdot$ describes the set of two separable qubits where the first qubit is in the $\mathbf{X}$ eigenstate. Similarly, $\boxdot \otimes \boxdot \otimes \mathbf{Z}$ describes a three qubit system in which the third qubit is in $\{|0\rangle, |1\rangle\}$ and separable from the rest of the system. Drawing on our $U_i^{\mathbf{I}}$ notation above, we use $\mathbf{X}_1^{\boxdot}$ and $\mathbf{Z}_3^{\boxdot}$ to describe the two types above, assuming the arity of the system is known. Additionally, for highly separable systems we will write $A \times B \times \cdots \times E$ for $A_1^{\boxdot} \cap B_2^{\boxdot} \cap \cdots \cap E_n^{\boxdot}$.

Per Corollary 4.2, we can freely interchange between $U_k^{\mathbf{I}}$ and $U_k^{\boxdot}$, where U is a single-qubit Pauli matrix. Separability judgments also distribute across intersections. For instance $\mathbf{X}_1^{\boxdot} \cap (\mathbf{X} \otimes \mathbf{Z} \otimes \mathbf{Z})$ is equal to $\mathbf{X}_1^{\boxdot} \cap (\boxdot \otimes \mathbf{Z} \otimes \mathbf{Z})$ or $(\mathbf{X}_1^{\boxdot} \cap (\mathbf{Z} \otimes \mathbf{Z})_{2,3}^{\boxdot})$.

This allows us to derive a new type from $CNOT$, $\mathbf{Z} \times \mathbf{X} \to \mathbf{Z} \times \mathbf{X}$. Our $\cap$-Arr-Dist rule (Figure 1), which follows directly from the subtyping rules for arrow and intersection[1], allows us to weaken $CNOT$'s type to $(\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{X}) \to (\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{X})$. Both appearances of $\mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{X}$ can be converted to $\mathbf{Z} \times \mathbf{X}$ (i.e. $\mathbf{Z}_1^{\boxdot} \cap \mathbf{X}_2^{\boxdot}$) which specifies that the qubits are separable, indicating that a $CNOT$ applied to a $\mathbf{Z}$ and $\mathbf{X}$ preserves separability.

4.2 Example: Discarding Qubits in Deutsch's Algorithm

Many quantum circuits introduce ancillary qubits that are used to perform some classical computation and are then discarded in a basis state. Several efforts have been made to verify this behaviour: The Quipper [Green et al. 2013] and Q# [Svore et al. 2018] languages allow us to *assert* that ancilla are separable and can be safely discarded, while $\mathcal{Q}$wire allows us to manually verify this [Rand et al. 2018]. More recently, Silq [Bichsel et al. 2020] 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 in the state $\mathbf{Z}_i^{\boxdot}$ for some i.

A simple example to demonstrate this ability to verify safe discard of auxiliary qubits is Deutsch's algorithm [Deutsch 1985]. 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.

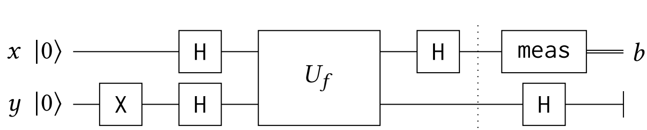

Fig. 2. Deutsch's algorithm to check if $f : \{0,1\} \to \{0,1\}$ is constant or balanced

[1]Thanks to Andreas Rossberg for pointing this out on Stack Overflow.

We want to show that the qubit y is never entangled with qubit x despite of the application of the oracle $U_f : |x\rangle\,|y\rangle \rightarrow |x\rangle\,|y \oplus f(x)\rangle$. In this case, it would be safe to discard qubit y just after the dotted line in Figure 2 (i.e., even before measurement destroys 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:

$$\mathtt{U_f\ 1\ 2} = \begin{cases} \mathtt{I\ 2} & \text{if } f(x) = 0 \\ \mathtt{X\ 2} & \text{if } f(x) = 1 \\ \mathtt{CNOT\ 1\ 2} & \text{if } f(x) = x \\ \mathtt{X\ 1;\ CNOT\ 1\ 2;\ X\ 1} & \text{if } f(x) = 1 - x. \end{cases} \tag{4}$$

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 when $\mathtt{U_f\ 1\ 2} \equiv \mathtt{CNOT\ 1\ 2}$. The input type for this circuit is two qubits initialized in the computational basis, or equivalently $\mathbf{Z} \times \mathbf{Z}$. Hence, we need to derive the output types for $\mathbf{Z}_1^{\boxdot}, \mathbf{Z}_2^{\boxdot}$ which we'll immediately convert to $\mathbf{Z}_1^{\mathbf{I}}, \mathbf{Z}_2^{\mathbf{I}}$ for ease of analysis. We look at the derivation for `deutsch` below, writing the gates on the left and the intermediate types on the right (with comments on the far right).

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

Using the rules for $\rightarrow$, $\cap$ and separability, we obtain:

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathtt{deutsch} : (\mathbf{Z}_2^{\mathbf{I}} \rightarrow \mathbf{X}_2^{\mathbf{I}}) \cap (\mathbf{Z}_1^{\mathbf{I}} \rightarrow \mathbf{Z} \otimes \mathbf{X})}{\mathtt{deutsch} : \mathbf{Z}_1^{\mathbf{I}} \cap \mathbf{Z}_2^{\mathbf{I}} \rightarrow (\mathbf{Z} \otimes \mathbf{X} \cap \mathbf{I} \otimes \mathbf{X})}}{\mathtt{deutsch} : \mathbf{Z}_1^{\mathbf{I}} \cap \mathbf{Z}_2^{\mathbf{I}} \rightarrow (\mathbf{Z} \otimes \mathbf{X} \cap \boxdot \otimes \mathbf{X})}}{\mathtt{deutsch} : \mathbf{Z}_1^{\mathbf{I}} \cap \mathbf{Z}_2^{\mathbf{I}} \rightarrow (\mathbf{Z} \otimes \boxdot \cap \boxdot \otimes \mathbf{X})}}{\mathtt{deutsch} : \mathbf{Z}_1^{\boxdot} \cap \mathbf{Z}_2^{\boxdot} \rightarrow (\mathbf{Z}_1^{\boxdot} \cap \mathbf{X}_2^{\boxdot})}$$

Hence, qubits x and y are separable even before measurement, and therefore y can be freely discarded before measuring x. It is common practice to "uncompute" auxiliary qubits to a computational basis state before discarding, hence the appearance of the final `H 2` which changes the output type to $\mathbf{Z}_1^{\boxdot} \cap \mathbf{Z}_2^{\boxdot}$. 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-Josza algorithm [Deutsch and Jozsa 1992] 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.

4.3 Multi-qubit separability

While Corollary 4.2 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 $S \subset [n]$ being separable from $[n] \setminus S$. Generalizing Proposition 4.1 will help us in this regard. The following fact about Pauli matrices, adapted from Nielsen and Chuang [2010, Prop. 10.5] by setting $n \leftarrow k, k \leftarrow 0$, will be useful for the proof.

Fact 1. *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 $U_{(1)}, \dots U_{(k)}$, the joint eigenspace for an eigenvalue tuple $(\lambda_1, \dots, \lambda_k)$ 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 $U_{(1)}, \dots, U_{(k)}$.

Fact 1 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 algorithm. For each pair of matrices $A = A_1 \otimes \dots \otimes A_k$, $B = B_1 \otimes \dots \otimes B_k$, the 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 $\{1, \dots, k\}$ where X, Z and XZ do not correspond in both matrices.

Proposition 4.3. *For independent, commutative, non-identity k-qubit matrices $U_{(1)}, \dots U_{(k)} \in \{\pm I, \pm X, \pm Y, \pm Z\}^{\otimes k}$ 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 bi-partition as

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

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

$$\begin{aligned}
\forall i \in \{1, \dots, k\} \quad (I \otimes U_{(i)}) \left|\delta\right\rangle &= \sum_j \alpha_j (I \left|\phi_j\right\rangle) \otimes (U_{(i)} \left|\gamma_j\right\rangle) \\
&= \lambda_i \sum_j \alpha_j \left|\phi_j\right\rangle \otimes \left|\gamma_j\right\rangle \\
&= \sum_j \alpha_j \left|\phi_j\right\rangle \otimes (\lambda_i \left|\gamma_j\right\rangle) \\
&\Rightarrow \forall i, j, \quad U_{(i)} \left|\gamma_j\right\rangle = \lambda_i \left|\gamma_j\right\rangle. \\
&\Rightarrow \forall i, j, \quad \lambda_i U_{(i)} \left|\gamma_j\right\rangle = \left|\gamma_j\right\rangle \quad \text{Since, } \lambda_i \in \{-1, 1\}. \qquad (5)
\end{aligned}$$

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 $\{\pm I, \pm X, \pm Y, \pm Z\}^{\otimes k}$. Then, from Fact 1, the joint eigenspace for the all-1s tuple has dimension 1. Specifically, there exists only a single $|\gamma\rangle$ that satisfies Equation (5). Hence, $K = 1$ contradicting the assumption that $|\delta\rangle$ is entangled across the $(n-k, k)$ qubit bi-partition. $\square$

Extending the short hand $U_i^{\mathbf{I}}$ to the multi-qubit setting where matrices have at least one non-trivial Pauli matrix present on qubits in the set $K \subset \{1, \dots, n\}$ where $0 < |K| < n$, we define $U_K^{\mathbf{I}}$ as:

$$U_K^{\mathbf{I}} := \begin{cases} U_i \in \{\pm I, \pm X, \pm Y, \pm Z\} & \text{if } i \in K \\ I & \text{otherwise.} \end{cases}$$

The $U_K^{\boxdot}$ notation follows accordingly. The idea to gather the non-trivial factors within a sub-system is not unique to our work and has been previously employed by Honda to determine the entangled components in his type system [Honda 2015].

Combining Propositions 3.1 and 4.3 we obtain the following corollary:

Corollary 4.4. *Let $K \subset \{1, \dots, n\}$ with $|K| = k$ and $\overline{K} := \{1, \dots, n\} \setminus K$. Every intersection type of the form $U_{K,(1)}^{\mathbf{I}} \cap \dots \cap U_{K,(k)}^{\mathbf{I}}$ where each of the k terms is independent and pair-wise commutes is separable across the bi-partition $(K, \overline{K})$. That is, the factors in K are separable from the $\overline{K}$ sub-system.*

The separability rules in Figure 1 can be extended to the multi-qubit separability scenario as follows. For $K \subset \{1, \dots, n\}$ with $|K| = k$ and $\overline{K} = \{1, \dots, n\} \setminus K$:

$$\bigcap_{i=1}^{k} U_{K,(i)}^{\mathbf{I}} = \bigcap_{i=1}^{k} U_{K,(i)}^{\boxdot} \text{ where the } U_{K,(i)}^{\mathbf{I}}\text{s are independent and pair-wise commuting,} \tag{6}$$

$$A_K^{\boxdot} \cap (A_K \otimes B_{\overline{K}}) = A_K^{\boxdot} \cap B_{\overline{K}}^{\boxdot},$$

$$A_K^{\boxdot} \cap (I_K \otimes B_{\overline{K}}) = A_K^{\boxdot} \cap B_{\overline{K}}^{\boxdot},$$

where A_K collects all the qubits in K and $B_{\overline{K}}$ collects all the qubits not in K.

4.4 Example: Superdense Coding

To illustrate the power of this simple system, consider the example of superdense coding as in Figure 3. Superdense coding allows Alice to convey two bits of information x and y, which we treat as qubits in the $\mathbf{Z}$ state, to Bob by sending a single qubit and consuming one EPR pair.

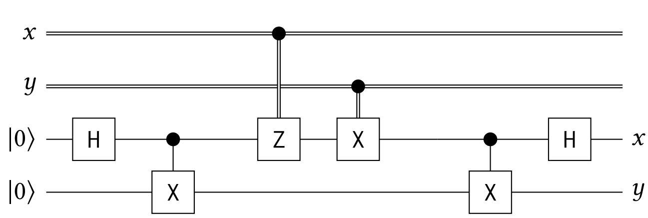

Fig. 3. Superdense Coding sending classical bits x and y from Alice to Bob.

The desired result type for this circuit is $\mathbf{Z} \times \mathbf{Z} \times \mathbf{Z} \times \mathbf{Z}$, showing that four classical bits are output. As we did for the Deutsch algorithm in 4.2, here we derive the output types for $\mathbf{Z}_1^{\boxdot}, \mathbf{Z}_2^{\boxdot}, \mathbf{Z}_3^{\boxdot}$ and $\mathbf{Z}_4^{\boxdot}$ which we'll immediately convert to $\mathbf{Z}_1^{\mathbf{I}}, \mathbf{Z}_2^{\mathbf{I}}, \mathbf{Z}_3^{\mathbf{I}}$ and $\mathbf{Z}_4^{\mathbf{I}}$. We can trivially derive that `superdense` has types $\mathbf{Z}_1^{\mathbf{I}} \to \mathbf{Z}_1^{\mathbf{I}}$ and $\mathbf{Z}_2^{\mathbf{I}} \to \mathbf{Z}_2^{\mathbf{I}}$ (since CZ and $CNOT$ have types $\mathbf{Z}_1^{\mathbf{I}} \to \mathbf{Z}_1^{\mathbf{I}}$), so we'll look at the derivation for $\mathbf{Z}_3^{\mathbf{I}}$:

```
Definition superdense :=
  INIT ;     I ⊗ I ⊗ Z ⊗ I   (* initial type *)
  H 3  ;     I ⊗ I ⊗ X ⊗ I   (* create Bell pair *)
  CNOT 3 4 ; I ⊗ I ⊗ X ⊗ X
  CZ 1 3 ;   Z ⊗ I ⊗ X ⊗ X   (* map bits onto Bell pair *)
  CNOT 2 3 ; Z ⊗ I ⊗ X ⊗ X
  CNOT 3 4 ; Z ⊗ I ⊗ X ⊗ I   (* decode qubits *)
  H 3        Z ⊗ I ⊗ Z ⊗ I
```

We can similarly derive $\texttt{superdense} : \mathbf{Z}_4^{\mathbf{I}} \to \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{Z}$. We can now combine all four derivations using our distributivity rules for $\to$ and $\cap$ as follows:

$$\cfrac{\cfrac{\cfrac{\texttt{superdense} : (\mathbf{Z}_1^{\mathbf{I}} \to \mathbf{Z}_1^{\mathbf{I}}) \cap (\mathbf{Z}_2^{\mathbf{I}} \to \mathbf{Z}_2^{\mathbf{I}}) \cap (\mathbf{Z}_3^{\mathbf{I}} \to \mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{Z}_4^{\mathbf{I}} \to \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{Z})}{\texttt{superdense} : \mathbf{Z}_1^{\mathbf{I}} \cap \mathbf{Z}_2^{\mathbf{I}} \cap \mathbf{Z}_3^{\mathbf{I}} \cap \mathbf{Z}_4^{\mathbf{I}} \to \mathbf{Z}_1^{\mathbf{I}} \cap \mathbf{Z}_2^{\mathbf{I}} \cap \mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I} \cap \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{Z}}}{\texttt{superdense} : \mathbf{Z}_1^{\boxdot} \cap \mathbf{Z}_2^{\boxdot} \cap \mathbf{Z}_3^{\boxdot} \cap \mathbf{Z}_4^{\boxdot} \to \mathbf{Z}_1^{\boxdot} \cap \mathbf{Z}_2^{\boxdot} \cap (\mathbf{Z} \otimes \mathbf{I})_{3,4}^{\boxdot} \cap (\mathbf{I} \otimes \mathbf{Z})_{3,4}^{\boxdot}}}{\texttt{superdense} : \mathbf{Z}_1^{\boxdot} \cap \mathbf{Z}_2^{\boxdot} \cap \mathbf{Z}_3^{\boxdot} \cap \mathbf{Z}_4^{\boxdot} \to \mathbf{Z}_1^{\boxdot} \cap \mathbf{Z}_2^{\boxdot} \cap \mathbf{Z}_3^{\boxdot} \cap \mathbf{Z}_4^{\boxdot}}$$

Hence `superdense` has type $\mathbf{Z} \times \mathbf{Z} \times \mathbf{Z} \times \mathbf{Z} \to \mathbf{Z} \times \mathbf{Z} \times \mathbf{Z} \times \mathbf{Z}$ demonstrating that it takes separable $\mathbf{Z}$ qubits to separable $\mathbf{Z}$ qubits (or bits to bits). Note that under our stronger interpretation of the type system, this says that `superdense` successfully transmits two +1 eigenvectors of Z (that is, $|0\rangle$s) and we can similarly show that it takes $\mathbf{Z} \times (-\mathbf{Z}) \times \mathbf{Z} \times \mathbf{Z} \to \mathbf{Z} \times (-\mathbf{Z}) \times \mathbf{Z} \times (-\mathbf{Z})$ and likewise for negating the other input.

5 NORMAL FORMS FOR INTERSECTION TYPES

5.1 Normalizing Terms

A key property that allows us to verify equality of intersection types is the existence of a canonical form in which to describe them. Our canonical form 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 $\mathbf{X} < \mathbf{Z}$ so $\mathbf{I} \otimes \mathbf{X}$ precedes $\mathbf{I} \otimes \mathbf{Z}$. Further, an intersection type 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 following rewrite rule will be useful to reduce types to their canonical forms: $A \cap B = A \cap A * B = A * B \cap B$. Given an n-qubit intersection type with m independent terms $A_1 \cap \ldots \cap A_m$, do the following:

(1) Replace all occurrences of $\boxdot$ with $\mathbf{I}$ in each term. Let the set of pivots, P be initialized to $\emptyset$.
(2) For each qubit $i = 1 \ldots n$:
  - For the first term $j \notin P$ such that $\mathbf{X}_i \in A_j$:[2]
    - Update $P \leftarrow P \cup \{j\}$.
    - For terms $k \neq j$, if $\mathbf{X}_i \in A_k$, rewrite $A_k \leftarrow A_j * A_k$.
  - If there is no term with $\mathbf{X}$ on qubit i , for the first term $j \notin P$ such that $\mathbf{Z}_i \in A_j$:
    - Update $P \leftarrow P \cup \{j\}$.
    - For terms $k \neq j$, if $\mathbf{Z}_i \in A_k$, rewrite $A_k \leftarrow A_j * A_k$.
  - If no term contains $\mathbf{X}$ or $\mathbf{Z}$ on qubit i proceed.
(3) We order terms in increasing order of pivots.

Notice that each term can be used as a pivot at most once and the canonical form will be unique.

[2]We use $\mathbf{X}_i \in A_j$ here to mean that the type for qubit i in A_j is $\mathbf{X}$ or $\mathbf{XZ}$.

Given a canonical n-qubit intersection type with m independent terms $A_1 \cap \ldots \cap A_m$, finding if a sub-system 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 it has type $\mathbf{I}$ in all terms and we conclude that K is not separable from the remaining system. If every qubit in K has a pivot, we run the following procedure:

- Let $A_{K,(1)}, \ldots, A_{K,(k)}$ be the k terms which have the pivots for qubits in K.
- For each $j = 2 \ldots k$, check that $A_{K,(1)}$ commutes with $A_{K,(j)}$ using the procedure outlined in §4.3. [3]
- For each $j = 1 \ldots k$, check that every qubit $i \in \overline{K}$ has type $\mathbf{I}$ in the term $A_{K,(j)}$.
- For each $j = 1 \ldots k$, set the type of every qubit $i \in \overline{K}$ to $\boxdot$ in the term $A_{K,(j)}$.

Example. 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 re-written 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, 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.

This allows us to easily apply our separability judgment. After box distribution, we obtain:

$$(\mathbf{X} \otimes \mathbf{X} \otimes \boxdot) \cap (\mathbf{Z} \otimes \mathbf{Z} \cap \boxdot) \cap (\boxdot \otimes \boxdot \otimes \mathbf{Z}),$$

which we may write as

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

5.2 Example: GHZ state, Entanglement Creation and Disentanglement

To demonstrate how we can track the possibly entangling and disentangling property of the CNOT gate, we can look at the example of creating the GHZ state $|000\rangle + |111\rangle$ starting from $|000\rangle$ and then disentangling it. A similar example was considered by Honda [2015] to demonstrate how his system can track when *CNOT* displays either its entangling or disentangling behaviour. One crucial difference is that Honda uses the denotational semantics of density matrices which, in practice, would scale badly with the size of the program being type checked. Our approach is closer to that of Perdrix [2007, 2008] in terms of design and scalability but capable of showing separability where the prior systems could not.

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

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

```
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 $\mathbf{Z}_2^{\boxdot}$ and $\mathbf{Z}_3^{\boxdot}$, we obtain the following type:

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

This type is non-trivial to read, but we can clearly see that entanglement is produced between the three qubits.

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

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

This isn't 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 $\mathbf{Z} \times (\mathbf{X} \otimes \mathbf{X} \cap \mathbf{Z} \otimes \mathbf{Z})$, that is, a $\mathbf{Z}$ qubit and a Bell pair.

If we then apply `CNOT 3 2` we get

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

to which we can apply distributivity and separability judgments to obtain

$$\mathbf{Z} \times \mathbf{Z} \times \mathbf{Z} \to \mathbf{Z} \times \mathbf{Z} \times \mathbf{X}$$

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

6 MEASUREMENT

It's challenging to turn Gottesman's semantics for measurement into a type system in light of the fact that 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 typechecking should be linear. Nonetheless, our normalization in §5 parallels that in the stabilizer formalism and the action of measurement on stabilizer groups is well understood [Gottesman 1998]. This produces a method for typechecking that is quadratic in the number of qubits in the worst case (see §8).

Formally measurement acts as follows; for ease of exposition, suppose we are measuring the first qubit in the computational basis (a Z-basis measurement). Using the same rewrite rules used for normalization, $A \cap B = A \ \cap \ A * B = A * B \ \cap \ B$, we do the following from our initial intersection:

(1) Use the rewrite rules to ensure there are only 0 or 1 terms in the intersection that involves $\mathbf{X}$ in the first position. If there is such a term, then remove it from the intersection.
(2) If there are no terms that have $\mathbf{X}$ in the first position, use the rewrite rules to ensure there are only 0 or 1 terms that have $\mathbf{Z}$ in the first position. If there is such a term, remove it from the intersection.
(3) Add $\mathbf{Z} \otimes \mathbf{I}^{n-1}$ to the intersection, which yields the post-measurement type.

Notice that this closely parallels the normalization procedure, where the key difference is that we are only interested in producing a pivot in the position of the measured qubit. By construction, at the position of the measured qubit each term in the intersection is either $\mathbf{I}$ or $\mathbf{Z}$. After normalization, these would all become $\mathbf{I}$, and so we see that we may take $\mathbf{Z} \otimes \boxdot^{n-1}$ instead of $\mathbf{Z} \otimes \mathbf{I}^{n-1}$ in the third step above.

Example. Continuing our analysis of the GHZ state from §5.2, the circuit `GHZ` had a co-domain of 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 and add $\mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{I}$ to get

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

Using these same rewrite rules, we can replace the second term with $\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I}$ and with that term replace the last with $\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Z}$, producing the output type as

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

7 BEYOND THE CLIFFORD GROUP

Universal quantum computation requires that we use gates outside the Clifford set, the two most studied candidates being the T $(\pi/8)$ and Toffoli gates. We start by giving the following type to T:

$$T : \mathbf{Z} \to \mathbf{Z}$$

Note that we don't give a type to T acting upon an $\mathbf{X}$ qubit. That shouldn't be surprising: In a classical type system the Boolean negation operator has type $\mathbb{B} \to \mathbb{B}$ but is not defined on natural numbers. Analogously, as T does not take $\mathbf{X}$ to a Pauli operator, it is not defined on $\mathbf{X}$ in our system.[4] We could alternatively create a Top type and declare that T has type $\mathbf{X} \to \top$ (as we did in [Rand et al. 2020]); we discuss the trade-offs of such an approach in Appendix A.

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.
```

Note that $T^\dagger$ is simply seven consecutive Ts, so like T, it preserves $\mathbf{Z}$ and cannot be applied to $\mathbf{X}$. Now consider `TOFFOLI`'s action on $\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X}$, with type annotations after every line:

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

[4]Note however, that T does implicitly have the type $\mathbf{I} \to \mathbf{I}$, as does every operator.

```
CNOT b c; T† c;        Z ⊗ I ⊗ Z
CNOT a c; T b; T c;    I ⊗ I ⊗ Z
H c;                   I ⊗ I ⊗ X
CNOT a b; T a; T† b;   I ⊗ I ⊗ X
CNOT a b.              I ⊗ I ⊗ X
```

The T's here are all identities, so only the behaviors of H (which takes $\mathbf{X}$ to $\mathbf{Z}$ and back) and $CNOT$ (which takes $\mathbf{I} \otimes \mathbf{Z}$ to $\mathbf{Z} \otimes \mathbf{Z}$ and back) are relevant.

The derivations for $\mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{I}$ and $\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I}$ are similar, leaving us with the following types for `TOFFOLI`:

$$\texttt{TOFFOLI} : \mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{I} \to \mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{I}$$
$$\texttt{TOFFOLI} : \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I} \to \mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{I}$$
$$\texttt{TOFFOLI} : \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X} \to \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X}$$

Using our technique for deriving judgments about separability, we can further derive the following type for the Toffoli gate

$$\texttt{TOFFOLI} : \mathbf{Z} \times \mathbf{Z} \times \mathbf{X} \to \mathbf{Z} \times \mathbf{Z} \times \mathbf{X}$$

which says that if you feed a Toffoli three separable qubits in the $\mathbf{Z}$, $\mathbf{Z}$ and $\mathbf{X}$ basis, you receive back three qubits in the same basis.

On the other hand, if you apply a Toffoli gate to $\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{Z}$, for instance, the H turns $\mathbf{Z}$ into $\mathbf{X}$, the subsequent $CNOT$ has no effect on the type, and $T^\dagger$ applied to an $\mathbf{X}$ isn't well-typed. Toffoli can always be given the type $\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{I} \to \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{I}$.

8 TYPECHECKING AND ITS COMPLEXITY

We can now present the algorithm for typechecking quantum circuits. Note that we say *typechecking* rather than *type inference*: Given that every circuit inhabits infinitely many types (though many will be equivalent), we cannot ascertain what type the user intended for it. (We could infer a maximally informative type, but this is unlikely to be human readable.) Instead, the user inputs a desired type and our procedure checks it. We will begin with a tailored algorithm for typechecking that illustrates all of the steps in the general typechecking procedure.

8.1 Checking fully separable types

Here we will assume that a user specifies that a given circuit is supposed to have the type $A_1 \times \cdots \times A_n \to B_1 \times \cdots \times B_n$, where each A_i, B_i is in $\{\pm\mathbf{X}, \pm\mathbf{Y}, \pm\mathbf{Z}\}$. We further assume all measurement is deferred until the end of the circuit [Nielsen and Chuang 2010, §4.4]. We will generalize this algorithm to a broader class of types afterwards.

Step 1: Remove separability judgments. Our first step is to remove the separability judgments (in the form of $\boxdot$ or $\times$ symbols) from the both sides of the judgment. We can do this by simply replacing each $U_i^{\boxdot}$ with $U_i^{\mathbf{I}}$ (Recall that $A_1 \times A_2$ is simply notation for $A_1^{\boxdot} \cap A_2^{\boxdot}$.)

Step 2: Distribute over arrows. Now that we have to check a type of the form $A_1 \cap \cdots \cap A_n \to B_1 \cap \cdots \cap B_n$ (repurposing our prior notation), we'll repeatedly apply the $\cap$-arrow distribution to obtain a goal of the form $A_1 \to A_1', A_2 \to A_2'$, etc. Note that each A_i' may or may not equal the corresponding B_i: The circuit `H 1; H 2` has type $\mathbf{X} \times \mathbf{Z} \to \mathbf{Z} \times \mathbf{X}$ which is derived from $(\mathbf{X}_1^{\mathbf{I}} \to \mathbf{Z}_1^{\mathbf{I}}) \cap (\mathbf{Z}_2^{\mathbf{I}} \to \mathbf{X}_2^{\mathbf{I}})$ (where $A_1' = \mathbf{Z}_1$), but `SWAP 0 1` has the same type, emerging from $(\mathbf{X}_1^{\mathbf{I}} \to \mathbf{Z}_2^{\mathbf{I}}) \cap (\mathbf{Z}_2^{\mathbf{I}} \to \mathbf{X}_1^{\mathbf{I}})$ (where $A_1' = \mathbf{Z}_2^{\mathbf{I}}$). Hence, for the following 2 steps, we will be doing type inference in the forward direction.

Step 3: Apply Gottesman's rules in the forward direction. To ascertain the values of $A'_1, \ldots, A'_n$, we can simply evolve $A_1, \ldots, A_n$ according to the basic rules from Gottesman's paper. For instance, applying an `H n` to a tensor product with $\mathbf{X}$ in the n^{th} position produces the same tensor updated to $\mathbf{Z}$ in the n^{th} position, while applying it to $\mathbf{XZ}$ yields $\mathbf{ZX} = -\mathbf{XZ}$. Applying a two qubit gate like CNOT takes an extra step of exchanging $A \otimes B$ for $(A \otimes \mathbf{I}) * (\mathbf{I} \otimes B)$ and applying the appropriate rules. Since we always normalize $*$-products to contain at most one $\mathbf{X}$ followed by at most one $\mathbf{Z}$ (e.g. $-\mathbf{X}$ or $i\mathbf{XZ}$ but never $\mathbf{Z}(-\mathbf{X})(i\mathbf{Z})$), gate application takes constant time.

Step 4: Measurement. Measurement is the first part of this procedure in which we cannot consider each A'_i independently, since measurement requires consideration of the full intersection of types. In this case we apply the measurement procedure of §6 for every measurement in the system.

Step 5: Normalization and Unification. Finally, we need to unify the A'_is that emerge from steps 2-4 with the B_is of our expanded initial types. We simply normalize both sides of the equation and check their equivalence up to intersection reordering.

8.2 Generalizing the algorithm

Of course, not every typing judgment has the form $A_1 \times \cdots \times A_n \to B_1 \times \cdots \times B_n$. In the general case, we will begin by checking that types are well-formed: terms in an intersection must all be of the same arity, and likewise across arrows. If there is no separability judgment in the terms, we simply skip step 1. Similarly, we will often be able to skip arrow distribution, when typechecking an expressive type like $CZ : (\mathbf{X} \otimes \mathbf{I} \to \mathbf{X} \otimes \mathbf{Z}) \cap (\mathbf{I} \otimes \mathbf{X} \to \mathbf{Z} \otimes \mathbf{X}) \cap (\mathbf{Z} \otimes \mathbf{I} \to \mathbf{Z} \otimes \mathbf{I}) \cap (\mathbf{I} \otimes \mathbf{Z} \to \mathbf{I} \otimes \mathbf{Z})$.

In the case with multiple separable subsystems, we must be careful before replacing $\boxdot$ with $\mathbf{I}$, as this is not generally valid. In this case, we check that the subsystems be spanned by independent, commuting factors and apply our rewrite rule for multi-qubit separability, eq. (6), in the reverse direction. (This excludes types like $\boxdot \otimes \mathbf{I} \to B$ as well.)

While we have assumed it for convenience, measurement need not be deferred until the end of the circuit as Steps 3 and 4 can be interleaved. In fact, in many protocols–such as those arising from error correction methods–one post-selects following operations based on the result of the measurement (see §9 below for more on this point). Note, however, that including many measurement operations in the circuit would increase the running time of our typechecking protocol.

In the absence of measurement and normalization, this entire procedure takes time linear in the number of gates G applied multiplied by the number of intersections terms m, as we can see from the linearity of the procedure in step 3. Measuring a single qubit scales as linear in the number of terms in its intersection. Normalization scales as $O(mn)$ for a type on n qubits with m terms. So with measurement and final normalization, the entire algorithm is upper bounded by $O(2nm+Gm+Mm+nm) \approx O((G+M+n)m)$ when we have G gate applications and M measurements. As m could be $2n$ in the worst case, this procedure could eventually scale as $O((G+M)n+n^2)$. When measurement is deferred to the end of the circuit, $M \leq n$, giving an $O(Gn+n^2)$ scaling.

9 APPLICATIONS AND FUTURE WORK

We think that this type system, along with possible extensions, is broadly applicable. Here we outline some of the possible uses of the type system along with (where necessary) extensions that will enable these uses.

Stabilizer Types and Quantum Error Correction. This aim of this paper is to define types for unitary operations, yet throughout, we have interpreted types such as $\mathbf{X} \otimes \mathbf{X}$ and $\mathbf{X} \otimes \mathbf{I}$ as being inhabited by certain states. By acting on such states, gates and circuits obtained an arrow type. But as we vary the input states on which unitaries act, they in fact obtain many different arrow types. We used the concept of intersection types to characterize this phenomenon. The nature of Pauli operators allows for a different treatment using the stabilizer formalism [Gottesman 1998].

As Pauli operators either commute or anticommute, for a type such as $(\mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{X} \otimes \mathbf{I})$ to be nonempty its terms must commute. In this example, the type contains $\{|{++}\rangle, |{+-}\rangle, |{-+}\rangle, |{--}\rangle\}$. But these are also eigenstates of $I \otimes X = (X \otimes X)(X \otimes I)$ and hence also of type $\mathbf{I} \otimes \mathbf{X} = (\mathbf{X} \otimes \mathbf{X}) * (\mathbf{X} \otimes \mathbf{I})$. Consequently our intersection type could be equally well presented as $(\mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{I} \otimes \mathbf{X})$ or $(\mathbf{I} \otimes \mathbf{X}) \cap (\mathbf{X} \otimes \mathbf{I})$, which is the genesis of our rewrite rule and normal form in §5. And so, we should not think of the intersection type $(\mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{X} \otimes \mathbf{I})$ as being determined by $\mathbf{X} \otimes \mathbf{X}$ and $\mathbf{X} \otimes \mathbf{I}$ alone, but rather by all the elements in the (commutative) group they generate.

Such a commutative group is typically called a stabilizer group. Formally, a stabilizer group is any commutative group of tensor products of Pauli operators having some common arity n that does not contain $-I^{\otimes n}$. This latter condition ensures the joint +1-eigenspace of all the operators in a stabilizer group exists, which then is called the stabilizer code associated to the group [Gottesman 1996]. The eigenspaces associated with other eigenvalue combinations are called syndrome spaces. Therefore our intersection type (presuming it does not include $-\mathbf{I}^n$) is realized as a stabilizer code and its associated syndrome spaces, hence capturing the notion of a Pauli frame [Knill 2005].

The Gottesman semantics of §2 then can be interpreted as a type theory for stabilizer groups and stabilizer codes, which is the direction taken by Honda [2015]. However, it is critically important to include signs in our types: the +1-eigenspace of a term in an intersection type contains the code space, while the −1-eigenspace is a syndrome space indicating an error has occurred in the system through some external cause. For example, if we treat $(\mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{X} \otimes \mathbf{I})$ as an intersection of signed types then the type is occupied only by $|{++}\rangle$, while $(-\mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{X} \otimes \mathbf{I})$ is $\{|{+-}\rangle\}$, and so on.

Syndrome measurement aims to detect (and ultimately correct) such errors, and thus we need to retain measurement outcomes in our types. In terms of the measurement process presented in §6 above, we would in step (3) add $\mathbf{Z} \otimes \mathbf{I}^{n-1}$ upon measuring +1 while adding $(-\mathbf{Z}) \otimes \mathbf{I}^{n-1}$ when measuring −1. This is beyond the capabilities of the type system we have discussed in this work. We may be able to treat such post-selection through dependent types. For instance in the example above have a post-measurement type $M(\mu)$ parametrized by the possible measurement outcomes.

Developing such a formalism would provide a type theory for stabilizer codes that includes encoding and decoding [Cleve and Gottesman 1997], and syndrome extraction [Steane 1997] circuits. Potential other applications could include analyzing circuits that implement non-Clifford gates on stabilizer codes [Paetznick and Reichardt 2013; Yoder 2017] and circuits that fault-tolerantly switch between codes [Colladay and Mueller 2018].

Resource Tracking. Resource monotones track the amount of resources contained in a type. For instance, at a very coarse level, one might quantify the entanglement in a state by counting the number of "ebits" needed to create the state. Here, a Bell state is of type $\mathbf{E} = (\mathbf{X} \otimes \mathbf{X}) \cap (\mathbf{Z} \otimes \mathbf{Z})$, which is counted as having one ebit. A separable type such as $\mathbf{X} \times \mathbf{Z}$ has a resource value 0 and hence no ebits. While the current system cannot handle such resource monotones, it seems plausible that using a suitable extension one could similarly calculate the resource cost of various operations. Say, by counting the number of $\mathbf{E}$ types used in a protocol. Then, superdense coding has an $\mathbf{E}$-count of 1 while entanglement swapping [Żukowski et al. 1993] will have an $\mathbf{E}$-count of 2.

Our types are too fine to capture the notion of local equivalence [Van den Nest et al. 2005]. Instead, one can coarsen to types generated from graph states [Fattal et al. 2004]. For example, all six entangled two qubit types are equivalent under the local operations of $H \otimes I$, $S \otimes I$, $I \otimes H$, and $I \otimes S$, and therefore, from an entanglement monotone perspective, they all contain the same amount of entanglement.

One can show that for three qubits, there are only two classes of entangled types, up to relabeling of the qubit indices [Bravyi et al. 2006]. Such computations are quite challenging at scale, and so methods to quantify the amount of entanglement using such states is a long-term goal of this project.

Provenance tracking. Another useful addition to the type system is *ownership*. Superdense coding is a central example of a class of quantum communication protocols. By annotating the typing judgments with ownership information and restricting multi-qubit operations to qubits under the same ownership, we can guarantee that superdense coding only transmits a single qubit, via a provided channel C. (Note that ownership types are a form of static information-flow control [Sabelfeld and Myers 2006].) With this additional typing information, we can give superdense coding the type

$$\mathbf{Z}_A \times \mathbf{Z}_A \times \mathbf{Z}_A \times \mathbf{Z}_B \to \mathbf{Z}_A \times \mathbf{Z}_A \times \mathbf{Z}_B \times \mathbf{Z}_B$$

indicating that a single qubit has passed from Alice's control to Bob's.

Acknowledgements

The first author would like to acknowledge the support of the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Quantum Testbed Pathfinder Program under Award Number DE-SC0019040. The third author was funded by EPiQC, an NSF Expedition in Computing, under grant CCF-1730449.

REFERENCES

Benjamin Bichsel, Maximilian Baader, Timon Gehr, and Martin Vechev. 2020. Silq: A High-Level Quantum Language with Safe Uncomputation and Intuitive Semantics. In *Proc. PLDI '20*. ACM, New York, NY, USA, 286–300. https://doi.org/10.1145/3385412.3386007

Sergey Bravyi, David Fattal, and Daniel Gottesman. 2006. GHZ extraction yield for multipartite stabilizer states. *J. Math. Phys.* 47, 6 (Jun 2006), 062106. https://doi.org/10.1063/1.2203431 arXiv:quant-ph/0504208

Richard Cleve and Daniel Gottesman. 1997. Efficient Computations of Encodings for Quantum Error Correction. *Phys. Rev. A* 56 (Jul 1997), 76–82. Issue 1. https://doi.org/10.1103/PhysRevA.56.76 arXiv:quant-ph/9607030

Kristina R Colladay and Erich J Mueller. 2018. Rewiring Stabilizer Codes. *New Journal of Physics* 20, 8 (2018), 083030. https://doi.org/10.1088/1367-2630/aad8dd arXiv:1707.09403

David Deutsch. 1985. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proc. R. Soc. Lond. A* 400, 1818 (1985), 97–117. https://doi.org/10.1098/rspa.1985.0070

David Deutsch and Richard Jozsa. 1992. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. A* 439, 1907 (1992), 553–558. https://doi.org/10.1098/rspa.1992.0167

David Fattal, Toby S Cubitt, Yoshihisa Yamamoto, Sergey Bravyi, and Isaac L Chuang. 2004. Entanglement in the stabilizer formalism. arXiv preprint. arXiv:quant-ph/0406168

Daniel Gottesman. 1996. Class of quantum error-correcting codes saturating the quantum Hamming bound. *Phys. Rev. A* 54, 3 (1996), 1862–1868. https://doi.org/10.1103/physreva.54.1862 arXiv:quant-ph/9604038

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

Alexander S. Green, Peter LeFanu Lumsdaine, Neil J. Ross, Peter Selinger, and Benoît Valiron. 2013. Quipper: A Scalable Quantum Programming Language. In *Proc. PLDI '13*. ACM, New York, NY, USA, 333–342. https://doi.org/10.1145/2491956.2462177 arXiv:1304.3390

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

Marek Żukowski, Anton Zeilinger, Michael A. Horne, and Aarthur K. Ekert. 1993. "Event-ready-detectors" Bell experiment via entanglement swapping. *Phys. Rev. Lett.* 71 (Dec 1993), 4287–4290. Issue 26. https://doi.org/10.1103/PhysRevLett.71.4287

E. Knill. 2005. Quantum computing with realistically noisy devices. *Nature* 434, 7029 (2005), 39–44. https://doi.org/10.1038/nature03350 arXiv:quant-ph/0410199

Michael A. Nielsen and Isaac L. Chuang. 2010. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, Cambridge.

Adam Paetznick and Ben W. Reichardt. 2013. Universal Fault-Tolerant Quantum Computation with Only Transversal Gates and Error Correction. *Phys. Rev. Lett.* 111 (Aug 2013), 090505. Issue 9. https://doi.org/10.1103/PhysRevLett.111.090505 arXiv:1304.3709

Simon Perdrix. 2007. Quantum Patterns and Types for Entanglement and Separability. *Electron. Notes Theor. Comput. Sci.* 170 (2007), 125–138. https://doi.org/10.1016/j.entcs.2006.12.015 Proc. QPL '05.

Simon Perdrix. 2008. Quantum Entanglement Analysis Based on Abstract Interpretation. In *Static Analysis*. Springer, Berlin, Heidelberg, 270–282. https://doi.org/10.1007/978-3-540-69166-2_18 arXiv:0801.4230

Benjamin C. Pierce. 2002. *Types and Programming Languages*. MIT Press, Cambridge, MA.

Robert Rand, Jennifer Paykin, Dong-Ho Lee, and Steve Zdancewic. 2018. ReQWIRE: Reasoning about Reversible Quantum Circuits. In *Proc. QPL '18*. Open Publishing Association, Waterloo, NSW, Australia, 299–312. https://doi.org/10.4204/EPTCS.287.17

Robert Rand, Aarthi Sundaram, Kartik Singhal, and Brad Lackey. 2020. Gottesman Types for Quantum Programs. In *Proc. QPL '20*. https://people.cs.uchicago.edu/~rand/qpl_2020.pdf To appear.

A. Sabelfeld and A. C. Myers. 2006. Language-based information-flow Security. *IEEE J. Sel. Areas Commun.* 21, 1 (9 2006), 5–19. https://doi.org/10.1109/JSAC.2002.806121

Andrew M. Steane. 1997. Active Stabilization, Quantum Computation, and Quantum State Synthesis. *Phys. Rev. Lett.* 78 (Mar 1997), 2252–2255. Issue 11. https://doi.org/10.1103/PhysRevLett.78.2252 arXiv:quant-ph/9611027

Krysta Svore, Alan Geller, Matthias Troyer, John Azariah, Christopher Granade, Bettina Heim, Vadym Kliuchnikov, Mariia Mykhailova, Andres Paz, and Martin Roetteler. 2018. Q#: Enabling Scalable Quantum Computing and Development with a High-level DSL. In *Proc. Real World Domain Specific Languages Workshop (RWDSL) 2018*. ACM, New York, NY, USA, Article 7, 10 pages. https://doi.org/10.1145/3183895.3183901 arXiv:1803.00652

Maarten Van den Nest, Jeroen Dehaene, and Bart De Moor. 2005. Local unitary versus local Clifford equivalence of stabilizer states. *Phys. Rev. A* 71 (Jun 2005), 062323. Issue 6. https://doi.org/10.1103/PhysRevA.71.062323 arXiv:quant-ph/0411115

Theodore J. Yoder. 2017. Universal fault-tolerant quantum computation with Bacon-Shor codes. arXiv preprint. arXiv:1705.01686

A WHY NOT TOP?

In §7 we gave the T gate the type $\mathbf{Z} \to \mathbf{Z}$ and didn't give it any type where $\mathbf{X}$ appears to the left of the arrow. While this isn't unusual for type systems in general, it is unusual for *our* type system, which given any Clifford circuit and input type, can provide an output type. Indeed, in an earlier version of this work [Rand et al. 2020], we included the following type for T:

$$T : \mathbf{X} \to \top$$

This type has the added advantage that it tells us what circuits leave the Clifford group. So why did why we drop this judgment?

To address this question, it's worth analyzing what $\top$ would mean under our various interpretations. Under the basic Gottesman interpretation of $T : \mathbf{X} \to \top$, applying an X before a T should be equal to applying an T after a $\top$. That is

$$TX\,|\psi\rangle = \top T\,|\psi\rangle$$

so $\top = TXT^\dagger$. Unfortunately, given that this isn't a Pauli operator, we would also have to add $H : \top \to \top$ and $S : \top \to \top$ to our type system, meaning that rather than representing some known matrix, $\top$ would have to represent an arbitrary matrix.

Similarly, in our eigenstate representation of Gottesman types, $U : \mathbf{Z} \to \top$ would mean that U takes $|0\rangle$ and $|1\rangle$ to the eigenstates of some unknown operator.

Finally, it's worth considering the behavior of $\top$ under intersection. Given that $\top$ is intended to represent a Top type (that is, one that subsumes all others) it should be the case that for any type $\mathbf{A}$, $\top \cap \mathbf{A} = \mathbf{A} = \mathbf{A} \cap \top$. However, our $\mathbf{I}$ type already has this property, meaning that $\top = \mathbf{I} \cap \top = \mathbf{I}$. Allowing us to replace $\top$ with $\mathbf{I}$ (or vice-versa) would trivially break the system as $\mathbf{X} \otimes \top$, which contains a variety of entangled pairs, would become the separable system $\mathbf{X} \otimes \mathbf{I}$.