# A Branching Distributed Temporal Logic for Reasoning about Quantum State Transformations

Luca Viganò, Marco Volpe, Margherita Zorzi

# A Branching Distributed Temporal Logic for Reasoning about Quantum State Transformations

Luca Viganò[1]

*Department of Informatics, King's College London, UK*

Marco Volpe

*INRIA-Saclay, Palaiseau, France*

Margherita Zorzi

*Dipartimento di Informatica, Università di Verona, Italy*

**Abstract**

The Distributed Temporal Logic DTL allows one to reason about temporal properties of a distributed system from the local point of view of the system's agents, which are assumed to execute independently and to interact by means of event sharing. In this paper, we introduce the Quantum Branching Distributed Temporal Logic QBDTL, a variant of DTL able to represent quantum state transformations in an abstract, qualitative way. In QBDTL, each agent represents a distinct quantum bit (the unit of quantum information theory), which evolves by means of quantum transformations and possibly interacts with other agents, and $n$-ary quantum operators act as communication/synchronization points between agents. We endow QBDTL with a DTL-style semantics, which fits the intrinsically distributed nature of quantum computing, we formalize a labeled deduction system for QBDTL, and we prove the soundness and completeness of this deduction system with respect to the given semantics. We give a number of examples and, finally, we discuss possible extensions of our logic in order to reason about entanglement phenomena.

*Keywords:* Quantum Computing, Quantum State Transformations, Temporal Logic, Distributed Temporal Logic, Natural Deduction

## 1. Introduction

### *1.1. Background and motivation*

The *Distributed Temporal Logic DTL* [14, 5, 6] allows one to reason about temporal properties of a distributed system from the local point of view of the system's agents:

[1]The work presented in this paper was partially supported by the EU FP7 Marie Curie PIRSES-GA-2012-318986 project "GeTFun: Generalizing Truth-Functionality". Part of this work was carried out while Luca Viganò and Marco Volpe were at the Dipartimento di Informatica, Università di Verona, Italy.

each asynchronous agent executes independently, evolves linearly along a time-line built upon some local events, and can interact with the other agents by means of event sharing. Distribution is implicit and properties of an entire system are formulated in terms of the local properties of the system's agents and their interaction. DTL's semantics was inspired by a conflict-free version of Winskel's *event structures* (see, e.g., [35]), enriched with information about sequential agents.

DTL has been initially proposed as a logic for specifying and reasoning about distributed information [14], but it has also been used in the context of security protocol analysis to reason about the interplay between protocol models and security properties [6]. In this paper, we show that, after a proper extension of the logic's syntax and semantics, DTL is also able to formally model quantum state transformations in an abstract, qualitative way.

*Quantum computing* is one of the most promising research fields of computer science as well as a concrete future technology (see [27] for a useful introduction to the basic notions of quantum computing as we here only very briefly summarize the notions that are relevant to our work in this paper). However, at least from the point of view of theoretical computer science, a number of foundational aspects are still underdeveloped: quantum complexity, quantum computability, quantum programming theory (and its logical account), quantum cryptography and security are all active but open research areas, which still require the development of ad hoc formal methods. These issues are complex to face since the physical model quantum computing is based on is sophisticated and all basic definitions and formal tools have to be reformulated in a non-standard way.

To illustrate this, and our contributions in this paper, in more detail, let us focus our attention on quantum data, in particular on the unit of quantum information, the *quantum bit* or *qubit*, for short. The qubit is the quantum counterpart of the classical bit and, mathematically, it is simply a normalized vector of the Hilbert space $\mathbb{C}^2$. Qubits can assume both classical values 0 and 1 (as the classical bit) and all their *superpositional values*, i.e., linear combinations such as $\alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ are called *amplitudes*, $|\alpha|^2 + |\beta|^2 = 1$ and $|c\rangle$, for $c \in \{0, 1\}$, is the so called *Dirac Notation*, which is simply a denotation of basis states (which corresponds to the classical values a bit can assume).

Intuitively, whereas a classical bit can only be 0 or 1, a quantum bit can assume both the value 0 and the value 1 (with a certain associated probability) at the same time. It is possible to modify a quantum bit in two ways:

- by means of a suitable class of algebraic operators called *unitary transformations* (that are also called *quantum gates* and are a class of algebraic operators enjoying some good properties, which represent the pure quantum computational steps) or
- by *measuring* it, i.e., probabilistically reducing it to 0 or 1.

In this paper, we deal only with unitary transformations, leaving measurement for future work.

The definition of a qubit can, of course, be generalized: a *quantum register* or *quantum state* is the representation of a system of $n$ qubits (mathematically, it is a

normalized vector of the Hilbert space $\mathbb{C}^{2^n}$). As for the single qubit, a quantum state can be modified by means of unitary algebraic operators.

Abstracting from any notion of control and considering only pure quantum transformations (i.e., unitary evolution of quantum states as computational steps), it seems to be interesting to provide a logical account of such a computation. The question then is: what is a logical approach suitable to represent quantum state evolution?

### *1.2. Contributions*

The main contribution of this paper is the formalization of a logic and of an associated deduction system that allows one to formally represent and reason about unitary transformations of quantum states from a temporal multi-agent system perspective. More specifically, we view our contributions as two-fold.

First, we define the *Quantum Branching Distributed Temporal Logic* QBDTL, a significant variant of DTL that we introduce here to represent quantum state transformations in an abstract, *qualitative* way. In QBDTL, we abstract from the value of the qubits: we are not interested in encoding into our system syntactical and semantical information about amplitudes or basis values 0 and 1 (in this way, we avoid any *quantitative* information) and we focus instead on the way qubits evolve by means of unitary transformations. Following DTL's central notion, in QBDTL we do not only consider globally quantum states but also, and in particular, the single unit of information, i.e., we maintain the local perspective of the qubit in the quantum computation.

In other words, in QBDTL each agent represents a distinct qubit, which is the object/subject of computation and which evolves in time by means of quantum transformations and possibly interacts with other agents/qubits.

There is a crucial difference between our QBDTL and the original DTL formulation. DTL is based on linear time life-cycles for agents. In QBDTL (and this provides an additional contribution of our work), we go beyond linearity and consider branching time since we want to be as general as possible: at each step of the temporal evolution of an agent/qubit, the accessibility relation between worlds in the subtended Kripke-style model aims to capture each possible unitary transformation that can be applied to the qubit. A world (a state in the temporal life-cycle of an agent) represents (an abstraction of) a 1-qubit quantum state. $n$-ary quantum operators, which act simultaneously on more than one qubit (such as control operators, which play a crucial role in quantum computing), act as communication/synchronization points between agents/qubits.

Second, we give a deduction system $\mathcal{N}$(QBDTL) for QBDTL. In order to deal with all the semantical notions (temporal, quantum and synchronization information), we follow the style of *labeled deduction* [19, 32, 33], a framework for giving uniform presentations of different non-classical logics, where labels allow one to explicitly encode in the syntax additional information, of a semantic or proof-theoretical nature, that is otherwise implicit in the logic one wants to capture.

In addition to the works on DTL, and in particular the labeled tableaux system given in [5], our starting points for $\mathcal{N}$(QBDTL) are the labeled natural deduction system for the logic *UB* (i.e., the until-free fragment of *CTL*) given in [11] and the approach developed in [23, 24], where a labeled modal deduction system with specific modalities able to describe quantum state transformations is given. Fittingly, in $\mathcal{N}$(QBDTL), we

consider composed labels $(i, x, q)$ that represent an agent/qubit $i$, a time instant $x$, and the quantum information $q$ in the underlying semantics. A further class of labels is used to represent paths in the life-cycles of the agents.

The rules of $\mathcal{N}(\mathsf{QBDTL})$ can then be divided into rules that formalize the *local* temporal evolution of an agent/qubit, and synchronization rules that are, in a sense, *global* as they lift the reasoning from the local perspective of the agent to the *distributed* perspective induced by agent's synchronizations.

### *1.3. Related Work*

It is important to observe that our QBDTL is not a quantum logic. Since the work of Birkhoff and von Neumann [10], various logics have been investigated as a means to formalize reasoning about propositions taking into account the principles of quantum theory, e.g., [13]. In general, it is possible to view quantum logic as a logical axiomatization of quantum theory, which provides an adequate foundation for a theory of reversible quantum processes, e.g., [25, 1, 2, 3, 4, 17, 18]. Research has focused also on automated reasoning (e.g., model checking for quantum systems as considered in [20]) and on formal analysis of quantum protocols (e.g., [22]). Our work moves from quite a different point of view, which, to reiterate, is the wish to provide a deduction system able to represent and reason about unitary transformations of quantum states from a temporal multi-agent system perspective and, as will become clear below, thereby provide a basis to reason about other, more complex properties of quantum states such as entanglement.

This paper extends and supersedes our preliminary account of QBDTL given in [34].

### *1.4. Organization*

We proceed as follows. In Section 2, we give a brief overview of the basic notions of quantum computing that are relevant for this paper. Then, after a discussion about aims and motivations of our approach (Section 3), in Section 4 we introduce the logic QBDTL and a DTL-style semantics, along with some examples. In Section 5 we define the natural deduction system $\mathcal{N}(\mathsf{QBDTL})$, providing some example derivations. In Section 6 and Section 7 we state and prove the Soundness Theorem and the Completeness Theorem (of $\mathcal{N}(\mathsf{QBDTL})$ with respect to the semantics), respectively. Section 8 is devoted to discussions about our ongoing and future works.

## 2. Quantum Computing in a Nutshell

The aim of this section is to expand on what we already discussed in the introduction and provide to the non-expert reader an overview of the basic notions of quantum computing, focusing on those that are relevant for this paper and our approach in general (in particular, the quantum meaning of unitary relations and the interpretation of propositional symbols, but also what future challenges lie ahead of us, such as the formalization of entanglement). Readers familiar with quantum computing are welcome to skip this section.

The most simple quantum system is a two-dimensional state space whose elements are called *quantum bits* or *qubits* for short. The qubit is thus the unit of quantum

information. The most direct way to represent a quantum bit is as a unitary vector in the 2-dimensional Hilbert space $\ell^2(\{0,1\})$, which is isomorphic to $\mathbb{C}^2$. We denote with $|0\rangle$ and $|1\rangle$ the elements of the computational basis of $\ell^2(\{0,1\})$. The states $|0\rangle$ and $|1\rangle$ of a qubit correspond to the Boolean constants 0 and 1, which are the only possible values of a classical bit.

A qubit, however, can assume other values, different from $|0\rangle$ and $|1\rangle$. In fact, every linear combination $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$, represents a possible qubit state. These states are said to be *superposed*, and the two values $\alpha$ and $\beta$ are called *amplitudes*. The amplitudes $\alpha$ and $\beta$ univocally represent the qubit with respect to the computational basis. Given a qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, we can thus also denote it by the vectorial notation $(\alpha\ \beta)^\top$, where $\top$ denotes the transposition of the vector. In particular, the vectorial representation of the elements of the computational basis $|0\rangle$ and $|1\rangle$ is: $(1\ 0)^\top$ represents $|0\rangle$ and $(0\ 1)^\top$ represents $|1\rangle$.

While we can determine the state of a classical bit, for a qubit we cannot establish with the same precision the values $\alpha$ and $\beta$: quantum mechanics says that a measurement of a qubit with state $\alpha|0\rangle + \beta|1\rangle$ has the effect of changing the state to $|0\rangle$ with probability $|\alpha|^2$ and to $|1\rangle$ with probability $|\beta|^2$. For example, if $|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$, one can fairly observe 0 or 1 with the same probability $|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$. Note, however, that in this brief survey on quantum computing, we will not enter into the details about the measurement/observation of the qubit(s), since in QBDTL we do not model agent evolution by measurement of quantum states; for a complete overview of measurement of qubits and the relationships between different kinds of measurement, see [27].

In order to define arbitrary sets of quantum data, we need a generalization of the notion of qubit, called *quantum register* or, more commonly, *quantum state* [31, 28, 12]. A quantum register can be viewed as a system of $n$ qubits and, mathematically, it is a normalized vector in the Hilbert space $\ell^2(\{0,1\}^n)$, where $\{0,1\}^n$ is a compact notation to represent any binary sequence of length $n$. The standard computational basis for $\ell^2(\{0,1\}^n)$ is $\mathcal{B} = \{|i\rangle \mid i \text{ is a binary string of length } n\}$.

As notation, for $b_i \in \{0,1\}$, we write $|b_1 \dots b_k\rangle$ for $|b_1\rangle \otimes \dots \otimes |b_k\rangle$, where $\otimes$ is the tensor product (see below). With a little abuse of language, we say that the number of qubits $n$ corresponds to the dimension of the space. Note that if the dimension is $n$, then the basis $\mathcal{B}$ contains $2^n$ elements, and each quantum state is a normalized linear combination of these elements:

$$\alpha_1 \underbrace{|00\dots0\rangle}_{n} + \alpha_2|00\dots1\rangle + \dots + \alpha_{2^n}|11\dots1\rangle$$

**Example 1.** *Let us consider a system of two qubits. Each 2-qubit quantum register is a normalized vector in* $\ell^2(\{0,1\}^2)$ *and the computational basis is* $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. *For example,* $\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{4}}|01\rangle + \frac{1}{\sqrt{8}}|10\rangle + \frac{1}{\sqrt{8}}|11\rangle$ *is a quantum register of two qubits and we can represent it as* $(\frac{1}{\sqrt{2}}\ \frac{1}{\sqrt{4}}\ \frac{1}{\sqrt{8}}\ \frac{1}{\sqrt{8}})^\top$.

A Hilbert space of dimension $n$ can be built from smaller Hilbert spaces by means of the *tensor product* $\otimes$. If $\mathcal{H}_1$ is a Hilbert space of dimension $k$ and $\mathcal{H}_2$ is a Hilbert space of dimension $m$, then $\mathcal{H}_3 = \mathcal{H}_1 \otimes \mathcal{H}_2$ is a Hilbert space of dimension $km$ (each element is a vector of $km$ coordinates obtained by "hooking" a vector in $\mathcal{H}_2$ to a vector

in $\mathcal{H}_1$). In other words, an $n$-qubit quantum register with $n \geq 2$ can be viewed as a composite system and, in general, it is possible to combine two (or more) distinct physical systems into a composite one. If the first system is in the state $|\phi_1\rangle$ (a vector in a Hilbert space $\mathcal{H}_1$) and the second system is in the state $|\phi_2\rangle$ (a vector in a Hilbert space $\mathcal{H}_2$), then the state of the combined system is $|\phi_1\rangle \otimes |\phi_2\rangle$ (a vector in a Hilbert space $\mathcal{H}_1 \otimes \mathcal{H}_2$). We will often omit the $\otimes$ symbol and write the joint state as $|\psi_1\rangle|\psi_2\rangle$ or as $|\psi_1\psi_2\rangle$.

Not all quantum states can be viewed as composite systems: this case occurs in presence of entanglement phenomena (to which we return below). Since normalized vectors of quantum data represent physical systems, the (discrete) evolution of systems can be viewed as a suitable transformation on Hilbert spaces. The evolution of a quantum register is *linear* and *unitary*. Giving an initial state $|\psi_1\rangle$, for each evolution to a state $|\psi_2\rangle$, there exists a *unitary* operator $U$ such that $|\psi_2\rangle = U(|\psi_1\rangle)$. Informally, "unitary" referred to an algebraic operator on a suitable space means that the normalization constraint of the amplitudes ($\sum_i |\alpha_i|^2 = 1$) is preserved during the transformation. Thus, a quantum physical system can be described in terms of linear operators and in a *deterministic* way.

In quantum computing, we refer to a unitary operator $U$ acting on a $n$-qubit quantum register as an $n$-qubit *quantum gate*. One can represent operators on the $2^n$-dimensional Hilbert space $\ell^2(\{0,1\}^n)$ with respect to the standard basis of $\mathbb{C}^{2^n}$ also as $2^n \times 2^n$ matrices, and it is possible to prove that to each unitary operator on a Hilbert space we can associate unequivocally an algebraic representation. Matrices that represent unitary operators enjoy some important properties: for example, they are easily invertible (*invertibility*, also called reversibility, is one of the peculiar features of quantum computing). The application of quantum gates to quantum registers thus represents the pure quantum computational step and captures the internal evolution of quantum systems.

From a computer science viewpoint, it is common to reason about quantum state transformations in terms of *quantum circuits* (see, e.g., [27, 26]). We have introduced qubits to store quantum information, in analogy with the classical case. We have also introduced operations acting on them, i.e., quantum gates, and we can think about quantum gates in analogy with gates in classical logic circuits. A quantum circuit on $n$ qubits implements a unitary operator on a Hilbert space of dimension $\mathbb{C}^{2^n}$ as a primitive collection of quantum gates, each implementing a unitary operator on $k$ (small) qubits. It is useful to represent quantum circuits graphically in terms of sequential and parallel composition of quantum gates and wires, as for Boolean circuits (still, in the quantum case, the graphical representation does not reflect the physical realization of the circuit).

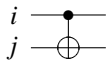
The most simple quantum gates act on a single qubit: they are operators on the space $\ell^2(\{0,1\})$, represented in $\mathbb{C}^2$ by $2 \times 2$ complex matrices. For example, the quantum gate $X$ is the unitary operator that represents the quantum counterpart of the *complementation gate*. Being a linear operator, $X$ maps a linear combination of inputs to the corresponding linear combination of outputs, i.e., $X$ maps the general qubit state $\alpha|0\rangle + \beta|1\rangle$ into the state $\alpha|1\rangle + \beta|0\rangle$.

Another fundamental unitary gate is the *Hadamard gate* $H$, which acts on the computational basis in the following way: $|0\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|1\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

The Hadamard gate is useful when we want to create a superposition starting from a classical state. It also holds that $H(H(|c\rangle)) = |c\rangle$ for $c = \{0, 1\}$.

1-qubit quantum gates can be used to build gates acting on $n$-qubit quantum states. If we have a 2-qubit quantum system, we can apply a 1-qubit quantum gate only to one component of the system, and we implicitly apply the identity operator (the identity matrix) $I$ to the other one. For example, suppose we want to apply $X$ to the first qubit. The 2-qubits input $|\psi_1\rangle \otimes |\psi_2\rangle$ gets mapped to $X|\psi_1\rangle \otimes I|\psi_2\rangle = (X \otimes I)|\psi_1\rangle \otimes |\psi_2\rangle$.

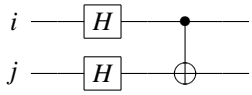
The *controlled-not (cnot)* is one of the most important quantum operators, usually represented graphically in the following way:




Intuitively, it takes two distinct quantum bits $i$ and $j$ as input and complements the *target* bit (the second one, $j$) if the *control* bit (the first one, $i$) is different from 0, and does not perform any action otherwise. Hence, its behavior on the computational basis is:

$$cnot|00\rangle = |00\rangle \quad cnot|10\rangle = |11\rangle$$
$$cnot|01\rangle = |01\rangle \quad cnot|11\rangle = |10\rangle$$

We can of course apply the cnot to superpositional states. For instance, if we apply cnot to the outcome of Hadamard gates applied to combinations of basis states $|0\rangle$ and $|1\rangle$, i.e.




then we have:

$$cnot(H(|0\rangle), H(|0\rangle)) = \tfrac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$
$$cnot(H(|0\rangle), H(|1\rangle)) = \tfrac{1}{2}(|00\rangle - |01\rangle - |10\rangle + |11\rangle)$$
$$cnot(H(|1\rangle), H(|0\rangle)) = \tfrac{1}{2}(|00\rangle + |01\rangle - |11\rangle - |10\rangle)$$
$$cnot(H(|1\rangle), H(|1\rangle)) = \tfrac{1}{2}(|00\rangle - |01\rangle - |11\rangle + |10\rangle)$$

Note that the control qubit is a "master" agent as its evolution is independent of the evolution of the target bit (if the first input of the cnot is $|\phi\rangle$, then the output is the same), whereas the target qubit is a "slave" agent as its evolution is controlled by the value of the first qubit. In some sense, a communication between the agents is required and the quantum circuit is a simple distributed system. By adopting this perspective, controlled operators like cnot act as "synchronization points" between agents, and this is indeed one of the main ideas we followed during the development of QBDTL.

Not all quantum states can be viewed as composite systems. In other words, if $|\psi\rangle$ is a state of a tensor product space $\mathcal{H}_1 \otimes \mathcal{H}_2$, it is not generally true that there exist $|\psi_1\rangle \in \mathcal{H}_1$ and $|\psi_2\rangle \in \mathcal{H}_2$ such that $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$. In general, it is not always possible to decompose an $n$-qubit register as the tensorial product of $n$ qubits. These non-decomposable registers are called *entangled* and enjoy properties that we cannot

find in any object of classical physics (and therefore in classical data). If $n$ qubits are entangled, they behave *as if connected*, independently of the real physical distance. The strength of quantum computation is essentially based on the existence of entangled states (see, for example, the *teleportation protocol* [27]).

**Example 2.** *The 2-qubit states* $|\psi_1\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$ *and* $|\psi_2\rangle = \frac{1}{\sqrt{2}}|01\rangle + \frac{1}{\sqrt{2}}|10\rangle$ *are entangled. The 2-qubit state* $|\phi\rangle = \alpha|00\rangle + \beta|01\rangle$ *is not entangled; rather, note that it is possible to rewrite it in the mathematically equivalent form* $|\phi\rangle = |0\rangle \otimes (\alpha|0\rangle + \beta|1\rangle)$.

A simple way to create an entangled state is to feed a cnot gate with a target qubit $|c\rangle$ and a particular control qubit, more precisely the output of the Hadamard gate applied to a base qubit, therefore a superposition $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ or $\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$. If, instead, we fed a cnot gate with two fair superpositional states (as above), that would not create an entangled state.

As we remarked above, in this paper we assume that no entanglement phenomena occur during the computation. Still, we have discussed entanglement here as it is a fundamental feature of quantum states and it has been one of the motivations for our approach in the first place. We believe that QBDTL will provide a suitable logic to formalize entanglement and we plan to explicitly model it in the future developments of our work (see also the discussion in Section 8).

## 3. Why Branching Temporal Logic and Synchronization?

In this section, we describe how it is possible to use temporal logic and synchronization rules (the core of the DTL approach) to reason in a simple way about quantum state transformations, whenever one is not interested in the encoding of the mathematical object that represents a quantum state (i.e., a vector in a suitable Hilbert space) but in the evolution itself as a sequence of transformations and in a notion of synchronization between different quantum bits.

Modal logics are a flexible instrument to describe qualitatively state transformations as they allow one to put the emphasis on the underlying "transition system" (the set of possible worlds of the Kripke semantics and the properties of the accessibility relations between them, which model the dynamical behavior of the system) rather than on the concrete meaning of the internal structures of possible worlds. This intuition was followed in [23, 24], where two pure modal systems were introduced and studied. In such systems, a world represents the abstraction of a quantum state and modal operators reflect general properties of quantum state transformations, since the subtended models are *S5*-models. The accessibility relation between worlds is therefore an equivalence relation, i.e., it enjoys reflexivity, symmetry and transitivity. This captures, in an abstract way, key properties of unary quantum operators: roughly speaking, reflexivity says that the class of the unitary operators includes the identity transformation; symmetry captures reversibility (it is always possible to reverse a quantum transformation, since the inverse operator is easily definable and is unitary); finally, transitivity models algebraic compositionality, i.e., the composition of two or more unitary operators is always a unitary operator [27].

The main difference between the modal systems proposed in [23, 24] and QBDTL is that whereas in the former case a world represents the abstraction of an *arbitrary* quantum state (i.e., a state that describes an arbitrary number $n$ of qubits), in the case of QBDTL we focus on the single qubit and on its transformation by means of *unary* quantum operators and on a notion of local formula built upon a local language. Moreover, we move from a modal to a temporal system: in some sense we "unfold" the accessibility relation between worlds obtaining, for each agent, a tree-like structure that represents the agent's local life-cycle. In this way, we "link" the subtended branching temporal model to the abstract transition system induced by all the unary quantum transformations possibly occurring in each world, which are uniformly modeled in the semantics and in the deduction system by an equivalence relation. Reflexivity, symmetry and transitivity can be plainly expressed in QBDTL: for example, symmetry can be abstractly captured by the labeled formula $(i, x, q) : p \supset \exists\bigcirc\exists\bigcirc p$, where $p$ is a propositional symbol, $\supset$ is implication and $\exists\bigcirc A$ expresses that the formula $A$ is true at the next time instant in some possible future.

A licit question at this stage is what is the meaning of the set of propositional symbols that QBDTL formulas are built upon. We maintain an abstract definition of the set (we simply say that it is a set of syntactic objects), following the style of DTL and also in the spirit of modal/temporal logic as we discussed above. Then, working with labeled expressions like $(i, x, q) : A$, where the formula $A$ is built by temporal operators, synchronization and propositional symbols, it is not actually crucial to say what propositional symbols stand for.[2] Still, it is important to consider what modal/temporal formulas, possible worlds and the accessibility relation stand for.

One could even choose to instantiate the set of propositional symbols to capture quantitative information about quantum states or general properties that permit one to reason about them. Let us provide here a simple example, related to the examples that we will give later. A possible choice (cf. Example 3) is to fix a set of atomic propositions representing mathematical descriptions of the qubit, i.e., a normalized vector in $\mathbb{C}^2$. In other words, given a qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, the *encoding* $\ulcorner|\psi\rangle\urcorner$ of this mathematical description is an atomic proposition. Let $s_i$ stand for a label $(i, x, q)$, take $p$ as $\ulcorner|\psi\rangle\urcorner$ and consider the labeled formula $s_i : p \supset \exists\Box p$ (whose derivation will be given in Figure 5 and where $\exists\Box p$ expresses that $p$ is true at every time instant in some possible future). This labeled formula can be intuitively interpreted as follows: a (potentially infinite) sequence of identity unitary transformations does not change the mathematical description of the qubit.

Now, let $p$ still be the encoding $\ulcorner|\psi\rangle\urcorner$ of a state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and let us consider again the labeled formula $(i, x, q) : p \supset \exists\bigcirc\exists\bigcirc p$, which fits a peculiar feature of quantum computation, i.e., reversibility. This labeled formula says that: if $p$ holds for $i$ in some state $x$, then there exists a temporal path such that, in two steps, $i$ reaches a new state in which $p$ still holds (i.e., the mathematical description of such a state is again $\alpha|0\rangle + \beta|1\rangle$). This models the fact that if one transforms a qubit state by means of a unitary operator $U$, then one can obtain again the same state by applying the adjoint

[2] In analogy, note, e.g., that temporal logics developed to deal with concurrent systems do not possess any concurrent feature.

$U^*$ of $U$, where, in the class of unitary operators, the adjoint corresponds to the inverse $U^{-1}$, and algebraically, one has $U^*(U(\alpha|0\rangle + \beta|1\rangle)) = U(U^*(\alpha|0\rangle + \beta|1\rangle)) = \alpha|0\rangle + \beta|1\rangle$, i.e., $U^*U = UU^* = I$, where $I$ is the identity operator. Looking for a concrete example, we can take $\alpha = \frac{1}{\sqrt{3}}$ and $\beta = \frac{\sqrt{2}}{\sqrt{3}}$ and instantiate $U$ to $X$, the complementation gate, which corresponds to an exchange between amplitudes of basis states. Among the temporal states reachable from $x$ there exists, in particular, the successor state in which $\overline{p}$ and $\exists\bigcirc p$ hold, where $\overline{p} = \ulcorner \frac{\sqrt{2}}{\sqrt{3}}|0\rangle + \frac{1}{\sqrt{3}}|1\rangle \urcorner$. We will provide further examples in the following sections.

We conclude this section by recalling that in quantum computing, it is useful to compose small states in order to obtain bigger quantum states (this operation has a precise algebraic meaning, see the previous section and [27]). Collecting agents, one can model quantum systems of $n$ qubits. In some sense, we can see a quantum state of $n$ qubits as a global state built upon the local states of the single qubits. Each qubit evolves independently but, in a realistic perspective, different qubits do not always evolve asynchronously, and so sometimes they interact, by means of $n$-ary quantum gates. This is modeled, in our system, by means of ad hoc "tools", properly adapted from DTL: by a special construct in the local language (an operator © named *calling*), it is possible to express the fact that an agent/qubit $i$ synchronizes with another agent/qubit $j$. This choice has a precise quantum meaning. In quantum computing, one can, of course, globally modify a set of $n$ qubits by means of $n$-ary algebraic operators. We view n-ary quantum gates as *synchronization* points between states of different lifecycles, i.e., between states of different qubits. The inputs of an n-ary quantum gate may each have previously been subject to a sequence of other transformations, i.e., in DTL terms, a sequence of events, and the gate itself then can be seen as a transformation event that is shared by the inputs. In this paper, we model this synchronization mechanism abstractly (since, as we said, we model unitary transformations by an equivalence relation), but it is possible to plan a concrete research direction based on the further development of this interpretation of $n$-ary gates as synchronization mechanisms. We will return to this in Section 8, where we give a more detailed discussion of our ongoing and future works.

## 4. The logic QBDTL

We introduce the Quantum Branching Distributed Temporal Logic QBDTL by presenting its syntax and semantics. As we mentioned above, QBDTL is a significant variant of DTL, with which it shares, of course, a number of similarities, but with respect to which it also sports several differences. For instance, we do not consider the sets of actions of the agents and, most importantly, in order to reason about quantum state transformations, QBDTL considers branching time as opposed to the linear time typical of DTL.

### *4.1. Syntax*

**Definition 1.** *Given a finite set* $\mathsf{Id} = \{i, j, \ldots\}$ *of* agent identifiers *and a set* $\mathsf{Prop} = \{p, p_1, p_2, \ldots\}$ *of* atomic propositions *(which characterize the current local states of*

*the agents), we define the* local language *of an agent* $i \in \mathsf{Id}$ *by the grammar*

$$\mathcal{L}_i ::= p \mid \bot \mid \mathcal{L}_i \supset \mathcal{L}_i \mid \exists\bigcirc\mathcal{L}_i \mid \exists\Box\mathcal{L}_i \mid \forall\Box\mathcal{L}_i \mid ©_j\,\mathcal{L}_j\,,$$

*where* $p \in \mathsf{Prop}$ *and* $j \in \mathsf{Id}$ *with* $i \neq j$.

*Local formulas*, as their names suggest, hold locally for the different agents. $\bot$ is *falsum* and $\supset$ is implication. As in DTL, the formula $©_j\,A$ means that agent $i$ has just communicated (i.e., synchronized) with agent $j$, for whom $A$ holds. We follow here the Peircean branching temporal logic *UB* [8] and only consider the temporal operators that are obtained as a combination of one single linear-time operator immediately preceded by one single path quantifier. More specifically, we consider here the Peircean operators

- $\exists\bigcirc$ (as we noted previously, $\exists\bigcirc A$ expresses that the formula $A$ in the scope of this operator is true at the next time instant in some possible future),
- $\exists\Box$ ("it is true at every time instant in some possible future") and
- $\forall\Box$ ("it is true at every time instant in every possible future").

For simplicity, in this work we do not consider the temporal operator until, although such an extension would not be problematic. Moreover, as usual, other connectives and temporal operators can be defined as abbreviations and will sometimes be explicitly used in the following.

**Definition 2.** *The* global language *of* QBDTL *is defined by the grammar*

$$\mathcal{L} ::= @_{i_1}\,\mathcal{L}_{i_1} \mid \ldots \mid @_{i_n}\,\mathcal{L}_{i_n}\,,$$

*where* $i_1, \ldots, i_n \in \mathsf{Id}$.

The *global formula* $@_{i_k}\,A$ means that $A$ holds for agent $i_k$.

### *4.2. Semantics*

The models of QBDTL are inspired by those of DTL and built upon a form of Winskel's event structures (cf. [35], where also the relationship to other concurrency models is discussed). There is, however, a fundamental difference with respect to the semantics that has (actually, with respect to the slightly different semantics that in the literature have) been given for DTL, which is based on distributed families of linear life-cycles local to each agent, i.e., countable, discrete and totally ordered local events. Since our logic QBDTL is inherently branching, we need to define its semantics accordingly, and we thus modify DTL's semantics as follows.

**Definition 3.** *Given an agent* $i \in \mathsf{Id}$*, a* branching local life-cycle *of* $i$ *is an* $\omega$*-tree, i.e., a pair* $\lambda_i = \langle \mathsf{Ev}_i, <_i \rangle$*, where* $\mathsf{Ev}_i$ *is the set of* local events of $i$ *and* $<_i \subseteq \mathsf{Ev}_i \times \mathsf{Ev}_i$ *is a binary relation such that:*

(i) $<_i$ *is transitive and irreflexive;*

(ii) *for each* $e \in \mathsf{Ev}_i$*, the set* $\{e' \in \mathsf{Ev}_i \mid e' <_i e\}$ *is linearly ordered by* $<_i$*;*

(iii) *there is a* $<_i$*-smallest element* $0_i$ *called the* root *of* $\lambda_i$*;*

(iv) *each maximal linearly* $<_i$*-ordered subset of* $\mathsf{Ev}_i$ *is order-isomorphic to the natural numbers.*

*We write* $e \rightarrow_i e'$ *to denote the fact that* $e'$ *is an immediate local successor of* $e$*, i.e.,* $e <_i e'$ *and there is no* $e''$ *such that* $e <_i e'' <_i e'$*. A* $\rightarrow_i$-path *is a sequence of local events* $(e_0, \ldots, e_n)$ *such that* $e_k \rightarrow_i e_{k+1}$ *for* $0 \leq k \leq n-1$*. An* $e$-branch $b$ of $i$ *is an infinite* $\rightarrow_i$*-path* $b = (e_0, e_1, \ldots)$ *such that* $e = e_0$ *and we write* $\rightarrow_i^b$ *to denote the restriction of* $\rightarrow_i$ *to* $b$*, i.e.,* $e' \rightarrow_i^b e''$ *iff* $e' = e_k$ *and* $e'' = e_{k+1}$ *for some* $k$*, and denote with* $\mathcal{B}_i$ *the set of all such* $\rightarrow_i^b$*. Further, we denote with* $\rightarrow_i^{b*}$ *the reflexive and transitive closure of* $\rightarrow_i^b$*.*

*A* local state *is a finite set* $\xi \in \mathsf{Ev}_i$ *down-closed for local causality, i.e., if* $e <_i e'$ *and* $e' \in \xi$ *then also* $e \in \xi$*. In general, each non-empty local state* $\xi$ *is reached by the occurrence of an event that we call* $last(\xi)$*, from the local state* $\xi \setminus \{last(\xi)\}$*. Given* $e \in \mathsf{Ev}_i$*, the set* $e{\downarrow}i = \{e' \in \mathsf{Ev}_i \mid e' \leq_i e\}$*, where* $\leq_i$ *denotes the reflexive closure of* $<_i$*, is always a local state. Moreover, if* $\xi$ *is non-empty, then* $last(\xi){\downarrow}i = \xi$*.*

*A* branching distributed life-cycle *is a family of local life-cycles*

$$\lambda = \{\lambda_i = \langle \mathsf{Ev}_i, <_i \rangle\}_{i \in \mathsf{Id}}$$

*such that:*

(i) $\leq = (\bigcup_{i \in \mathsf{Id}} \leq_i)^*$ *defines a partial order of* global causality *on the set of events* $\mathsf{Ev} = \bigcup_{i \in \mathsf{Id}} \mathsf{Ev}_i$*;*

(ii) *if* $e, e' \in \mathsf{Ev}_i$ *and* $e \leq e'$ *then* $e \leq_i e'$*.*

Condition (i) ensures that a distributed life-cycle respects global compatibility, i.e., there is no $e \in \mathsf{Ev}_i \cap \mathsf{Ev}_j$ such that $e <_i e'$ but $e' <_j e$, while condition (ii) ensures that synchronization $\leq$-relates two events of an agent $i$ only if there exists a $0_i$-branch in which both the events occur.

**Definition 4.** *An S5* Kripke frame *is a pair* $\langle Q, \mathcal{U} \rangle$*, where* $Q$ *is a non-empty set of* qubit states *and* $\mathcal{U}$ *is a binary equivalence relation on* $Q$*, i.e.,* $\mathcal{U} : Q \to Q$ *is reflexive, symmetric and transitive.*

*An S5* Kripke model *is a triple* $\mathcal{M} = \langle Q, \mathcal{U}, \mathcal{V} \rangle$*, where* $\langle Q, \mathcal{U} \rangle$ *is an S5 Kripke frame and* $\mathcal{V} : Q \to \mathcal{P}(\mathsf{Prop})$ *is a valuation function assigning to each qubit state in* $Q$ *a set of atomic propositions.*

*A* QBDTL model *is a triple* $\mu = \langle \lambda, \mathcal{M}, \pi \rangle$*, where* $\lambda = \{\lambda_i\}_{i \in \mathsf{Id}}$ *is a distributed life-cycle,* $\mathcal{M} = \langle Q, \mathcal{U}, \mathcal{V} \rangle$ *is an S5 Kripke model and* $\pi = \{\pi_i\}_{i \in \mathsf{Id}}$ *is a family of local functions associating to each local state a qubit state in* $Q$*; for each* $i \in \mathsf{Id}$ *and set* $\Xi_i$ *of local states of* $i$*, the function* $\pi_i : \Xi_i \to Q$ *is such that:*

(i) *if* $\xi, \xi' \in \Xi_i$*,* $last(\xi) \rightarrow_i last(\xi')$*,* $\pi(\xi) = q$ *and* $\pi(\xi') = q'$*, then* $q \mathcal{U} q'$*;*

(ii) *if* $q, q' \in Q$*,* $q \mathcal{U} q'$ *and* $\pi(\xi) = q$*, then there exists* $\xi' \in \Xi_i$ *such that* $last(\xi) \rightarrow_i last(\xi')$ *and* $\pi(\xi') = q'$*.*

*We denote* $\langle \lambda_i, \mathcal{M}, \pi_i \rangle$ *by* $\mu_i$.

*The* global satisfaction relation *is defined by:*

$$\models^{\mu} @_i A \ \textit{iff} \ \models_i^{\mu_i} A \ \textit{iff} \ \models_i^{\mu_i,\xi} A \textit{ for every } \xi \in \Xi_i \,,$$

*where the* local satisfaction relation at a local state $\xi$ *of* $i$ *is defined by:*

$$\begin{array}{lll}
\not\models_i^{\mu_i,\xi} \bot & & \\
\models_i^{\mu_i,\xi} p & \textit{iff} & p \in \mathcal{V}(\pi_i(\xi)), \textit{ for } p \in \mathsf{Prop} \\
\models_i^{\mu_i,\xi} A \supset B & \textit{iff} & \models_i^{\mu_i,\xi} A \textit{ implies } \models_i^{\mu_i,\xi} B \\
\models_i^{\mu_i,\xi} \forall\Box A & \textit{iff} & \textit{for all } \xi', \textit{last}(\xi) \leq_i \textit{last}(\xi') \textit{ implies } \models_i^{\mu_i,\xi'} A \\
\models_i^{\mu_i,\xi} \exists\Box A & \textit{iff} & \textit{there exists a last}(\xi)\textit{-branch } b \textit{ such that for all } \xi', \\
 & & \textit{last}(\xi) \rightarrow_i^{b*} \textit{last}(\xi') \textit{ implies } \models_i^{\mu_i,\xi'} A \\
\models_i^{\mu_i,\xi} \exists\bigcirc A & \textit{iff} & \textit{there exists } \xi' \textit{ such that last}(\xi) \rightarrow_i \textit{last}(\xi') \textit{ and } \models_i^{\mu_i,\xi'} A \\
\models_i^{\mu_i,\xi} \copyright_j A & \textit{iff} & \textit{last}(\xi) \in \mathsf{Ev}_j \textit{ and } \models_j^{\mu_j, last(\xi)\downarrow j} A
\end{array}$$

*By extension, we define:*

$$\begin{array}{lll}
\models^{\mu} \Gamma & \textit{iff} & \models^{\mu} A \textit{ for all } A \in \Gamma \\
\Gamma \models^{\mu} A & \textit{iff} & \models^{\mu} \Gamma \textit{ implies } \models^{\mu} A \\
\Gamma \models A & \textit{iff} & \Gamma \models^{\mu} A \textit{ for each QBDTL model } \mu
\end{array}$$

### *4.3. An example of QBDTL "in action"*

Now that we have introduced the syntax and semantics of QBDTL, we can give a first example of QBDTL in action, i.e., how it can be used to formalize reasoning about quantum computations in a qualitative way. As pointed out in [6], one of the main features of distributed temporal logics is their versatility: their abstract and parametric formulation permits a number of heterogeneous "specializations" to capture different agent-oriented systems and applications. As a first concrete example, we follow the ideas sketched in Section 3 and propose a concrete instance of the set Prop of propositional symbols in order to provide a description of the behavior of a simple quantum circuit.

We begin by introducing some notation and assumptions about quantum computations that our examples in this paper rely on. We focus on a *universal class* of quantum gates: we take into consideration all the unary gates plus the cnot. We call $\mathcal{G}$ this class of unitary operators. Given an instance $C$ of the cnot gate, we write $C^{\langle i \rangle}$ to denote the *restriction of the "action"* (in terms of transformations) of $C$ on the qubit $i$. Note that we are not affirming that the transformations of the two agents (by means of the cnot) are independent of each other (indeed, this would negate the definition of the cnot, at least from the point of view of the target qubit). Rather, we are only observing that, after the control, two distinct unary transformations occur: the identity transformation leaves unchanged the control agent, whereas, the induced operation (eventually) complements the target agent. Following this perspective and assuming (as we do in this paper) that no entanglement phenomena occur, we can then "factorize" global transformations into local transformations and, as a consequence, we can plainly reason about global events in terms of local events of the local life-cycles of the agents.

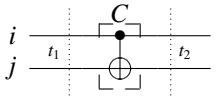
**Example 3 (QBDTL "in action" — I).** *The first step of any concrete example aiming at showing QBDTL in action is to fix the set* Prop*. Let us consider the standard algebraic axiomatization of quantum mechanics and the Hilbert space formalism [30]. As we remarked above, the mathematical state of a single qubit is represented by an element (a normalized vector) of* $\mathbb{C}^2$*, the* 1*-dimensional complex space. Let us call* $\mathbf{Q}$ *the* set of all the normalized vectors in $\mathbb{C}^2$*. We assume that each propositional symbol* $p \in$ Prop *is the encoding of a normalized vector in* $\mathbf{Q}$*: given* $|\phi\rangle \in \mathbf{Q}$*, we write* $\ulcorner|\phi\rangle\urcorner$ *to denote its* encoding into the QBDTL syntax*. More formally, for this example, we can define the set Prop of propositional symbols to contain:*

- *the set of strings of the shape* $\ulcorner\alpha_1|0\rangle + \alpha_2|1\rangle\urcorner$*, where* $\alpha_1|0\rangle + \alpha_2|1\rangle$ *is the representation of an element of* $\mathbf{Q}$ *with respect to the standard computational basis, and*
- *the set of string of the shape* $\ulcorner U_1(U_2(\cdots(U_n(\alpha_1|0\rangle + \alpha_2|1\rangle))\cdots))\urcorner$*, where each* $U_j$ *is either a unary unitary transformation or the* restriction *of a cnot operator and* $\alpha_1|0\rangle + \alpha_2|1\rangle$ *is defined as above.*

*Note that, as discussed in [24], different propositional symbols can describe equivalent quantum states. For example,* $\ulcorner H(|0\rangle)\urcorner$ *(where H is the* Hadamard gate*) and* $\ulcorner \frac{1}{2}(|0\rangle + |1\rangle)\urcorner$ *represent the same quantum state. For simplicity, but without loss of generality, in this paper whenever we consider one propositional symbol we actually take it as representative for its equivalence class. Extending our logic to explicitly consider these equivalences would be quite straightforward but fairly tedious and notationally cumbersome.*

*Note also that, as we remarked above, our central idea is to see event sharing as synchronization points between different agent life cycles, i.e., as controlled gate occurrences. We thus use the operator* © *to syntactically model synchronization/control. This interpretation has an evidence both in QDTL derivations and at the semantical level. In particular, we will get the evidence that: if no non-unary gate occurs in a computation involving different agents, no synchronizations (and, syntactically, no calling operator* $©_k$*) happen.*

*The simplest quantum circuit (generated by* $\mathbf{G}$*) acting on two agents is the one built upon a single occurrence of the cnot gate. Consider the following figure*




*and suppose that C is an instance of the cnot gate, and that the control input i and the target input j are set to* $|1\rangle$ *and* $|0\rangle$*, respectively. By definition, the negation of the target qubit is performed. For the sake of illustration, but of course with a slight abuse of notation, the figure shows also two time instants* $t_1$ *and* $t_2$*. This allows us to consider a significative set of formulas that are built from the propositional symbols* $\ulcorner|1\rangle\urcorner$ *and* $\ulcorner|0\rangle\urcorner$*, and that describe the behavior of C on inputs* $|1\rangle$ *and* $|0\rangle$ *during their temporal evolution.*

$$\text{at time } t_1: \qquad @_i\,(\ulcorner|1\rangle\urcorner) \;\; \text{and} \;\; @_j\,(\ulcorner|0\rangle\urcorner) \tag{1}$$

$$\textit{at time } t_1: \quad @_i\,(\ulcorner|1\rangle\urcorner \wedge \exists\bigcirc\ulcorner|1\rangle\urcorner) \tag{2}$$

$$\textit{at time } t_1: \quad @_j\,((\ulcorner|0\rangle\urcorner \wedge \copyright_i(\ulcorner|1\rangle\urcorner)) \supset \exists\bigcirc\ulcorner|1\rangle\urcorner) \tag{3}$$

$$\textit{at time } t_2: \quad @_i\,(\ulcorner|1\rangle\urcorner) \;\; \textit{and} \;\; @_j\,(\ulcorner|1\rangle\urcorner) \tag{4}$$

*The intuitive meaning of these formulas is as follows. Formulas (1) and (4) describe the input and output states, respectively. Formula (2) says that the first agent (corresponding in our interpretation to the control qubit) has input value* $|0\rangle$ *and moreover there exists a successor (temporal) state (along the branch belonging to the computation captured by the circuit among all the possible computations) in which the state remains unchanged (remember that the cnot operator does not act on the control qubit). Formula (3) describes the* control/synchronization *between agents, corresponding to the occurrence of the binary gate; it is syntactically significative to model synchronization from the* target agent's perspective*, i.e., by means of a formula of the local language* $\mathcal{L}_j$ *where, by means of the calling operator, we can express the mathematical state of the control agent through the suitable propositional symbols). In other words, the formula expresses the fact that a control (a* QBDTL *calling) and the related operation occur as a consequence of agent synchronization.*

*In Examples 4 and 5, we will continue this discussion and, moreover, consider a more complex quantum circuit, explaining also how the semantical notions introduced above can be concretely interpreted in quantum computation modeling.*

## 5. A deduction system for QBDTL

### *5.1. Syntax of the labeled logic*

In order to formalize our labeled natural deduction system $\mathcal{N}$(QBDTL), we extend the syntax and semantics of QBDTL by introducing four kinds of *labels* (that represent agents, states, quantum information and paths in the underlying semantics) and by defining labeled and relational formulas.

First of all, we use the agent identifiers in Id as labels. Further, we assume given two fixed denumerable sets of labels $\mathsf{Lab}_S$ and $\mathsf{Lab}_Q$. Intuitively, the labels $x, y, z, \ldots$ in $\mathsf{Lab}_S$ refer to local states of an agent, whereas the labels $q, q', q_1, \ldots$ in $\mathsf{Lab}_Q$ refer to the quantum information concerning an agent.

**Definition 5.** *A* labeled formula *is a formula of the form*

$$(i, x, q) : A\,,$$

*where* $(i, x, q)$ *is a* composed label *with* $i \in \mathsf{Id}$*,* $x \in \mathsf{Lab}_S$ *and* $q \in \mathsf{Lab}_Q$*, and A is a formula in the local language* $\mathcal{L}_i$ *of the agent i.*

Note that we do not use the operator @ inside labeled formulas as it is implicitly expressed by the first element of the composed label. For instance, in order to show that a global formula $@_i A$ is valid, we will prove that the labeled formula $(i, x, q) : A$, for arbitrary $x$ and $q$, is derivable in our system.

In $\mathcal{N}$(QBDTL), we also need formulas modeling the relation between the states referred by the labels. We thus assume given a further set of labels $\mathsf{Lab}_B$, whose elements will be denoted by $\lhd, \lhd_1, \lhd_2, \ldots$, which intuitively refer to the successor relation

between local states in the local life-cycle of an agent $i$ along a given branch. We then define:

**Definition 6.** $\mathsf{Lab}_B^+ = \mathsf{Lab}_B \cup \{r(i, x, \bigstar A) \mid i \in \mathsf{Id}, x \in \mathsf{Lab}_S, \bigstar \in \{\Box, \bigcirc\}, A \in \mathcal{L}_i\}$.

The labels in $\mathsf{Lab}_B^+ \setminus \mathsf{Lab}_B$ are used to refer to successor relations between local states along distinct branches. For instance, the label $r(i, x, \Box A)$ is used to denote a particular branch, in the local life-cycle of $i$, which starts in the node denoted by $x$ and along which $A$ always holds (provided that such a branch exists). We write $R$, $R_1$, $R_2$, ... to denote generic elements of $\mathsf{Lab}_B^+$ and we use $R^*$ to refer to the reflexive and transitive closure of $R$. Finally, we use the symbol $U$ to refer to the relation modeling unary quantum transformations and the symbol $\bowtie$ to denote that the local states of two agents are synchronized on a given event. Then, we can define relational formulas as follows:

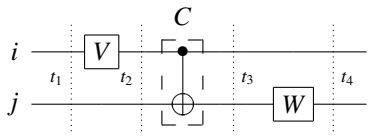
**Definition 7.** *A* relational formula *is a formula of the form*

- $(i, x, q)\ R\ (i, y, q')$*, or*
- $(i, x, q)\ R^*\ (i, y, q')$*, or*
- $(i, x, q) \bowtie (j, y, q')$*, or*
- $q\ U\ q'$*,*

*where* $i, j \in \mathsf{Id}$*,* $x, y \in \mathsf{Lab}_S$*,* $R \in \mathsf{Lab}_B^+$*,* $q, q' \in \mathsf{Lab}_Q$*.*

In the following, for simplicity, we sometimes use metavariables of the form $s_i$, possibly superscripted, to refer to composed labels of the form $(i, x, q)$.

**Example 4 (QBDTL "in action" — II).** *To illustrate further the power of QBDTL to formalize quantum computations, we consider a more complex quantum circuit Circ*




*where V and W are quantum gates and C is a cnot. We can interpret Circ as a* finite *set of paths belonging to a distributed life-cycle for agents i and j; in some sense, Circ can be viewed as a sub-computation of a (potentially) infinite quantum computation involving i and j.*

*Recall, from Definition 3, the notion of e-branch b of an agent i. We write* $\rightarrow_i^{\downarrow Circ}$ *to denote the finite restriction of the particular e-branch corresponding to the transformation occurring to i by means of the computation Circ. We write* $Ev_i^{\downarrow Circ}$ *to denote the (possibly finite) set of events "concerning" the Circ computation. In other words, we focus on the dynamical evolution of the qubit i by means of the transformations corresponding to the quantum gates of Circ. To this end, we interpret each local event either as a pair* $\langle input, gate\rangle$ *where* $input \in \mathbb{C}^2$ *and* $gate \in \mathcal{G}$*, or as an input value (i.e., the quantum state of the agents before the circuit evaluation) that we call* $|\phi\rangle_{in}$*. Note*

*that, in the former case, we describe the outcome of a gate (the current state) by the pair $\langle input, gate\rangle$, whereas in the latter we are fixing the circuit's input value.*

*Suppose now that, at the time $t_1$, the agents $i$ and $j$ are in the states $|c\rangle$ and $|d\rangle$ (for $c, d \in \{0, 1\}$), respectively. We can describe the "progress" of the two agents in terms of increasing sets of collected events (which "keeps track" of the subcomputation occurred up to the current time) as follows. For i, we have*

$$\emptyset \rightarrow \{|c\rangle_{in}\} \rightarrow \{|c\rangle_{in}, \langle\, |c\rangle_{in}, V\,\rangle\} \rightarrow \{|c\rangle_{in}, \langle\, |c\rangle_{in}, V\,\rangle, \langle\, V(|c\rangle_{in}), C^{\langle i\rangle}\,\rangle\}...$$

*and for agent $j$, we have*

$$\emptyset \rightarrow \{|d\rangle_{in}\}) \rightarrow \{|d\rangle_{in}, \langle\, |d\rangle_{in}, C^{\langle j\rangle}\,\rangle\} \rightarrow \{|d\rangle_{in}, \langle\, |d\rangle_{in}, C^{\langle j\rangle}\,\rangle, \langle\, C^{\langle j\rangle}(|d\rangle_{in}), W\,\rangle\}...$$

*We can now consider the set of propositional symbols*

$$\{\ulcorner|c\rangle\urcorner, \ulcorner|d\rangle\urcorner, \ulcorner V(|c\rangle)\urcorner, \ulcorner C^{\langle i\rangle}(V(|c\rangle))\urcorner, \ulcorner C^{\langle j\rangle}(|d\rangle)\urcorner, \ulcorner W(C^{\langle j\rangle}(|d\rangle))\urcorner\}$$

*to build the following labeled formulas to describe the circuit Circ:*

$$\text{at time } t_1: \quad (i, x_1, q_1) : \ulcorner|c\rangle\urcorner \text{ and } (j, y_1, q'_1) : \ulcorner|d\rangle\urcorner \tag{5}$$

$$\text{at time } t_1: \quad (i, x_1, q_1) : \ulcorner|c\rangle\urcorner \wedge \exists\bigcirc\ulcorner V(|c\rangle)\urcorner \tag{6}$$

$$\text{at time } t_2: \quad (i, x_2, q_2) : \ulcorner V(|c\rangle)\urcorner \wedge \exists\bigcirc\ulcorner V(|c\rangle)\urcorner \tag{7}$$

$$\text{at time } t_2: \quad (i, x_2, q_2) : \ulcorner V(|c\rangle)\urcorner \wedge \exists\bigcirc[\copyright_j(\ulcorner d\urcorner)] \tag{8}$$

$$\text{at time } t_2: \quad (j, y_2, q'_2) : \ulcorner|d\rangle\urcorner \wedge \exists\bigcirc[\copyright_i(V(|c\rangle))] \tag{9}$$

$$\text{at time } t_3: \quad (i, x_3, q_3) : \ulcorner V(|c\rangle)\urcorner \tag{10}$$

$$\text{at time } t_4: \quad (i, x_3, q_3) : \ulcorner V(|c\rangle)\urcorner \text{ and } (j, y_3, q'_3) : \ulcorner W(C^{\langle j\rangle}(|d\rangle))\urcorner \tag{11}$$

*We describe the intuitive meaning of some of these formulas. Formulas (5) and (11) describe the input and output states, respectively. Formula (6) says that the first agent has input value $|c\rangle$ (i.e., the propositional symbol $\ulcorner|c\rangle\urcorner$ holds) and moreover there exists a successor temporal state (along the branch belonging to the computation captured by the circuit among all the possible computations) in which $\ulcorner V(|c\rangle)\urcorner$ holds. Formula (7) says that this successor temporal state has a successor in which $\ulcorner V(|c\rangle)\urcorner$ still holds, i.e., the quantum state remains unchanged (remember that the cnot operator does not act on the control qubit). Formulas (8) and (9) say that for the agents i and j, for which $\ulcorner V(|c\rangle)\urcorner$ and $\ulcorner|d\rangle\urcorner$ respectively hold, there exists a successor state in which a synchronization with the other agent occurs.*

*The sub-circuit corresponding to the cnot instance C can be described again in terms of synchronization (from the viewpoint of the target qubit/agent) as done in Example 3. By the definition of the cnot operator there are two cases. First, suppose that $V(|c\rangle) = |0\rangle$, which means that no complementation of the target qubit state occurs. Then, we have $(j, y_2, q'_2) : (\ulcorner|d\rangle\urcorner \wedge \copyright_i(\ulcorner V(|c\rangle\urcorner))) \supset \exists\bigcirc\ulcorner|d\rangle\urcorner$. Second, suppose that $V(|c\rangle) \neq |0\rangle$ (and remember that we are not dealing with entangled states). Then, we have $(j, y_2, q'_2) : (\ulcorner|d\rangle\urcorner \wedge \copyright_i(\ulcorner V(|c\rangle\urcorner))) \supset \exists\bigcirc\ulcorner\,\overline{|d\rangle}\,\urcorner$, where $\overline{|d\rangle}$ denotes the complementation of the state $|d\rangle$.*

*5.2. Semantics of the labeled logic*

In order to give a semantics for our labeled system, we need to define explicitly an interpretation of the labels.

**Definition 8.** *Given a QBDTL model $\mu$, an* interpretation function *is a triple $\mathcal{I} = \langle \mathcal{I}_S, \mathcal{I}_Q, \mathcal{I}_B \rangle$, where:*

- $\mathcal{I}_S = \{\mathcal{I}_S^i\}_{i \in \mathsf{Id}}$ *is a set of functions such that* $\mathcal{I}_S^i : \mathsf{Lab}_S \to \Xi_i$ *for each* $i \in \mathsf{Id}$*;*
- $\mathcal{I}_Q : \mathsf{Lab}_Q \to Q$*;*
- $\mathcal{I}_B = \{\mathcal{I}_B^i\}_{i \in \mathsf{Id}}$ *is a set of functions such that* $\mathcal{I}_B^i : \mathsf{Lab}_B^+ \to \mathcal{B}_i$ *for each* $i \in \mathsf{Id}$*, and if* $r(i, x, \bigstar A) \in \mathsf{Lab}_B^+ \setminus \mathsf{Lab}_B$*, then:*
  - $\mathcal{I}_B^i(r(i, x, \bigstar A)) = \to_i^b$ *for some* $\mathcal{I}_S^i(x)$*-branch* $b$*;*
  - *if* $\models^{\mu, \mathcal{I}_S^i(x)} \exists \bigstar A$*, then for all* $\xi \in \Xi_i$*:*
    - *if* $\bigstar = \bigcirc$*, then* $last(\mathcal{I}_S^i(x))\ \mathcal{I}_B^i(r(i, x, \bigstar A))\ last(\xi)$ *implies* $\models^{\mu, \xi} A$*;*
    - *if* $\bigstar = \Box$*, then* $last(\mathcal{I}_S^i(x))\ \mathcal{I}_B^i(r(i, x, \bigstar A))^*\ last(\xi)$ *implies* $\models^{\mu, \xi} A$*.*

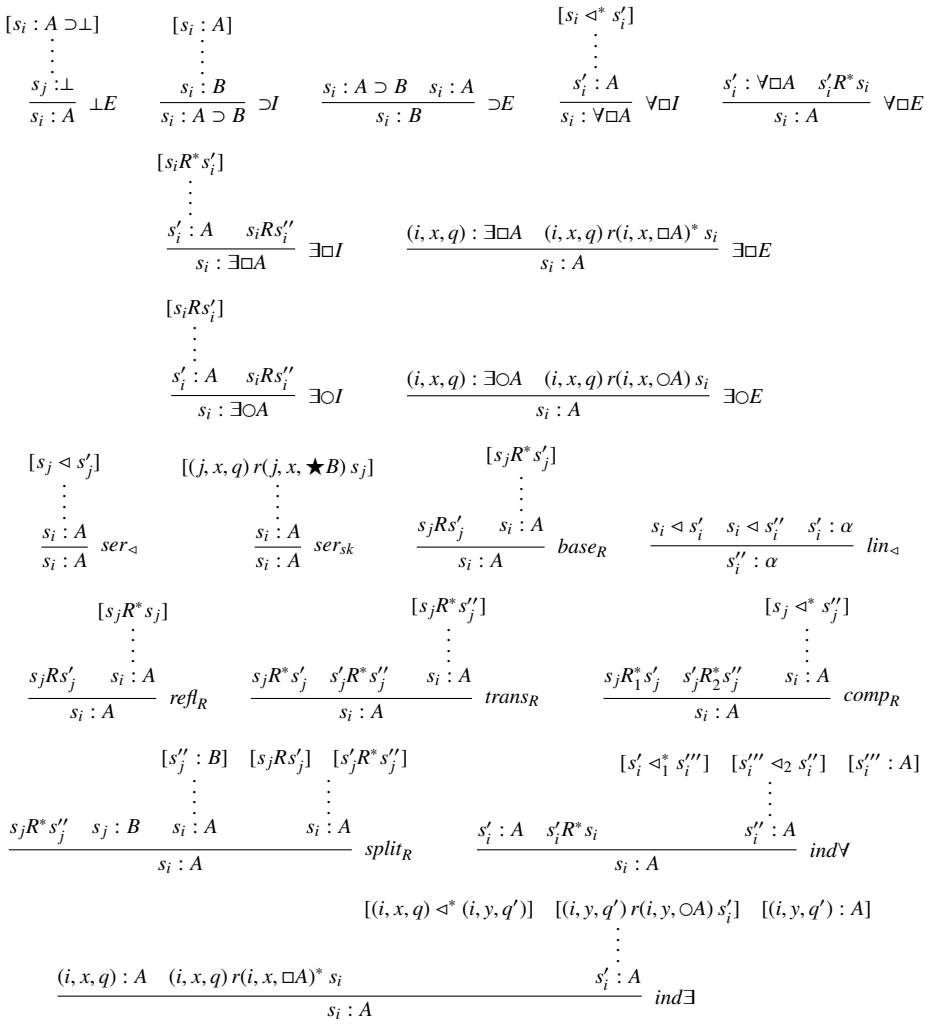
The notion of interpretation allows us to extend the truth relation to labeled formulas, as well as define truth of relational formulas.

**Definition 9.** *Given a QBDTL model $\mu$ and an interpretation function $\mathcal{I} = \langle \mathcal{I}_S, \mathcal{I}_Q, \mathcal{I}_B \rangle$ on it,* truth for a labeled or relational formula $\gamma$ *is defined as follows:*

| | | |
|---|---|---|
| $\models^{\mu, \mathcal{I}} (i, x, q) : A$ | *iff* | $\mu_i, \mathcal{I}_S^i(x) \models_i A$ *and* $\pi_i(\mathcal{I}_S^i(x)) = \mathcal{I}_Q(q)$ |
| $\models^{\mu, \mathcal{I}} (i, x, q)\ R\ (i, y, q')$ | *iff* | $last(\mathcal{I}_S^i(x))\ \mathcal{I}_B^i(R)\ last(\mathcal{I}_S^i(y)), \pi_i(\mathcal{I}_S^i(x)) = \mathcal{I}_Q(q)$ *and* $\pi_i(\mathcal{I}_S^i(y)) = \mathcal{I}_Q(q')$ |
| $\models^{\mu, \mathcal{I}} (i, x, q)\ R^*\ (i, y, q')$ | *iff* | $last(\mathcal{I}_S^i(x))\ \mathcal{I}_B^i(R)^*\ last(\mathcal{I}_S^i(y)), \pi_i(\mathcal{I}_S^i(x)) = \mathcal{I}_Q(q)$ *and* $\pi_i(\mathcal{I}_S^i(y)) = \mathcal{I}_Q(q')$ |
| $\models^{\mu, \mathcal{I}} (i, x, q) \bowtie (j, y, q')$ | *iff* | $last(\mathcal{I}_S^i(x)) = last(\mathcal{I}_S^j(y)), \pi_i(\mathcal{I}_S^i(x)) = \mathcal{I}_Q(q)$ *and* $\pi_j(\mathcal{I}_S^j(y)) = \mathcal{I}_Q(q')$ |
| $\models^{\mu, \mathcal{I}} q\ U\ q'$ | *iff* | $\mathcal{I}_Q(q)\ \mathcal{U}\ \mathcal{I}_Q(q')$ |

*When* $\models^{\mu, \mathcal{I}} \gamma$*, for $\gamma$ a labeled or relational formula, we say that $\gamma$ is* true *in $\mu$ according to $\mathcal{I}$. By extension:*

| | | |
|---|---|---|
| $\models^{\mu, \mathcal{I}} \Gamma$ | *iff* | $\models^{\mu, \mathcal{I}} \gamma$ *for all* $\gamma \in \Gamma$ |
| $\Gamma \models^{\mu, \mathcal{I}} \gamma$ | *iff* | $\models^{\mu, \mathcal{I}} \Gamma$ *implies* $\models^{\mu, \mathcal{I}} \gamma$ |
| $\models^{\mu} \gamma$ | *iff* | *for every interpretation function* $\mathcal{I}$*,* $\models^{\mu, \mathcal{I}} \gamma$ |
| $\models^{\mu} \Gamma$ | *iff* | *for every interpretation function* $\mathcal{I}$*,* $\models^{\mu, \mathcal{I}} \Gamma$ |
| $\Gamma \models \gamma$ | *iff* | *for every QBDTL model* $\mathcal{M}$ *and interpretation function* $\mathcal{I}$*,* $\Gamma \models^{\mu, \mathcal{I}} \gamma$ |



In $\forall\Box I$, $\exists\Box I$ and $\exists\bigcirc I$, where $s'_i \equiv (i, x, q)$, the labels $x$ and $q$ are fresh. Moreover, in $\forall\Box I$, $\lhd$ is fresh.
In $ser_{\lhd}$, where $s'_j \equiv (j, x, q)$, the labels $x$, $q$ and $\lhd$ are fresh.
In $ser_{sk}$, where $s_j \equiv (j, y, q')$, the labels $y$ and $q'$ are fresh.
In $comp_R$, $\lhd$ is fresh.
In $ind\forall$, where $s''_i \equiv (i, x, q)$ and $s'''_i \equiv (i, y, q')$, the labels $x$, $y$, $q$, $q'$, $\lhd_1$ and $\lhd_2$ are fresh.
In $ind\exists$, where $s'_i \equiv (i, z, q'')$, the labels $y$, $z$, $q'$, $q''$ and $\lhd$ are fresh.

Figure 1: The rules of $\mathcal{N}$(QBDTL): local life-cycle rules

$$\frac{s_j : A \quad s_i \bowtie s_j}{s_i : ©_j A}\ ©I \qquad \frac{s_i : ©_j A \qquad \begin{array}{c}[s_i \bowtie s_j][s_j : A]\\ \vdots \\ s_k : A\end{array}}{s_k : A}\ ©E$$

$$\frac{s_j \bowtie s_k \qquad \begin{array}{c}[s_k \bowtie s_j]\\ \vdots \\ s_i : A\end{array}}{s_i : A}\ symm_{\bowtie} \qquad \frac{s_j \bowtie s_k \quad s_k \bowtie s_l \qquad \begin{array}{c}[s_j \bowtie s_l]\\ \vdots \\ s_i : A\end{array}}{s_i : A}\ trans_{\bowtie}$$

$$\frac{s_i \bowtie s_{j_1} \quad s_{j_1} R s'_{j_1} \quad s'_{j_1} R^* s''_{j_1} \quad s''_{j_1} \bowtie s_{j_2} \quad \ldots \quad s''_{j_n} \bowtie s''_i \qquad \begin{array}{c}[s_i \lhd s'_i][s'_i \lhd^* s''_i]\\ \vdots \\ s_k : A\end{array}}{s_k : A}\ comp_{\bowtie}$$

In $©I$ and $©E$, $i \neq j$. In $©E$, where $s_j \equiv (j, x, q)$, the labels $x$ and $q$ are fresh. In $comp_{\bowtie}$, $\lhd$ is fresh.

Figure 2: The rules of $\mathcal{N}$(QBDTL): distributed life-cycle rules

$$\frac{\begin{array}{c}[q\ U\ q]\\ \vdots \\ s_i : A\end{array}}{s_i : A}\ refl_U \qquad \frac{q\ U\ q' \qquad \begin{array}{c}[q'\ U\ q]\\ \vdots \\ s_i : A\end{array}}{s_i : A}\ symm_U$$

$$\frac{q\ U\ q' \quad q'\ U\ q'' \qquad \begin{array}{c}[q\ U\ q'']\\ \vdots \\ s_i : A\end{array}}{s_i : A}\ trans_U \qquad \frac{(i, x, q) : p \quad \gamma(j, y, q)}{(j, y, q) : p}\ prop$$

In *prop*, $\gamma(j, y, q)$ is a (labeled or relational) formula where $(j, y, q)$ occurs and $p \in \mathsf{Prop}$ is an atomic proposition.

Figure 3: The rules of $\mathcal{N}$(QBDTL): quantum transformation rules

### *5.3. The rules of $\mathcal{N}$(QBDTL)*

The rules of $\mathcal{N}$(QBDTL) are given in Figures 1–4. We can classify them into four categories: (i) *local life-cycle rules* (inspired to the deduction system for the logic UB given in [11]), (ii) *distributed life-cycle rules* (reminiscent of the global labeled tableaux developed for DTL in [5]), (iii) *quantum transformations rules* (actually a fragment of the deduction systems studied in [24]) and (iv) *interaction rules*.

$$\frac{q\ U\ q' \quad \gamma(i,x,q) \qquad \begin{array}{c}[(i,x,q) \lhd (i,y,q')] \\ \vdots \\ s_j : A\end{array}}{s_j : A}\ U{\Rightarrow}R \qquad \frac{(i,x,q) \lhd^* (i,y,q') \qquad \begin{array}{c}[q\ U\ q'] \\ \vdots \\ s_j : A\end{array}}{s_j : A}\ R{\Rightarrow}U$$

In $U{\Rightarrow}R$, $\gamma(i,x,q)$ is a (labeled or relational) formula where $(i,x,q)$ occurs. Moreover, $y$ is fresh.

Figure 4: The rules of $\mathcal{N}$(QBDTL): interaction rules

*Local life-cycle rules (Figure 1)*

These rules all infer formulas "local" to an agent $i$, i.e., labeled with $s_i$. We can divide them further into rules for classical connectives ($\bot E$, $\supset I$ and $\supset E$), rules for temporal operators ($\forall\Box I$, $\forall\Box E$, $\exists\Box I$, $\exists\Box E$, $\exists\bigcirc I$ and $\exists\bigcirc E$), relational rules ($ser_{\lhd}$, $ser_{sk}$, $base_R$, $lin_{\lhd}$, $refl_R$, $trans_R$ and $comp_R$) and induction rules ($ind\forall$ and $ind\exists$).

*Rules for classical connectives.* The rule $\bot E$ is a labeled version of *reductio ad absurdum*, where we do not enforce Prawitz's side condition that $A \neq \bot$ and we do not constrain the "world" in which we derive a contradiction to be the same as in the assumption. The rules $\supset I$ and $\supset E$ are the labeled version of the standard [29] natural deduction rules for implication introduction and elimination.

*Rules for temporal operators.* The rules for the introduction and the elimination of $\forall\Box$, $\exists\Box$ and $\exists\bigcirc$ follow the same structure as the rules for introduction and elimination of $\Box$ in labeled systems for modal logics. Let us consider $\forall\Box I$; the idea is that the meaning of $s_i : \forall\Box A$ is given by the metalevel implication $s_i \lhd^* s_i' \Longrightarrow s_i' : A$ for an arbitrary path denoted by the relation $\lhd$ and an arbitrary $s_i'$ $\lhd^*$-accessible from $s_i$. The arbitrariness, i.e., the *freshness*, of both the path denoted by $\lhd$ and $s_i'$ is ensured by the side-conditions of the rule, e.g., $s_i$ must be different from $s_i$ and not occur in any assumption on which $s_i' : A$ depends other than the discharged assumption $s_i \lhd^* s_i'$.

Introductions of $\exists\Box$ and $\exists\bigcirc$ follow the same principle, but relax the freshness condition on the label denoting the relation, thus allowing us to reason on a single specific path. Note that in this case a further premise ($s_i R s_i''$) is required: such a premise works as a "witness", in the sense that it ensures that the relation $R$ considered is indeed a relation passing through the state $s_i$.

For what concerns the elimination rules, the intuition behind $\forall\Box E$ is that if $\forall\Box A$ holds in a state $s_i'$ and $s_i$ is accessible from $s_i'$ (along some path), then it is possible to conclude that $A$ holds in $s_i$. The case of $\exists\Box E$ and $\exists\bigcirc E$ is similar but complicated by the fact that the universal linear-time operator ($\Box$ or $\bigcirc$) is preceded by an existential path quantifier ($\exists$), which prevents us from inferring the conclusion for a successor along an arbitrary relation. Our solution is based on the idea (originally proposed in [11]) of using Skolem functions as names for particular relations, e.g., $r(i,x,\Box A)$ denotes a relation passing at $x$ and such that if $\exists\Box A$ holds in $x$, then $A$ holds at each successor of $x$ along $r(x,\Box A)$.

*Relational rules.* Relational rules allow for modeling properties of the accessibility relations.[3] The rule $base_R$ expresses the fact that for each relation $R$, $R^*$ contains $R$; i.e., $base_R$ says that if (i) $s_j$ is such that there is some $R$-accessible $s'_j$ and (ii) from the assumption that $s'_j$ is also $R^*$-accessible from $s_j$ we can infer some labeled formula $s_i : A$ (where $s_i$ might be different from $s_j$ and $s'_j$), then we can discharge the assumption $s_j R^* s'_j$ and conclude that indeed $s_i : A$ holds. $refl_R$ and $trans_R$ model reflexivity and transitivity of each relation, respectively, whereas $comp_R$ states that it is possible to compose two relations, i.e., if $s_j R_1^* s'_j$ and $s'_j R_2^* s''_j$, then there exists a third relation $\lhd^*$ such that $s_j \lhd^* s''_j$. We also have two rules capturing two different aspects of the seriality of the relations. $ser_\lhd$ captures the fact that, given a state $s_j$, there is at least a relation passing through $s_j$ and a successor along that relation. $ser_{sk}$ says that, given a state $s_j$ and a Skolem function $r(j, x, \bigstar B)$, there exists a successor of $s_j$ along that relation. Finally, $split_R$ states that if $s_i R^* s'_i$ holds, then either $s_i$ and $s'_i$ coincide or $s'_i$ is a proper successor of $s_i$.

*Induction rules.* Finally, we have two rules that model the induction principle underlying the relation between $R$ and $R^*$. This modeling of the induction principle is inspired to the one proposed in [11] and it is reminiscent of deduction systems for Peano Arithmetic. An example of use of the rule $ind\exists$ can be found in Figure 5, as we discuss below.

*Distributed life-cycle rules (Figure 2)*

The rules for communication ($\copyright I$ and $\copyright E$) follow quite closely the semantics. Consider, e.g., $\copyright I$: if agent $i$ in state $s_i$ synchronizes with agent $j$ in state $s_j$, and $A$ holds for $j$ in that state, then $i$ just communicated with agent $j$. The rules for synchronization are also quite intuitive, except maybe $comp_\bowtie$. Intuitively, $comp_\bowtie$ models a notion of compatibility between different synchronizations that involves the same agents and reflects condition (ii) in the definition of branching distributed life-cycle. It is a rule schema that can be applied for any finite $n$.

*Quantum transformations rules (Figure 3)*

The rules $refl_U$, $symm_U$, $trans_U$ formalize, quite straightforwardly, the reflexivity, symmetry and transitivity of the $U$ relation, in order to uniformly model the class of algebraic unitary operators as an equivalence relation. This captures, in an abstract way, key properties of quantum operators. Roughly speaking: reflexivity says that the class of the unitary operators includes the identity transformation; symmetry captures reversibility (it is always possible to reverse a quantum transformation, since the inverse operator is easily definable and is unitary [27]); finally, transitivity models algebraic compositionality, i.e., the composition of two or more unitary operators is always a unitary operator.

The rule *prop* says that the third element in a composed label fully captures the quantum information contained in a state; thus, if two composed labels $(i, x, q)$ and

[3]Note that in these rules we use relational formulas as auxiliary formulas in order to derive labeled formulas. Rules treating relational formulas as full-fledged first class formulas, which can be assumed and derived in the rules, could also be defined in the style of [33].

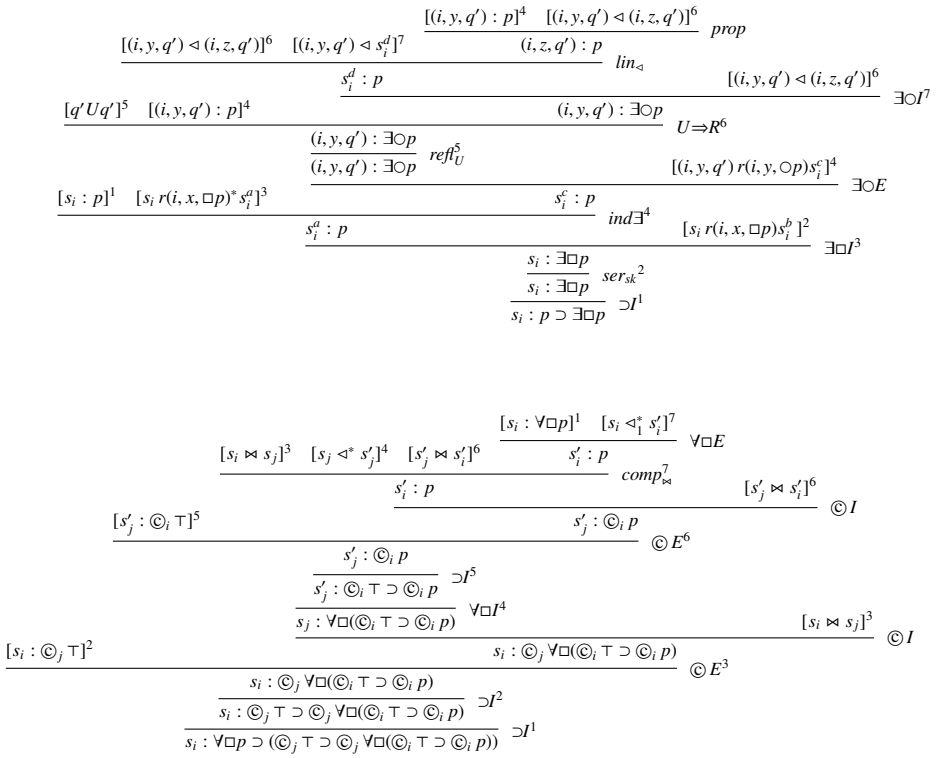


Figure 5: Example derivations

$(j, y, q)$ share the same $q$, each atomic proposition holding in $(i, x, q)$ must also hold in $(j, y, q)$.

*Interaction rules (Figure 4)*

The rules $U{\Rightarrow}R$ and $R{\Rightarrow}U$ model the interaction between $U$ and $R$ and express respectively the conditions (i) and (ii) in the definition of function $\pi_i$ of QBDTL models. More specifically, $U{\Rightarrow}R$ says that if $qUq'$ and the label $(i, x, q)$ occurs in the labeled or relational formula $\gamma(j, x, q)$, then $(i, x, q)$ has a $\lhd$-successor $(i, y, q')$; this means that the local state labeled by $y$ is an immediate successor of the state labeled by $x$ in local life-cycle of the agent $i$, along a given branch. The rule $R{\Rightarrow}U$ captures the fact that if $(i, y, q')$ is a $\lhd^*$-successor of $(i, x, q)$ then also the quantum labels $q$ and $q'$ have to be related by $U$.

### *5.4. Derivations*

Given the rules in Figure 1–4, the notions of *derivation*, *conclusion*, *open* and *discharged assumption* are the standard ones for natural deduction systems (see, e.g., [21], pp. 127-129).

**Definition 10.** *We write*

$$\Gamma \vdash_{\mathcal{N}(\mathsf{QBDTL})} (i, x, q) : A$$

*to say that there exists a derivation of* $(i, x, q) : A$ *in the system* $\mathcal{N}(\mathsf{QBDTL})$ *whose open assumptions are all contained in the set of (labeled and relational) formulas* $\Gamma$*. A derivation of* $(i, x, q) : A$ *in* $\mathcal{N}(\mathsf{QBDTL})$ *where all the assumptions are discharged is a* proof *of* $(i, x, q) : A$ *in* $\mathcal{N}(\mathsf{QBDTL})$ *and we then say that* $(i, x, q) : A$ *is a* theorem *of* $\mathcal{N}(\mathsf{QBDTL})$ *and write* $\vdash_{\mathcal{N}(\mathsf{QBDTL})} (i, x, q) : A$*.*

Figure 5 contains two examples of derivations (actually, proofs). The first is based on the fact that it is always possible to apply the identity transformation to a qubit. It follows that if a qubit is in a state where an atomic proposition $p$ holds, then there exists a path along which $p$ always holds.

The formula derived in the second example describes a property of the synchronization between qubits and can be read as a consequence of condition (ii) in the definition of a distributed life-cycle. If the qubit $i$ is in a state from which a proposition $p$ always holds in the future, then if $i$ synchronizes with $j$, i.e., the two qubits are combined by means of some $n$-ary quantum operator, and after that, $j$ synchronizes with $i$ again, we end up in a state of $i$ where $p$ still holds. Note that in this derivation we use the *verum* $\top$ as an abbreviation for $\bot \supset \bot$.

**Example 5 (QBDTL "in action" — III).** *We now give a derivation that describes the circuit of Example 3. The aim of this example is to show that our interpretation of synchronizations as controlled operations is significative also at level of deduction. Let us consider the following formula, which describes the circuit's behavior from the perspective of agent j:*

$$(j, x, q) : \forall\Box((\ulcorner|0\rangle\urcorner \wedge ©_i\ulcorner|1\rangle\urcorner) \supset \exists\bigcirc(\ulcorner|1\rangle\urcorner \wedge (©_i\ulcorner|1\rangle\urcorner))) \tag{12}$$

*where we suppose that x is the starting temporal state that belongs to the path corresponding to the circuit. Intuitively, the antecedent and the consequent of the implication describe the input-output behavior of the circuit; the "action" of the controlled gate can be read as the sharing of a time instant by the agents involved in the calling.*

*We need also to specify a further set of assumptions, in order to model the required input conditions. This will be explicitly done by the following labelled and relational formulas:*

$$(j, x, q) \lhd^* (j, x_j^1, q_j^1) \tag{13}$$

$$(j, x_j^1, q_j^1) : \ulcorner|0\rangle\urcorner \tag{14}$$

$$(j, x_j^1, q_j^1) \bowtie (i, x_i^1, q_i^1) \tag{15}$$

$$(i, x_i^1, q_i^1) : \ulcorner|1\rangle\urcorner \tag{16}$$

*Informally, we are expressing the following notions: starting from the current state x, we focus on a state in the future (13) in which* $\ulcorner|0\rangle\urcorner$ *holds for j (14); moreover, at the same instant, j is synchronized with the agent i (15) and* $\ulcorner|1\rangle\urcorner$ *holds for i (16).*

*Starting from these assumptions, we are able to derive the labeled formula* $(j, x, q) : \exists\bigcirc\ulcorner|1\rangle\urcorner$*, which actually says that there exists a successor temporal state (among all the possible immediate successors) that corresponds to evaluation by means of cnot:*

$$\dfrac{\dfrac{\dfrac{\dfrac{s_j : \forall\Box((\ulcorner|0\rangle\urcorner \wedge \copyright_i\ulcorner|1\rangle\urcorner) \supset \exists\bigcirc(\ulcorner|1\rangle\urcorner \wedge (\copyright_i\ulcorner|1\rangle\urcorner))) \quad s_j \lhd^* s_j^1}{s_j^1 : ((\ulcorner|0\rangle\urcorner \wedge \copyright_i\ulcorner|1\rangle\urcorner) \supset \exists\bigcirc(\ulcorner|1\rangle\urcorner \wedge (\copyright_i\ulcorner|1\rangle\urcorner)))}\,\forall\Box E \quad \dfrac{s_j^1 : \ulcorner|0\rangle\urcorner \quad \dfrac{s_i^1 : \ulcorner|1\rangle\urcorner \quad s_j^1 \bowtie s_i^1}{s_j^1 : \copyright_i\ulcorner|1\rangle\urcorner}\,\copyright I}{s_j^1 : \ulcorner|0\rangle\urcorner \wedge \copyright_i\ulcorner|1\rangle\urcorner}\,\wedge I}{s_j^1 : \exists\bigcirc(\ulcorner|1\rangle\urcorner \wedge \copyright_i\ulcorner|1\rangle\urcorner)}\,\supset E \quad [s_j^1 r(j, x_j^1, \bigcirc A) s_j^2]^2}{\dfrac{s_j^2 : \ulcorner|1\rangle\urcorner \wedge \copyright_i\ulcorner|1\rangle\urcorner}{s_j^2 : \ulcorner|1\rangle\urcorner}\,\wedge E \quad [s_j^1 r(j, x_j^1, \bigcirc A) s_j^n]^1}\,\exists\bigcirc I^2}{\dfrac{s_j^1 : \exists\bigcirc\ulcorner|1\rangle\urcorner}{s_j^1 : \exists\bigcirc\ulcorner|1\rangle\urcorner}\,ser_{sk}{}^1}\,\exists\bigcirc E$$

*where* $s_j \equiv (j, x, q)$*,* $s_j^1 \equiv (j, x_j^1, q_j^1)$*,* $s_j^2 \equiv (j, x_j^2, q_j^2)$*,* $s_i^1 \equiv (i, x_i^1, q_i^1)$ *and* $A \equiv (\ulcorner|1\rangle\urcorner \wedge \copyright_i\ulcorner|1\rangle\urcorner)$*.*

*Note that it would not be possible to derive the conclusion* $s_j^1 : \exists\bigcirc\ulcorner|1\rangle\urcorner$ *only by means of the assumptions describing the input conditions (13), (14), (15) and (16), i.e., without a further premise specifying the behavior of cnot.*

*This example suggests a possible use of* QBDTL *and of the associated deduction system. While some basic properties of quantum transformations, e.g., reversibility, are embodied in the logic, it is also possible to use the formalism provided by* QBDTL *in order to model the existence and the behavior of specific gates, as we did here in the case of cnot. By adding the representation of such gates as assumptions, a modeler is allowed to consider (and to use the deduction system to reason about) more specific scenarios.*

## 6. Soundness

The system $\mathcal{N}$(QBDTL) is sound with respect to the given semantics, as can be shown by adapting standard proof techniques for labeled natural deduction systems [7, 33].

**Theorem 1 (Soundness).** *For every set* $\Gamma$ *of labeled and relational formulas and every labeled formula* $(i, x, q) : A$*, it holds that* $\Gamma \vdash_{\mathcal{N}(QBDTL)} (i, x, q) : A \;\Rightarrow\; \Gamma \models (i, x, q) : A$*.*

Proof. The proof proceeds by induction on the structure of the derivation of $(i, x, q) : A$. The base case is when $(i, x, q) : A \in \Gamma$ and is trivial. There is one step case for every rule (where, for what concerns local life-cycle rules, we refer to [11], whose treatment can be quite easily adapted to work here). We show a few representative step cases.

Consider the case when the last rule applied is *prop*:

$$\dfrac{\begin{array}{cc}\Pi_1 & \Pi_2\\ (i, x, q) : p & \gamma(j, y, q)\end{array}}{(j, y, q) : p}\;prop$$

where $\Pi_1$ is a proof of $(i, x, q) : p$ from hypotheses in $\Gamma_1$ and $\Pi_2$ is a proof of $\gamma(j, y, q)$ from hypotheses in $\Gamma_2$, for some sets $\Gamma_1$, $\Gamma_2$ of formulas. By the induction hypothesis, for each model $\mu = \langle\lambda, \mathcal{M}, \pi\rangle$ and interpretation function $\mathcal{I}$, if $\models^{\mu,\mathcal{I}} \Gamma_1$ then

$\models^{\mu,\mathcal{I}} (i,x,q) : p$ and if $\models^{\mu,\mathcal{I}} \Gamma_2$ then $\models^{\mu,\mathcal{I}} \gamma(j,y,q)$. We consider an $\mathcal{I}$ and a $\mu$ such that $\models^{\mu,\mathcal{I}} \Gamma = \Gamma_1 \cup \Gamma_2$, and show that $\models^{\mu,\mathcal{I}} (j,y,q) : p$. As a consequence of the induction hypothesis, we get: *(i)* $\models_i^{\mu_i,\mathcal{I}^i_S(x)} p$; *(ii)* $\pi_i(\mathcal{I}^i_S(x)) = \mathcal{I}_Q(q)$; and *(iii)* $\pi_j(\mathcal{I}^j_S(y)) = \mathcal{I}_Q(q)$. It follows from (*i*) that $p \in \mathcal{V}(\pi_i(\mathcal{I}^i_S(x)))$, i.e., by (*ii*), $p \in \mathcal{V}(\mathcal{I}_Q(q))$ and, by (*iii*), $p \in \mathcal{V}(\pi_j(\mathcal{I}^j_S(y)))$. By definition, we have $\models^{\mu,\mathcal{I}^j_S(y)} p$, from which we infer $\models^{\mu,\mathcal{I}} (j,y,q) : p$.

Now consider the case of an application of $\copyright I$:

$$\frac{\begin{array}{c}\Pi\\ (j,y,q') : A \quad (i,x,q) \bowtie (j,y,q')\end{array}}{(i,x,q) : \copyright_j A}\ \copyright I$$

where $\Pi$ is a proof of $(j,y,q') : A$ from hypotheses in $\Gamma_1$. By the induction hypothesis, we have $\Gamma_1 \models (j,y,q') : A$. We want to show that $\Gamma = \Gamma_1 \cup \{(i,x,q) \bowtie (j,y,q')\} \models (i,x,q) : \copyright_j A$. Let us consider an arbitrary QBDTL model $\mu = \langle \lambda, \mathcal{M}, \pi \rangle$ and an interpretation function $\mathcal{I}$, and assume that $\models^{\mu,\mathcal{I}} \Gamma$ holds. This implies $last(\mathcal{I}^i_S(x)) = last(\mathcal{I}^j_S(y))$ and $\pi_i(\mathcal{I}^i_S(x)) = \mathcal{I}_Q(q)$. By the induction hypothesis, we also obtain $\models^{\mu,\mathcal{I}} (j,y,q') : A$, which yields $\models_j^{\mu_j,\mathcal{I}^j_S(y)} A$. By the definition of local satisfaction relation, we infer $\models_i^{\mu_i,\mathcal{I}^i_S(x)} \copyright_j A$ and then $\models^{\mu,\mathcal{I}} (i,x,q) : \copyright_j A$. Since $\mu$ and $\mathcal{I}$ are arbitrary, we can conclude $\Gamma \models (i,x,q) : \copyright_j A$.

Finally, consider the case of an application of $R{\Rightarrow}U$:

$$\frac{\begin{array}{cc}(i,y,q') \lhd^* (i,z,q'') & \begin{array}{c}[q'Uq'']\\ \Pi\\ (j,x,q) : A\end{array}\end{array}}{(j,x,q) : A}\ R{\Rightarrow}U$$

where $\Pi$ is a proof of $(j,x,q) : A$ from hypotheses in $\Gamma_1 \cup \{q'Uq''\}$ for some set $\Gamma_1$ of formulas. By the induction hypothesis, we have $\Gamma_1 \cup \{q'Uq''\} \models (j,x,q) : A$. We want to show that $\Gamma = \Gamma_1 \cup \{(i,y,q') \lhd^* (i,z,q'')\} \models (j,x,q) : A$. Let us consider arbitrary $\mu = \langle \lambda, \mathcal{M}, \pi \rangle$ and $\mathcal{I}$, and assume that $\models^{\mu,\mathcal{I}} \Gamma$ holds. This implies $\models^{\mu,\mathcal{I}} (i,y,q') \lhd^* (i,z,q'')$, from which we infer: $last(\mathcal{I}^i_S(y))\ \mathcal{I}^i_B(\lhd)^*\ last(\mathcal{I}^i_S(z))$; $\pi_i(\mathcal{I}^i_S(y)) = \mathcal{I}_Q(q')$; and $\pi_i(\mathcal{I}^i_S(z)) = \mathcal{I}_Q(q'')$. By condition (*i*) in the definition of a QBDTL model and the property of universality behind connected components of *S5* models, this yields $\mathcal{I}_Q(q')\mathcal{U}\mathcal{I}_Q(q'')$ and thus $\models^{\mu,\mathcal{I}} q'Uq''$. By the induction hypothesis, we obtain $\models^{\mu,\mathcal{I}} (j,x,q) : A$. Since $\mu$ and $\mathcal{I}$ are arbitrary, we can conclude $\Gamma \models (j,x,q) : A$. △

## 7. Completeness

Completeness of $\mathcal{N}$(QBDTL) is proved (i) by defining a decision procedure based on semantic tableaux and (ii) by showing a relation between the "closure" conditions in the tableaux and derivability of formulas in $\mathcal{N}$(QBDTL). The structure of the proof is inspired mainly to the one given in [8] for an Hilbert-style axiomatization of the logic *UB*. Similar proofs are presented in [16, 15] for the logic *CTL*. In our setting, adaptations are required in order to deal with labeled formulas, with the synchronization mechanisms and operators and with the *S5*-like modal logic that determines the valuation of atomic propositions.

*7.1. A semantic tableau for QBDTL*

*Hintikka structures*

We define a *structure* as a pair $(S, \bowtie_S)$, where $S$ is a set of triples $\{\langle S_i, R_i, P_i\rangle\}_{i\in \mathsf{Id}}$ specifying for each agent $i$ a set $S_i$ of states, an assignment $P_i$ of formulas to states and a binary relation $R_i$ on $S_i$, and $\bowtie_S$ is a binary relation on $\{s \in S_i \mid i \in \mathsf{Id}\}$, used in the construction to connect states of distinct agents. We require that there exists a subset $\mathsf{Prop}'$ of $\mathsf{Prop}$ such that each $P_i$ is complete with respect to $\mathsf{Prop}'$, i.e., for each atomic proposition $p \in \mathsf{Prop}'$ and state $s \in S_i$, either $p \in P_i(s)$ or $\neg p \in P_i(s)$, and such that all the formulas assigned to states by $P_i$ are built over atoms in $\mathsf{Prop}'$. Finally, we require that a structure respects the following conditions:

- for each $i$ and $s \in S_i$, there exists a $t \in S_i$ such that $sR_it$;
- there exists an *S5* model $\mathcal{M} = \langle \mathcal{Q}, \mathcal{U}, \mathcal{V}\rangle$, where $\mathcal{V}$ is defined from $\mathcal{Q}$ to $\mathcal{P}(\mathsf{Prop}')$, such that for each $i$ and each $s \in S_i$, a world $q_s \in \mathcal{Q}$ is assigned to $s$, and this assignment is such that: $(i)$ for each $p \in \mathsf{Prop}'$, $p \in P_i(s)$ iff $p \in \mathcal{V}(q_s)$, and $(ii)$ $sR_it$ iff $q_s\mathcal{U}q_t$.

We define $R = \bigcup_{i\in \mathsf{Id}} R_i$ as the union of the $R_i$ and, for simplicity, given $s \in S_i$, we will often write $A \in s$ to denote $A \in P_i(s)$ and $s \in S$ if $s \in S_i$ for some $i$.

As in [8], we implicitly reduce possible double negations and replace negated modalities by their duals. We define $\alpha$-formulas and $\beta$-formulas as the following local formulas:[4]

| $\alpha$ | $\alpha_1$ | $\alpha_2$ |
|---|---|---|
| $A_1 \wedge A_2$ | $A_1$ | $A_2$ |
| $\forall\Box A$ | $A$ | $\forall\bigcirc\forall\Box A$ |
| $\exists\Box A$ | $A$ | $\exists\bigcirc\exists\Box A$ |

| $\beta$ | $\beta_1$ | $\beta_2$ |
|---|---|---|
| $A_1 \vee A_2$ | $A_1$ | $A_2$ |
| $\forall\Diamond A$ | $A$ | $\forall\bigcirc\forall\Diamond A$ |
| $\exists\Diamond A$ | $A$ | $\exists\bigcirc\exists\Diamond A$ |

Given a local formula $A$, let $At(A) = \{p_1, \ldots, p_n\}$ be the set of atomic propositions occurring in $A$ (for simplicity, we can assume such a set to be ordered). We call an *atomic configuration for A* any formula of the form $\hat{p_1} \wedge \ldots \wedge \hat{p_n}$, where each $\hat{p_i}$ is either $p_i$ or $\neg p_i$. Let $\Delta_A = \{B \mid B \text{ is an atomic configuration for } A\}$. Then, we define

$$In_A = \forall\Box\exists\bigcirc\top \wedge (\forall\Box \bigvee_{B\in\Delta_A} B) \wedge \bigwedge_{B\in\Delta_A} (\exists\Diamond B \supset \forall\Box(\exists\bigcirc B)).$$

The formula $In_A$ will be used to ensure that: (first conjunct) the construction of a tableau for $A$ gives rise to models where seriality is satisfied; (second conjunct) the

[4]We are aware of the clash with the amplitudes, but we prefer to overload $\alpha$ and $\beta$ in order to keep the standard terminologies of both quantum computing and tableaux.

set of atomic propositions in $At(A)$ holding in each node is specified during the tableau construction; and (third conjunct) the assignments of atomic propositions to worlds is ruled by an underlying *S5*-model.

**Definition 11.** *A* Hintikka structure *is a structure* $(S, \bowtie_S)$ *such that for each state* $s \in S$ *in it:*

1. $\neg A \in s \Rightarrow A \notin s$;
2. $\alpha \in s \Rightarrow \alpha_1 \in s$ *and* $\alpha_2 \in s$;
3. $\beta \in s \Rightarrow \beta_1 \in s$ *or* $\beta_2 \in s$;
4. $\forall \bigcirc A \in s \Rightarrow$ *for all t such that sRt,* $A \in t$;
5. $\exists \bigcirc A \in s \Rightarrow$ *for some t such that sRt,* $A \in t$;
6. $\exists \Diamond A \in s \Rightarrow$ *there exists an s-branch b and for some* $t \in b$, $A \in t$;
7. $\forall \Diamond A \in s \Rightarrow$ *for each s-branch b there exists* $t \in b$ *such that* $A \in t$;
8. $©_j A \in s \Rightarrow$ *there exists* $t \in S_j$ *such that* $s \bowtie_S t$ *and* $A \in t$.

*We say that* $(S, \bowtie_S)$ *is a* Hintikka structure for a global formula $@_i A$ *if* $(A \wedge In_A) \in s$ *for some* $s \in S_i$.

**Theorem 2.** *A global formula* $@_j A$ *is satisfiable iff there is a Hintikka structure for* $@_j A$.

Proof. ($\Rightarrow$) For each agent $i$, the branching distributed local life-cycle can be easily turned into a local structure $\langle S_i, R_i, P_i \rangle$ and the sharing of events between agents can be used to define the relation $\bowtie_S$. For this direction, just observe that the conditions defining a Hintikka structure are less restrictive than those defining a QBDTL model.

($\Leftarrow$) Let $(S, \bowtie_S)$, where $S$ is a set of triples $\{\langle S_i, R_i, P_i \rangle\}_{i \in \mathsf{Id}}$, be a Hintikka structure. By definition, there is an associated *S5* model $\mathcal{M} = \langle \mathcal{Q}, \mathcal{U}, \mathcal{V} \rangle$ defined over a subset $\mathsf{Prop}'$ of the set $\mathsf{Prop}$ of atomic propositions. We can extend $\mathcal{M}$ by defining $p \in \mathcal{V}(q)$ for each $q \in \mathcal{Q}$ and for each $p \in \mathsf{Prop} \setminus \mathsf{Prop}'$. Now we define a QBDTL model $\mu = \langle \lambda, \mathcal{M}, \pi \rangle$, where $\lambda = \{\lambda_i\}_{i \in \mathsf{Id}}$ and each $\lambda_i$ is obtained by unwinding the corresponding local structure $\langle S_i, R_i, P_i \rangle$. $\pi$ is simply defined by using the mapping between states in $S$ and worlds in $\mathcal{M}$. Finally, the sharing of events in $\lambda$ is induced by the relation $\bowtie_S$. The proof that the model $\mu$ obtained is indeed a model for $@_j A$ is quite standard and proceeds by induction on the length of the formula. The only new case is the one concerning communication formulas and is handled by using condition 8 of Definition 11. △

*Tableau construction*

Here we will first describe a procedure for constructing a semantic tableau, and then show how to obtain a Hintikka structure from the tableau.

Given $i, j \in \mathsf{Id}$ with $i \neq j$, an *agent formula* is a formula of the form $i : A$, where $A$ is a formula in the local language of $i$; a *communication formula* is a formula of the form $[i : A]_j$ and a *synchronization formula* is a formula of the form $i \bowtie j$. A *tableau formula* is an agent formula, a communication formula or a synchronization formula. We will sometimes refer to agent and communication formulas concerning an agent $i$ as *i-formulas*.

The tableau consists of a set of nodes and oriented edges between nodes, where each node is labelled with a set of tableau formulas. Given a node $n$, we will denote by $L_n$ the *set of formulas labeling* $n$, and, for simplicity, often write $\varphi \in n$ for $\varphi \in L_n$. We say that an agent $i$ *occurs* in $n$ if at least one tableau formula labeled with $i$ occurs in $n$. An agent $i$ is *blocked* in $n$ if there exists a synchronization formula $[j : A]_i \in L_n$. We denote with $Cl_{\bowtie}(n)$ the *symmetric and transitive closure of the relation induced by the synchronization formulas in* $L_n$, e.g., if $(i \bowtie j), (j \bowtie k) \in L_n$, then $(i \bowtie k) \in Cl_{\bowtie}(n)$.

The tableau $\mathcal{T}$ for a global formula $@_i A_0$ is constructed inductively by labeling a root node $n_0$ with $i : A_0 \wedge In_{A_0}$ and by applying the following rules to nodes $n$ that are leaves of $\mathcal{T}$:

- $R_\alpha$: if $i : \alpha \in L_n$, then create a son $n_1$ of $n$ and define $L_{n_1} = L_n \cup \{i : \alpha_1, i : \alpha_2\}$;
- $R_\beta$: if $i : \beta \in L_n$, then create two sons $n_1$ and $n_2$ of $n$ and define $L_{n_l} = L_n \cup \{i : \beta_l\}$ for $l \in \{1, 2\}$;
- $R_{\bowtie}$: if $[i : \exists\Diamond A]_j \in L_n$, then:
  - create a son $n_1$ of $n$ and define $L_{n_1} = L_n \cup \{i : A, i \bowtie j\}$;
  - if $(i \bowtie j) \notin Cl_{\bowtie}(L_n)$, then also create a second son $n_2$ of $n$ and define $L_{n_2} = L_n \cup \{[i : \exists\bigcirc\exists\Diamond A]_j\}$.
- $R_{©}$: if $i : ©_j A \in L_n$, then create a son $n_1$ of $n$ and define:
  - $L_{n_1} = L_n \cup \{j : A \wedge In_{A_0}, i \bowtie j\}$ if $j$ does not occur in $n$;
  - $L_{n_1} = L_n \cup \{j : A\}$ if $(i \bowtie j) \in Cl_{\bowtie}(L_n)$;
  - $L_{n_1} = L_n \cup \{[j : \exists\Diamond A]_i\}$ otherwise;
- $R_{\bigcirc}$: this rule is applied only when applications of $R_\alpha$, $R_\beta$ and $R_{©}$ would not generate a new node. Let $\Gamma_i$ be the set of formulas in $L_n$ that do not involve $i$. For each agent $i$ occurring in $n$ and not blocked, such that $V_i = \{i : \exists\bigcirc A_1, \ldots, i : \exists\bigcirc A_k, [i : \exists\bigcirc B_1]_{j_1}, \ldots, [i : \exists\bigcirc B_h]_{j_h}, i : \forall\bigcirc C_1, \ldots, i : \forall\bigcirc C_m\}$ is the set of $i$-nexttime formulas in $L_n$, we create:
  - $k$ sons $n_l$ of $n$, for $1 \leq l \leq k$, and define $L_{n_l} = \{i : A_l, i : C_1, \ldots, i : C_m\} \cup \Gamma_i$; and
  - $h$ sons $n_l$ of $n$, for $1 \leq l \leq h$, and define $L_{n_l} = \{[i : B_l]_{j_l}, i : C_1, \ldots, i : C_m\} \cup \Gamma_i$;

  we will refer to the $k + h$ nodes created in this step as *$i$-sons* of $n$.

With respect to the construction in [8], we note that here we need some further rules. $R_{\bowtie}$ can be seen as a special case of $R_\beta$, where the creation of a witness for the formula expressing an eventuality also adds to the node a synchronization formula stating that the two agents have just communicated. $R_{©}$ is a special case of $R_\alpha$ that forces a communication to happen if the two agents are currently synchronized or postpones such a synchronization to some future node otherwise. Finally, we note that in our case $R_{\bigcirc}$ embodies also a $\beta$-step that represents the choice of the agent to be reactivated next.

We have two termination rules in the construction of a tableau:

- $Cl_1$: if $A \in L_n$ and $\neg A \in L_n$, then $n$ is a *closed* node and no further rules are applied to it;
- $Cl_2$: if a node $n_1$ is being created by an application of $R_\bigcirc$ to $n$ and there exists an ancestor $n'$ (also generated by $R_\bigcirc$) of $n$ such that $L_{n'} = L_{n_1}$ then do not create $n_1$ but create an edge oriented from $n$ to $n'$.

**Lemma 1.** *The construction of a tableau $\mathcal{T}$ for a tableau formula $\varphi$ terminates.*

Proof. Just observe that the set of formulas that can occur in nodes of $\mathcal{T}$ is finite and then the closure rule $Cl_2$ will eventually prevent the creation of new nodes. In fact, each logical formula is a subformula of $\varphi$, a negation of a subformula or a dual of a subformula (possibly prefixed by $\forall\bigcirc$ or $\exists\bigcirc$ and possibly in the form of a communication formula). Since the set of agent labels occurring in $\varphi$ is finite, the set of synchronization formulas is also finite. △

*Marking algorithm*

In the following, we will refer to a node $n$ as an $\alpha/\beta/©/\bowtie/\bigcirc$-node if a rule $R_\alpha/R_\beta/R_©/R_\bowtie/R_\bigcirc$ has been applied to such a node during the tableau construction. In particular, in the case of a $\bigcirc$-node $n$, we can be more specific and say that it is a $\bigcirc_i$-node for any agent $i$ that occurs in $n$ and is not blocked. Moreover, we say that a formula is a *future formula* if it has the form $i : \forall\Diamond A$, $i : \forall\bigcirc\forall\Diamond A$, $i : \exists\Diamond A$, $i : \exists\bigcirc\exists\Diamond A$, $[i : \exists\Diamond A]_j$ or $[i : \exists\bigcirc\exists\Diamond A]_j$.

The possible presence of cycles in a tableau $\mathcal{T}$ makes it not trivial to determine whether it is closed or not. Here we will define a two-step algorithm for marking the nodes of $\mathcal{T}$. In the end of the process, we will have that it is possible to extract from $\mathcal{T}$ a Hintikka structure for the root formula iff the root of $\mathcal{T}$ is not marked.

1. Apply iteratively the following rules as long as possible:
   - $M_{Cl}$: Mark every closed leaf.
   - $M_\alpha$: If $n$ is an $\alpha$-node and its son is marked, then mark $n$.
   - $M_\beta$: If $n$ is a $\beta$-node and both its sons are marked, then mark $n$.
   - $M_©$: If $n$ is a ©-node and its son is marked, then mark $n$.
   - $M_\bowtie$: If $n$ is a $\bowtie$-node and all its sons are marked, then mark $n$.
   - $M_\bigcirc$: If $n$ is a $\bigcirc$-node and for each agent $i$ occurring in $n$ and not blocked, at least one of the $i$-sons of $n$ is marked, then mark $n$.

   We can define a structure $(S, \bowtie_S)$ as follows. For each $i \in \mathsf{Id}$, $S$ contains a triple $\{S_i, R_i, P_i\}$ such that:
   - for each unmarked $\bigcirc_i$-node $n$ in $\mathcal{T}$, there is a state $n^i$ in $S_i$;
   - given $n^i \in S_i$, $A \in P_i(n^i)$ iff $A \in L_n$;
   - given $n^i_1, n^i_2 \in S_i$, $n^i_1 R_i n^i_2$ if there is a path $(n_1, m_0, \ldots, m_k, n_2)$ in $\mathcal{T}$ such that $m_0, \ldots, m_k$ are not $\bigcirc_i$-nodes.

   Finally, given $n^i_1 \in S_i$ and $n^j_2 \in S_j$, we say that $(n^i_1, n^j_2) \in \bowtie_S$ iff $n_1 \bowtie n_2 \in Cl_\bowtie(n_1)$ or $n_1 \bowtie n_2 \in Cl_\bowtie(n_2)$.

2. Apply the ranking algorithm described below to the structure obtained and mark each node containing an unranked occurrence of a future formula. If the resulting structure is such that the rules in step 1 become applicable again, then go back to step 1.

*Ranking algorithm*

The application of step 1 in the marking algorithm given above generates a structure that does not necessarily satisfy conditions 6 and 7 of Definition 11. This is due to the fact that, in the presence of cycles, the creation of a witness for a future eventuality could be postponed indefinitely.

Here we define a ranking algorithm aimed at facing this issue. Let $\mathcal{T}$ be a tableau obtained by applying step 1 of the marking algorithm above and let $N$ be the set of unmarked nodes in $\mathcal{T}$. The following algorithm will assign a positive rank $\rho(q, n)$ to some of the occurrences $q$ of future formulas in $n$. Intuitively, $\rho(q, n)$ represents the distance between $n$ and the node where a witness for the future formula $q$ occurs.

Let $q \in L_n$ be a not ranked future formula. The ranking algorithm consists of the following rules:

- $R_1$: If $n$ is a $\beta$-node or a $\bowtie$-node for $q$ and $n_1 \in N$, then $\rho(q, n) = 1$;
- $R_2$: If $n$ is a $\beta$-node or a $\bowtie$-node for $q$, $q'$ is the formula occurrence generated for the node $n_2$ and $\rho(q', n_2)$ is defined, then $\rho(q, n) = \rho(q, n_2) + 1$;
- $R_3$: If $n$ is a $\beta$-node or a $\bowtie$-node not for $q$ and $\rho(q, n_k)$ is defined for some $k \in \{1, 2\}$, then $\rho(q, n) = \rho(q, n_k) + 1$;
- $R_4$: If $n$ is an $\alpha$-node or a ©-node and $\rho(q, n_1)$ is defined, then $\rho(q, n) = \rho(q, n_1) + 1$;
- $R_5$: If $n$ is a $\bigcirc$-node and $q = i : \forall\bigcirc\forall\Diamond A$, then:
  - if $\rho(i : \forall\Diamond A, n_k)$ is defined for all $i$-sons $n_k$ of $n$, then $\rho(q, n) = max\{\rho(i : \forall\Diamond A, n_k)\}_k + 1$;
  - otherwise, if for some $j$ occurring and not blocked in $n$, $\rho(i : \forall\bigcirc\forall\Diamond A, n_k)$ is defined for all $j$-sons $n_k$ of $n$, then $\rho(q, n) = max\{\rho(i : \forall\bigcirc\forall\Diamond A, n_k)\}_k + 1$.
- $R_6$: If $n$ is a $\bigcirc$-node and $q = i : \exists\bigcirc\exists\Diamond A/[i : \exists\bigcirc\exists\Diamond A]_j$, then:
  - if $\rho(i : \exists\Diamond A/[i : \exists\Diamond A]_j, n_k)$ is defined for some $i$-son $n_k$ of $n$, then $\rho(q, n) = \rho(i : \exists\Diamond A/[i : \exists\Diamond A]_j, n_k) + 1$;
  - otherwise, if for some agent $h$ occurring and not blocked in $n$, $\rho(i : \exists\bigcirc\exists\Diamond A/[i : \exists\bigcirc\exists\Diamond A]_j, n_k)$ is defined for some $h$-son $n_k$ of $n$, then $\rho(q, n) = \rho(i : \exists\bigcirc\exists\Diamond A/[i : \exists\bigcirc\exists\Diamond A]_j, n_k) + 1$.

The ranking algorithm terminates since the set of future formulas in $N$ is finite and each formula is ranked at most once.

*From tableaux to Hintikka structures*

Here we show how a tableau constructed as described above for a satisfiable formula can be turned into a Hintikka structure.

**Theorem 3.** *Let $\mathcal{T}$ be a tableau for $@_i A$. If the root of $\mathcal{T}$ is not marked, then there exists a Hintikka structure for $@_i A$.*

Proof. The proof of this theorem mirrors the one of the analogous theorem in [8]; here we will just give a sketch of it and highlight some differences arising in our setting. Given a tableau for $@_i A$, a structure can be constructed as described in step 1 of the marking algorithm. Some care needs to be taken in order to deal with $\beta$-nodes. It may happen that, when unwinding a cycle of the tableau, some future formula is not fulfilled. This can be avoided by adopting a "fair" policy, which in the presence of a $\beta$-node alternates the choice between the two sons in such a way that if there is a branch containing an infinite number of occurrences of a $\beta$-node, then each of its sons occurs infinitely many times in the branch. Such a construction, the details of which can be found in [8], guarantees that conditions 6 and 7 of Definition 11 are satisfied. Finally, we note that the formula $In_A$, added as a conjunct to the root of the tableau, ensures that the valuation of atomic propositions meets the requirement of being determined by an underlying *S5* model. △

*7.2. Completeness of $\mathcal{N}$(QBDTL) via tableaux*

Here we will first prove a result concerning the relation between the formulas contained in a marked node of a tableau, built according to the procedure described in Section 7.1, and the formulas derivable in $\mathcal{N}$(QBDTL). Then we will use such a result in order to prove the weak completeness of $\mathcal{N}$(QBDTL).

Given a tableau $\mathcal{T}$ and a node $n$ of $\mathcal{T}$, the $\mathcal{N}$(QBDTL) *translation* $L_n^*$ of $L_n$ is obtained from $L_n$ by replacing, for each agent $i$, each $i$-formula $i : A$ or $[i : A]_j$ with the formula $(i, x_i, q_i) : A$ for some $x_i$ and $q_i$. We say that a node $n$ of $\mathcal{T}$ is *inconsistent* if $L_n^* \vdash_{\mathcal{N}(\mathsf{QBDTL})} \bot$.

**Lemma 2.** *If n is a marked node of a tableau, then n is inconsistent.*

Proof. The structure of this proof also follows quite closely the one given in [8] with respect to a Hilbert-style axiomatization for the logic *UB*. Here we recall the general structure and add some remarks concerning our case. We notice that $\mathcal{N}$(QBDTL) includes a subsystem that has been proved complete for *UB* [11]; thus for what concerns branching-time reasoning in the proof, we can just rely on the fact that the axioms used in [8] can be derived by using the rules of $\mathcal{N}$(QBDTL).

The proof proceeds by showing that the marked leaves of the tableau are inconsistent nodes (this is trivial as a leaf must contain both $A$ and $\neg A$ for some $A$) and by proving that the property of being inconsistent is propagated at each step when one moves towards the root. In particular, this is easily shown for those nodes that are marked through the step 1 of the marking algorithm. We just notice, as a peculiarity of our setting, that in the case of a ©-node $n$, if the son is generated by adding a formula of the form $[j : \exists\Diamond A]_i$, then for proving that $n$ is inconsistent we need to refer to the node,

possibly lower in the tableau than the son of $n$, where the synchronization between $i$ and $j$ actually occurs.

When nodes are marked through step 2 of the marking algorithm, a more complex construction is needed in order to show that there is a cycle that never fulfills an eventuality. This requires the definition of an invariant formula as a disjunction of the formulas characterizing each of the nodes in the cycle. In this case, inconsistency of the nodes is proved by using the induction rules $ind\forall$ and $ind\exists$.

Finally, we note that formulas of the form $(i, x, q) : In_A$ for any $A$ are derivable in our system by using the rule $ser\lhd$ for seriality, the quantum transformation rules $refl_U$, $symm_U$, $trans_U$ and $prop$, together with the interaction rules $U{\Rightarrow}R$ and $R{\Rightarrow}U$. This is used for proving that the initial node is inconsistent as well as in the case of ©-nodes that introduce new agents. △

The following theorem states that the converse of Theorem 3 holds, thus establishing the fact that the tableau procedure given in Section 7.1 is a decision procedure for QBDTL.

**Theorem 4.** *Let $@_i A$ be a satisfiable formula. Then the root $n_0$ of the tableau for $@_i A$ is not marked.*

Proof. Assume $@_i A$ is satisfiable, i.e., $@_i \neg A$ is not valid. By the soundness of $\mathcal{N}$(QBDTL), it follows $\nvdash_{\mathcal{N}(\mathsf{QBDTL})} (i, x, q) : A$, for arbitrary $x$ and $q$. Then, by Lemma 2, the root of the tableau for $@_i A$ is not marked. △

**Theorem 5 (Weak completeness).** *For every QBDTL formula $A$ and composed label $(i, x, q)$, if $\models (i, x, q) : A$, then $\vdash_{\mathcal{N}(\mathsf{QBDTL})} (i, x, q) : A$.*

Proof. Let $\models (i, x, q) : A$. Then $(i, x, q) : \neg A$ is unsatisfiable. By Theorems 2 and 3, the root of any tableau for $@_i \neg A$ is marked. By Lemma 2, we conclude $\vdash_{\mathcal{N}(\mathsf{QBDTL})} (i, x, q) : A$. △

## 8. Concluding Remarks

We have given the syntax and semantics of our Quantum Branching Distributed Temporal Logic QBDTL along with the deduction system $\mathcal{N}$(QBDTL) and examples of its application. Moreover, we have proved that $\mathcal{N}$(QBDTL) is sound and complete with respect to the given semantics. The proof of soundness is a standard induction, whereas the proof of the completeness theorem required a non-trivial refinement and extension of the complex proof, based on the definition of a tableau system, proposed in [8] for the branching-time logic *UB*.

In this paper, we have modeled quantum state transformations from an abstract perspective: in QBDTL, no reference to a specific quantum computation or to a notion of input/output of values is required. This allowed us to design a manageable high-level formalization oriented to modeling the behavior of quantum systems, but it is probably not enough if one wants to capture more complex properties such as entanglement. This does not mean that one has to completely convert the qualitative approach into a quantitative one (following the "philosophy" of quantum logic, cf. the discussion

in the introduction and the following sections). We believe that a distributed logic is a promising tool not only for the simple description of quantum states, but also to model the correct amount of quantitative information needed to capture properties like entanglement. In some sense, we aim at integrating into the QBDTL high-level perspective, able to model the "control" of quantum computation (which treats qubits and quantum gates as black-boxes), more detailed information about quantum data, so that it is possible to "look inside" the qubits and specifically model the quantitative behavior of some interesting unitary operators.

In QBDTL, we are as general as possible with respect to the application of transformations: in a local-life cycle the subtended temporal transition tree represents at each step all the possible unary gates that can be applied to the current state, while the synchronization mechanism between agents models all possible $n$-ary operators. It is well known that one can fix a complete computational basis (finite or infinite) of unitary operators and represent other operators in terms of the elements of such a basis. An infinite complete basis can be defined by taking all unary operators and the binary quantum gate cnot. As we remarked above, when the control qubit assumes some specific superpositional value (e.g., $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$), the output of the cnot is an entangled state. This suggests that restricting our perspective about arbitrary $n$-ary gates as synchronization operators to a single binary gate, the cnot, and lifting syntax and semantics to capture its behavior would provide us with all the ingredients needed to model entanglement. Following this standpoint, one can now view synchronizations exactly as control operators: a target qubit has to synchronize (by sharing an event) with the control qubit in order to perform its own, controlled evolution. Moreover, we observe that the notion of synchronization, in presence of entanglement, assumes a non-local (with respect to time) meaning: a synchronization that creates entanglement does not only represent the sharing of local events, but it also influences the subsequent events in the local life cycle of the involved agents. We thus aim to make the connection between agent synchronization and (possible) entanglement of qubits explicit.

More specifically, if an agent/qubit belongs to an entangled state, it is not possible to "keep out" its individual mathematical description, since the global mathematical description of an entangled state cannot be expressed in terms of local descriptions. Hence, in order to speak about entanglement, we need new syntactical constructs. A side effect is that also the interpretation of propositional symbols provided in Section 3 and in the examples has to be lifted in order to deal with the impossibility to express a local mathematical status. We are working at extending the logic with ad hoc propositional symbol of the shape $entg_i$, which models exactly the information that the agent $i$ is momentarily entangled and no local information about its quantum states can be provided. This is why we believe that this extension will have an interesting impact also on the notion of synchronization.

Finally, we are considering the explicit modeling inside QBDTL of measurement steps, which can be seen as a further class of transformations. We believe that these extensions will also enable the use of our approach for the analysis of quantum security protocols, which are based on entanglement phenomena [9], along the lines of what has been done with respect to classical security protocols by using DTL [6].

## References


[1] S. Abramsky and R. Duncan. A categorical quantum logic. *Math. Structures Comput. Sci.*, 16(3):469–489, 2006.

[2] A. Baltag and S. Smets. The logic of quantum programs. In *Proceedings of the 2nd QPL*, 2004.

[3] A. Baltag and S. Smets. LQP: the dynamic logic of quantum information. *Math. Structures Comput. Sci.*, 16(3):491–525, 2006.

[4] A. Baltag and S. Smets. Quantum logic as a dynamic logic. *Synthese*, 179(2):285–306, 2011.

[5] D. Basin, C. Caleiro, J. Ramos, and L. Viganò. Labelled tableaux for distributed temporal logic. *J. Log. and Comput.*, 19(6):1245–1279, Dec. 2009.

[6] D. Basin, C. Caleiro, J. Ramos, and L. Viganò. Distributed Temporal Logic for the Analysis of Security Protocol Models. *Theor. Comput. Sci.*, 412(31):4007–4043, July 2011.

[7] D. A. Basin, S. Matthews, and L. Viganò. Natural deduction for non-classical logics. *Studia Logica*, 60(1):119–160, 1998.

[8] M. Ben-Ari, A. Pnueli, and Z. Manna. The temporal logic of branching time. *Acta Inf.*, 20:207–226, 1983.

[9] C. H. Bennett and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing*, pages 175–179, 1984.

[10] G. Birkhoff and J. von Neumann. The logic of quantum mechanics. *Ann. of Math. (2)*, 37(4):823–843, 1936.

[11] C. Caleiro, L. Viganò, and M. Volpe. A labeled deduction system for the logic UB. In C. Sánchez, K. B. Venable, and E. Zimányi, editors, *Proceedings of the 20th International Symposium on Temporal Representation and Reasoning*, pages 45–53. IEEE Computer Society, 2013.

[12] U. Dal Lago, A. Masini, and M. Zorzi. On a measurement-free quantum lambda calculus with classical control. *Math. Structures Comput. Sci.*, 19(2):297–335, 2009.

[13] M. L. Dalla Chiara. Quantum logic. In *Handbook of Philosophical Logic III*: 427–469. Reidel, 1986.

[14] H.-D. Ehrich and C. Caleiro. Specifying communication in distributed information systems. *Acta Informatica*, 36:591–616, 2000.

[15] E. A. Emerson. Temporal and Modal Logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 995–1072. MIT Press, 1990.

[16] E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *J. Comput. Syst. Sci.*, 30(1):1–24, 1985.

[17] K. Engesser, D. M. Gabbay, and D. Lehmann. *A New Approach to Quantum Logic*. College Publications, 2007.

[18] K. Engesser, D. M. Gabbay, and D. Lehmann, editors. *Handbook of Quantum Logic and Quantum Structures*. Elsevier, 2009.

[19] D. M. Gabbay. *Labelled Deductive Systems*, volume 1. Clarendon Press, 1996.

[20] S. J. Gay, R. Nagarajan, and N. Papanikolaou. QMC: A Model Checker for Quantum Systems. In *Proceedings of CAV*, LNCS 5213. Springer, 2008.

[21] J.-Y. Girard. *Proof theory and logical complexity. Vol. 1*. Bibliopolis, 1987.

[22] P. Kouvaros and A. Lomuscio. Automatic verification of parameterised multi-agent systems. In *Proceedings of AAMAS*, 2013.

[23] A. Masini, L. Viganò, and M. Zorzi. A qualitative modal representation of quantum state transformations. In *Proceedings of the 38th ISMVL*. IEEE CS Press, 2008.

[24] A. Masini, L. Viganò, and M. Zorzi. Modal Deduction Systems for Quantum State Transformations. *Multiple-Valued Logic and Soft Computing*, 17(5-6):475–519, 2011.

[25] P. Mittelstaedt. The modal logic of quantum logic. *J. Philos. Logic*, 8(4):479–504, 1979.

[26] M. Nakahara and T. Ohmi. *Quantum Computing - From Linear Algebra to Physical Realizations.* CRC Press, 2008.

[27] M. Nielsen and I. Chuang. *Quantum computation and quantum information*. Cambridge University Press, 2000.

[28] H. Nishimura and M. Ozawa. Perfect computational equivalence between quantum turing machines and finitely generated uniform quantum circuit families. *Quantum Inf. Process.*, 8(1):13–24, 2009.

[29] D. Prawitz. *Natural Deduction: a Proof-Theoretical Study*. Almquist and Wiskell, 1965.

[30] S. Roman. *Advanced linear algebra.* Springer, 2008.

[31] P. Selinger and B. Valiron. A lambda calculus for quantum computation with classical control. *Math. Structures Comput. Sci.*, 16(3):527–552, 2006.

[32] A. K. Simpson. *The Proof Theory and Semantics of Intuitionistic Modal Logic*. PhD thesis, School of Informatics, University of Edinburgh, 1994.

[33] L. Viganò. *Labelled Non-Classical Logics*. Kluwer Academic Publishers, 2000.

[34] L. Viganò, M. Volpe, and M. Zorzi. Quantum state transformations and branching distributed temporal logic. In *Proceedings of the 21st International Workshop on Logic, Language, Information, and Computation (WoLLIC)*, LNCS 8652, pages 1–19. Springer, 2014.

[35] G. Winskel and M. Nielsen. Event structures. In *Handbook of Logic in Computer Science IV*. Oxford University Press, 1995.