# A Static Analysis of Entanglement

Nicola Assolini$^{1[0000-0002-6754-6206]}$, Alessandra Di Pierro$^{1[0000-0003-4173-7941]}$, and Isabella Mastroeni$^{1[0000-0003-1213-536X]}$

Dipartimento di Informatica
Università di Verona, Verona, Italy
{nicola.assolini,alessandra.dipierro,
isabella.mastroeni}@univr.it

**Abstract.** Managing quantum variables in quantum programs presents specific challenges due to the possible occurrence of *entanglement*, the quantum mechanical phenomenon for which two variables can reach a state where they cannot be separated into two distinct individual states. Such a phenomenon may lead to critical issues due to unintended measurements, which may alter the outcome of computations involving entangled variables. To address this problem, we propose a static analysis based on the abstract interpretation framework to soundly and automatically detect entanglement occurring in quantum programs. By constructing an abstract domain for the entanglement property, our analysis identifies cases where side effects from quantum operations may produce unwanted entanglement, thus reducing the possibility of unintended computational side effects.

## 1 Introduction

Quantum programming languages play an important role in quantum software development and are essential for effectively using a quantum computer for problem-solving [17,13]. Quantum programming language design and implementation provide crucial support to the rapid technological progress driven by the efforts of a number of companies striving to build large-scale quantum computers. The nature of quantum computation introduces specific challenges when working with these languages [3]. One such challenge comes from the fact that, unlike classical variables, quantum variables represent information encoded in quantum states, i.e., vectors within a Hilbert space; this inevitably implies the need for specific approaches, different from the classical case, for their abstraction.

Quantum computation is characterised by two fundamental principles: superposition and entanglement. Unlike classical systems, where a state exists in a single, definite configuration, a quantum state can simultaneously exist in a superposition of multiple classical states. Entanglement occurs when some particles become so strongly correlated through a computation that they form an 'inseparable' state, i.e., a state where they are no longer identifiable in their individual states. For program variables, this means that every time we act on a variable

$q$, we also alter the state of the other variables entangled with $q$. Entanglement is crucial for many applications such as quantum communication protocols (e.g. quantum teleportation [20, Chapter 5]), but can be problematic in quantum programming, particularly when combined with the principle of implicit measurement [22, Section 4.4]. Therefore, analysing and tracking entanglement is essential for effectively reasoning about quantum programs. A particularly important task is identifying sets of entangled variables, namely sets of variables, such that whenever we operate on one variable, we may alter other variables in the same set. Crucial for this task is to identify which variables are in a quantum state and which are not.

In this work, we introduce a new abstract domain for entanglement, which refines the abstract domain introduced in [2] by incorporating additional labels that abstract the quantum states of variables. These labels allow us to define a static analysis, which is able to identify entangled variables and approximate the variable's state. Our static analysis not only determines the sets of entangled variables but also distinguishes a particular relation, which we call *direct inseparability*, between entangled variables capturing the case of entangled states of the form $\alpha\,|00\dots0\rangle + \beta\,|11\dots1\rangle$. These states called GHZ from the names of their inventors[1], are particularly interesting because they can be 'disentangled' in a very easy way. Our work improves the accuracy of existing methods that use a similar but less refined domain, such as the approach in [27], while avoiding the computational challenges associated with more precise abstract domains that grow exponentially with the number of qubits [18]. Additionally, our analysis is flexible enough to be applied to all imperative quantum languages (e.g. Quipper [15], QWire [24], Qiskit [1], qrisp [31] and Guppy [28]) rather than being limited to specific quantum circuits as in [30,29].

In the following sections, we will first provide an overview of the essential background of quantum computation (Section 2) and introduce the programming language used to define our analysis (Section 3). We then refine the properties introduced in our previous work [2] (Section 4) and define a new abstract domain (Section 5). In Section 6, we present the abstract semantics, and in Section 7, we show how to compute the analysis on a control flow graph. Finally, Section 8 discusses related work and Section 9 concludes the paper with a summary of our findings and potential directions for future research.

## 2 Quantum Computation

This section briefly recalls the main aspects of quantum computation related to the entanglement phenomenon. In doing so, we will refer to the circuit model of computation. In a quantum circuit, wires represent quantum bits, or qubits, rather than bits. Thus, a qubit replaces the classical unit of information (the bit) in the quantum computation model, generalising the two only possible values 0

[1] Danny Greenberg, Mike Horne, and Anton Zeilinger experimentally created this three-particle entanglement showing that quantum mechanics is not compatible with Einstein's theory of 'hidden variables'.

and 1 of a bit to any vector in a complex Hilbert space (the quantum system), with 0 and 1 as basis vector. The typical notation of such vectors (or states of a qubit) is the Dirac *ket* notation, according to which $|0\rangle = (1,0)^T$ and $|1\rangle = (0,1)^T$ indicate the basis states 0 and 1 and, in general, $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ denotes a linear combination or *superposition* state. The numbers $\alpha$ and $\beta$ are complex numbers called probability amplitudes since, from them, we can infer the probability of the state resulting in 0 or 1 after measuring the system. Such probabilities are obtained as $|\alpha|^2$ and $|\beta|^2$, which explains why quantum states must be unitary, i.e., $|\alpha|^2 + |\beta|^2 = 1$ must hold.

Implementing significant and powerful quantum algorithms requires performing quantum computation on circuits that are more complex than a single qubit operation and involve $n$ qubit states with $n > 1$. A $n$ qubit state corresponds to a unitary vector in the $2^n$-dimensional Hilbert space ($\mathcal{H}^{2^n}$), obtained by composing by tensor product ($\otimes$) the vector space of the single qubits, each living in a 2-dimensional complex Hilbert space ($\mathcal{H}^2$) [22, Chapter 2]. For instance, the space of two qubits is $\mathcal{H}^4 = \mathcal{H}^2 \otimes \mathcal{H}^2$ and a generic state $|\psi\rangle$ in $\mathcal{H}^4$ can be written as $|\psi\rangle = \alpha_0|00\rangle + \alpha_1|01\rangle + \alpha_2|10\rangle + \alpha_3|11\rangle$ where all $\alpha_i$ are complex numbers.

## 2.1 Measurement

Quantum measurement is an operation that allows us to extract a classical result from a quantum superposition $|\psi\rangle$. This operation transforms the quantum state into a classical one by breaking the quantum coherence (and so the quantum nature) of the state. Therefore, measurement is typically applied as the last operation in a quantum circuit to get the final (classical and probabilistic) result of the coherent (i.e., in superposition) evolution of the quantum system represented by the circuit. Formally, quantum measurement on the state space of the quantum system is represented using measurement operators $\{M_m\}$, where $m$ corresponds to the possible outcomes of the measurement. If the system is in the quantum state $|\psi\rangle$ before the measurement, the probability of obtaining outcome $m$ is given by $p(m) = \|M_m|\psi\rangle\|^2$, where $\|\cdot\|$ is the vector norm[2], and the system state after the measurement is $\frac{M_m|\psi\rangle}{\sqrt{p(m)}}$. For instance, given one qubit, the measurement operators are $M_0 = |0\rangle\langle 0|$ and $M_1 = |1\rangle\langle 1|$, corresponding to the outcomes 0 and 1. If the state of the qubit before the measurement is $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the probability of measuring 0 is: $p(0) = \|M_0\psi\|^2 = |\alpha|^2$ and the probability of measuring 1 is $p(1) = \|M_1\psi\|^2 = |\beta|^2$. After the measurement, if outcome 0 is observed, the state collapses to $\frac{M_0|\psi\rangle}{|\alpha|} = |0\rangle$ while if outcome 1 is observed, the state collapses to $\frac{M_1|\psi\rangle}{|\beta|} = |1\rangle$.

[2] We refer here to the Hilbert space vector norm defined as $\|\,|\psi\rangle\,\| = \sqrt{\langle\psi|\psi\rangle}$, where $\langle\psi|$ is the conjugate transpose of $|\psi\rangle$ and $\langle x|y\rangle$ is the inner product between vector $|x\rangle$ and vector $|y\rangle$.

### 2.2 Entanglement

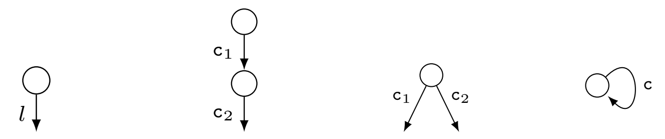
The behaviour of quantum circuits is determined by the laws of quantum mechanics and undergoes the effect of an important quantum phenomenon with no classical counterparts, namely *entanglement*. This can be intuitively described as an application of the superposition principle to a system composed of two or more subsystems. It occurs when statistically correlated measurement outcomes are observed as the effect of two subsequent quantum measurements, one on each subsystem. More concretely, the term entanglement describes a situation in which two particles, designated as $x$ and $y$, which form a composite system, become strongly correlated. This occurs as a result of a computational process that generates a superposition of product states for both particles. This superposition implies that the state of the composite system cannot be described without considering the other particle's state. Consequently, if measurements are made on an entangled state $ab + cd$, where $a$ and $c$ are two possible states of $x$ and $b$ and $d$ are two possible states of $y$, then if $x$ is found in state $a$, $y$ must be in state $b$; similarly if $x$ is found in state $c$, $y$ must be in state $d$.

As an example, the state $1/\sqrt{2}(|00\rangle + |11\rangle)$ in the Hilbert space $\mathcal{H}^2 \otimes \mathcal{H}^2$ is entangled because it cannot be expressed as a tensor product of the individual states of the two-component qubits. In this state, if one qubit is measured and found to be in the state $|0\rangle$, the other qubit will instantaneously collapse to the state $|0\rangle$ as well, and similarly for the state $|1\rangle$. In some cases, measuring a qubit of an entangled pair alters the other, keeping it in a quantum state. For instance, consider the entangled state $1/2(|00\rangle + |01\rangle + |10\rangle - |11\rangle)$. If the first qubit is measured and found in the state $|0\rangle$, the other qubit will instantaneously collapse to the state $1/\sqrt{2}(|0\rangle + |1\rangle)$ and, similarly, if the first qubit after the measurement collapses to $|1\rangle$, then the other one will be in state $1/\sqrt{2}(|0\rangle - |1\rangle)$.

### 2.3 Quantum Variables

A quantum variable is the high-level abstraction of the state of a quantum register. Therefore, its type is the dimension of the Hilbert space to which those states belong, namely $2^n$ for a $n$-qubit quantum register. Thus, the abstraction of a qubit is a quantum variable $q$ of type 2, whose states are vectors in a two-dimensional complex Hilbert space $\mathcal{H}_q$. Following [37], we construct the space of values for a set $Q = \{q_i\}$ of quantum variables $q_i$ as the Hilbert space $\mathcal{H}_Q = \bigotimes_i \mathcal{H}_{q_i}$ obtained by composing via tensor product the space of each variable. Given a quantum program characterised by a set $Q$ of quantum variables, we say that the Hilbert space $\mathcal{H}_Q$ is the program space, and the semantics of the program can be described using vectors and operators in the space $\mathcal{H}_Q$ [37].

We write $|\psi\rangle_{q_i}$ to indicate that $q_i$ represents the state $|\psi\rangle$ in $\mathcal{H}_{q_i}$. For entangled states, such for example $1/\sqrt{2}(|01\rangle + |10\rangle)$, we write $(1/\sqrt{2}(|01\rangle + |10\rangle))_{p,q}$ to indicate that $p$ and $q$ represent, respectively, the first and the second variable of the entangled pair. In this case, the state is an inseparable vector in the space $\mathcal{H}_p \otimes \mathcal{H}_q$. We use the same notation to represent linear operators on the program Hilbert space. Given an operator $U$ in the Hilbert space $\mathcal{H}_Q$ and a set of variables



(a) $l$ (b) $\mathsf{c}_1;\mathsf{c}_2$ (c) $\mathsf{c}_1 \oplus \mathsf{c}_2$ (d) $\mathsf{c}^*$

Fig. 1: Graphical representation of the language $\mathsf{c}$ constructs. The CFG (a), where $l \in \mathsf{label}$, represents a single statement, the CFG (b) corresponds to the sequential composition, (c) represents the branching $\mathsf{c} \oplus \mathsf{c}$ and (d) corresponds to the iteration $\mathsf{c}^*$.

$v \subseteq Q$, we write $U_v$ to indicate the operator acts as $U$ on the variables in $v$ and acts as the identity on the other variables of $Q$. For instance, $H_q$ is the unitary operator in $\mathcal{H}_Q$ that acts as the Hadamard gate($H$) on $q$ and as the identity on the other variables. In the same way, $CX_{p,q}$ is the operator corresponding to the control-not operators on $p$ and $q$ and the identity on the other variables.

## 3 Control Flow Graph Language

Static analysis is usually performed using the control flow graph (CFG) representation of programs [32]. We follow [23,6,3] and consider the CFG language defined by the syntax:

$$\mathsf{c} \ ::= \mathsf{label} \ \big| \ \mathsf{c};\mathsf{c} \ \big| \ \mathsf{c} \oplus \mathsf{c} \ \big| \ \mathsf{c}^*, \tag{1}$$

where the term $\mathsf{c};\mathsf{c}$ represents sequential composition; the term $\mathsf{c} \oplus \mathsf{c}$ represents branching; the term $\mathsf{c}^*$ is the Kleene closure of the $n$ time composition $c^n = \mathsf{c};\ldots\mathsf{c};$, where $n \in \mathbb{N}$, and the term $\mathsf{label}$ represents the labels that correspond to the statements of the analysed language. The CFG associated with a program is a graph with a start node corresponding to the program entry point, an end node corresponding to the exit point, and all other nodes corresponding to intermediate points in the execution of the program; each edge of the graph has a label that represents the change produced by the execution of an instruction of the language that is analysed. Hence, $\mathsf{label}$ defines the language of the programs we are analysing, while the other elements in the syntactic category $\mathsf{c}$ describe how we compose the edges of the program CFG depending on their labels as displayed in Figure 1. If $N$ is the set of program points, the CFG will be defined as sets of edges, with a label $l \in \mathsf{label}$, between nodes in $N$, i.e., as subsets of $N \times \mathsf{label} \times N$. In this way, we can define a new analysis simply by providing abstract semantics for the instructions defining $\mathsf{label}$.

We will define our analysis by referring to a minimal quantum language characterised by quantum statements on quantum variables with a control flow based on measurement. This means that the branching in the CFG is guided by the probabilistic result of a quantum measurement. Given a finite set of quantum variables $Q$ and $q, p \in Q$ we define the language $\mathsf{label}$ as follows:

$$\mathsf{label} ::= \mathtt{NonZero}(q) \Big| \ \mathtt{Zero}(q) \ \Big| \ \mathtt{skip} \ \Big| \ \mathtt{h}(q) \Big| \ \mathtt{t}(q) \Big| \ \mathtt{cx}(p, q), \tag{2}$$

where $\mathtt{Zero}(q)$ ($\mathtt{NonZero}(q)$) indicates that the measurement of $q$ returns 0 (1); the statements, $\mathtt{h}$, $\mathtt{t}$, and $\mathtt{cx}$ indicate, respectively, the Hadamard gate, the $T$ gate and the control-not gate [22, Chapter 4]. Considering only these three quantum operations is not a limitation because they allow us to cover all possible quantum computations [5]. Moreover, without loss of generality, we can assume that every variable $q_i \in Q$ corresponds to a 1-qubit register initialised to the state $|0\rangle$. In our language, we can write both `while` and `if then else` statements [35, Chapter 14, Exercise 14.4]:

$$\begin{aligned} \mathtt{if}\,(q)\,\mathtt{then}\,\mathtt{c}_1\,\mathtt{else}\,\mathtt{c}_2 &\equiv (\mathtt{NonZero}(q); \mathtt{c}_1) \oplus (\mathtt{Zero}(q); \mathtt{c}_2) \\ \mathtt{while}\,(q)\,\mathtt{do}\,\mathtt{c} &\equiv (\mathtt{NonZero}(q); \mathtt{c})^*; \mathtt{Zero}(q) \end{aligned}$$

Thus, our language is equivalent to the quantum while language that is used in [36,37,25], and to the one that is used to define the other two entanglement analyses based on abstract semantics [18,27].

### 3.1 Collecting semantics

Let $Q = \{q_i\}_n$ be the set of variables, we call $\mathcal{H}_Q = \bigotimes_i^n \mathcal{H}_{q_i}$ the $n$-qubit Hilbert space, i.e., a space of dimension $2^n$. Let $V_Q$ be the set of all vectors $|\psi\rangle \in \mathcal{H}_Q$, we define the collecting semantics as a function $[\![\, \cdot \,]\!] : \wp(V_Q) \to \wp(V_Q)$. First, given a set $v \in \wp(V_Q)$, we define the collecting semantics for each instruction of the language $\mathsf{label}$ as follows:

$$\begin{aligned} [\![\, \mathtt{skip} \,]\!] v &= v \\ [\![\, \mathtt{h}(q) \,]\!] v &= \left\{ \ H_q \,|\psi\rangle \ \middle| \ |\psi\rangle \in v \ \right\} \\ [\![\, \mathtt{t}(q) \,]\!] v &= \left\{ \ T_q \,|\psi\rangle \ \middle| \ |\psi\rangle \in v \ \right\} \\ [\![\, \mathtt{cx}(p, q) \,]\!] v &= \left\{ \ CX_{p,q} \,|\psi\rangle \ \middle| \ |\psi\rangle \in v \ \right\} \\ [\![\, \mathtt{NonZero}(q) \,]\!] v &= \left\{ \ \frac{M_{1q}\,|\psi\rangle}{\|M_{1q}\,|\psi\rangle\,\|} \ \middle| \ \|M_{1q}\,|\psi\rangle\,\|^2 > 0, |\psi\rangle \in v \ \right\} \\ [\![\, \mathtt{Zero}(q) \,]\!] v &= \left\{ \ \frac{M_{0q}\,|\psi\rangle}{\|M_{0q}\,|\psi\rangle\,\|} \ \middle| \ \|M_{0q}\,|\psi\rangle\,\|^2 > 0, |\psi\rangle \in v \ \right\} \end{aligned}$$

where $H_q$ and $T_q$ are the unitary operators in $\mathcal{H}_Q$ that correspond to the gate Hadamard and T applied to $q$, $CX_{p,q}$ is the unitary that corresponds to control-not on $p$ and $q$ (where $p$ is the controller and $q$ the target) and the identity on the other variables while $\mathtt{NonZero}(q)$ and $\mathtt{Zero}(q)$ corresponds to measurement 1 and 0 on $q$. For instance $[\![\, \mathtt{NonZero}(q) \,]\!] \{|1\rangle_q\} = \{|1\rangle_q\}$ while $[\![\, \mathtt{Zero}(q) \,]\!] \{|1\rangle_q\} = \varnothing$.

Finally, we can define the collecting semantics for the whole language:

$$\begin{aligned} \llbracket\, \mathtt{c}_1; \mathtt{c}_2 \,\rrbracket v &= \llbracket\, \mathtt{c}_2 \,\rrbracket(\llbracket\, \mathtt{c}_1 \,\rrbracket v) \\ \llbracket\, \mathtt{c}_1 \oplus \mathtt{c}_2 \,\rrbracket v &= \llbracket\, \mathtt{c}_1 \,\rrbracket v \cup \llbracket\, \mathtt{c}_2 \,\rrbracket v \\ \llbracket\, \mathtt{c}^* \,\rrbracket v &= \bigcup_n \llbracket\, \mathtt{c}^n \,\rrbracket v. \end{aligned} \tag{3}$$

With this collecting semantics, we only want to represent the set of all states that a program can return as a result. For this reason, when we consider the measurement, we follow a conservative approach by collecting all possible results, ignoring the probability with which these results occur.

## 4 Characterising Entanglement

In this section, we recall the properties of *separability* and of *direct inseparability*[3] [2], and the abstract domain proposed to represent them. Here, the idea is to refine this domain to make it suitable for performing a static analysis for soundly detecting entanglement. To define an abstract domain which is able to capture the entanglement property of quantum variables, we introduce a characterisation of this property by means of an equivalence relation. For bipartite systems (e.g. two qubits), entanglement and separability are dual concepts. In fact, a composite quantum state $|\psi\rangle_{q_1,q_2} \in \mathcal{H}_{q_1} \otimes \mathcal{H}_{q_2}$ is separable if and only if it can be written as a tensor product $|\psi\rangle_{q_1,q_2} = |\phi_{q_1}\rangle \otimes |\phi_{q_2}\rangle$ for some states $|\phi_{q_1}\rangle \in \mathcal{H}_{q_1}$ and $|\phi_{q_2}\rangle \in \mathcal{H}_{q_2}$. A state $|\psi\rangle_{q_1,q_2}$ is entangled if and only if it is *not* separable. However, the scenario becomes more complex when considering systems consisting of three or more subsystems. Entanglement in such systems corresponds to inseparability across the entire system, but various degrees of entanglement can occur within subsystems.

Some metrics have been introduced to analyse the entanglement of these systems, such as *entanglement monotones* and *entanglement measures* [33], which quantify entanglement between subsystems. In [10], the entanglement of two subsystems $S_1$, $S_2$ is measured in terms of an entanglement monotone function $E_{|\psi\rangle}(S_1, S_2)$, such that $E_{|\psi\rangle}(S_1, S_2) = 0$ if and only if $S_1$ and $S_1$ taken in isolation are not entangled in the global system state $|\psi\rangle$. For instance, the 3-qubits state $|\psi\rangle_{q_1,q_2,q_3} = {}^1\!/\!_2(|000\rangle + |001\rangle + |011\rangle + |111\rangle)_{q_1,q_2,q_3}$ is *fully inseparable* since it cannot be decomposed via the tensor product ($|\psi\rangle = |\phi_1\rangle \otimes |\phi_2\rangle$ for any $|\phi_1\rangle$ and $|\phi_2\rangle$). In fact, the entanglement monotone $E_{|\psi\rangle}(q_i, \{q_j, q_k\})$ always differs from zero for all $i, j, k \in \{1, 2, 3\}$. However, if we measure the entanglement between pairs of qubits, we have $E_{|\psi\rangle}(q_1, q_2) > 0$, $E_{|\psi\rangle}(q_2, q_3) > 0$ but $E_{|\psi\rangle}(q_1, q_3) = 0$, that is the qubits $q_1$ and $q_3$ taken in isolation are a separable subsystem. This example shows that the entanglement is not transitive, i.e., the fact that $q_1$ is entangled with $q_2$ and $q_2$ with $q_3$ does not imply that $q_1$ is entangled with $q_3$.

In our analysis, we are interested in understanding when a set of variables is fully inseparable or whether the variables are separable in some way. When two

[3] In [2] we call the direct inseparability property as being at the same level

variables are separable (and thus not entangled), we know we can measure one without altering the other. Thus, in our analysis, we speak about *separability* and we consider its dual notion *inseparability* instead of entanglement. Let us formally define the separability of two variables in a multi-variable state.

**Definition 1 (Separability).** *Let $Q$ be the set of variables in a state $|\psi\rangle_Q$. Two variables $q_1, q_2 \in Q$ are* separable *if the state $|\psi\rangle_Q$ can be written as $|\psi\rangle_Q = |\phi_1\rangle_{Q_1} \otimes |\phi_2\rangle_{Q_2}$, where $Q_1, Q_2 \subset Q$, $q_1 \in Q_1$ and $q_2 \in Q_2$. Otherwise, we say that $q_1, q_2$ are* inseparable*. Given a set $v \in \wp(V_Q)$, two variables $q_1, q_2$ are inseparable in $v$ if they are inseparable in* at least one state *$|\psi\rangle_Q \in v$.*

There exists a particular relation between inseparable variables. For instance, let us consider the state $|\psi\rangle_{a,b,c} = (|000\rangle + |110\rangle + |001\rangle - |111\rangle)_{a,b,c}$, where $a, b, c$ are inseparable. On an intuitive level, it can be seen that the three variables are not related in the same way. In fact, $a$ and $b$ are more closely related to each other than either $a$ with $c$ or $b$ with $c$: if we measure $a$ we obtain one of the two states: $(|00\rangle + |01\rangle)_{b,c}$ or $(|10\rangle - |11\rangle)_{b,c}$, where $b$ has collapsed to a base state (0 or 1) in both states while $c$ is still in superposition. Instead, if we measure $c$ we obtain: $(|00\rangle + |11\rangle)_{a,b}$ or $(|00\rangle - |11\rangle)_{a,b}$ where $a$ and $b$ are in a entangled and superpose state. When two variables are related as $a$ and $b$ in this example, we say that they are *directly inseparable.*

**Definition 2 (Direct Inseparability).** *Given $a$ and $b \in Q$ in a state $|\psi\rangle_Q$, we say that $a$ and $b$ are* directly inseparable*(d-inseparable) if, by measuring one of them, the other also collapses to a base state. Two variables $q_1, q_2 \in Q$ are d-inseparable in $v \in \wp(V_Q)$ if they are d-inseparable in* all states *$|\psi\rangle_Q \in v$.*

Being d-inseparable is a useful property when reasoning about entanglement. In fact, if we apply a controlled not ($CX$) between two d-inseparable variables, we will always 'disentangle' the target variable. For instance, if we apply $CX_{a,b}$, where $a$ is the controller and $b$ is the target, the state $|\psi\rangle_{a,b,c}$ defined above, we obtain:

$$\begin{aligned} CX_{a,b}(|\psi\rangle_{a,b,c}) &= (|000\rangle + |100\rangle + (|001\rangle - |101\rangle)_{a,b,c} = \\ &= ((|0\rangle + |1\rangle)\,|0\rangle + (|0\rangle - |1\rangle)\,|1\rangle)_{a,c} \otimes |0\rangle_b\,. \end{aligned} \tag{4}$$

In other words, we have separated $b$ from the other variables. Instead if we apply $CX_{a,c}$ we obtain:

$$CX_{a,c}(|\psi\rangle_{a,b,c}) = (|000\rangle + |111\rangle + |001\rangle - |110\rangle)_{a,b,c}, \tag{5}$$

and we do not 'disentangle' $a$ since $c$ and $a$ are not d-inseparable.

As shown in [2], inseparability and d-inseparability are equivalence relations.

## 5 An Abstract Domain for Entanglement

The generation of entanglement depends on the values of the variables, which, therefore, must be taken into account when defining the elements of our abstract domain. Thus, we define an abstract state as consisting of two parts: the first is based on sets of variables representing inseparability and d-inseparability. In contrast, the second consists of a function that associates each variable with a specific label indicating the variable's state.

**The Inseparability and D-Inseparability Domain** Inseparability and d-inseparability are both equivalence relations; thus, given a set of variables $Q$, we can represent both properties on $Q$ by partitions of $Q$. For instance, consider the state $|\psi\rangle_{a,b,c,d} = ((|00\rangle + |11\rangle)\,|0\rangle + (|00\rangle - |11\rangle)\,|1\rangle)_{a,b,c} \otimes |1\rangle_d$. We can build the partition $(\{a,b,c\}\{d\})$ that represents the inseparable variables in $|\psi\rangle_{a,b,c,d}$ and another partition $(\{a,b\},\{c\},\{d\})$ that identifies the d-inseparable variables. Since being d-inseparable implies being inseparable, the d-inseparable partition is always included in the inseparable one. We encode this information in our abstract states by representing them as a list of numbered sets (e.g., $[(\{a,b\},0),(\{c\},0),(\{d\},1)]$), where the smaller partition $((\{a,b\},\{c\},\{d\}))$ represents which variables are d-inseparable, while by merging sets with the same number, we obtain the partition representing inseparability $((\{a,b,c\}\{d\}))$.

**Definition 3.** *Given a set of quantum variables $Q$, we define the abstract state $\mathcal{E}^Q$ as the set of tuples*

$$\mathcal{E}^Q = \left\{\ (e,k) \,\middle|\, e \in \wp(Q) \text{ and } k \in \mathbb{N}\ \right\},$$

*where $\forall\,(e,k),(e',k') \in \mathcal{E}^Q, e \cap e' = \varnothing$ and $\bigcup_{(e,k)\in\mathcal{E}^Q} e = Q$. In other words, the sets $e$ form a partition of $Q$.*

We call $\mathbb{E}^Q \subset \wp(\wp(Q) \times \mathbb{N})$ the abstract domain of all possible $\mathcal{E}^Q$.

To better refer to the abstract state, we introduce the following notation. Given an abstract state $\mathcal{E}^Q$, we write $E_k$, using a capital letter, to refer to the set $E_k = \bigcup_{(e,k)\in\mathcal{E}^Q} e$, i.e., the union of all $e$ with the same index $k$. For instance, if $\mathcal{E}^{\{a,b,c,d,e\}} = [(\{a,b\},0),(\{c\},0),(\{d,e\},1)]$, $E_0 = \{a,b,c\}$ while $E_1 = \{d,e\}$. Using this notation, we introduce a partial order in $\mathbb{E}^Q$.

**Definition 4 ($\mathbb{E}^Q, \leqslant_{\mathbb{E}}$).** *Given $\mathcal{E}_1^Q, \mathcal{E}_2^Q \in \mathbb{E}^Q$. $\mathcal{E}_1^Q \leqslant_{\mathbb{E}} \mathcal{E}_2^Q$ iff $\forall (e_2,k) \in \mathcal{E}_2^Q$, $\exists\,(e_1,k') \in \mathcal{E}_1^Q$ such that $e_2 \subseteq e_1$ and $\forall E_k \in \mathcal{E}_1^Q$, $\exists\,E_h \in \mathcal{E}_2^Q$ such that $E_k \subseteq E_h$.*

We write $\vee_{\mathbb{E}}$ and $\wedge_{\mathbb{E}}$ to refer to the least upper bound (lub) and the greatest lower bound (glb) induced by $\leqslant_{\mathbb{E}}$, and the resulting domain is a complete lattice. For instance, $[(\{a,b,c\},0),(\{d\},1)] \leqslant_{\mathbb{E}} [(\{a,b\},0),(\{c\},0),(\{d\},1)] \leqslant_{\mathbb{E}}$ $[(\{a\},0),(\{b\},0),(\{c\},0),(\{d\},1)] \leqslant_{\mathbb{E}} [(\{a\},0),(\{b\},0),(\{c\},0),(\{d\},0)]$, and $[(\{a,b\},0),(\{c\},1)] \vee_{\mathbb{E}} [(\{a\},0),(\{b,c\},1)] = [(\{a\},0),(\{b\},0),(\{c\},0)]$.

To ensure soundness, we overestimate the non-separabilities and determine which variables are potentially inseparable. On the other hand, since being d-inseparable implies special effects in relation to control-not and measurement, we underestimate the d-inseparability property to make sure we do not introduce errors in the abstract semantics.

We have defined an abstract domain that allows us to represent inseparability and d-inseparability. However, we need a final ingredient to define the abstract semantics: some elements abstracting the variables' state.

**Labels:** We introduce some labels that represent some specific states that are relevant to entanglement abstraction. The $CX$ gate does not introduce entanglement if the controller is in a classical state ($|0\rangle$ or $|1\rangle$) or the target is in a uniform superposition ($^1/\sqrt{2}\,|0\rangle \pm |1\rangle$). To track these two states, we introduce two labels that represent two sets of states: $Z = \{\phi\,|b\rangle\}$ (the set of classical values), and $X = \{\phi(^1/\sqrt{2}\,|b\rangle \pm {}^1/\sqrt{2}\,|\bar{b}\rangle)\}$ (the set of values in uniform superposition), where $\phi$ represents a global phase, $b$ is a binary string and $\bar{b}$ is the negation of $b$ (e.g. if $b = 0$ then $\bar{b} = 1$ and if $b = 010$ then $\bar{b} = 101$). Moreover, we introduce three other labels:

$$P = \{\phi(^1/\sqrt{2}\,|b\rangle \pm {}^{i+1}/\sqrt{2}\,|\bar{b}\rangle)\} \qquad Y = \{\phi(^1/\sqrt{2}\,|b\rangle \pm {}^{i}/\sqrt{2}\,|\bar{b}\rangle)\}$$
$$R = \{\phi(^1/\sqrt{2}\,|b\rangle \pm {}^{i-1}/\sqrt{2}\,|\bar{b}\rangle)\}.$$

We need these labels to represent the semantics of the gate `t`. In particular, $\mathtt{t}X = P$, $\mathtt{t}P = Y$, $\mathtt{t}Y = R$ and $\mathtt{t}R = X$ while $\mathtt{t}Z = Z$. Then, we add a final label to represent states that are not classical, i.e., those that are definitely in superposition: $S = \{\phi(\alpha\,|b\rangle \pm \beta\,|\bar{b}\rangle) \mid |\alpha|^2 + |\beta|^2 = 1 \text{ and } \alpha \neq 0 \wedge \beta \neq 0\}$. Finally, we define $\bot_{\mathcal{L}}$ as the empty set and $\top_{\mathcal{L}}$ as the set of all possible vectors. We can order these labels by inclusion, constructing a lattice $(\mathcal{L}, \leqslant_{\mathcal{L}})$, represented in Figure 2.

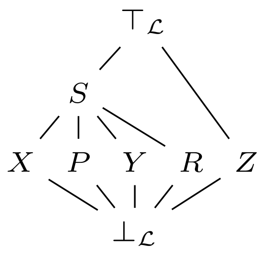


Fig. 2: Lattice $(\mathcal{L}, \leqslant_{\mathcal{L}})$

## 5.1 Put all together: The Abstract Domain

Now, we include the labels in the definition of the abstract domain. When variables are inseparable, their state cannot be described as a combination of individual states. Recall that, in an abstract state $\mathcal{E}^Q$, a set of inseparable variables is the union of all sets $e$ with the same index $k$. We introduce a labelling function to associate each set of inseparable variables with a label. Formally, we define the labelling function as $\lambda : \mathbb{N} \to \mathcal{L}$, and we call $\Lambda = \mathbb{N} \times \mathcal{L}$ the domain of the labelling function. Consequently, we reformulate the definition of the abstract state, including the labelling function, as follows.

**Definition 5.** *Given a set of quantum variables $Q$, let be $\mathcal{E}^Q \in \mathbb{E}^Q$ and $\lambda \in \Lambda$, we define an abstract state as the pair $(\mathcal{E}^Q, \lambda)$.*

We define $\mathbb{A}^Q = \mathbb{E}^Q \times \Lambda$ as the abstract domains. For instance, let us consider a set of variables $Q = \{a, b, c, d\}$ in the state

$$|\psi\rangle_Q = ({}^1\!/\!_2(|0\rangle + |1\rangle)\,|0\rangle + {}^i\!/\!_2(|0\rangle - |1\rangle)\,|1\rangle)_{a,b} \otimes {}^1\!/\!\sqrt{2}(|10\rangle + |01\rangle)_{c,d}.$$

First, we construct the partition corresponding to the sets of inseparable variables, i.e., the sets $\{a, b\}$ and $\{c, d\}$. Once these two sets have been identified, we construct the abstract state by identifying which variables are d-inseparable, obtaining the abstract state: $[(\{a\}, 0)(\{b\}, 0)(\{d, c\}, 1)]$. The last step is to label the sets of inseparable variables. In particular, given a set of variables in a state $|\phi\rangle$, we choose the smallest label $L$ such that $|\phi\rangle \in L$. We see that the state of the variables $a, b$ is contained only in $\top_{\mathcal{L}}$ while the state of $c, d$ is contained in $X, S, \top_{\mathcal{L}}$, consequently the final state will be equal to: $(\mathcal{E}^Q, \lambda) = ([(\{a\}, 0)(\{b\}, 0)(\{c, d\}, 1)], \{0 : \top_{\mathcal{L}}, 1 : X\}$.

At this point, if we want to compute the set of concrete states represented by an abstract state, starting from the obtain $(\mathcal{E}^Q, \lambda)$, we consider $\mathcal{E}^Q$. In this case, we know that there are two sets $\{a, b\}$, $\{c, d\}$ of inseparable variables, so the abstract state corresponds to a set of concrete states in the form $|\phi_1\rangle_{a,b} \otimes |\phi_2\rangle_{c,d}$. Then, we can get more precise information about $|\phi_1\rangle$ and $|\phi_2\rangle$ by checking which variables are d-inseparable, that is, $c$ and $d$. In particular, we know that the concrete states can be expressed by the set $\{|\Psi\rangle \mid |\Psi\rangle = (|\phi_1\rangle_{a,b} \otimes (|b\rangle + |\overline{b}\rangle)_{c,d}\}$. Now, we complete the concretisation by checking the information contained in the labels. By the labels we know that $a, b$ can be in any quantum state while $c, d$ are in the form of ${}^1\!/\!\sqrt{2}(|b\rangle + |\overline{b}\rangle)_{c,d}$. Finally, by intersecting this information with the previous one, we obtain the set: $\{|\psi\rangle \mid |\psi\rangle = (|\phi_1\rangle_{a,b} \otimes {}^1\!/\!\sqrt{2}(\alpha\,|b\rangle + \beta\,|\overline{b}\rangle)_{c,d}\}$, which represents the concretisation $\gamma((\mathcal{E}^Q, \lambda))$ of $(\mathcal{E}^Q, \lambda)$ and, of course, $|\psi\rangle \in \gamma((\mathcal{E}^Q, \lambda))$. Note that labels represent the state of a single variable or a state of $n$ d-inseparable variables. So, if a set of inseparable variables is not labelled as $\top_{\mathcal{L}}$, $Z$ or $\bot_{\mathcal{L}}$, it means that it corresponds either to a single variable or to all d-inseparable variables.

Based on the partial order $\leqslant_{\mathbb{E}}$ defined in Definition 4, we can define an ordering in $\mathbb{A}^Q$.

**Definition 6 ($\mathbb{A}^Q, \sqsubseteq$).** *Given $(\mathcal{E}_1^Q, \lambda_1), (\mathcal{E}_2^Q, \lambda_2) \in \mathbb{A}^Q$, $(\mathcal{E}_1^Q, \lambda_1) \sqsubseteq (\mathcal{E}_2^Q, \lambda_2)$ if and only if*

$$\mathcal{E}_1^Q \leqslant_{\mathbb{E}} \mathcal{E}_2^Q \ \wedge\ \forall E_h \in \mathcal{E}_2, \begin{cases} \lambda(k) \leqslant_{\mathcal{L}} \lambda(h) & \textit{if } \exists E_k \in \mathcal{E}_1 \textit{such that } E_h = E_k \\ \lambda(h) = \top_{\mathcal{L}} & \textit{otherwise} \end{cases}$$

We call $\sqcup$ and $\sqcap$ the lub and glb induced by the order. Since the lattices $(\mathbb{E}^Q, \leqslant_{\mathbb{E}})$ and $(\mathcal{L}, \leqslant_{\mathcal{L}})$ are finite and defined by inclusion operators, they are complete lattices. Thereby, also the lattice $((\mathcal{E}^Q, \lambda), \sqsubseteq)$ is complete.

For instance, consider two abstract states $(\mathcal{E}_1^Q, \lambda_1) = ([(\{p, q\}, 0)], 0 : X)$ and $(\mathcal{E}_2^Q, \lambda_2) = ([(\{p, q\}, 0)], 0 : Y)$; the lub between them is $(\mathcal{E}_3^Q, \lambda_3) = ([(\{p, q\}, 0)],$

$0 : S$). In this case, since $p, q$ are d-inseparable in both states, they are also in the lub, and we label the partition by the lub between the labels (we are in the 'if' case of the Definition 6). In fact, $(\mathcal{E}_1^Q, \lambda_1)$ represent the set of states $\{|\psi\rangle \mid \phi(1/\sqrt{2}\,|b\rangle \pm 1/\sqrt{2}\,|\overline{b}\rangle)\}$ and $(\mathcal{E}_2^Q, \lambda_2)$ represent the set of states $\{|\psi\rangle \mid \phi(1/\sqrt{2}\,|b\rangle \pm i/\sqrt{2}\,|\overline{b}\rangle)\}$, where $b \in \{00, 01, 10, 11\}$, and the abstraction of the union of these two sets is $(\mathcal{E}_3^Q, \lambda_3)$.
On the other hand, if we consider the states $(\mathcal{E}_4^Q, \lambda_4) = ([(\{p, q\}, 0), (\{t\}, 1)], \{0 : X, 1 : X\})$ and $(\mathcal{E}_5^Q, \lambda_5) = ([(\{p\}, 0), (\{q, t\}, 1)], \{0 : X, 1 : X\}$, the lub between them is $(\mathcal{E}_6^Q, \lambda_6) = ([(\{p\}, 0), (\{q\}, 0), (\{t\}, 0)], \{0 : \top_{\mathcal{L}}\}$. In fact, $(\mathcal{E}_4^Q, \lambda_4)$ represent the set of states $\{|\psi\rangle \mid \phi(1/\sqrt{2}\,|b\rangle \pm 1/\sqrt{2}\,|\overline{b}\rangle)_{p,q} \otimes 1/\sqrt{2}\,|0\rangle \pm 1/\sqrt{2}\,|1\rangle)_t\}$ and $(\mathcal{E}_5^Q, \lambda_5)$ represent the set of states $\{|\psi\rangle \mid \phi(1/\sqrt{2}\,|0\rangle \pm 1/\sqrt{2}\,|1\rangle)_p \otimes 1/\sqrt{2}\,|b\rangle \pm 1/\sqrt{2}\,|\overline{b}\rangle)_{q,t}\}$. If we join these sets, we obtain a set of states where $p, q, t$ are inseparable. However, $p$ and $q$ are only d-inseparable in some states, while $q$ and $t$ are d-inseparable in others. So, in general, we can only say that $p, q, t$ are inseparable and not d-inseparable, and the only possible label for these states is $\top_{\mathcal{L}}$.

**Concretisation Function** Now, we can introduce some formalism to define the concretisation function $\gamma : \mathbb{A} \to \wp(V_Q)$.

**Definition 7.** *Let $Q$ be a set of variables, and $\pi \subset \wp(Q)$ a partition of $Q$. Given a state $|\psi\rangle_Q$, we say that the variables in $Q$ are $\pi$-separable if $|\psi\rangle_Q = \bigotimes_{p\in\pi} |\phi\rangle_p$, i.e., their joint state can be decomposed into a product of states across a partition $\pi$ of the variables.*

For instance, given a state $|\psi\rangle_{q_1,q_2,q_3}$ and a partition $\pi = \{\{q_1, q_2\}\{q_3\}\}$, $q_1, q_2, q_3$ are $\pi$-separable if and only if we can write $|\psi\rangle_{q_1,q_2,q_3}$ as $|\phi_1\rangle_{q_1,q_2} \otimes |\phi_2\rangle_{q_3}$.

Given an abstract state $(\mathcal{E}^Q, \lambda)$, we write $\{E_k\}$ to indicate the sets that are obtained by $\mathcal{E}^Q = \{(e, k)\}$ joining the sets $e$ with the same $k$. Recall that $\{E_k\}$ is a partition that represents the sets of inseparable variables.

**Definition 8.** *Given a set of variables $Q$, we say that $|\psi\rangle_Q \triangleright (\mathcal{E}^Q, \lambda)$ (that is, $|\psi\rangle_Q$ is abstracted by $(\mathcal{E}^Q, \lambda)$) if and only if*

- *$|\psi\rangle_Q$ is $\{E_k\}$-separable;*
- *$\forall (e, k) \in \mathcal{E}^Q$, $q_i, q_j \in e \Rightarrow q_j$ and $q_i$ are d-inseparable in $|\psi\rangle_Q$;*
- *given $|\psi\rangle_Q = \bigotimes_k |\psi\rangle_{E_k}$, for all $k$, $|\psi\rangle_{E_k} \in \lambda(k)$.*

In other words, we say that an abstract state $\mathcal{E}^Q$ abstracts a concrete state if and only if the abstract state over-approximates the set of inseparable variables, under-approximates the d-inseparability, and all separable sub-states that compose the state are represented by labels. Now we have all the elements to define the concretisation function $\gamma : \mathbb{A} \to \wp(V_Q)$:

$$\gamma(\mathcal{E}^Q, \lambda) = \{|\psi\rangle_Q \mid |\psi\rangle_Q \triangleright (\mathcal{E}^Q, \lambda)\}.$$

**Theorem 1.** *Given a set of abstract states $\{(\mathcal{E}_j^Q, \lambda_j)\}_j$, let $I = \bigcap_j \gamma(\mathcal{E}_j^Q, \lambda_j)$, $\exists$ an abstract state $(\mathcal{E}_I^Q, \lambda_I)$ such that $\gamma((\mathcal{E}_I^Q, \lambda_I)) = I$.*

*Proof (Sketch).* $\gamma(\mathcal{E}_j^Q, \lambda_j)$ is the set of state in $\mathcal{H}_Q$ that are abstracted by $(\mathcal{E}_j^Q, \lambda_j)$, so $I$ is the set of states that are abstracted by all $(\mathcal{E}_j^Q, \lambda_j)$. Consequently, we if $(\mathcal{E}_I^Q, \lambda_I) = \bigsqcap_j (\mathcal{E}_j^Q, \lambda_j)$ then $\gamma((\mathcal{E}_I^Q, \lambda_I)) = I$.

Theorem 1 proves that $\mathbb{A}$ is isomorphic to a Moore family of $\wp(V_Q)$. This means that exist a function $\alpha_l : \wp(V_Q) \to \mathbb{A}$ such that $\langle \mathbb{A}, \alpha_l, \gamma_l, \wp(V_Q) \rangle$ forms a Galois Insertion [34,8,7].

## 6 An Abstract Semantics

In this section, we define the abstract semantics for our analysis. We first need to introduce some additional notation to better represent the operations on abstract states. Let us consider a generic abstract state $(\mathcal{E}^Q, \lambda)$, where $\mathcal{E}^Q = \{(e_i, k_i)\}_i$. We write $\mathcal{E}^Q(q)$ to refer to the pair $(e, k) \in \mathcal{E}^Q$ such that $q \in e$, while we have $\mathcal{E}^Q\{q\}$ to refer to set $E_k \in \mathcal{E}^Q$ that contains $q$. For instance, if $\mathcal{E}_1 = [(\{q\}, 0), (\{t\}, 0), (\{r\}, 1)]$ then $\mathcal{E}_1(q) = (\{q\}, 0)$ and $\mathcal{E}_1\{q\} = \{q, t\}$. We write $\mathcal{E}^Q[q_1 + q_2]$ to mark $q_1$ and $q_2$ inseparable in $\mathcal{E}^Q$ (e.g., $\mathcal{E}_1[r + q]$ is equal to $[(\{q\}, 1), (\{t\}, 1), (\{r\}, 1)]$). Note that if we mark $q$ and $r$ as inseparable, this also affects $t$ due to the transitivity of the inseparability. $\mathcal{E}^Q[q_1 \uplus q_2]$ means that we marked $q_1$ and $q_2$ as d-inseparable, so given $\mathcal{E}_1$ from above, $\mathcal{E}_1[q \uplus t] = [(\{q, t\}, 0), (\{r\}, 1)]$. Note that $\mathcal{E}[q_1 \uplus q_2]$ implies applying also $\mathcal{E}[q_1 + q_2]$. $\mathcal{E}^Q[\sim q]$ denotes the removal of $q$ from the set of variables d-inseparable from itself, and $\mathcal{E}^Q[\neg q]$ denotes that we make $q$ separable from the rest of the variables. For instance, given $\mathcal{E}_2 = [(\{q, r\}, 0), (\{t\}, 0)]$, $\mathcal{E}_2[\sim q]$ is equal to $[(\{r\}, 0), (\{q\}, 0), (\{t\}, 0)]$ and $\mathcal{E}_2[\neg t]$ correspond to $[(\{q, r\}, 0), (\{t\}, 1)]$. Of course, $\mathcal{E}_2[\neg q_1]$ implies $\mathcal{E}_2[\sim q_1]$. Finally, to ease the use of the labelling function, given a variable $q$, let $\mathcal{E}^Q(q) = (e, k)$, we write $\lambda(q)$ to refer to $\lambda(k)$. Additionally, we write $\lambda[q \leftarrow L]$ to state that we change the label referred to the index $k$ associated with $q$, setting it equal to $L$. For instance, given $(\mathcal{E}^{q,p,t}, \lambda) = ([(\{q\}, 0)(\{p\}, 0), (\{t\}, 1)], \{0 : \top_{\mathcal{L}}, 1 : X\})$, $\lambda(q) = \lambda(p)$ correspond to $\lambda(0) = \top_{\mathcal{L}}$, $\lambda(t) = X$, since it corresponds to $\lambda(1)$, and $(\mathcal{E}^{q,p,t}, \lambda[q \leftarrow Y])$ is equal to $([(\{q\}, 0)(\{p\}, 0), (\{t\}, 1)], \{0 : Y, 1 : X\})$. In general, when we speak about a label associated with a variable $q$, we implicitly refer to the label associated with the index associated with $q$.

Before defining the abstract semantics of the language, we define four abstract operations $H_q^\sharp, T_q^\sharp, CX_{p,q}^\sharp, M_q^\sharp : \mathbb{A}^Q \to \mathbb{A}^Q$, that correspond to the abstraction of the unitary operation $H, T, CX$ and the measurement respectively.

For all the abstract operations it holds that if $(\mathcal{E}^Q, \lambda) = \bot$, then $G_v^\sharp(\mathcal{E}^Q, \lambda) = (\mathcal{E}^Q, \lambda)$. If fact $\gamma(\bot) = \varnothing$ and $G_v \varnothing = \varnothing$.

*T gate* $(T_q^\sharp)$ This gate does not make a difference if we apply it to a single or a group of d-inseparable variables. Formally,

$$T_q^\sharp(\mathcal{E}^Q, \lambda) = (\mathcal{E}^Q, \lambda[q \leftarrow V]),$$

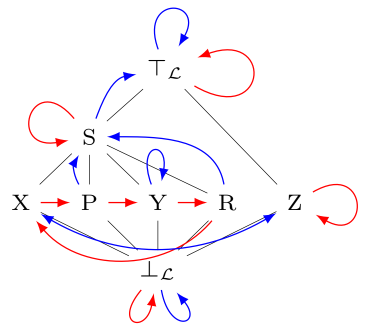


Fig. 3: The red arrow indicates the semantics of the abstract operation $T_q^\sharp$ while the blue one indicates the semantics of the abstract operation $H_q^\sharp$.

where $V$ can be derived from the red arrows in Figure 3.

*Hadamard gate ($H_q^\sharp$)* The Hadamard gate distinguishes whether $q$ is entangled with other variables. Formally, the abstract semantics is defined as follows:

$$H_q^\sharp(\mathcal{E}^Q, \lambda) = \begin{cases} (\mathcal{E}^Q, \lambda[q \leftarrow V]) & |\mathcal{E}^Q\{q\}| = 1, \\ (\mathcal{E}^Q(\sim q), \lambda[q \leftarrow \top_{\mathcal{L}}]) & \text{otherwise.} \end{cases}$$

In particular, if $q$ is separable from the other variables (i.e., $|\mathcal{E}^Q\{q\}| = 1$), $V$ can be derived from the blue arrows in Figure 3. If $q$ is inseparable (i.e., $|\mathcal{E}^Q\{q\}| > 1$), applying Hadamard to it produces a state that we can only label with $\top_{\mathcal{L}}$ and the variable $q$ is no longer d-inseparable. For instance, given $|\psi\rangle_{p,q,t} = (1/\sqrt{2}(|000\rangle + |111\rangle)_{p,q,t}$, $p, q, t$ are d-inseparable and their state can be labelled as $X$. Then, applying $H_q\,|\psi\rangle_{p,q,t} = (|000\rangle + |010\rangle + |101\rangle + |111\rangle)_{p,q,t}$, $q$ is not is no longer d-inseparable from $p, t$, and the state is only labelable by $\top_{\mathcal{L}}$.

*Controlled-Not gate ($CX_{c,t}^\sharp$)* The $CX$ gate can introduce or nullify entanglement, so we need to consider different cases according to the state of $c$ and $t$. Given an abstract state $(\mathcal{E}^Q, \lambda)$, we can define the abstract semantics as follows:

- if $\lambda(c) = Z$, i.e., the controller is in a base value, the $CX$ corresponds to a classical controlled not, so it does not introduce or nullify entanglement and it does not change labels, thus:

$$CX_{c,t}^\sharp(\mathcal{E}^Q, \lambda) = (\mathcal{E}^Q, \lambda);$$

- if $t$ is separable from other variables (i.e. $|\mathcal{E}^Q\{t\}| = 1$) we check the state of $c$ and $t$:

$$CX_{c,t}^\sharp(\mathcal{E}^Q, \lambda) = \begin{cases} (\mathcal{E}^Q, \lambda) & \text{if } \lambda(t) = X, \\ (\mathcal{E}[c \uplus t], \lambda) & \text{if } \lambda(c) \neq \top_{\mathcal{L}} \wedge \lambda(t) = Z, \\ ((\mathcal{E}[\sim t])[c + t], \lambda[t \leftarrow \top_{\mathcal{L}}]) & \text{otherwise.} \end{cases}$$

In particular, if the target is in uniform superposition, the $CX$ gate corresponds to the identity.
If the controller is surely in superposition (i.e. $\lambda(c) \neq \top_{\mathcal{L}}$ and $\lambda(c) \neq Z$) and the target is a classical value, the $CX$ gate makes $t$ d-inseparable from $c$. Moreover, when we set $t$ d-inseparable from $c$, $t$ automatically gets the label of $c$. For instance, if we have three variables, $q, c, t$ in the state $|\psi_1\rangle = {}^1\!/\!\sqrt{2}(|00\rangle + |11\rangle)_{q,c} \otimes |1\rangle_t$, applying $CX_{c,t}\,|\psi_1\rangle$ we obtain ${}^1\!/\!\sqrt{2}(|001\rangle + |110\rangle)_{q,c,t}$ in which $q, c, t$ are d-inseparable. Then, given $Q = \{q, c, t\}$ and $(\mathcal{E}_1^Q, \lambda_1) = \{(\{q, c\}, 0), (\{t\}, 1)\}, \{0 : X, 1 : Z\})$ such that $|\psi_1\rangle \in \gamma(\mathcal{E}_1^Q, \lambda_1)$, $CX^\sharp_{c,t}(\mathcal{E}_1^Q, \lambda_1) = (\{(\{q, c, t\}, 0)\}, \{0 : X\})$ and $CX_{c,t}\,|\psi_1\rangle \in \gamma(CX^\sharp_{c,t}(\mathcal{E}_1^Q, \lambda_1))$.
Finally, if none of the above conditions is fulfilled, we need to approximate the relation between $c$ and $t$. To maintain the soundness, we mark $c$ and $t$ as inseparable without setting $c$ and $t$ d-inseparable. Moreover, we mark $c$ and $t$ equal to $\top_{\mathcal{L}}$ (recall that, since $t$ and $c$ are inseparable, they are related to the same $k$, so writing $\lambda[t \leftarrow \top_{\mathcal{L}}]$ or $\lambda[c \leftarrow \top_{\mathcal{L}}]$ produce the same effects).

– if $t$ is inseparable from other variables, we need to check if the $CX$ gate nullifies some entanglements:

$$CX^\sharp_{c,t}(\mathcal{E}^Q, \lambda) = \begin{cases} (\mathcal{E}[\neg t], \lambda[t \leftarrow Z]) & \mathcal{E}^Q(c) = \mathcal{E}^Q(t) \\ (\mathcal{E}[\sim t], \lambda[t \leftarrow \top_{\mathcal{L}}]) & \text{otherwise} \end{cases}.$$

In particular, if $c$ and $t$ are d-inseparable ($\mathcal{E}^Q(c) = \mathcal{E}^Q(t)$), we 'disentangle' $t$, as we see in Equation 4, and, we set the disentangled variable separable from the rest, labelling it as $Z$. Otherwise, we do not know the exact effect of the gate since, as we have shown in Equation 5, the $CX$ alters the entangled state. Thus, to maintain soundness, we mark $t$ as not d-inseparable from the other variables and label it as $\top_{\mathcal{L}}$. For instance, given $(\mathcal{E}_1^Q, \lambda_1) = ([(\{a, b\}, 0), (\{c\}, 0)], \{0 : \top_{\mathcal{L}}\})$, $CX^\sharp_{a,b}(\mathcal{E}_1^Q, \lambda_1) = ([(\{a\}, 0), (\{c\}, 0), (\{b\}, 1)],$ $\{0 : \top_{\mathcal{L}}, 1 : Z\})$ while $CX^\sharp_{c,a}(\mathcal{E}_1^Q, \lambda_1) = ([(\{a\}, 0), (\{c\}, 0), (\{b\}, 0)], \{0 : \top_{\mathcal{L}}\})$. Given $|\psi\rangle_{a,b,c}$, $CX_{a,b}\,|\psi\rangle_{a,b,c}$ and $CX_{c,a}\,|\psi\rangle_{a,b,c}$ from Equation 4 and Equation 5, note that $|\psi\rangle_{a,b,c} \in \gamma((\mathcal{E}_1^Q, \lambda_1))$, $CX_{a,b}\,|\psi\rangle_{a,b,c} \in \gamma(CX^\sharp_{c,a}(\mathcal{E}_1^Q, \lambda_1))$ and $CX_{c,a}\,|\psi\rangle_{a,b,c} \in \gamma(CX^\sharp_{a,b}(\mathcal{E}_1^Q, \lambda_1))$.

*Measurement* ($M^\sharp_q$) In this case, we need to approximate which variables may be affected by the measurement. The semantics of measurement is formally defined as:

$$M^\sharp_q(\mathcal{E}^Q, \lambda) = (\mathcal{E}[\neg Q], L[Q \leftarrow Z, T \leftarrow \top_{\mathcal{L}}]),$$

where $Q = \mathcal{E}^Q(q)$ and $T = (\mathcal{E}^Q\{q\} \setminus Q)$. In particular, when a variable $q$ is separable, $Q = \{q\}$ and $T = \varnothing$ and the measurement simply makes $q$ collapse to a base state. Instead, if $q$ is inseparable from other variables, all variables d-inseparable from $q$ collapse to $Z$ (making them separable), while all other variables inseparable and not d-inseparable from $q$ are altered in a way that cannot be modelled given an abstract state. For this reason, we can only label these variables with $\top_{\mathcal{L}}$.

*Language Abstract Semantics* Now we have all the ingredients to define the abstract semantics of the language, which we define as a function $[\![\cdot]\!]^\sharp : \mathbb{A}^Q \to \mathbb{A}^Q$, for each instruction of our language:

$$\begin{aligned}
[\![ \mathit{skip} ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= (\mathcal{E}^Q,\lambda)\\
[\![ \mathtt{h}(q) ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= H_q^\sharp(\mathcal{E}^Q,\lambda)\\
[\![ \mathtt{t}(q) ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= T_q^\sharp(\mathcal{E}^Q,\lambda)\\
[\![ \mathtt{cx}(p,q) ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= CX_{p,q}^\sharp(\mathcal{E}^Q,\lambda)\\
[\![ \mathtt{NonZero}(b) ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= M_q^\sharp(\mathcal{E}^Q,\lambda)\\
[\![ \mathtt{Zero}(b) ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= M_q^\sharp(\mathcal{E}^Q,\lambda)\\
[\![ \mathtt{c}_1;\mathtt{c}_1 ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= [\![ \mathtt{c}_2 ]\!]^\sharp([\![ \mathtt{c}_1 ]\!]^\sharp(\mathcal{E}^Q,\lambda))\\
[\![ \mathtt{c}_1 \oplus \mathtt{c}_2 ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= [\![ \mathtt{c}_1 ]\!]^\sharp \sqcup [\![ \mathtt{c}_2 ]\!]^\sharp\\
[\![ \mathtt{c}^* ]\!]^\sharp(\mathcal{E}^Q,\lambda) &= \bigsqcup_n [\![ \mathtt{c}^n ]\!]^\sharp(\mathcal{E}^Q,\lambda).
\end{aligned} \tag{6}$$

where $H_q^\sharp, T_q^\sharp, CX_{p,q}^\sharp, M_q^\sharp : \mathbb{A}^Q \to \mathbb{A}^Q$ represent the abstract semantics of gates and measurement.

Finally, we formulate the soundness of our abstraction in terms of the concretisation function $\gamma$ [8].

**Proposition 1 (Soundness).** *Let $Q$ be a set of variables, $\forall l \in$ label, $\forall(\mathcal{E}^Q,\lambda) \in \mathbb{A}$, $[\![ l ]\!] \circ \gamma(\mathcal{E}^Q,\lambda) \subseteq \gamma \circ [\![ l ]\!]^\sharp(\mathcal{E}^Q,\lambda)$.*

Evidence in support of this proposition is shown by the examples in Sect. 7.1. Since every label is sound, by induction on c we can prove that $[\![ \mathtt{c} ]\!] \circ \gamma(\mathcal{E}^Q,\lambda) \subseteq \gamma \circ [\![ \mathtt{c} ]\!]^\sharp(\mathcal{E}^Q,\lambda)$.

## 7 Computing the Analysis

To compute the analysis on the CFG, we need to compute the abstract state $(\mathcal{E}^Q,\lambda)$ for each node of the CFG, namely at each program point of the analysed program [32]. The analysis we propose is forward; namely, the state $(\mathcal{E}^Q,\lambda)$ at node $u$, denoted $(\mathcal{E}^Q,\lambda)[u]$, depends on the pairs $\{(\mathcal{E}_i^Q,\lambda_i)\}$ of its predecessors and the label semantics of the edges entering in $u$. Given a CFG $G$, for all node $u$ in $G$ (program points of the represented program), we define the following system of equations:

$$(\mathcal{E}^Q,\lambda)[u] = \begin{cases} (\mathcal{E}_0^Q,\lambda_0) & \text{if } u = \textbf{start}\\ \bigsqcup \left\{ [\![ l ]\!]^\sharp(\mathcal{E}^Q,\lambda)[u] \;\middle|\; (u,l,v) \in G \right\} & \text{otherwise} \end{cases} \tag{7}$$

where $(\mathcal{E}_0^Q,\lambda_0)$ is the initial state. Since we assume that all variables are initialised to $|0\rangle$, the initial state is the state where all variables are separable and labelled as $Z$. So if $Q = \{a,b,c\}$ then $(\mathcal{E}_0^Q,\lambda_0) = ([(\{a\},0),(\{b\},1),(\{c\},2)],\{0,1,2,: Z\})$.

This system can be solved by the least fixed point obtaining the best solution for each program point [21].

**Proposition 2.** *For all statement $l \in$ label, the abstract semantics $[\![\, l \,]\!]^\sharp : \mathbb{A} \to \mathbb{A}$ is monotonic w.r.t. $\sqsubseteq$.*

Since the semantics is monotonic, it is granted that we reach the fix-point.

We provide a prototype of our procedure[4] that analyses the quantum language used to present the analysis. Together with the prototype, we provide examples showing how our analysis works in various scenarios. In particular, we analyse the examples contained in [30](superdense coding [4], Deutsch algorithmn [9] and the Creation and disentanglement of the GHZ state) and in [27] (teleportation circuit and GHZ), obtaining the same results as [30] and improving [27]. We also provide some examples showing how we lose precision in the presence of control flow, showing when our analysis is sound but incomplete.

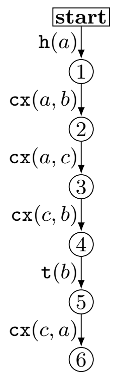
### 7.1 Showing the Analysis

Consider the example in Figure 4. In Figure 4, we show the CFG to the program $dGHZ ::= \mathtt{h}(a); \mathtt{cx}(a,b); \mathtt{cx}(a,c); \mathtt{cx}(c,b); \mathtt{t}(b); \mathtt{cx}(c,a)$ displaying for each program point the concrete state in blue and the abstract state in black. This program creates the GHZ states up to node **3**, then 'disentangles' $b$ with the first `cx`, then changes the relatives phase with the `t` gate and then disentangles $c$ with the last `cx`. In the abstract domain up to node **3**, we construct the state in which $\{a, b, c\}$ are d-inseparable. Then we are able to keep track in the abstract state the entanglement cancellation made by the `cx` gate in edges $(\mathbf{3}, \mathtt{cx}(a,b), \mathbf{4})$ and $(\mathbf{5}, \mathtt{cx}(c,a), \mathbf{6})$ and the phase change made by the `t` gate in edge $(\mathbf{4}, \mathtt{t}(b), \mathbf{5})$. We show how our analysis works with control flow in Figure 5. We consider the program $prog ::= \mathtt{h}(a); \mathtt{cx}(a,b); \mathtt{h}(c); \mathtt{if}\ (b)\ \mathtt{then}\ \{\mathtt{cx}(a,c)\}\ \mathtt{else}\ \{\mathtt{cx}(c,b)\}$, writing $|\phi\rangle$ to indicate the state ${}^1\!/\!\sqrt{2}(|0\rangle + |1\rangle)$ and $|\varphi\rangle$ to indicate ${}^1\!/\!\sqrt{2}(|0\rangle - |1\rangle)$. In this example, we start in node **1** with a state where $a$ and $b$ form a Bell state and are therefore abstracted as being entangled d-inseparable. In nodes **2** and **3**, due to the measurement of $b$, both $a$ and $b$ collapse to a basis state. Since $a$ and $b$ are d-inseparable, they are both labelled with $Z$. In node **4**, the concrete state is obtained by merging the semantics of the two paths, while the abstract state is the lub between $CX^\sharp_{a,c}([(\{a\},0),(\{b\},1),(\{c\},2)],\{(0,1){:}\,Z,2{:}\,X\})$ and $CX^\sharp_{c,b}([(\{a\},0),(\{b\},1),(\{c\},2)],\{(0,1){:}\,Z,2:X\})$, which correspond to: $([(\{a\},0),(\{b\},1),(\{c\},2)],\{(0,1){:}\,Z,2:X\}) \sqcup ([(\{a\},0),(\{b,c\},1)],\{(0){:}\,Z,1:X\}) = [(\{a\},0),(\{c\},1),(\{b\},1)],\{0{:}\,Z,1{:}\,\top_{\mathcal{L}}\}$. In both examples, using the labels, we can approximate the variable's state during the execution of the program.

## 8 Related Works

Yu et al. [38] propose an abstraction of quantum domains using the abstract interpretation formalism to achieve an abstract simulation of quantum circuits.

[4] The following GitHub repository NicolaAssolini98/EntaglementAnalysis contains our prototype implemented in Python



(a)

| | Concrete state | Abstract state |
|---|---|---|
| **Start** | $\{\lvert 000\rangle_{a,b,c}\}$ | $[(\{a\},0),(\{b\},1),(\{c\},2)],\{(0,1,2):Z\})$ |
| **1** | $\{{}^1/\sqrt{2}(\lvert 0\rangle + \lvert 1\rangle)_a \lvert 00\rangle_{b,c}\}$ | $[(\{a\},0),(\{b\},1),(\{c\},2)],\{0:X,(1,2):Z\})$ |
| **2** | $\{{}^1/\sqrt{2}(\lvert 00\rangle + \lvert 11\rangle)_{a,b} \lvert 0\rangle_c\}$ | $[(\{a,b\},0),(\{c\},2)],\{0:X,2:Z\})$ |
| **3** | $\{{}^1/\sqrt{2}(\lvert 000\rangle + \lvert 111\rangle)_{a,b,c}\}$ | $[(\{a,b,c\},0)],\{0:X\})$ |
| **4** | $\{{}^1/\sqrt{2}(\lvert 00\rangle + \lvert 11\rangle)_{a,c} \lvert 0\rangle_b\}$ | $[(\{a,c\},0),(\{b\},1)],\{0:X,1:Z\})$ |
| **5** | $\{{}^{i\,+\,1}/\sqrt{2}(\lvert 00\rangle + \lvert 11\rangle)_{a,c} \lvert 0\rangle_b\}$ | $[(\{a,c\},0),(\{b\},1)],\{0:P,1:Z\})$ |
| **6** | $\{{}^{i\,+\,1}/\sqrt{2}(\lvert 0\rangle + \lvert 1\rangle)_c \lvert 0\rangle_b \lvert 0\rangle_c\}$ | $[(\{c\},0),(\{b\},1),(\{a\},2)],\{0:P,(1,2):Z\})$ |

(b)

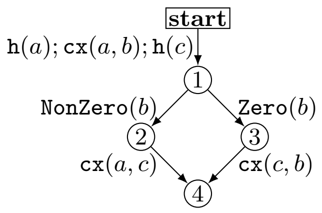
Fig. 4: The $dGHZ$ CFG (a), and a table representing concrete and abstract states for each node (b).

Another work [11] explores the relation between quantum Hoare logic, quantum incorrectness logic and abstract interpretation in the context of quantum programs.

An entanglement analysis was introduced in [26,27] for a simple while language that uses abstract semantics based on partitions. In this work, Perdrix uses partitions to represent entangled variables and two labels ($X$ and $Z$) to support the analysis. Moreover, in this approach, the labels are related to single variables, not to partitions, and every entangled variable is labelled as $\top$. In this way, this approach fails to detect when a gate or a measurement removes entanglement, to track the state of entangled variables, and to approximate the $T$ gate. For instance, in the examples in Figure 4 from the previous section, the Perdrix analysis computes in node **3** the abstract state $(\{a, b, c\}, \{a, b, c \to \top_{\mathcal{L}}\})$ that as no information. Consequently, after node 4, the three variables will always be considered entangled regardless of the gates applied.

Other systems have been developed to detect entanglement. Honda's approach [18] is based on an abstract domain that uses abstract density matrices. In this way, it is possible to abstract more information about entanglement, but the space of the abstract states explodes with the program's size. In [30,29], Rand introduces a type system based on Gottesman's [14] representation of Clifford gates ($H$,$S$,$CX$). This approach proposes an entanglement analysis, although working at the circuit level (i.e., no control flow) and limited to Clifford gates and measurement. This approach cannot be applied to the program in Figure 5 due to the presence of the `if` statement and to the program in Figure 4 due to the presence of the $T$ gate. Since our approach is based on abstract interpretation instead of type systems, it can be easily integrated with other analyses to improve precision.

The design of the language Twist [39] includes the verification of the separability of states, which is based on a type system with annotation (that must be



(a)

| | Concrete state | Abstract state |
|---|---|---|
| **Start** | $\{\|000\rangle_{a,b,c}\}$ | $[(\{a\},0),(\{b\},1),(\{c\},2)],\{(0,1,2):Z\})$ |
| **1** | $\{{}^{1}/\sqrt{2}(\|00\rangle+\|11\rangle)_{a,b}\,\|\phi\rangle_c\}$ | $[(\{a,b\},0),(\{c\},1)],\{0:X,1:Z\})$ |
| **2** | $\{\|11\phi\rangle_{a,b,c}\}$ | $[(\{a\},0),(\{b\},1),(\{c\},2)],\{(0,1):Z,2:X\})$ |
| **3** | $\{\|00\phi\rangle_{a,b,c}\}$ | $[(\{a\},0),(\{b\},1),(\{c\},2)],\{(0,1):Z,2:X\})$ |
| **4** | $\{\|11\varphi\rangle_{a,b,c}, (\|0\rangle_a\, {}^{1}/\sqrt{2}\,\|00\rangle+\|11\rangle)_{a,b,c}\}$ | $[(\{a\},0),(\{c\},1),(\{b\},1)],\{0:Z,1:\top_{\mathcal{L}}\})$ |

(b)

Fig. 5: The *prog* CFG (a), and a table representing concrete and abstract states for each node (b), where $|\phi\rangle = {}^{1}/\sqrt{2}(|0\rangle + |1\rangle)$ and $|\varphi\rangle = {}^{1}/\sqrt{2}(|0\rangle - |1\rangle)$

inserted manually) and dynamic checking (that uses classical simulation). Also, the Scaffold compiler [19] includes an analysis of entanglement, which works at the circuit level, considering only the circuit composed by controlled-$NOT$ gates.

## 9 Conclusion

In this paper, we have introduced a static analysis for quantum programming languages, building on the abstract domain for entanglement analysis developed in [2]. By extending the abstract domain with additional labels, we provide a more precise and practical method for analysing quantum entanglement. Our static analysis framework not only identifies entangled variables but also provides an abstract description of the program state during a computation, which can potentially be used as a basis for other analyses. Our approach improves other analyses proposed in the literature, such as [27], by enhancing precision while avoiding the exponential growth in computational complexity [18]. Additionally, the adaptability of our method within a while-language framework makes it a versatile tool for various quantum programming scenarios. In summary, our work contributes to the effective reasoning about quantum programs by providing a robust framework for entanglement analysis and quantum state approximation.

As a future development, we aim to implement our analysis so that it can be used in real quantum programming languages such as Qiskit [1], Qrisp [31], Guppy [28] or Isq [16]. To this end, we plan to integrate our approach into existing analysis tools, such as LiSA [12]. Future research will also explore further refinements to our analysis framework, e.g. by considering abstractions of the entanglement property along the lines of [10]. Moreover, we plan to define a probabilistic version of our analysis to better deal with the probabilistic control flow.

## References

1. Aleksandrowicz, G., et al.: Qiskit: An Open-source Framework for Quantum Computing (Jan 2019). https://doi.org/10.5281/zenodo.2562111, `https://doi.org/10.5281/zenodo.2562111`
2. Assolini, N., Di Pierro, A., Mastroeni, I.: Abstracting entanglement. In: Proceedings of the 10th ACM SIGPLAN International Workshop on Numerical and Symbolic Abstract Domains (NSAD '24) (2024). https://doi.org/AC10.1145/3689609.3689998
3. Assolini, N., Di Pierro, A., Mastroeni, I.: Static analysis of quantum programs. In: Static Analysis: 31th International Symposium, SAS 2024, Pasadena, USA, October 20-22, 2024. Proceedings. Springer (2024)
4. Bennett, C.H., Wiesner, S.J.: Communication via one- and two-particle operators on einstein-podolsky-rosen states. Phys. Rev. Lett. **69**, 2881–2884 (Nov 1992). https://doi.org/10.1103/PhysRevLett.69.2881, `https://link.aps.org/doi/10.1103/PhysRevLett.69.2881`
5. Boykin, P., Mor, T., Pulver, M., Roychowdhury, V., Vatan, F.: A new universal and fault-tolerant quantum basis. Information Processing Letters **75**(3), 101–107 (2000). https://doi.org/https://doi.org/10.1016/S0020-0190(00)00084-3, `https://www.sciencedirect.com/science/article/pii/S0020019000000843`
6. Bruni, R., Giacobazzi, R., Gori, R., Ranzato, F.: A correctness and incorrectness program logic. J. ACM **70**(2) (Mar 2023). https://doi.org/10.1145/3582267, `https://doi.org/10.1145/3582267`
7. Cousot, P., Cousot, R.: Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages. p. 238–252. POPL '77, Association for Computing Machinery, New York, NY, USA (1977). https://doi.org/10.1145/512950.512973, `https://doi.org/10.1145/512950.512973`
8. Cousot, P., Cousot, R.: Systematic design of program analysis frameworks. In: Proceedings of the 6th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages. p. 269–282. POPL '79, Association for Computing Machinery, New York, NY, USA (1979). https://doi.org/10.1145/567752.567778, `https://doi.org/10.1145/567752.567778`
9. Deutsch, D., Penrose, R.: Quantum theory, the church–turing principle and the universal quantum computer. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences **400**(1818), 97–117 (1985). https://doi.org/10.1098/rspa.1985.0070, `https://royalsocietypublishing.org/doi/abs/10.1098/rspa.1985.0070`
10. Di Pierro, A., Mancini, S., Memarzadeh, L., Mengoni, R.: Homological analysis of multi-qubit entanglement. Europhysics Letters **123**(3), 30006 (sep 2018). https://doi.org/10.1209/0295-5075/123/30006, `https://dx.doi.org/10.1209/0295-5075/123/30006`
11. Feng, Y., Li, S.: Abstract interpretation, Hoare logic, and incorrectness logic for quantum programs. Information and Computation **294**, 105077

(2023). https://doi.org/https://doi.org/10.1016/j.ic.2023.105077, `https://www.sciencedirect.com/science/article/pii/S0890540123000809`

12. Ferrara, P., Negrini, L., Arceri, V., Cortesi, A.: Static analysis for dummies: experiencing LiSA. In: Proceedings of the 10th ACM SIGPLAN International Workshop on the State Of the Art in Program Analysis. p. 1–6. SOAP 2021, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3460946.3464316, `https://doi.org/10.1145/3460946.3464316`
13. Fürntratt, H., Schnabl, P., Krebs, F., Unterberger, R., Zeiner, H.: Towards higher abstraction levels in quantum computing. In: International Conference on Service-Oriented Computing. pp. 162–173. Springer (2023). https://doi.org/10.1007/978-981-97-0989-2_13, `https://doi.org/10.1007/978-981-97-0989-2_13`
14. Gottesman, D.: The heisenberg representation of quantum computers. arXiv preprint quant-ph/9807006 (1998)
15. Green, A.S., Lumsdaine, P.L., Ross, N.J., Selinger, P., Valiron, B.: An introduction to quantum programming in quipper. In: Reversible Computation: 5th International Conference, RC 2013, Victoria, BC, Canada, July 4-5, 2013. Proceedings 5. pp. 110–124. Springer (2013). https://doi.org/10.1007/978-3-642-38986-3_10, `https://doi.org/10.1007/978-3-642-38986-3_10`
16. Guo, J., Lou, H., Yu, J., Li, R., Fang, W., Liu, J., Long, P., Ying, S., Ying, M.: isq: An integrated software stack for quantum programming. IEEE Transactions on Quantum Engineering **4**, 1–16 (2023). https://doi.org/10.1109/TQE.2023.3275868
17. Heim, B., Soeken, M., Marshall, S., Granade, C., Roetteler, M., Geller, A., Troyer, M., Svore, K.: Quantum programming languages. Nature Reviews Physics **2**(12), 709–722 (2020). https://doi.org/10.1038/s42254-020-00245-7
18. Honda, K.: Analysis of quantum entanglement in quantum programs using stabilizer formalism. arXiv preprint arXiv:1511.01572 (2015)
19. JavadiAbhari, A., Patil, S., Kudrow, D., Heckey, J., Lvov, A., Chong, F.T., Martonosi, M.: Scaffcc: a framework for compilation and analysis of quantum computing programs. In: Proceedings of the 11th ACM Conference on Computing Frontiers. CF '14, Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2597917.2597939, `https://doi.org/10.1145/2597917.2597939`
20. Kaye, P., Laflamme, R., Mosca, M.: An introduction to quantum computing. OUP Oxford (2006)
21. Khedker, U.P., Sanyal, A., Sathe, B.: Data Flow Analysis: Theory and Practice. CRC Press, 1st edn. (2009). https://doi.org/10.1201/9780849332517, `https://doi.org/10.1201/9780849332517`
22. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press (2010)
23. O'Hearn, P.W.: Incorrectness logic. Proc. ACM Program. Lang. **4**(POPL) (Dec 2019). https://doi.org/10.1145/3371078, `https://doi.org/10.1145/3371078`
24. Paykin, J., Rand, R., Zdancewic, S.: Qwire: a core language for quantum circuits. SIGPLAN Not. **52**(1), 846–858 (Jan 2017). https://doi.org/10.1145/3093333.3009894, `https://doi.org/10.1145/3093333.3009894`
25. Peng, Y., Ying, M., Wu, X.: Algebraic reasoning of quantum programs via non-idempotent kleene algebra. In: Proceedings of the 43rd ACM SIGPLAN International Conference on Programming Language Design and Implementation. p. 657–670. PLDI 2022, Association for Computing Machinery, New York,

NY, USA (2022). https://doi.org/10.1145/3519939.3523713, https://doi.org/10.1145/3519939.3523713
26. Perdrix, S.: Quantum patterns and types for entanglement and separability. Electronic Notes in Theoretical Computer Science **170**, 125–138 (2007). https://doi.org/https://doi.org/10.1016/j.entcs.2006.12.015, https://www.sciencedirect.com/science/article/pii/S157106610700059X, proceedings of the 3rd International Workshop on Quantum Programming Languages (QPL 2005)
27. Perdrix, S.: Quantum entanglement analysis based on abstract interpretation. In: Alpuente, M., Vidal, G. (eds.) Static Analysis. pp. 270–282. Springer Berlin Heidelberg, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-69166-2_18
28. Quantinuum: guppylang (2024), https://github.com/CQCL/guppylang
29. Rand, R., Sundaram, A., Singhal, K., Lackey, B.: Extending gottesman types beyond the clifford group. In: The Second International Workshop on Programming Languages for Quantum Computing (PLanQC 2021) (2021)
30. Rand, R., Sundaram, A., Singhal, K., Lackey, B.: Gottesman types for quantum programs. arXiv preprint arXiv:2109.02197 (2021)
31. Seidel, R., Bock, S., Zander, R., Petrič, M., Steinmann, N., Tcholtchev, N., Hauswirth, M.: Qrisp: A framework for compilable high-level programming of gate-based quantum computers (2024), https://arxiv.org/abs/2406.14792
32. Seidl, H., Wilhelm, R., Hack, S.: Compiler Design: Analysis and Transformation. Springer (2012)
33. Vidal, G.: Entanglement monotones. Journal of Modern Optics **47**(2-3), 355–376 (2000). https://doi.org/10.1080/09500340008244048, https://www.tandfonline.com/doi/abs/10.1080/09500340008244048
34. Ward, M.: The closure operators of a lattice. Annals of Mathematics **43**(2), 191–196 (1942). https://doi.org/10.2307/1968865, http://www.jstor.org/stable/1968865
35. Winskel, G.: The formal semantics of programming languages: an introduction. MIT press (1993)
36. Ying, M.: Floyd–hoare logic for quantum programs. ACM Trans. Program. Lang. Syst. **33**(6) (Jan 2012). https://doi.org/10.1145/2049706.2049708, https://doi.org/10.1145/2049706.2049708
37. Ying, M.: Foundations of quantum programming. Morgan Kaufmann (2016)
38. Yu, N., Palsberg, J.: Quantum abstract interpretation. In: Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation. p. 542–558. PLDI 2021, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3453483.3454061, https://doi.org/10.1145/3453483.3454061
39. Yuan, C., McNally, C., Carbin, M.: Twist: sound reasoning for purity and entanglement in quantum programs. Proc. ACM Program. Lang. **6**(POPL) (Jan 2022). https://doi.org/10.1145/3498691, https://doi.org/10.1145/3498691