An Algebraic Language for Specifying Quantum Networks

ANITA BUCKLEY, USI Lugano, Switzerland
PAVEL CHUPRIKOV, USI Lugano, Switzerland
RODRIGO OTONI, USI Lugano, Switzerland
ROBERT SOULÉ, Yale University, USA
ROBERT RAND, University of Chicago, USA
PATRICK EUGSTER, USI Lugano, Switzerland

Quantum networks connect quantum capable nodes in order to achieve capabilities that are impossible only using classical information. Their fundamental unit of communication is the *Bell pair*, which consists of two entangled quantum bits. Unfortunately, Bell pairs are fragile and difficult to transmit directly, necessitating a network of repeaters, along with software and hardware that can ensure the desired results. Challenging intrinsic features of quantum networks, such as dealing with resource competition, motivate formal reasoning about quantum network protocols. To this end, we developed BellKAT, a novel specification language for quantum networks based upon Kleene algebra. To cater to the specific needs of quantum networks, we designed an algebraic structure, called BellSKA, which we use as the basis of BellKAT's denotational semantics. BellKAT's constructs describe entanglement distribution rules that allow for modular specification. We give BellKAT a sound and complete equational theory, allowing us to verify network protocols. We provide a prototype tool to showcase the expressiveness of BellKAT and how to optimize and verify networks in practice.

CCS Concepts: • **Networks** → **Formal specifications**; • **Hardware** → *Quantum technologies*; • **Theory of computation** → **Formal languages and automata theory**.

Additional Key Words and Phrases: Kleene algebra, quantum networks, entanglement

ACM Reference Format:
Anita Buckley, Pavel Chuprikov, Rodrigo Otoni, Robert Soulé, Robert Rand, and Patrick Eugster. 2024. An Algebraic Language for Specifying Quantum Networks. *Proc. ACM Program. Lang.* 8, PLDI, Article 200 (June 2024), 23 pages. https://doi.org/10.1145/3656430

1 INTRODUCTION

Quantum networks are distributed systems providing communication services to distributed quantum applications. They bring numerous advantages over what is possible in a classical setting, improving the capabilities of existing applications and allowing for fundamentally new ones to arise. Most notable benefits are related to enhanced communication capabilities leading to increased security, with examples including unconditionally secure client-server communication, blind cloud computing, and secure multi-party computation [Gyongyosi and Imre 2022; Pirandola et al. 2020; Wang et al. 2023]. Distribution is also essential to expand quantum computation beyond capabilities of individual quantum-enabled computers to quantum clusters [Kozlowski and Wehner 2019].

The basic unit of communication between two nodes in a quantum network is a distributed *Bell pair* or *EPR pair* (named after Bell [1964] and Einstein, Podolsky, and Rosen [1935]) – a pair of quantum bits (qubits), one at each node, that are *entangled*. Entangled qubits are correlated

Authors' addresses: Anita Buckley, USI Lugano, Lugano, Switzerland, anita.buckley@usi.ch; Pavel Chuprikov, USI Lugano, Lugano, Switzerland, pavel.chuprikov@usi.ch; Rodrigo Otoni, USI Lugano, Lugano, Switzerland, otonir@usi.ch; Robert Soulé, Yale University, New Haven, USA, robert.soule@yale.edu; Robert Rand, University of Chicago, Chicago, USA, rand@uchicago.edu; Patrick Eugster, USI Lugano, Lugano, Switzerland, eugstp@usi.ch.

in a much stronger way than can be achieved with classical information. As entanglement is a fundamentally quantum property, quantum networks must operate within the constraints of quantum hardware, one of which is decoherence – quick degradation of quantum state quality over time. The issues attached to decoherence are compounded with the fact that it is not possible to copy unknown quantum states, which together with noise and qubit loss represent major obstacles to realizing long-distance quantum communication in the spirit of store-and-forward as in classical networks. These factors turn end-to-end distribution of Bell pairs, the core quantum network service, into a stateful task that requires non-trivial runtime coordination between *distributed* nodes. Moreover, it includes steps like distillation or initial entanglement generation that have intrinsically high probability of failure.

he need for distributed coordination, statefulness, and failure-prone primitive operations all contribute to the complex behavior of quantum network *protocols* – distributed programs that govern end-to-end distribution of Bell pairs among remote nodes [Illiano et al. 2022; Kozlowski et al. 2023]. The scarcity of resources in quantum networks (e.g., memory and communication qubits) prompts intensive resource sharing and competition among quantum network protocols executing in parallel, further increasing protocol complexity. This makes *formal reasoning* about the network's behavior critical to enable protocol optimization, efficient compilation to hardware, and safe co-existence of multiple protocols, in addition to the verification of correctness properties of individual protocols (e.g., that the Bell pairs are indeed being generated between the right nodes). Quantum networks already require tight coordination, and are thus a natural fit for *logically* centralized architectures, similar to software-defined networking (SDN), which allow reasoning about global behavior.

To enable global behavior analysis of quantum network protocols, we propose a novel specification language, called BellKAT. We take inspiration from the extensive body of work done with regard to specification of classical networks, particularly NetKAT [Anderson et al. 2014], but present a language with distinct features that cater to the fundamentally new way in which communication occurs in a quantum setting. BellKAT is built on a solid mathematical foundation, called BellSKA, which is a novel algebraic structure enabling equational reasoning about quantum network protocols. The BellSKA structure is in turn based on Kleene algebra (KA) [Kozen 1994], specifically synchronous Kleene algebra (SKA) [Prisacariu 2010], and is designed to tackle round-based behavior, which is inherent to quantum networks as currently envisioned by the Quantum Internet Research Group (QIRG)[1] of the Internet Research Task Force (IRTF). With BellKAT, it is possible to specify and check properties such as reachability and traffic isolation, as well as manage network resources by predicting occurrences and effects of race conditions. BellKAT can also form the foundation for a unified high-level interface between control and data plane in quantum networks, similar to what OpenFlow [McKeown et al. 2008], and later P4 [Bosshart et al. 2014] and P4-Runtime [P4 API Working Group 2021] became for classical networks.

In addition to formally defining BellKAT, we present soundness and completeness results for its axioms with respect to its denotational semantics. Concretely, we prove soundness and completeness for both a single round, with respect to end-to-end behavior, and for multiple rounds, with respect to execution traces. We design BellKAT to favor expressiveness in order to faithfully represent quantum network behavior. Lastly, we implemented a prototype tool that enables the practical specification of protocols in BellKAT and allows users to verify the effects of protocol executions.

To summarize, the contributions of this paper are the following:

(1) We propose a novel language to specify quantum networks, BellKAT, and the algebraic structure that underpins it, BellSKA.

[1]See https://irtf.org/qirg.

(2) We prove multi- and single-round soundness and completeness of BellKAT's axioms w.r.t. their corresponding semantics, with these results forming the basis for equational reasoning.
(3) We show that the equality of isolated protocol executions is decidable.
(4) We present a prototype tool that showcases how automated reasoning of quantum network specifications written in BellKAT can be carried out.

The remainder of the paper is structured as follows: In Section 2, we introduce the necessary background and provide a literature review of quantum networks and of approaches for specification and verification of networks. In Section 3, we present an overview of our formalization. In Section 4, we formally describe all aspects of BellKAT and BellSKA and prove their properties. In Section 5, we discuss the relevant quantum network properties and how they can be verified. Finally, in Section 6, we present our closing remarks and future work.

2 BACKGROUND AND RELATED WORK

In this section, we first introduce the basic concepts surrounding quantum networks, together with their advantages and limitations. Then, we describe the concrete network model proposed by the IRTF's QIRG. Finally, we discuss existing approaches for network specification and verification.

Quantum Networks. Quantum networks are governed by the laws of quantum mechanics, which impose constraints on their design while enabling fundamentally new capabilities that are impossible when only using classical information. The *no-cloning theorem* prevents copying unknown quantum states without irreversibly altering them [Nielsen and Chuang 2011]. This means that it is impossible to forward quantum information following the receive-copy-retransmit paradigm of classical network switches. On the positive side, the no-cloning theorem makes quantum communication inherently secure, allowing for novel applications that are resistant to eavesdropping and man-in-the-middle attacks [Pirandola et al. 2020].

Our work focuses on the core service provided by quantum networks, namely generation and distribution of entangled quantum states. Bell pairs, also called Bell states, form the basis of communication, since all distributed quantum applications (teleportation being most notable) can be built on top of (distributed) Bell pairs [Briegel et al. 1998; Kozlowski et al. 2023]. Bell pairs are maximally entangled states, having the strongest possible quantum correlations among two-qubits, which makes them easier to create, distribute, and apply error handling to. For instance, with the entanglement-based quantum key distribution (QKD) protocol E91 [Ekert 1991], which has inherent source-independent security, it is possible to avoid the trusted relays that pose security risks in long-distance implementations of the original QKD protocol BB84 [Bennett and Brassard 2014].

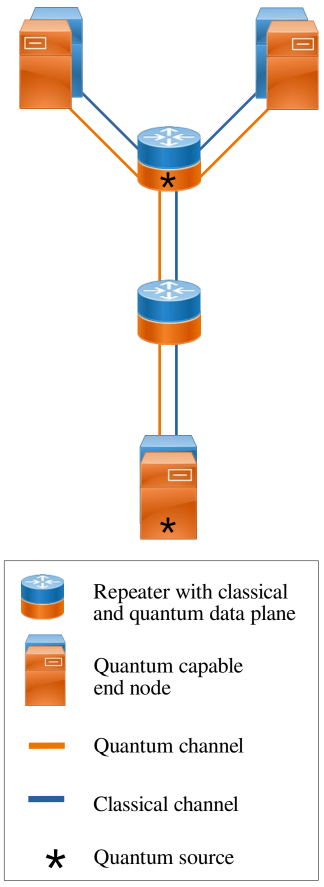


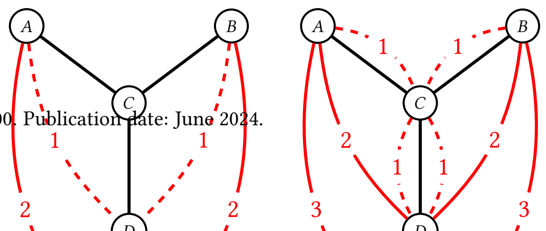
Fig. 1. Illustration of a quantum network with five nodes.

In the following, we provide a high-level overview of key components of a quantum network [Kozlowski and Wehner 2019], which are illustrated in Figure 1. Quantum applications are run on quantum capable **end nodes**. They must be capable of receiving and processing entangled pairs of qubits. Most architectures rely on hardware that uses a dedicated subset of qubits, called *communication qubits*, to generate distributed entanglement; once a Bell pair is generated, the constituent qubits can be transferred into memory. A Bell pair is first generated locally by a **quantum source**, and then one or both of the entangled qubits are transmitted over **quantum channels**. However, the probability that a photon representing a qubit reaches the target node by direct transmission decreases exponentially with the distance. Hence, entanglement distribution over long distances is implemented using **quantum repeaters**, making them the core active building blocks of quantum networks [Briegel et al. 1998; Towsley 2021]. A quantum repeater acts as an intermediary node between two other nodes, consuming the Bell pairs it shares with each of the other two nodes in order to create a new Bell pair connecting them. To illustrate, the top two end nodes in the network shown in Figure 1 can be entangled via the repeater to which they are connected. This physical process is known as *entanglement swapping*, and it can be extended with multiple quantum repeaters acting as intermediate nodes. Decoherence (quantum state degradation) is addressed by *entanglement distillation* (also called purification), the process of generating a single Bell state from two or more imperfect entangled states. When distillation succeeds, the quality of the state is improved. Distillation may, however, probabilistically fail, thus it substantially increases the resource demands [Pompili et al. 2021]. In order to distinguish between successful attempts and failures, *heralded* schemes are deployed that announce when attempts succeed [Wehner et al. 2018]. The final crucial components are **classical channels**, as entanglement generation schemes depend on tight synchronization and timely signaling among remote network entities.

Network Model. This paragraph describes our network model for end-to-end Bell pair creation, which follows the principles of quantum Internet outlined by IRTF's QIRG [Kozlowski et al. 2023].

A quantum network has a classical control plane, as well as two data planes – one classical and one quantum. The control plane is responsible for discovering the network topology, managing resources, and coordinating the actions of the nodes. In addition, it also manages routing and signaling. The classical data plane handles the forwarding of classical packets, while the quantum data plane oversees the generation and distribution of Bell pairs. Several authors propose to embed quantum networks within classical networks and use the existing infrastructure to send and receive control messages [Illiano et al. 2022; Kozlowski and Wehner 2019; Rabbie et al. 2022]. This may be achieved by adding a quantum data plane to the classical data plane to build repeaters, and by using both classical and quantum physical channels to connect quantum-capable nodes. End-to-end Bell pair distribution between distant nodes is a stateful, distributed task. The task is initiated by a set of requests for Bell pair creation, each indicating the two endpoints and quality of service parameters. For each request, a *quantum virtual circuit* [Illiano et al. 2022] between the corresponding end-points needs to be created, which entails identifying available paths between those end-points. An entanglement routing scheme then (with the use of a traffic engineering function, taking into account the capacity of the routers and channels, and the resources already consumed by other virtual circuits) computes the optimal path, i.e., the best sequence of repeaters and links that guarantees the requested quality of service. Finally, the entanglement *generating rules* are installed into the data plane of each quantum repeater on the paths.

This work focuses on the specification of these generating rules, which is the way entanglement generating protocols are implemented, enabling formal reasoning. Such specification requires sensible hardware abstractions for quantum networks, similar to those found in classical networks [Anderson et al. 2014]. The following abstract building blocks, which we call actions, are the cornerstone of the specification of the generating rules: create a Bell pair locally at a source, transmit qubits of



a Bell pair over a quantum channel, <u>sw</u>ap Bell pairs via repeaters, and <u>di</u>still Bell pairs. We describe these actions using the following example, to specify two different entanglement generating protocols for the network in Figure 1. The two protocols in Figure 2 generate Bell pairs between nodes A and E and nodes B and E, which we denote as $A{\sim}E$ and $B{\sim}E$, but with different capabilities at the source C. At node E, both protocols act in the same manner, creating two Bell pairs and sending half of each to D. Protocol (a) has node C distribute a Bell pair between A and D and another between B and D, to obtain $A{\sim}D$ and $B{\sim}D$, then performs two swaps at D with $E{\sim}D$, resulting in $A{\sim}E$ and $B{\sim}E$. In contrast, protocol (b) transmits half of each Bell pair created at C to a neighbor and keeps the other half in C's memory, leading to $C{\sim}A$, $C{\sim}B$, and two copies of $C{\sim}D$, then performs two swaps at C, to obtain $A{\sim}D$ and $B{\sim}D$, and finally two swaps at D, resulting in $A{\sim}E$ and $B{\sim}E$. In Figure 2 the physical paths connecting nodes A and B with node E are drawn in black, and the red virtual links depict the order in which Bell pairs are being generated.

Adding parallelism among subprotocols can immediately lead to contention between them, as illustrated with the next example. Assume that, due to network constraints, the first round of protocol (a) only succeeds in transmitting Bell pairs $A{\sim}D$, $B{\sim}D$, and one copy of $D{\sim}E$ (instead of two). The missing $D{\sim}E$ Bell pair will lead to resource competition. Then, when the second round performs two swaps at D in parallel (one that requires $A{\sim}D$, $D{\sim}E$ to produce $A{\sim}E$ and the other that requires $B{\sim}D$, $D{\sim}E$ to produce $B{\sim}E$), only one of the two swaps will succeed (i.e., either $A{\sim}E$ or $B{\sim}E$ will be produced). On the other hand, if the second round performs the two swaps sequentially on the same input Bell pairs $A{\sim}D$, $B{\sim}D$, $D{\sim}E$, e.g., the swap that aims to produce $B{\sim}E$ is called after the swap for $A{\sim}E$, the first swap always succeeds and in this case outputs $A{\sim}E$.

Given the intricacies above, we strive to answer the following questions, which naturally arise:

- Does protocol (a) always produce Bell pairs $A{\sim}E$ and $B{\sim}E$?
- Are protocols (a) and (b) equivalent?
- Is protocol (a) an optimized version of protocol (b)?

Algebraic Specification. It is natural to ask whether we can draw analogies with existing approaches for the specification and verification of classical networks. In order to benefit from strong mathematical foundations, we opt for an algebraic approach. In classical networks, this line of research originated with the seminal work of Anderson et al. [2014] on NetKAT, a high-level programming language and logic for specifying and reasoning about packet-switched networks. NetKAT is an instance of Kleene algebra with tests (KAT) whose equational theory is sound and complete with respect to its denotational semantics. The foundation of KAT is Kleene algebra (KA) [Kozen 1994], which has been used for decades as the algebraic structure of finite automata and regular events. KAT is an extension of KA with Boolean actions (called tests) that increases its expressiveness, to the extent that KAT subsumes propositional Hoare logic [Kozen 1997; Kozen and Smith 1997]. Quantum actions, the building blocks of our language, are fundamentally different from NetKAT actions, whose assignments and tests are abstractions for packet field modifications and filters, respectively. In addition, *quantum packets* that represent Bell pairs (network resources) in quantum networks have no counterpart in classical networks, which instead contain classical packets (information carriers). Furthermore, in quantum networks concurrent behaviors cannot be ignored – contrary to the forwarding of packets in classical networks – since, to produce a single end-to-end Bell pair, we need to create and distribute many entangled pairs among the intermediate nodes. On top of that, multiple nodes simultaneously compete for the same Bell pairs. These features, combined with the fact that our actions take several Bell pairs as inputs while NetKAT programs act instead on a single input packet at a time, prevent us from drawing direct analogies with the stateless network model of NetKAT.

Concurrent NetKAT (CNetKAT) [Wagemaker et al. 2022] is an algebraic language for modeling and analyzing stateful, concurrent classical networks. CNetKAT combines the language models of NetKAT and partially-observable concurrent Kleene algebra (POCKA) [Wagemaker et al. 2020], in which tests are replaced with observations that are more suitable for addressing concurrency [Kappé et al. 2020]. Parallel CNetKAT programs can execute at different speeds, leading to arbitrary interleavings. Due to resource and time constraints in quantum networks, we need tight synchronization that limits interleaving, and the protocol can be seen as progressing in rounds. Synchronous Kleene algebra (SKA) [Prisacariu 2010] allows for an alternative way of handling concurrency by layering synchronous actions into rounds, providing a formalism more suitable to our setting. However, pure SKA is not capable of faithfully expressing the complex orderings of actions in entanglement-generating protocols.

By contrast, Peng et al. [2022] have successfully applied KA to reason about quantum programs, although not in a distributed setting. Challenges related to quantum applications in a distributed setting have been discussed by Buckley et al. [2023], but no solutions have been proposed.

In summary, compositionality of algebraic structures fits well with the need for scalable and robust quantum network architectures. However, there are many features that existing classical KA structures cannot cater to, as discussed above. The nature of Bell pairs, being distributed across two different network nodes and as constituting undirected network resources, further contrasts with classical packet forwarding.

3 BELLKAT OVERVIEW

This section presents the principles of our network specification approach. We use the quantum network illustrated in Figure 1 and detailed in Figure 2 as a running example, which motivates and introduces the key elements of our language. Abstractly, a quantum network protocol can be thought of as an automaton that coordinates the distribution of entangled qubits across different nodes, along both physical and virtual quantum links. This distribution is done through generating rules, which are faithful abstractions of hardware behavior.

Network behavior. We divide the entanglement generating protocols in the spirit of Van Meter and Touch [2013] into *rounds*, each representing a time window. Rounds contain actions, which we refer to as basic actions, that are executed synchronously, i.e., they are performed in the same time window. Progression from one round to the next is represented by sequential composition, while iteration across rounds is encoded using the Kleene star. Basic actions of a single round can only act on the set of Bell pairs present in the network at the start of the corresponding time window, with race conditions emerging if resources are insufficient, i.e., not enough Bell pairs are available. In order for an individual basic action to be successfully executed it must first *acquire* a specific set of Bell pairs, said to be *required* by that action, from those available in the corresponding round, and after that use these Bell pairs to generate new entangled pairs. If the required set of Bell pairs cannot be acquired due to insufficient number of Bell pairs, the action is not executed and no Bell pairs are consumed, leaving them available to other actions in the same round. If the action acquires the required Bell pairs but fails to generate a new pair, the acquired Bell pairs are destroyed. A heralding classical signal is sent from the quantum data plane to acknowledge the success or failure of each action. The next round then proceeds in the same manner, acting on the set of Bell pairs either produced or not consumed by the prior round.

Bell pairs. The fundamental unit in quantum networks are Bell pairs, like packets are in classical networks. Yet, unlike packets, qubits carry no headers, therefore control information needs to be sent via separate classical channels. The nodes then correlate this information with the qubits stored in their memory. Another difference is that a Bell pair consists of two qubits distributed across two nodes, and these nodes must coordinate to ensure that they are operating on qubits that belong to the same Bell pair. The identity of nodes entangled via a Bell pair should be properly shared across the network, hence we assume that nodes have unique efficiently representable identifiers. We write $A{\sim}C$ or $C{\sim}A$ to denote a Bell pair between nodes A and C. For a given qubit in a Bell pair, the node of the other qubit can dynamically change with each action at runtime, making actions stateful, as opposed to the classical mostly stateless packet switching.

Actions. A *basic action* has form $r \triangleright o$, whose effect entails consuming a multiset of required Bell pairs r and producing a multiset of Bell pairs o. For example, a swap of $A{\sim}D$ and $D{\sim}E$ at node D, denoted $\mathsf{sw}\langle A{\sim}E \mathbin{@} D\rangle$, is represented as $\{\!\{A{\sim}D, D{\sim}E\}\!\} \triangleright \{\!\{A{\sim}E\}\!\}$, and a local creation at node E, denoted $\mathsf{cr}\langle E\rangle$, can be represented as $\emptyset \triangleright \{\!\{E{\sim}E\}\!\}$. Similarly, $\mathsf{tr}\langle C \rightarrow A{\sim}D\rangle$ represents physically forwarding one qubit of the Bell pair $C{\sim}C$ to node A and the other qubit to node D, and $\mathsf{tr}\langle C \rightarrow C{\sim}A\rangle$ represents physically forwarding one qubit of the Bell pair $C{\sim}C$ to node D and keeping the other qubit in C's memory; the former can be written as $\{\!\{C{\sim}C\}\!\} \triangleright \{\!\{A{\sim}D\}\!\}$, and the latter as $\{\!\{C{\sim}C\}\!\} \triangleright \{\!\{C{\sim}A\}\!\}$. Modeling failures of actions is necessary to capture decoherence and loss, as well as inherently probabilistic operations like distillation, where $\mathsf{di}\langle A{\sim}D\rangle$ inputs two copies of $A{\sim}D$ and returns $\{\!\{A{\sim}D\}\!\}$ or $\emptyset$. We model such failures as $r \triangleright o + r \triangleright \emptyset$, where $+$ represents nondeterministic choice. We remark that our actions also abstract away the control operations over the classical network. For example, Bell state measurement performed in the repeater during entanglement swapping requires two bits of classical control signals to be exchanged.

Policies. BellKAT policies are specifications of entanglement generating protocols. Intuitively, policies p and q can be thought of as functions that take a multiset of Bell pairs as input and return two multisets of Bell pairs: those that were produced and those that were not consumed. Within a single round, the produced Bell pairs and the Bell pairs that were not consumed are kept separate, since the fresh Bell pairs cannot be consumed in the round in which they were generated due to timing constraints. When the round is finished, all Bell pairs in the network are together made available to the next round. Concretely, single round policies are functions from $\mathcal{M}(\mathsf{BP})$ to $\mathcal{P}(\mathcal{M}(\mathsf{BP}) \times \mathcal{M}(\mathsf{BP}))$, and multi-round policies are functions from $\mathcal{M}(\mathsf{BP})$ to $\mathcal{P}(\mathcal{M}(\mathsf{BP}))$; where elements of $\mathcal{M}(\mathsf{BP})$ are multisets of Bell pairs, and the ranges are powersets due to nondeterminism. In order to build more sophisticated policies, we introduce policy composition operators. When acting on the input multiset, the union operator $(p + q)$ yields the union of the sets produced by p and q. The sequential composition operator $(p \mathbin{;} q)$ first applies p to the input multiset and then applies q to each multiset produced by p. The Kleene star operator $(p^{\star})$ expresses iteration. Furthermore, operators $(p \cdot q)$ and $(p \parallel q)$ model ordered and parallel composition of policies, respectively, that occur synchronously within a single round. Here, operator $\cdot$ imposes that p has preference over q in accessing the available Bell pairs, while $\|$ allows for resource competition.

Policies of our running example. The entanglement generating protocol in Figure 2a can be expressed with the following policy:

$$\begin{aligned}&\big(\mathsf{cr}\langle C\rangle \parallel \mathsf{cr}\langle C\rangle \parallel \mathsf{cr}\langle E\rangle \parallel \mathsf{cr}\langle E\rangle\big);\\ &\big(\mathsf{tr}\langle C \to A{\sim}D\rangle \parallel \mathsf{tr}\langle C \to B{\sim}D\rangle \parallel \mathsf{tr}\langle E \to E{\sim}D\rangle \parallel \mathsf{tr}\langle E \to E{\sim}D\rangle\big);\\ &\big(\mathsf{sw}\langle A{\sim}E \,@\, D\rangle \parallel \mathsf{sw}\langle B{\sim}E \,@\, D\rangle\big)\end{aligned} \tag{P1}$$

Similarly, the generating protocol in Figure 2b can be expressed with the following policy:

$$\begin{aligned}&\big(\mathsf{cr}\langle C\rangle \parallel \mathsf{cr}\langle C\rangle \parallel \mathsf{cr}\langle C\rangle \parallel \mathsf{cr}\langle C\rangle\big);\\ &\big(\mathsf{tr}\langle C \to C{\sim}A\rangle \parallel \mathsf{tr}\langle C \to C{\sim}B\rangle \parallel \mathsf{tr}\langle C \to C{\sim}D\rangle \parallel \mathsf{tr}\langle C \to C{\sim}D\rangle \parallel \mathsf{cr}\langle E\rangle \parallel \mathsf{cr}\langle E\rangle\big);\\ &\big(\mathsf{sw}\langle A{\sim}D \,@\, C\rangle \parallel \mathsf{sw}\langle B{\sim}D \,@\, C\rangle \parallel \mathsf{tr}\langle E \to E{\sim}D\rangle \parallel \mathsf{tr}\langle E \to E{\sim}D\rangle\big);\\ &\big(\mathsf{sw}\langle A{\sim}E \,@\, D\rangle \parallel \mathsf{sw}\langle B{\sim}E \,@\, D\rangle\big)\end{aligned} \tag{P2}$$

Histories. Quantum histories record the behaviors that the generating rules produce. Concretely, they capture the order of operations in a given execution of the protocol. To illustrate, the execution histories of protocols P1 and P2 can be seen in Figure 3 below. Unlike NetKAT histories, which encode paths of classical packets, our histories record the basic actions that execute successfully. Histories are not needed for protocol implementation and execution, they are, however, very useful when carrying out verification tasks, as detailed in Section 5.

Tests. Our policies can be guarded by tests, which act as additional explicit checks over the available Bell pairs. These tests check for the absence of multiset elements, in addition to the checking for required Bell pairs that is inherent to every action. This allows us to capture conditional behaviors with expressions of form $[t]p + [t']p'$, with $[t]p$ denoting that policy p is guarded by test t.

Iterative policies with Kleene star. Due to the probabilistic nature of operations, early generations of quantum networks will inevitably employ the strategy of repeated attempts of distillation and creation [Van Meter et al. 2011]. This was demonstrated by Pompili et al. [2021] realizing the first multinode quantum network, in which a pair of directly connected quantum nodes repeatedly attempts to generate an entangled pair until the heralding signal announcing success is received. Their protocol involves repeated rounds of distillation – in our language, this iterative behavior is expressed

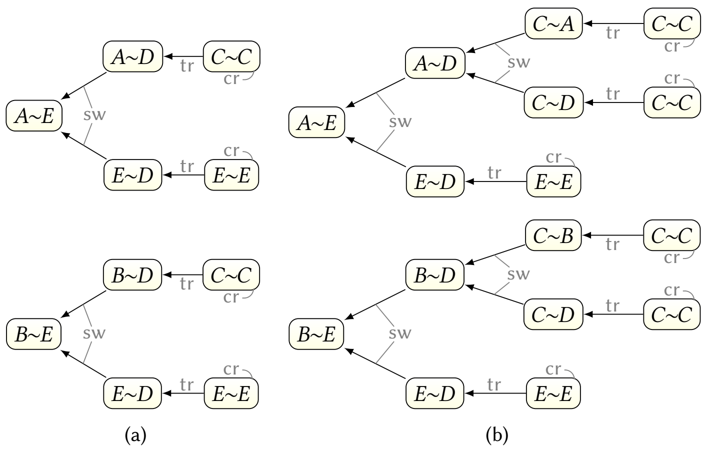


Fig. 3. Sample execution histories of the two protocols in Figure 2, generating Bell pairs $A{\sim}E$ and $B{\sim}E$ in different ways. The histories shown in (a) have three rounds and one swap each, and the histories shown in (b) have four rounds and two swaps each. Actions are annotated in gray, but they are not part of the execution histories.

with Kleene star. To showcase the expressiveness of our language, we now specify the protocol of Pompili et al. as a guarded iterative policy. We make use of the network in Figure 2 (a), which has all the necessary components. The goal is end-to-end entanglement distribution between nodes A and E, by swapping $A{\sim}D$ and $E{\sim}D$ at node D. In this scenario, however, before performing the swap, we improve the quality of entangled states $A{\sim}D$ and $E{\sim}D$ with distillation. To achieve this, we first transmit two Bell pairs from the source C to the nodes A and D, and then distill $A{\sim}D$. The distillation requires two copies of $A{\sim}D$ to produce one copy of the same Bell pair of a higher quality. This is expressed by the policy p_d:

$$p_d = \big(\mathsf{cr}\langle C\rangle \parallel \mathsf{cr}\langle C\rangle\big) \mathbin{;} \big(\mathsf{tr}\langle C \to A{\sim}D\rangle \parallel \mathsf{tr}\langle C \to A{\sim}D\rangle\big) \mathbin{;} \mathsf{di}\langle A{\sim}D\rangle$$

Since distillation is an inherently probabilistic operation, p_d must be repeatedly executed until the success signal arrives. With b denoting the test that checks for the absence of $A{\sim}D$, the while-loop of repeated executions is expressed with Kleene star as specified in the policy $p_d \mathbin{;} ([b]\, p_d)^\star$. Similarly to p_d, an improved Bell state $E{\sim}D$ can be generated by guarded iterations of the policy p'_d:

$$p'_d = \big(\mathsf{cr}\langle E\rangle \parallel \mathsf{cr}\langle E\rangle\big) \mathbin{;} \big(\mathsf{tr}\langle E \to E{\sim}D\rangle \parallel \mathsf{tr}\langle E \to E{\sim}D\rangle\big) \mathbin{;} \mathsf{di}\langle E{\sim}D\rangle$$

This leads to the policy $p'_d \mathbin{;} ([b']\, p'_d)^\star$, where b' tests for the absence of $E{\sim}D$. Then the repeater swap protocol of Pompili et al. is expressed with the policy below:

$$\Big(\big(p_d \mathbin{;} ([b]\, p_d)^\star\big) \parallel \big(p'_d \mathbin{;} ([b']\, p'_d)^\star\big)\Big) \mathbin{;} \mathsf{sw}\langle A{\sim}E \mathbin{@} D\rangle \tag{P3}$$

4 LANGUAGE

BellKAT is designed to be a simple but expressive specification language for quantum networks. Its semantics satisfies the axioms of our BellSKA algebraic structure together with additional axioms that capture domain-specific features of entanglement distribution in quantum networks. This section presents the syntax, semantics, and equational theory in a formal manner. For brevity, we omit the detailed proofs, they can be found in the long version of this paper.[2]

4.1 Preliminaries

A Bell pair bp is represented by an unordered pair of nodes. We assume a finite number of nodes $A_1, \ldots, A_k$. Bell pairs may have additional classical metadata, like tags denoting the action by which they were produced, or a timestamp (which we omit here for simplicity). A quantum network must keep track of the Bell pairs it contains. If a multiset $a \in \mathcal{M}(\mathsf{BP})$ contains n_{ij} Bell pairs $A_i{\sim}A_j$, we say that the multiplicity of $A_i{\sim}A_j$ in a is n_{ij}. We will be using the common terminology of multisets (also called msets or bags) and relations between them. In particular, we write $a \uplus a'$ for additive union of multisets and $a \backslash a'$ for multiset difference. When nodes perform a basic action $r \triangleright o$, they only need a partial view of the Bell pairs in the network, in order to determine whether the network contains the required multiset of Bell pairs r. This permits us to define the network state as a partial function $A_i{\sim}A_j \mapsto n_{ij}$. We will use the terms multiset and (total) network state interchangeably.

Tests act as guards for policies. They are positive Boolean terms over *atomic propositions*, denoting multiset absence, with an additional operation $\uplus$. Here, test $b \in \mathcal{M}(\mathsf{BP})$ has the semantics that $b \not\subseteq a$ for a given input multiset a. In particular, $\emptyset$ is a valid test which is false on any multiset. On the other hand, $\mathbb{1}$ signifies the test which is true on any multiset, i.e., *no test*.

An *atomic action* $[t]r \blacktriangleright o \in \Pi$ behaves the same as a basic action when its required Bell pairs are available and test t succeeds, meaning that it consumes the multiset r and outputs the multiset o together with the unconsumed Bell pairs. On the other hand, when this is not the case, the action aborts, resulting in no output. Atomic actions are the core building blocks (i.e.,

[2]Available at https://swystems.usi.ch/files/BellKAT-PLDI24-long.pdf.

cannot be decomposed) of our language and thus form the basis for algebraic reasoning. The language users, however, will express their protocol using only basic actions and guards, as illustrated in policies P1, P2, and P3. We note that basic actions are broken down into atomic components as follows: $r \triangleright o \triangleq [\mathbb{1}]r \blacktriangleright o + [r]\emptyset \blacktriangleright \emptyset$. (In principle, users could use atomic actions directly to specify quantum protocols, but we advise against it, since it may lead to mistakes like expressing protocols that unintentionally abort or do not correspond to valid quantum operations.) Constant policy 0 acts as abort, and constant policy 1 displays no-op behavior, which is different in different contexts – within a single round it acts as *skip*, whereas in multi-rounds it represents the absence of actions, which we refer to as *no-round*.

4.2 Overview

The diagram in Figure 4 overviews the key components of BellKAT's syntax and semantics and relations among them. $\llparenthesis - \rrparenthesis$ interprets policies in P_s consisting of a single round. Standard interpretation $I(-)$ transforms a policy into a set of strings of atomic actions. The semantics $\llbracket - \rrbracket$ of multi-round policies P is defined through $\llbracket - \rrbracket_I$ that converts each string in Π^* to a sequential composition of atomic actions. Formal definitions are elaborated on below and detailed in Figure 5. If p is a single round policy, $I(p)$ is a finite set of strings of length one.

$$\begin{tikzcd}
 & \mathrm{P} \arrow[rr, shift left, "I(-)"] \arrow[d] \arrow[ddrr, "\llbracket - \rrbracket"] & & \mathcal{P}(\Pi^*) \arrow[ll, dashed, shift left] \arrow[dd, "\llbracket - \rrbracket_I"] \\
\mathrm{P}_s \arrow[ddr] & \mathrm{P}_s \arrow[l, "N(-)"'] \arrow[dd, shift right, "\llparenthesis - \rrparenthesis"'] & & \\
 & & & \mathcal{M}(\mathrm{BP}) \to \mathcal{P}(\mathcal{M}(\mathrm{BP})) \\
 & \mathcal{M}(\mathrm{BP}) \to \mathcal{P}(\mathcal{M}(\mathrm{BP}) \times \mathcal{M}(\mathrm{BP})) \arrow[uu, dashed, shift right] \arrow[urr] & &
\end{tikzcd}$$

Fig. 4. Overview of BellKAT key ingredients: arrows $\longrightarrow$ and $\dashrightarrow$ indicate soundness and completeness, $\longrightarrow$ is restriction from multi-round to single round policies. Single round meaning factors ($\longrightarrow$) thorough $N(-)$, which modifies a single round policy into its normal form. $\longrightarrow$ signifies merging the freshly produced with unused Bell pairs.

4.3 Syntax

The complete BellKAT syntax is given in Figure 5. Atomic actions $[t]r \blacktriangleright o$ together with constants 0 and 1 form the constituents of policies. Users will typically write protocols as (guarded) policies consisting of basic actions $r \triangleright o$. We provide shorthand notations for the most common basic actions:

swap	$\mathrm{sw}\langle A{\sim}B \,@\, C\rangle$	$\triangleq$	$\{\!\{A{\sim}C, B{\sim}C\}\!\} \triangleright \{\!\{A{\sim}B\}\!\}$
transmit	$\mathrm{tr}\langle A \to B{\sim}C\rangle$	$\triangleq$	$\{\!\{A{\sim}A\}\!\} \triangleright \{\!\{B{\sim}C\}\!\}$
create	$\mathrm{cr}\langle A\rangle$	$\triangleq$	$\emptyset \triangleright \{\!\{A{\sim}A\}\!\}$
wait	$\mathrm{wait}\langle r\rangle$	$\triangleq$	$r \triangleright r$
fail	$\mathrm{fail}\langle r\rangle$	$\triangleq$	$r \triangleright \emptyset$

With our syntax, it is possible to express basic actions that may fail to generate new Bell pairs even if there are enough required Bell pairs in the network. We write such policies as $r \triangleright o + r \triangleright \emptyset \triangleq r \triangleright o + \mathrm{fail}\langle r\rangle$ where, if the required Bell pairs r are available, either the multiset of new Bell pairs o is created or the action fails, in both cases consuming r. For example, for distillation, which is inherently probabilistic, we use the following shorthand notation:

$$\text{distill} \quad \mathrm{di}\langle A{\sim}B\rangle \triangleq \{\!\{A{\sim}B, A{\sim}B\}\!\} \triangleright \{\!\{A{\sim}B\}\!\} + \{\!\{A{\sim}B, A{\sim}B\}\!\} \triangleright \emptyset$$

Basic actions enable users to specify many other quantum operations, for instance, create a Bell pair between neighboring nodes directly, or variants of distillation that require more that two Bell pairs.

We follow the conventional precedence of the operations: * > ; > $\cdot$ > $\|$ > +, with * binding the tightest and + the weakest. For example, $p_1 \parallel p_2 ; p_3 + p_4 \cdot p_5^\star$ is parsed as $(p_1 \parallel (p_2 ; p_3)) + (p_4 \cdot (p_5^\star))$.

Syntax

Nodes	$\mathsf{N} ::=$	$A, B, C, \dots$	
Bell pairs	$\mathsf{BP} \ni bp ::=$	$\mathsf{N}{\sim}\mathsf{N}$	
Multisets	$\mathcal{M}(\mathsf{BP}) \ni a, b, r, o ::=$	$\{\!\{ bp_1, \dots, bp_k \}\!\}$	
Tests	$T \ni t, t' ::=$	$\mathbb{1}$	*no test*
	$\mid$	b	*multiset absence*
	$\mid$	$t \wedge t'$	*conjunction*
	$\mid$	$t \vee t'$	*disjunction*
	$\mid$	$t \uplus b$	*multiset union*
Atomic actions	$\Pi \ni \pi, x, y ::=$	$[t]r \blacktriangleright o$	
Policies	$\mathsf{P} \ni p, q ::=$	0	*abort*
	$\mid$	1	*skip or no-round*
	$\mid$	π	*atomic action*
	$\mid$	$r \triangleright o$	*basic action*
	$\mid$	$[t]p$	*guarded policy*
	$\mid$	$p + q$	*nondeterministic choice*
	$\mid$	$p \cdot q$	*ordered composition*
	$\mid$	$p \parallel q$	*parallel composition*
	$\mid$	$p \ ; \ q$	*sequential composition*
	$\mid$	$p^{\star}$	*Kleene star*
Basic actions	$r \triangleright o ::=$	$[\mathbb{1}]r \blacktriangleright o + [r]\emptyset \blacktriangleright \emptyset$	
Guarded policy	$[t]p ::=$	$[t]\emptyset \blacktriangleright \emptyset \cdot p$	

Test semantics

$$\langle\!| t |\!\rangle \in \mathcal{M}(\mathsf{BP}) \longrightarrow \{\top, \bot\}$$

$$\begin{array}{lcl@{\qquad}lcl}
\langle\!| \mathbb{1} |\!\rangle a & \triangleq & \top & \langle\!| t \uplus b |\!\rangle a & \triangleq & (\langle\!| t |\!\rangle a \setminus b \wedge b \subseteq a) \vee \langle\!| b |\!\rangle a \\
\langle\!| b |\!\rangle a & \triangleq & b \not\subseteq a & \langle\!| t \,\Box\, t' |\!\rangle a & \triangleq & \langle\!| t |\!\rangle a \,\Box\, \langle\!| t' |\!\rangle a, \text{ with } \Box \text{ is either } \wedge \text{ or } \vee
\end{array}$$

Single round semantics

$$\begin{array}{rcl}
(\!| p |\!) & \in & \mathcal{M}(\mathsf{BP}) \longrightarrow \mathcal{P}(\mathcal{M}(\mathsf{BP}) \times \mathcal{M}(\mathsf{BP})) \\
(\!| 0 |\!) a & \triangleq & \emptyset \\
(\!| 1 |\!) a & \triangleq & \{ \emptyset \bowtie a \} \\
(\!| [t]r \blacktriangleright o |\!) a & \triangleq & \begin{cases} \{ o \bowtie a \setminus r \} & \text{if } r \subseteq a \text{ and } \langle\!| t |\!\rangle a = \top \\ \emptyset & \text{otherwise} \end{cases} \\
(\!| p + q |\!) a & \triangleq & (\!| p |\!) a \cup (\!| q |\!) a \\
(\!| p \cdot q |\!) a & \triangleq & ((\!| p |\!) \cdot (\!| q |\!)) a \\
(\!| p \parallel q |\!) a & \triangleq & ((\!| p |\!) \parallel (\!| q |\!)) a
\end{array}$$

Multi-round semantics

$$\begin{array}{rcl}
[\![p]\!] & \in & \mathcal{M}(\mathsf{BP}) \longrightarrow \mathcal{P}(\mathcal{M}(\mathsf{BP})) \\
[\![\omega]\!]_I & \in & \mathcal{M}(\mathsf{BP}) \longrightarrow \mathcal{P}(\mathcal{M}(\mathsf{BP})), \text{ where } \omega = \pi_1 \fatsemi \pi_2 \fatsemi \dots \fatsemi \pi_k \\
[\![p]\!] a & \triangleq & \bigcup_{\omega \in I(p)} [\![\omega]\!]_I a \\
[\![\epsilon]\!]_I a & \triangleq & \{ a \} \\
[\![[t]r \blacktriangleright o]\!]_I a & \triangleq & \begin{cases} \{ o \uplus a \setminus r \} & \text{if } r \subseteq a \text{ and } \langle\!| t |\!\rangle a = \top \\ \emptyset & \text{otherwise} \end{cases} \\
[\![\pi_1 \fatsemi \pi_2 \fatsemi \dots \fatsemi \pi_k]\!]_I a & \triangleq & ([\![\pi_1]\!]_I \bullet [\![\pi_2 \fatsemi \dots \fatsemi \pi_k]\!]_I) a
\end{array}$$

Fig. 5. Syntax and semantics. $a \bowtie b$ and $a \uplus b$ are a pair and a multiset union, and $\bullet$ stands for Kleisli composition for $\mathcal{P}(-)$ monad. Semantics of $\cdot$ and $\parallel$ in $(\!| - |\!)$ are detailed in defs. 4.2-4.3, whereas $[\![-]\!]$ (including ;) is defined via standard interpretation I (cf. Sec. 4.6.2), where $I(p)$ is a set of π strings concatenated by $\fatsemi$.

4.4 Axioms

In this section we introduce our algebraic structure BellSKA, which is the foundation of the BellKAT language, and its related equational theory. All the axioms are listed in Figure 6.

KA axioms

$(p+q)+r \equiv p+(q+r)$	KA-Plus-Assoc	$p\,;1 \equiv p$	KA-Seq-One
$p+q \equiv q+p$	KA-Plus-Comm	$1\,;p \equiv p$	KA-One-Seq
$p+0 \equiv p$	KA-Plus-Zero	$0\,;p \equiv 0$	KA-Zero-Seq
$p+p \equiv p$	KA-Plus-Idem	$p\,;0 \equiv 0$	KA-Seq-Zero
$(p\,;q)\,;r \equiv p\,;(q\,;r)$	KA-Seq-Assoc	$1+p\,;p^\star \equiv p^\star$	KA-Unroll-L
$p\,;(q+r) \equiv p\,;q+p\,;r$	KA-Seq-Dist-L	$p\,;r \leq r \Rightarrow p^\star\,;r \leq r$	KA-Lfp-L
$(p+q)\,;r \equiv p\,;r+q\,;r$	KA-Seq-Dist-R	$1+p^\star\,;p \equiv p^\star$	KA-Unroll-R
		$r\,;p \leq r \Rightarrow r\,;p^\star \leq r$	KA-Lfp-R

SKA axioms for $\|$

$(p \parallel q) \parallel r \equiv p \parallel (q \parallel r)$	SKA-Prl-Assoc	$p \parallel q \equiv q \parallel p$	SKA-Prl-Comm
$p \parallel (q+r) \equiv p \parallel q + p \parallel r$	SKA-Prl-Dist	$1 \parallel p \equiv p$	SKA-One-Prl
$(x\,;p) \parallel (y\,;q) \equiv (x \parallel y)\,;(p \parallel q)$	SKA-Prl-Seq	$0 \parallel p \equiv 0$	SKA-Zero-Prl

SKA axioms for $\cdot$

$(p \cdot q) \cdot r \equiv p \cdot (q \cdot r)$	SKA-Ord-Assoc	$1 \cdot p \equiv p$	SKA-One-Ord
$p \cdot (q+r) \equiv p \cdot q + p \cdot r$	SKA-Ord-Dist-L	$p \cdot 1 \equiv p$	SKA-Ord-One
$(p+q) \cdot r \equiv p \cdot r + q \cdot r$	SKA-Ord-Dist-R	$0 \cdot p \equiv 0$	SKA-Zero-Ord
$(x\,;p) \cdot (y\,;q) \equiv (x \cdot y)\,;(p \cdot q)$	SKA-Ord-Seq	$p \cdot 0 \equiv 0$	SKA-Ord-Zero

Boolean axioms (in addition to monotone axioms)

$\mathbb{1} \uplus b \equiv \mathbb{1}$	Bool-One-U	$(t \wedge t') \uplus b \equiv t \uplus b \wedge t' \uplus b$	Bool-Conj-U-Dist
$b \wedge (b \uplus b') \equiv b$	Bool-Conj-Subset	$(t \vee t') \uplus b \equiv t \uplus b \vee t' \uplus b$	Bool-Disj-U-Dist
$b \vee b' \equiv b \cup b'$	Bool-Disj-U		

Network axioms

$[t]r \blacktriangleright o \cdot [t']r' \blacktriangleright o' \equiv [t \wedge (t' \uplus r)]\hat{r} \blacktriangleright \hat{o}$	if $\hat{r} = r \uplus r'$ and $\hat{o} = o \uplus o'$	Net-Ord
$[t]r \blacktriangleright o \parallel [t']r' \blacktriangleright o' \equiv [(t \uplus r') \wedge (t' \uplus r)]\hat{r} \blacktriangleright \hat{o}$	if $\hat{r} = r \uplus r'$ and $\hat{o} = o \uplus o'$	Net-Prl

Single round axioms

$[\mathbb{1}]\emptyset \blacktriangleright \emptyset \equiv 1$	Sr-One	$(p \parallel p') \cdot (q \parallel q') \leq (p \cdot q) \parallel (p' \cdot q')$	Sr-Exc
$[\emptyset]r \blacktriangleright o \equiv 0$	Sr-Zero	$[b \wedge t]r \blacktriangleright o \equiv [(r \cup b) \wedge t]r \blacktriangleright o$	Sr-Can
		$[t]r \blacktriangleright o + [t']r \blacktriangleright o \equiv [t \vee t']r \blacktriangleright o$	Sr-Plus

Fig. 6. BellKAT equational axioms. Set union $\cup$ of multisets by definition takes the maximum cardinality of each element, in contrast to multiset union $\uplus$ which is the sum of cardinalities of each element. Single round BellKAT axioms exclude the axioms containing operators ; or $\star$. Multi-round BellKAT axioms exclude the axioms starting with Sr.

Definition 4.1. BellSKA is an algebraic structure $(\mathsf{P}, +, \cdot, \|, ;, ^\star, 0, 1, \Pi)$ obtained from a Kleene algebra by adding operations $\|$ and $\cdot$ for synchronous composition of actions. Formally, BellSKA satisfies the KA and SKA axioms in Figure 6, where $\Pi \subseteq \mathsf{P}$ is closed under $\cdot$ and $\|$.

BellSKA is a KA designed to tackle multi-round behavior, by modeling sequential progress throughout rounds. BellSKA contains two SKAs, catering to two different synchronous behaviors that arise from sequential and parallel compositions of atomic actions within single rounds.

Formally, $(\mathsf{P}, 0, 1, +, ;, ^\star)$ is a KA, meaning that P is an idempotent semiring under $(+, ;, 0, 1)$ together with Kleene star axioms. (We use the standard abbreviation $p \leq q$ for $p + q = q$.) In addition, there are extra axioms for parallel and ordered compositions, such that $(\mathsf{P}, 0, 1, +, ;, ^\star, \|)$ is a non-idempotent SKA, and $(\mathsf{P}, 0, 1, +, ;, ^\star, \cdot)$ is an SKA that is neither commutative nor idempotent.

BellKAT is an instantiation of the BellSKA algebraic structure for our multi-round quantum network model. A key aspect of BellSKA are the axioms Net-Prl (stating that π and π' can be applied in any order) and Net-Ord (stating that the former action always acts first) combine language symbols in such a manner that $\pi \| \pi' \in \Pi$ and $\pi \cdot \pi' \in \Pi$.

Tests follow the monotone axioms of Boolean algebra, with the additional axioms listed in Figure 6. Tests are predicates over multisets of Bell pairs. In BellKAT, tests are part of atomic actions, unlike in KAT, where Boolean algebra is a subalgebra in KA. A guarded policy $[t]p$ is provably equivalent to the expression that adds an additional test to the first round of p. For example, axioms SKA-Ord-Dist-L and Net-Ord imply:

$$[t]r \triangleright o \triangleq [t]\emptyset \blacktriangleright \emptyset \cdot r \triangleright o \equiv [t]\emptyset \blacktriangleright \emptyset \cdot ([\mathbb{1}]r \blacktriangleright o + [r]\emptyset \blacktriangleright \emptyset) \equiv [t]r \blacktriangleright o + [t \wedge r]\emptyset \blacktriangleright \emptyset$$

Single round policies are the terms constructed from basic actions, with the following grammar:

$$\mathsf{P}_s \ni p ::= 0 \mid 1 \mid [t]r \blacktriangleright o \mid p + p \mid p \cdot p \mid p \| p$$

Single round policies are composed with single round BellKAT axioms in Figure 6. The resulting algebraic structure $(\mathsf{P}_s, +, \cdot, \|, 0, 1)$ is a *trioid*, i.e., $(\mathsf{P}_s, +, \cdot, 0, 1)$ is an idempotent semiring and $(\mathsf{P}_s, +, \|, 0, 1)$ is a commutative idempotent semiring. However, trioid axioms are not complete for the underlying single round BellKAT model. To establish single round completeness, we need to add axioms that relate atomic actions which have equivalent single round behaviors. Axiom Sr-Exc, called the *exchange law*, relates the ordered and parallel structures within single rounds (however, it does not hold for multi-round policies). Concurrency in BellKAT is governed by the synchrony laws SKA-Prl-Seq and SKA-Ord-Seq that relate the sequential and synchronous algebraic structures of multi-round policies. BellKAT's round-by-round architecture provides simple equational reasoning, and at the same time its semantics faithfully expresses the behaviors of quantum networks. We compare BellKAT to other concurrent KAs in the long version of the paper.

The next section deals with the semantics of single round policies, and the following section deals with the semantics of multi-round policies. Importantly, in Section 4.6 we prove that BellKAT's equational theory is sound and complete with respect to the standard interpretation, which permits us to express multi-round policies as sets containing sequential compositions of atomic actions.

4.5 Semantics of Single Round Policies

The denotational semantics $(\!| - |\!) : \mathsf{P} \to \mathcal{M}(\mathsf{BP}) \to \mathcal{P}(\mathcal{M}(\mathsf{BP}) \times \mathcal{M}(\mathsf{BP}))$ of single round policies is defined in Figure 5. Semantically, a policy denotes a function that takes a multiset of Bell pairs as input and returns a set of pairs of multisets as output. Each returned pair of multisets (written $b \bowtie b'$) has the freshly created Bell pairs as its first element and the Bell pairs that were not acted on as its second element. The output set can be empty, which models aborting behavior, or it can contain a number of multiset pairs, each one modeling a possible way in which entangled states are distributed between end nodes. In particular, a basic action (which constitutes the user-facing syntax of single round policies) is interpreted as a function that acts by the rule:

$$(\!| r \triangleright o |\!) : \ a \ \mapsto \begin{cases} \{\, o \bowtie a \backslash r \,\} & \text{if } r \subseteq a \\ \{\, \emptyset \bowtie a \,\} & \text{otherwise} \end{cases}$$

It creates Bell pairs o if the input multiset a contains the required Bell pairs r, otherwise the entire a is passed on. Similarly, an action which may fail is expressed with $r \triangleright o + r \triangleright \emptyset$ and produces:

$$a \ \mapsto \begin{cases} \{\, o \bowtie a \backslash r \,\} \bigcup \{\, \emptyset \bowtie a \backslash r \,\} & \text{if } r \subseteq a \\ \{\, \emptyset \bowtie a \,\} & \text{otherwise} \end{cases}$$

In contrast with basic action, semantics of an atomic policy $[t]r \blacktriangleright o$ produces the empty set if either test $r \subseteq a$ or $\langle\!| t |\!\rangle a$ fails. Semantically, tests act as guards of atomic actions to which they are atomically tied. The intuition behind our definition of $\langle\!| t \uplus b |\!\rangle$ is that $\langle\!| b' \uplus b |\!\rangle a$ checks for multiset absence $b' \uplus b \not\subseteq a$ (one can prove that the definition $(\langle\!| b' |\!\rangle a \backslash b \wedge b \subseteq a) \vee \langle\!| b |\!\rangle a$ is symmetric), and the general case follows by distributivity of $\uplus$ over $\wedge$ and $\vee$. The relation between the test t and the inclusion test for r in $[t]r \blacktriangleright o$ is captured with the SR-CAN axiom. Tests can model filters with the action $(\!| \, [t]\emptyset \blacktriangleright \emptyset \, |\!)$. Constant 0 aborts on every input and behaves as the action $[\emptyset]r \blacktriangleright o$ with arbitrary r and o, since test $\emptyset$ will only succeed for a given a if $\emptyset \not\subseteq a$. Constant 1 acts as skip and has the same behavior as $[\mathbb{1}]\emptyset \blacktriangleright \emptyset$ – it requires no Bell pairs and produces no Bell pairs. Thus we declare these equivalences as single round axioms SR-ZERO and SR-ONE. However, in Section 4.6.2 we elaborate why they cannot hold for multi-round policies.

The union operation + denotes a function that produces the union of the sets generated by the operands. Axiom SR-PLUS relates union with disjunction. Ordered composition $\cdot$ models actions that occur in a fixed sequential order within a single round. Intuitively, the actions occurring later in the policy expression can act only on the Bell pairs that have not been used by the previous actions. Contrawise, parallel composition $\|$ connects the policies which are to be executed in the same round with no specified order.

Definition 4.2. Consider functions $f, g : \mathcal{M}(\mathrm{BP}) \to \mathcal{P}(\mathcal{M}(\mathrm{BP}) \times \mathcal{M}(\mathrm{BP}))$. We define $f \cdot g$ by using the Kleisli composition of functions on the right component. The exact definition is as follows:

$$f \cdot g : \ a \ \longmapsto \bigcup_{b \bowtie a \backslash a_f \in f(a)} \{\, b \uplus b' \bowtie a \backslash (a_f \uplus a_g) \ \mid \ b' \bowtie a \backslash (a_f \uplus a_g) \in g(a \backslash a_f) \,\}$$

Definition 4.3. For functions $f, g : \mathcal{M}(\mathrm{BP}) \to \mathcal{P}(\mathcal{M}(\mathrm{BP}) \times \mathcal{M}(\mathrm{BP}))$, we define $f \,\|\, g$ as:

$$f \,\|\, g : \ a \ \longmapsto \bigcup_{a_f \uplus a_g \subseteq a} \left\{ b \uplus b' \bowtie a \backslash (a_f \uplus a_g) \ \middle| \ \begin{array}{l} b \bowtie a \backslash (a_f \uplus a_g) \in f(a \backslash a_g), \\ b' \bowtie a \backslash (a_f \uplus a_g) \in g(a \backslash a_f) \end{array} \right\}$$

The next example illustrates the difference between $\|$ and $\cdot$ when there is resource contention.

Example 4.1. Consider the third round policy P1 in Figure 2a: $q = \mathrm{sw}\langle A{\sim}E \,@\, D \rangle \,\|\, \mathrm{sw}\langle B{\sim}E \,@\, D \rangle$. Then, $(\!| q |\!) = (\!| \mathrm{sw}\langle A{\sim}E \,@\, D \rangle |\!) \,\|\, (\!| \mathrm{sw}\langle B{\sim}E \,@\, D \rangle |\!)$, and for the input $a = \{\!\{ A{\sim}D, B{\sim}D, E{\sim}D \}\!\}$ it holds:

$$(\!| q |\!) a = \{\, \{\!\{A{\sim}E\}\!\} \bowtie \{\!\{B{\sim}D\}\!\}, \{\!\{B{\sim}E\}\!\} \bowtie \{\!\{A{\sim}D\}\!\} \,\}$$

If we replace parallel composition with ordered composition, then $q' = \mathrm{sw}\langle A{\sim}E \,@\, D \rangle \cdot \mathrm{sw}\langle B{\sim}E \,@\, D \rangle$ always attempts to create $A{\sim}E$ before $B{\sim}E$. For example, from the same multiset a, it produces:

$$(\!| q' |\!) a = \{\, \{\!\{A{\sim}E\}\!\} \bowtie \{\!\{B{\sim}D\}\!\} \,\}$$

On the other hand, q and q' produce the same Bell pairs from the input $b = \{\!\{ A{\sim}D, B{\sim}D, E{\sim}D, E{\sim}D \}\!\}$, since both basic actions find the required Bell pairs in b: $(\!| q |\!) b = (\!| q' |\!) b = \{\, \{\!\{A{\sim}E, B{\sim}E\}\!\} \bowtie \emptyset \,\}$.

4.5.1 Soundness of Single Round. This section proves the soundness of single round BellKAT axioms with respect to the denotational semantics of a single round. More formally, Corollary 4.1 states that every equivalence provable using the BellKAT axioms also holds in the denotational model. That is, $\vdash p \equiv q \Rightarrow (\!| p |\!) = (\!| q |\!)$, where $\vdash$ denotes provability in BellKAT.

Definition 4.4. Set $\mathcal{F} \subseteq \mathcal{M}(\mathrm{BP}) \to \mathcal{P}(\mathcal{M}(\mathrm{BP}) \times \mathcal{M}(\mathrm{BP}))$ is the minimal set generated by $(\!| [t]r \blacktriangleright o |\!)$, $(\!| 1 |\!)$, $(\!| 0 |\!)$ for any r and o, that is closed under the operations $\cdot$ and $\|$ from Definition 4.2 and Definition 4.3, and under $+$, defined as $(f+g)(a) \triangleq f(a) \cup g(a)$.

We warm up with observations that aid the reasoning about semantic functions in $\mathcal{F}$. The following lemmas show that $\mathcal{F}$ satisfies BellKAT's axioms, which implies the soundness of single round policies.

Lemma 4.1. *For any $f \in \mathcal{F}$ the following properties hold:*

(1) $b \bowtie a' \in f(a) \Rightarrow a' \subseteq a$

(2) *For any $r \subseteq a' \subseteq a$ we have $b \bowtie a \setminus r \in f(a) \Rightarrow b \bowtie a' \setminus r \in f(a')$*

Lemma 4.2. *Functions in $\mathcal{F}$ satisfy the trioid axioms in Figure 6 - these are the KA axioms involving the $+$ and the SKA axioms involving $\cdot$ and $\|$, excluding the axioms that contain ; or *.*

Lemma 4.3. *Test semantics is sound. This means, $t \equiv t'$ implies $\langle\!| t |\!\rangle a = \langle\!| t' |\!\rangle a$ for all multisets a.*

Lemma 4.4. *The $(\!| - |\!)$ meaning of network axioms and single round axioms in Figure 6 is sound.*

Lemma 4.5 (Exchange law). *For $f, f', g, g' \in \mathcal{F}$ it holds: $(f \| f') \cdot (g \| g') \leq (f \cdot g) \| (f' \cdot g')$*

Lemma 4.2, Lemma 4.4, and Lemma 4.5 imply the soundness of single round policies.

Corollary 4.1. *BellKAT axioms are sound w.r.t. the denotational semantics of a single round.*

We conclude with an example, showing that parallel composition of actions does not simply reduce to interleaving. Accordingly, $f \cdot (g \| (f' \cdot g')) + f' \cdot ((f \cdot g) \| g') \neq (f \cdot g) \| (f' \cdot g')$.

Example 4.2. Consider the following basic actions and the multiset $a = \{\!\{C{\sim}C, E{\sim}E, C{\sim}E, C{\sim}E\}\!\}$:

$$f = \{\!\{C{\sim}C\}\!\} \triangleright \{\!\{C{\sim}D\}\!\} \quad g = \{\!\{E{\sim}E, C{\sim}E\}\!\} \triangleright \{\!\{C{\sim}E\}\!\} \quad f' = \{\!\{E{\sim}E\}\!\} \triangleright \{\!\{E{\sim}D\}\!\} \quad g' = \{\!\{C{\sim}C, C{\sim}E\}\!\} \triangleright \{\!\{C{\sim}E\}\!\}$$

Notice that $\{\!\{C{\sim}E, C{\sim}E\}\!\} \bowtie \emptyset \in ((f \cdot g) \| (f' \cdot g'))(a)$, since the entire a can be consumed by $f \cdot g$ acting on $\{\!\{E{\sim}E, C{\sim}E\}\!\} = a \setminus \{\!\{C{\sim}C, C{\sim}E\}\!\}$ and by $f' \cdot g'$ acting on $\{\!\{C{\sim}C, C{\sim}E\}\!\} = a \setminus \{\!\{E{\sim}E, C{\sim}E\}\!\}$.

In contrast, in $f \cdot (g \| (f' \cdot g'))$ function f acts first on a consuming $C{\sim}C$, meaning that g' cannot act on $a \setminus \{\!\{C{\sim}C\}\!\}$. Similarly, in $f' \cdot ((f \cdot g) \| g')$, f' acting on a prevents g from acting on $a \setminus \{\!\{E{\sim}E\}\!\}$. This proves that, $\forall\ b' \bowtie a' \in (f \cdot (g \| (f' \cdot g')) + f' \cdot ((f \cdot g) \| g'))(a)$, either $C{\sim}D \in b'$ or $E{\sim}D \in b'$.

4.5.2 Completeness of Single Round. Next we prove the completeness of BellKAT axioms with respect to the denotational semantics of a single round. This means that single round BellKAT expressions which are semantically equal, are provably equivalent by BellKAT axioms. In order to prove completeness, we will find a normal form of policies that captures their semantic meaning.

Definition 4.5 (Normal form of tests). A test is in normal form if it is a finite conjunction of multiset absences (by convention, empty conjunction is test $\mathbb{1}$), $t = \bigwedge b$, where $b \in \mathcal{M}(\mathrm{BP})$, and for no two multisets b and b' in t the inclusion $b \subseteq b'$ holds.

Lemma 4.6. *Every test t is equivalent to a test in normal form $N(t)$. Moreover, if tests in normal form have the same test semantics, they are syntactically identical (up to permutations of conjuncts).*

Definition 4.6 (Canonical form of tests). Let t be a test and $N(t) = \bigwedge b$ its normal form. The normalized test of $\bigwedge(r \cup b)$ is called canonical form of t with respect to r. Canonical form of $\mathbb{1}$ is $\mathbb{1}$.

Lemma 4.7. *Let $\pi = [t]r \blacktriangleright o$ and $\pi' = [t']r \blacktriangleright o$ be atomic actions. Then $(\!|\, \pi \,|\!) = (\!|\, \pi' \,|\!)$ if and only the canonical forms of t and t' with respect to r coincide.*

Definition 4.7 (Normal form of policies). A policy p is in normal form if it is a finite sum, s.t. every summand has a unique (r, o) pair with the corresponding t in canonical form w.r.t. r and $t \neq r$:

$$p = \sum [t]r \blacktriangleright o$$

A corollary of Lemma 4.7 is that an atomic action $[t]r \blacktriangleright o$ aborts if and only if the canonical form of t is r. This ensures that normal form of a policy is unique as stated in the next lemma.

Lemma 4.8. *Every single round policy is normalizable, i.e., it is provably equivalent to a policy in normal form. Furthermore, policies in normal form with the same single round semantics coincide.*

The proofs of the above lemmas (provided in the long version of the paper, where we also include examples of policies in normal form) follow by rigorously applying the definitions.

Proposition 4.1 (Completeness). *Let p, q be single round policies such that $(\!|\, p \,|\!) = (\!|\, q \,|\!)$ Then p and q are provably equivalent by the BellKAT axioms.*

Proof. By the definition of single round policies, $(\!|\, p \,|\!)$ and $(\!|\, q \,|\!)$ are in $\mathcal{F}$. By Lemma 4.8 policies p and q are provably equivalent to their normal form $\vdash p \equiv N(p)$ and $\vdash q \equiv N(q)$. Then soundness in Corollary 4.1 yields $(\!|\, p \,|\!) = (\!|\, N(p) \,|\!)$ and $(\!|\, q \,|\!) = (\!|\, N(q) \,|\!)$. Completeness then follows from the implication $(\!|\, N(p) \,|\!) = (\!|\, N(q) \,|\!) \Rightarrow N(p) = N(q)$ proven in Lemma 4.8. □

4.6 Semantics of Multi-Round Policies

In this section we tackle the standard issue with the algebraic models dealing with concurrency, also encountered by Wagemaker et al. [2020, 2022]: some behaviors of an executed policy can only be observed when executed concurrently with another policy, and not in isolation. This goes against the algebraic approach, which requires capturing policy behavior in all contexts. Hence, in the sequel, we include complete execution traces in the semantics that do not directly correspond to the observable end-to-end behavior. We present the *standard interpretation* of policies by defining a homomorphism that maps a policy (as an expression in BellSKA and obeying axioms in Figure 6) into a synchronous set of strings of atomic actions.

4.6.1 Soundness and Completeness of BellSKA.

Definition 4.8 (Synchronous policy sets). In a quantum network, consider the set of atomic actions (denote them by $x, y \in \Pi$). *String policies* over Π are strings of atomic actions including the empty string ϵ (denote them by $u, v \in \Pi^*$). A *synchronous policy set* is a set of policy strings in $\mathcal{P}(\Pi^*)$ (denoted by U, V). Consider the following definitions and operations on synchronous policy sets,

$$0 \triangleq \emptyset \qquad U + V \triangleq U \cup V \qquad U \cdot V \triangleq \{\, u \circ v \mid u \in U, v \in V \,\}$$
$$1 \triangleq \{\epsilon\} \qquad U \,;\, V \triangleq \{u \mathbin{⨾} v \mid u \in U, v \in V\} \qquad U \parallel V \triangleq \{\, u \mathbin{|||} v \mid u \in U, v \in V \,\}$$
$$U^\star \triangleq \cup_{n \geq 0} U^n$$

where $u ⨾ v$ denotes the concatenation of strings u and v in Π^*, with *layer-by-layer ordered composition* $u \circ v \in \Pi^*$ and *layer-by-layer parallel composition* $u \mathbin{|||} v \in \Pi^*$, defined respectively by the rules,

$$u \circ \epsilon \triangleq u \triangleq \epsilon \circ u \qquad \text{and} \qquad u \mathbin{|||} \epsilon \triangleq u \triangleq \epsilon \mathbin{|||} u$$
$$(x ⨾ u) \circ (y ⨾ v) \triangleq (x \cdot y) ⨾ (u \circ v) \qquad (x ⨾ u) \mathbin{|||} (y ⨾ v) \triangleq (x \parallel y) ⨾ (u \mathbin{|||} v)$$

where $x \cdot y$ and $x \parallel y$ are the atomic action obtained by the axioms Net-Ord and Net-Prl in Figure 6.

The powers of U are defined recursively as $U^0 \triangleq \{\epsilon\}$ and $U^n \triangleq U \,;\, U^{n-1}$. By convention $U^\star$ always contains the empty string, thus when $U = \emptyset$ we set $U^\star = \{\epsilon\}$.

Theorem 4.1. *Any set of synchronous policy sets that contains* 0 *and* 1 *and is closed under the operations of Definition 4.8 is a BellSKA.*

Theorem 4.1 shows that any subalgebra of $\mathcal{P}(\Pi^*)$ is also a BellSKA (see Definition 4.1). Let M_{BellSKA} be the smallest algebra that contains $0, 1$ and all $\pi \in \Pi$ in a given quantum network.

Definition 4.9 (Standard interpretation). Consider the set of policies P as a BellSKA term algebra. Standard interpretation $I\colon \mathsf{P} \longrightarrow M_{\text{BellSKA}}$ maps the generators of P by the rule $I(\pi) = \{\, \pi \,\}$ and $I(1) = \{\epsilon\}$, $I(0) = \emptyset$, and is then homomorphically extended as:

$$I(p{+}q) = I(p){+}I(q),\ I(p \parallel q) = I(p) \parallel I(q),\ I(p{\cdot}q) = I(p){\cdot}I(q),\ I(p;q) = I(p);I(q),\ I(p^\star) = I(p)^\star$$

Standard interpretation provides a deterministic algorithm for obtaining a model for BellSKA policies. Indeed, for a given policy we recursively apply homomorphism I to obtain a set of synchronous strings, as illustrated on the next example.

Example 4.3. Consider policies p and q that are sequential compositions of atomic actions,

$$p = ([\mathbb{1}]\emptyset \blacktriangleright \{\!\{C{\sim}C\}\!\})\,;([\{\!\{C{\sim}C\}\!\}]\emptyset \blacktriangleright \{\!\{C{\sim}C\}\!\}) \quad q = ([\mathbb{1}]\emptyset \blacktriangleright \{\!\{E{\sim}E\}\!\})\,;([\mathbb{1}]\{\!\{C{\sim}C\}\!\} \blacktriangleright \{\!\{C{\sim}D\}\!\})$$

therefore they are interpreted as singletons. Then, $I(p \parallel q)$ is given by the Net-Prl axiom:

$$I(p \parallel q) = I(p) \parallel I(q) = \{\, ([\mathbb{1}]\emptyset \blacktriangleright \{\!\{C{\sim}C, E{\sim}E\}\!\}) \fatsemi ([\{\!\{C{\sim}C, C{\sim}C\}\!\}]\{\!\{C{\sim}C\}\!\} \blacktriangleright \{\!\{C{\sim}C, C{\sim}D\}\!\}) \,\}$$

The theorem below shows that $I(p)$ is regular for any policy p.

Theorem 4.2 (completeness w.r.t. standard interpretation). *Policies $p, q \in \mathsf{P}$ are equal under the standard interpretation if and only if they are provably equivalent using BellKAT's axioms. That is, $I(p) = I(q)$ if and only if $\vdash p \equiv q$.*

The automata constructed in the proof of completeness can be also used to decide if $I(p) = I(q)$.

4.6.2 Multi-Round Policies as Functions. The denotational semantics of multi-round BellKAT is defined in Figure 5. In summary, $[\![-]\!]\colon \mathsf{P} \to \mathcal{M}(\mathsf{BP}) \to \mathcal{P}(\mathcal{M}(\mathsf{BP}))$ is defined through the standard interpretation as $[\![p]\!]a \triangleq \bigcup_{\omega \in I(p)} [\![\omega]\!]_I a$, where $[\![-]\!]_I\colon \Pi^* \to \mathcal{M}(\mathsf{BP}) \to \mathcal{P}(\mathcal{M}(\mathsf{BP}))$ is recursively defined as $[\![\omega]\!]_I \triangleq [\![\pi_1]\!]_I \bullet [\![\pi_2 \fatsemi \ldots \fatsemi \pi_k]\!]_I$, with $(\bullet)$ denoting the Kleisli composition:

$$\begin{aligned} f \bullet g :\ \mathcal{M}(\mathsf{BP}) &\longrightarrow \mathcal{P}(\mathcal{M}(\mathsf{BP})) \\ a &\longmapsto \big\{c \mid \text{there exists } b \in f(a) \text{ s.t. } c \in g(b)\big\} = \bigcup_{b \in f(a)} g(b) \end{aligned}$$

Next we show that executions of provably equivalent policies produce the same Bell pairs.

Theorem 4.3 (Soundness of multi-round policies). *If policies $p, q \in \mathsf{P}$ are equivalent under BellKAT's axioms, then their denotational semantics coincide. That is, $\vdash p \equiv q \implies [\![p]\!] = [\![q]\!]$.*

Proof. The soundness of multi-round policies follows from the soundness of both the standard interpretation (Theorem 4.1) and single round policies (Corollary 4.1). A key aspect is that all $\pi \in \Pi$ are considered as actions that require time, whereas 1 is considered as the absence of actions (hence the name no-round). This is why we exclude the Sr-One axiom from multi-round policies. □

Remark 4.1 (Atomic actions $\pi \in \mathsf{P}$ vs. $\pi \in \mathsf{P}_s$). For an atomic action π, its multi-round and single round semantics are closely related. Syntactically, we replace $\uplus$ with $\bowtie$ in the definitions of $[\![\pi]\!]$ and $(\!|\, \pi \,|\!)$. This means that $\bowtie$ separates the freshly created Bell pairs from the unused Bell pairs within a single round, as opposed to $\uplus$ that combines both multisets to be offered to the next round.

The following example illustrates that the nature of quantum networks does not permit compositional observational reasoning at the level of end-to-end behavior when actually executed. The end-to-end behavior is captured by our multi-round denotational semantics, which assumes isolated execution. This is consistent with our quantum network architecture, which, before execution, installs the policy into the network as generating rules. It is precisely the behavior of these rules that we want to analyze – any future policy modifications, including composition with other policies, would require a separate analysis. We construct a policy whose semantics is abort, however when run in parallel with another policy, their combined meaning produces Bell pairs.

Example 4.4. Recall the policies p and q in Example 4.3. From $I(p)$ we read that the first round always creates $C{\sim}C$, and the test in the second round only passes the multisets which do not contain $C{\sim}C$, thus $[\![p]\!] = [\![0]\!] = \emptyset$ is abort. On the other hand, we showed that $I(p \parallel q)$ is a singleton containing $\omega = \pi_1 \fatsemi \pi_2 = ([\mathbb{1}]\emptyset \blacktriangleright \{\!\{C{\sim}C, E{\sim}E\}\!\}) \fatsemi ([\{\!\{C{\sim}C, C{\sim}C\}\!\}]\{\!\{C{\sim}C\}\!\} \blacktriangleright \{\!\{C{\sim}C, C{\sim}D\}\!\})$. This leads to $[\![p \parallel q]\!] = [\![\pi_1]\!]_I \bullet [\![\pi_2]\!]_I$, which on the empty input produces:

$$[\![p \parallel q]\!]\emptyset = [\![\pi_2]\!]_I\{\!\{C{\sim}C, E{\sim}E\}\!\} = \{\ \{\!\{C{\sim}C, C{\sim}D, E{\sim}E\}\!\}\ \}$$

This illustrates that no set of axioms is sound and complete w.r.t. $[\![-]\!]$, else the application of Leibniz rule of inference on $p \equiv 0$ would lead to $p \parallel q \equiv 0 \parallel q \equiv 0$, which contradics $p \parallel q \not\equiv 0$.

The next example shows that BellKAT axioms are not complete w.r.t. $[\![-]\!]$, even for the fragment of policies generated with basic actions $r \triangleright o$. The following policies have clearly distinct second rounds, however semantically they behave the same on all the inputs provided by the first round.

Example 4.5 (Completeness w.r.t. isolated execution). Policies $p = (\mathrm{cr}\langle C\rangle \parallel \mathrm{cr}\langle C\rangle)\ ;\ (\mathrm{tr}\langle C \to A{\sim}D\rangle \cdot \mathrm{tr}\langle C \to B{\sim}D\rangle)$ and $q = (\mathrm{cr}\langle C\rangle \parallel \mathrm{cr}\langle C\rangle)\ ;\ (\mathrm{tr}\langle C \to A{\sim}D\rangle \parallel \mathrm{tr}\langle C \to B{\sim}D\rangle)$ have the same meaning. Indeed, on any input both first rounds generate two copies of $C{\sim}C$, and then both second rounds transmit to $A{\sim}D$ and $B{\sim}D$. This means, $[\![p]\!]a = [\![q]\!]a = \{\!\{A{\sim}D, B{\sim}D\}\!\} \uplus a$. However, in the equational theory of BellKAT, $p \not\equiv q$ since $\mathrm{tr}\langle C \to A{\sim}D\rangle \cdot \mathrm{tr}\langle C \to B{\sim}D\rangle \not\equiv \mathrm{tr}\langle C \to A{\sim}D\rangle \parallel \mathrm{tr}\langle C \to B{\sim}D\rangle$.

4.6.3 Equivalence Checking for Isolated Policy Behaviors. This section presents the tools to reason about actual end-to-end execution of policies, at which point it is appropriate to factor in the system constraints, in particular, finite qubit memory at the end nodes. We explicitly deal with atomic actions whose execution leads to constraint violation. Specifically, we introduce an *invalid* network state $\perp$ and a notion of *valid policies* (policies that do not reach invalid states).

Example 4.5 with $[\![p]\!] = [\![q]\!]$ and $p \not\equiv q$, shows that checking whether $[\![p]\!] = [\![q]\!]$ requires techniques beyond the equational reasoning we have been focused on so far. On a high level, $[\![-]\!]$ is designed to faithfully model the network's behavior, i.e., how protocols represented by policies are executed; thus, some information about how a policy would behave when composed with other policies gets lost, as illustrated in Example 4.4. Such information should be present for soundness and completeness to hold as the congruence rule of inference implies that equivalent functions must behave equivalently in all contexts. Concretely, Theorem 4.2 guarantees that equational theory is complete w.r.t. the standard interpretation $I(-)$, which is thus not a proper reflection of $[\![-]\!]$.

The meanings $I(-)$ and $[\![-]\!]$ cater for different applications, namely, standard interpretation is useful for policy optimization (capturing the behavior in all contexts) and end-to-end meaning facilitates policy verification (whether the policies in isolation do what they were meant to). The difference between the meanings stems from the shared nature of resources (Bell pairs) that policies use and produce; through these resources the policies affect each other's behavior when composed.

To reason about $[\![-]\!]$, we keep track of the current network state (multiset of Bell pairs). There is only a finite number of valid network states, denoted by $\mathcal{N} \subseteq \mathcal{M}(\mathrm{BP})$, as both the number of nodes in the network and the number of Bell pairs between any two nodes is bounded. Since $\mathcal{N}$ captures the network hardware constraints (e.g., number of available memory qubits), we fix it globally to simplify the definitions. Moreover, we allow further restriction on the set of *initial* network states $\mathcal{N}_0 \subseteq \mathcal{N}$ in which policies start execution. $\mathcal{N}_0$ specifies properties of the initial network state, e.g., $\mathcal{N}_0 = \emptyset$ signifies that there are no Bell pairs present yet. Hence, the goal is to check whether for all $a \in \mathcal{N}_0$ we have $[\![p]\!]a = [\![q]\!]a$, denoted as $[\![p]\!] =_{\mathcal{N}_0} [\![q]\!]$.

Definition 4.10. A BellKAT policy p is *valid* with respect to $\mathcal{N}_0 \subseteq \mathcal{N}$ if and only if any network state, encountered during any execution of p on an input from $\mathcal{N}_0$, is also in $\mathcal{N}$.

If we consider a finite $\Pi' \subseteq \Pi$, we can build a transition system $\mathcal{G}(\Pi')$ with $\mathcal{N}_\perp = \mathcal{N} \cup \{\perp\}$ as states and Π' as actions; states in $\mathcal{N}_0$ are initial, states in $\mathcal{N}$ are terminal, and $\perp$ is an invalid network state, for $a, a' \in \mathcal{N}$ there is a transition from a to a' labelled with π iff $a' \in [\![\pi]\!]a$, and there is a transition from a to $\perp$ labelled with π iff $[\![\pi]\!]a \setminus \mathcal{N} \neq \emptyset$. At the same time, for a policy p we can build an automaton $\mathcal{A}(p)$ as in the proof of Theorem 4.2, which can be seen as another transition system with the set of actions $\Pi_p = \{\pi \in \omega \mid \omega \in I(p)\}$. Finally, we can build a transition system $\mathcal{G}(\Pi_p) \parallel_{\Pi_p} \mathcal{A}(p)$ that is a parallel composition of $\mathcal{G}$ and $\mathcal{A}(p)$ with handshaking on the set of actions Π_p (see [Milner 1989]), i.e., the set of states is $\mathcal{N} \times \mathrm{states}(\mathcal{A}(p))$ and there is a transition from (a, s) to (a', s') labelled with π iff there are π-labelled transitions from a to a' in $\mathcal{G}$ and from s to s' in $\mathcal{A}(p)$. Such a transition system captures isolated behavior of p.

These definitions of transitions in $\mathcal{G}(\Pi_p)$ and $\mathcal{A}(p)$ yield the next lemma. As a consequence, we obtain a tool to reason about the equality of policies w.r.t. $[\![-]\!]$ in Theorem 4.4. Furthermore, in Theorem 4.5 we also show that the validity of a policy is decidable. The actual implementation of the decision procedure, does not explicitly build $\mathcal{G}(\Pi_p)$ due to its large size, but constructs $\mathcal{G}(\Pi_p) \parallel_{\Pi_p} \mathcal{A}(p)$ directly, adding states from $\mathcal{N}$ in a lazy manner.

Lemma 4.9. *Let $p \in P$, $a \in \mathcal{N}_0$, and $a' \in \mathcal{N}$. Then, $a' \in [\![p]\!]a$ if and only if there is an execution of $\mathcal{G}(\Pi_p) \parallel_{\Pi_p} \mathcal{A}(p)$ starting in state (a, s) and ending in (a', s') for some s and s' in* $\mathrm{states}(\mathcal{A}(p))$.

Theorem 4.4. *If p and q are valid policies with respect to $\mathcal{N}_0 \subseteq \mathcal{N}$, then $[\![p]\!] =_{\mathcal{N}_0} [\![q]\!]$ is decidable.*

Proof. The question of $[\![p]\!] =_{\mathcal{N}_0} [\![q]\!]$ can be reduced to checking whether for any $a \in \mathcal{N}_0$, $a' \in \mathcal{N}$, $a' \in [\![p]\!]a \Leftrightarrow a' \in [\![q]\!]a$. Since $\mathcal{N}$ is finite, the latter can be answered with Lemma 4.9. □

Theorem 4.5. *Policy p is valid if an only if there is* no *execution in $\mathcal{G}(\Pi_p) \parallel_{\Pi_p} \mathcal{A}(p)$ ending in the state $(\perp, s')$ for some s' in $\mathcal{A}(p)$.*

5 QUANTUM NETWORK VERIFICATION

The limitations of hardware, such as low rates of Bell pair generation, short memory lifetimes, and limited numbers of communication qubits, make competition for resources unavoidable. This competition is the main motivation for formal reasoning about quantum network properties.

BellKAT's equational theory and the decidability result, following from Theorem 4.2, can be used to verify that policies are equivalent in all contexts, facilitating modular policy optimization, while the decidability result in Theorem 4.4 can be used for network verification by translating certain network properties to checking equivalences between end-to-end policy behaviors.

BellKAT serves as an intermediate layer when verifying distributed quantum applications, separating them from low level implementations of quantum actions. Users specify protocols as BellKAT policies, typically starting with a round of `create` actions. Before deploying the policies on a quantum network, users will verify that resource sharing, arising either from concurrency within the policy or due to composition with other policies, will not impede the desired Bell pair generation.

Verification tasks. The following properties translate naturally from classical networks to the quantum setting of entanglement distribution.

- *Reachability.* The most basic property of interest is whether the execution of a policy is able to generate the requested entanglement between end nodes.
- *Waypoint correctness.* We may wish to guarantee that an entanglement generating protocol always performs the swapping operation through certain nodes.
- *Traffic (protocol) isolation.* Composition of policies may lead to undesired behaviors, such as race conditions. In light of this, it is desirable to prove non-interference properties that ensure isolation between executions of the composed policies.
- *Compilation.* Establishing the correctness of the compilation process is a necessary final step for ensuring correct deployment.

The following properties, which do not have a clear counterpart in classical networks, are posed as resource constraint problems.

- *Resource utilization.* What is the number of required memory locations and communication qubits? For how many rounds must Bell pairs be kept in the memory?
- *Quality of service.* Does the network have the required capacity (i.e., number of created end-to-end Bell pairs per second), and what is the confidence in their quality?
- *Network state access.* Can we minimize the number of costly accesses to the network global state in a policy? Such optimization can significantly reduce the coordination effort.

From histories it is possible to read whether an underlying protocol obeys the hardware constraints (e.g., the number of communication and memory qubits, as illustrated in Figure 3), and also suggest how to optimize resource allocation over rounds. It is worth noting that Bell pairs between the same two nodes are indistinguishable for most applications, which can lead to more efficient provisioning of resources. In addition, the information recorded in histories could shed some light on the order among communication channels, as investigated by Chandra et al. [2022].

Decidability and Verification. Some of the tasks above can be solved by a directed application of the decidability results of Section 4.6. An example of verifying the reachability property would be to check whether policy p always or never generates an entangled pair $A{\sim}B$; concretely, we can check if $(p \,;\, [\mathbb{1}]\{\!\{A{\sim}B\}\!\} \blacktriangleright \{\!\{A{\sim}B\}\!\}) \equiv_{\mathcal{N}_0} p$ or $(p \,;\, [\{\!\{A{\sim}B\}\!\}]\emptyset \blacktriangleright \emptyset) \equiv_{\mathcal{N}_0} p$ using Theorem 4.4. We can also verify resource utilization, with an important practical task being to analyze memory requirements of a policy p, achieved by trying different sets of valid states $\mathcal{N}$ while keeping track of whether p remains valid (see Theorem 4.5). In addition, to verify correctness of the compilation procedure $\texttt{compile} : \mathsf{P} \to \mathsf{P}$ on a policy p, we can again use Theorem 4.4 to ensure $p \equiv_{\mathcal{N}_0} \texttt{compile}(p)$. A single optimization step $\texttt{opt} : \mathsf{P} \to \mathsf{P}$ can be checked with $\vdash p \equiv \texttt{opt}(p)$ using Theorem 4.2.

Prototype implementation. For a given policy, BellKAT histories are records of the successful basic actions $r \triangleright o$ and the order in which they occur. We implemented a prototype in Haskell which produces a set of histories from a given policy. For improved visualization, the prototype can also illustrate histories as shown in Figure 3. Furthermore, it can check the validity and equality of policies by implementing the decision procedure described in Section 4.6.3. This procedure allows us to verify concrete properties, such as those discussed in the previous paragraph. A last feature of note is our prototype's ability to perform *network slicing*, which facilitates modular construction of policies by tagging them with unique identifiers in order to keep them differentiated, similar to the concepts of NetKAT slices and boxes by Brunet and Pym [2020]. By appending such (classical) metadata to Bell pairs it is possible to model basic interactions between control plane and classical and quantum data planes. Our prototype is open source and freely available as an online artifact.[3]

[3]Artifact available at Zenodo [Chuprikov 2024].

6 CONCLUSION

Successful integration of classical and quantum networks will provide novel solutions for secure communication tasks, pave the way to distributed quantum computing, and enable other large scale applications of quantum communication technologies. Significant research and engineering efforts are still required until quantum networks reach full functionality. Our work focuses on the specification of entanglement generating protocols, taking into account the distinctive features of entanglement as the main communication resource. With BellKAT, we provide a foundational model for quantum network programming languages in a threefold manner. (1) We present a solid algebraic foundation, called BellSKA, on which the BellKAT language and logic is based. BellKAT's axioms faithfully encode the network behavior and allow for equational reasoning. (2) We showcase the expressiveness of BellKAT by specifying a number of entanglement generating protocols, including the only long distance repeater protocol currently realized in practice [Pompili et al. 2021]. (3) We implemented a prototype to support the practical specification of protocols and verification of relevant properties. The capabilities of our prototype are complementary to those of existing simulators like NetSquid [Coopmans et al. 2021].

The BellKAT formalism and its underlying BellSKA structure open exciting new research avenues, including (i) the formalization of probabilistic phenomena of quantum networks, (ii) the extension of BellKAT to handle quantum states other than Bell pairs, (iii) the investigation of additional BellKAT semantic models to cater to more verification tasks, and (iv) the exploration of possible uses of BellSKA. For (i), we envision extending BellKAT with probabilistic semantics by adding a random choice operation ($+_p$) for specifying probabilities, similar to the work of Foster et al. [2016] and Smolka et al. [2019b]. For (ii), actions could, for instance, be generalized to handle the transmission of single qubits $\{\!\{A\}\!\} \triangleright \{\!\{B\}\!\}$, or the creation of EPR pairs from tripartite states $\{\!\{A{\sim}B{\sim}C\}\!\} \triangleright \{\, \{\!\{A{\sim}B\}\!\}, \{\!\{A{\sim}C\}\!\}, \{\!\{B{\sim}C\}\!\} \,\}$ with the distillation process of Dür et al. [2000]. For (iii), a potential semantic extension could record the state of the network at the end of each round, allowing for the capture of intermediate results. For (iv) we could consider possible instantiations of BellSKA to handle other systems exhibiting 2-dimensional behavior (e.g., bulk-synchronous parallel model [Valiant 1990] or hardware design [Halbwachs et al. 1991]) and, furthermore, identify a guarded fragment of BellKAT that is regular, similar to the work of Smolka et al. [2019a].

ACKNOWLEDGMENTS

This material is based upon work supported by Hasler Foundation grant number 23086, Swiss National Science Foundation (SNF) grant number 197353, US Air Force Office of Scientific Research (AFOSR) award number FA95502110051, and US National Science Foundation (NSF) Expedition in Computing (EPiQC) grant number CCF-1730449.

REFERENCES

Carolyn Jane Anderson, Nate Foster, Arjun Guha, Jean-Baptiste Jeannin, Dexter Kozen, Cole Schlesinger, and David Walker. 2014. NetKAT: Semantic Foundations for Networks. *SIGPLAN Notices* 49, 1 (2014), 113–126. https://doi.org/10.1145/2578855.2535862

John Stewart Bell. 1964. On the Einstein Podolsky Rosen Paradox. *Physics Physique Fizika* 1, 3 (1964), 195–200. https://doi.org/10.1103/PhysicsPhysiqueFizika.1.195

Charles H. Bennett and Gilles Brassard. 2014. Quantum Cryptography: Public Key Distribution and Coin Tossing. *Theoretical Computer Science* 560 (2014), 7–11. https://doi.org/10.1016/j.tcs.2014.05.025

Pat Bosshart, Dan Daly, Glen Gibb, Martin Izzard, Nick McKeown, Jennifer Rexford, Cole Schlesinger, Dan Talayco, Amin Vahdat, George Varghese, and David Walker. 2014. P4: Programming Protocol-Independent Packet Processors. *ACM SIGCOMM Computer Communication Review* 44, 3 (2014), 87–95. https://doi.org/10.1145/2656877.2656890

Hans Jürgen Briegel, Wolfgang Dür, Juan Ignacio Cirac, and Peter Zoller. 1998. Quantum Repeaters: The Role of Imperfect Local Operations in Quantum Communication. *Physical Review Letters* 81, 26 (1998), 5932–5935. https://doi.org/10.1103/

PhysRevLett.81.5932

Paul Brunet and David Pym. 2020. Pomsets with Boxes: Protection, Separation, and Locality in Concurrent Kleene Algebra. In *5th International Conference on Formal Structures for Computation and Deduction*. 1–16. https://doi.org/10.4230/LIPIcs.FSCD.2020.8

Anita Buckley, Pavel Chuprikov, Rodrigo Otoni, Robert Rand, Robert Soulé, and Patrick Eugster. 2023. Towards an Algebraic Specification of Quantum Networks. In *1st Workshop on Quantum Networks and Distributed Quantum Computing*. 7–12. https://doi.org/10.1145/3610251.3610557

Daryus Chandra, Marcello Caleffi, and Angela Sara Cacciapuoti. 2022. The Entanglement-Assisted Communication Capacity Over Quantum Trajectories. *IEEE Transactions on Wireless Communications* 21, 6 (2022), 3632–3647. https://doi.org/10.1109/TWC.2021.3122962

Pavel Chuprikov. 2024. Artifact for the article An Algebraic Language for Specifying Quantum Networks. https://doi.org/10.5281/zenodo.10909730

Tim Coopmans, Robert Knegjens, Axel Dahlberg, et al. 2021. NetSquid, a NETwork Simulator for QUantum Information using Discrete events. *Communications Physics* 4, 164 (2021), 1–15. https://doi.org/10.1038/s42005-021-00647-8

Wolfgang Dür, Guifre Vidal, and Juan I. Cirac. 2000. Three Qubits can be Entangled in Two Inequivalent Ways. *Physical Review A* 62, 6 (2000), 1–12. https://doi.org/10.1103/PhysRevA.62.062314

Albert Einstein, Boris Podolsky, and Nathan Rosen. 1935. Can Quantum-Mechanical Description of Physical Reality Be Considered Complete? *Physical Review Online Archive* 47, 10 (1935), 777–780. https://doi.org/10.1103/PhysRev.47.777

Artur K. Ekert. 1991. Quantum Cryptography based on Bell's Theorem. *Physical Review Letters* 67, 6 (1991), 661–663. https://doi.org/10.1103/PhysRevLett.67.661

Nate Foster, Dexter Kozen, Konstantinos Mamouras, Mark Reitblatt, and Alexandra Silva. 2016. Probabilistic NetKAT. In *25th European Symposium on Programming Languages and Systems*. 282–309. https://doi.org/10.1007/978-3-662-49498-1_12

Laszlo Gyongyosi and Sandor Imre. 2022. Advances in the Quantum Internet. *Commun. ACM* 65, 8 (2022), 52–63. https://doi.org/10.1145/3524455

Nicolas Halbwachs, Paul Caspi, Pascal Raymond, and Daniel Pilaud. 1991. The Synchronous Data Flow Programming Language LUSTRE. *Proc. IEEE* 79, 9 (1991), 1305–1320. https://doi.org/10.1109/5.97300

Jessica Illiano, Marcello Caleffi, Antonio Manzalini, and Angela Sara Cacciapuoti. 2022. Quantum Internet Protocol Stack: A Comprehensive Survey. *Computer Networks* 213, 109092 (2022), 1–26. https://doi.org/10.1016/j.comnet.2022.109092

Tobias Kappé, Paul Brunet, Alexandra Silva, Jana Wagemaker, and Fabio Zanasi. 2020. Concurrent Kleene Algebra with Observations: From Hypotheses to Completeness. In *23rd International Conference on the Foundations of Software Science and Computation Structures*. 381–400. https://doi.org/10.1007/978-3-030-45231-5_20

Dexter Kozen. 1994. A Completeness Theorem for Kleene Algebras and the Algebra of Regular Events. *Information and Computation* 110, 2 (1994), 366–390. https://doi.org/10.1006/inco.1994.1037

Dexter Kozen. 1997. Kleene Algebra with Tests. *ACM Transactions on Programming Languages and Systems* 19, 3 (1997), 427–443. https://doi.org/10.1145/256167.256195

Dexter Kozen and Frederick Smith. 1997. Kleene Algebra with Tests: Completeness and Decidability. In *10th International Workshop on Computer Science Logic*. 244–259. https://doi.org/10.1007/3-540-63172-0_43

Wojciech Kozlowski and Stephanie Wehner. 2019. Towards Large-Scale Quantum Networks. In *6th Annual ACM International Conference on Nanoscale Computing and Communication*. 1–7. https://doi.org/10.1145/3345312.3345497

Wojciech Kozlowski, Stephanie Wehner, Rodney Van Meter, Bruno Rijsman, Angela Sara Cacciapuoti, Marcello Caleffi, and Shota Nagayama. 2023. Architectural Principles for a Quantum Internet. RFC 9340. https://doi.org/10.17487/RFC9340

Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Computer Communication Review* 38, 2 (2008), 69–74. https://doi.org/10.1145/1355734.1355746

Robin Milner. 1989. *Communication and Concurrency*. Prentice-Hall, Inc.

Michael A. Nielsen and Isaac L. Chuang. 2011. *Quantum Computation and Quantum Information*. Cambridge University Press.

P4 API Working Group. 2021. P4 Runtime Specification. https://p4.org/p4-spec/p4runtime/main/P4Runtime-Spec.html

Yuxiang Peng, Mingsheng Ying, and Xiaodi Wu. 2022. Algebraic Reasoning of Quantum Programs via Non-Idempotent Kleene Algebra. In *43rd ACM SIGPLAN International Conference on Programming Language Design and Implementation*. 657–670. https://doi.org/10.1145/3519939.3523713

Stefano Pirandola, Ulrik Lund Andersen, Leonardo Banchi, et al. 2020. Advances in Quantum Cryptography. *Advances in Optics and Photonics* 12, 4 (2020), 1012–1236. https://doi.org/10.1364/AOP.361502

Matteo Pompili, Sophie L. N. Hermans, Simon Baier, et al. 2021. Realization of a Multinode Quantum Network of Remote Solid-State Qubits. *Science* 372, 6539 (2021), 259–264. https://doi.org/10.1126/science.abg1919

Cristian Prisacariu. 2010. Synchronous Kleene Algebra. *The Journal of Logic and Algebraic Programming* 79, 7 (2010), 608–635. https://doi.org/10.1016/j.jlap.2010.07.009

Julian Rabbie, Kaushik Chakraborty, Guus Avis, and Stephanie Wehner. 2022. Designing Quantum Networks Using Preexisting Infrastructure. *npj Quantum Information* 8, 5 (2022), 1–12. https://doi.org/10.1038/s41534-021-00501-3

Steffen Smolka, Nate Foster, Justin Hsu, Tobias Kappé, Dexter Kozen, and Alexandra Silva. 2019a. Guarded Kleene Algebra with Tests: Verification of Uninterpreted Programs in Nearly Linear Time. *Proceedings of the ACM on Programming Languages* 4, POPL (2019), 1–28. https://doi.org/10.1145/3371129

Steffen Smolka, Praveen Kumar, David M. Kahn, Nate Foster, Justin Hsu, Dexter Kozen, and Alexandra Silva. 2019b. Scalable Verification of Probabilistic Networks. In *40th ACM SIGPLAN Conference on Programming Language Design and Implementation*. 190–203. https://doi.org/10.1145/3314221.3314639

Don Towsley. 2021. The Quantum Internet: Recent Advances and Challenges. Keynote at the 29th IEEE International Conference on Network Protocols. https://icnp21.cs.ucr.edu

Leslie G. Valiant. 1990. A Bridging Model for Parallel Computation. *Commun. ACM* 33, 8 (1990), 103–111. https://doi.org/10.1145/79173.79181

Rodney Van Meter and Joe Touch. 2013. Designing Quantum Repeater Networks. *IEEE Communications Magazine* 51, 8 (2013), 64–71. https://doi.org/10.1109/MCOM.2013.6576340

Rodney Van Meter, Joe Touch, and Clare Horsman. 2011. Recursive Quantum Repeater Networks. *Progress in Informatics* 8 (2011), 65–79. https://doi.org/10.2201/NiiPi.2011.8.8

Jana Wagemaker, Paul Brunet, Simon Docherty, Tobias Kappé, Jurriaan Rot, and Alexandra Silva. 2020. Partially Observable Concurrent Kleene Algebra. In *31st International Conference on Concurrency Theory*. 1–22. https://doi.org/10.4230/LIPIcs.CONCUR.2020.20

Jana Wagemaker, Nate Foster, Tobias Kappé, Dexter Kozen, Jurriaan Rot, and Alexandra Silva. 2022. Concurrent NetKAT. In *31st European Symposium on Programming*. 575–602. https://doi.org/10.1007/978-3-030-99336-8_21

Chonggang Wang, Akbar Rahman, Ruidong Li, Melchior Aelmans, and Kaushik Chakraborty. 2023. *Application Scenarios for the Quantum Internet*. Technical Report. Internet Engineering Task Force. https://datatracker.ietf.org/doc/draft-irtf-qirg-quantum-internet-use-cases/16

Stephanie Wehner, David Elkouss, and Ronald Hanson. 2018. Quantum Internet: A Vision for the Road Ahead. *Science* 362, 6412 (2018), 1–9. https://doi.org/10.1126/science.aam9288

Received 2023-11-16; accepted 2024-03-31