Measurements in Proof Nets as Higher-Order Quantum Circuits

Akira Yoshimizu[1], Ichiro Hasuo[1], Claudia Faggian[2], and Ugo Dal Lago[3]

[1] University of Tokyo, Japan
[2] CNRS and Université Paris Diderot, Paris 7, France
[3] Università di Bologna, Italy

Abstract. We build on the series of work by Dal Lago and coauthors and identify proof nets (of linear logic) as higher-order quantum circuits. By accommodating quantum measurement using additive slices, we obtain a comprehensive framework for programming and interpreting quantum computation. Specifically, we introduce a quantum lambda calculus MLLqm and define its geometry of interaction (GoI) semantics—in the style of token machines—via the translation of terms into proof nets. Its soundness, i.e. invariance under reduction of proof nets, is established. The calculus MLLqm attains a pleasant balance between expressivity (it is higher-order and accommodates all quantum operations) and concreteness of models (given as token machines, i.e. in the form of automata).

1 Introduction

Quantum Programming Languages. Quantum computation and quantum communication have been attracting growing attention. The former achieves real breakthrough in computational power—at least for some classes of problems, such as the integer factorization problem (Shor's algorithm) and search problems. While it is often disputed if quantum computation is physically realizable, quantum communication is close to actual deployment in real-world applications. By exploiting the nonlocal character of quantum phenomena (notably *quantum entanglement*), quantum cryptography protocols accomplish *perfect security* that do not rely on any computational assumptions (like Diffie-Hellman).

Compared to the algorithmic aspects, the theory of *quantum programming* is relatively new. For example, quantum algorithms are most often expressed in *quantum circuits* that lack structuring means like recursion or higher-order functions. Consequently we have seen some proposals for quantum programming languages including QCL [19], quantum lambda calculi [21, 23] and most recently Quipper [10]: QCL is imperative and the others are functional.

Our interests are in a quantum lambda calculus as a prototype of functional quantum programming languages. The functional style comes with several advantages. For one, a type system based on resource-sensitive *linear logic* [6] can force *no-cloning* of quantum states via type safety [23]. Moreover, various techniques for classical functional programming can often be "transferred" to the quantum setting, since they are formulated in an abstract mathematical language and hence are generic. For example,

Z. Shao (Ed.): ESOP 2014, LNCS 8410, pp. 371–391, 2014.

in [11, 16, 21] various semantical techniques in the classical setting—such as linear-nonlinear adjunctions, categorical geometry of interaction, and presheaf completion—are applied to quantum calculi, exploiting the categorical genericity of these techniques.

From Quantum Circuits to Proof Nets. The current work relies on another rich body of techniques that are developed in the linear logic community. Specifically we follow the line of [3, 4] where, roughly speaking,

proof nets are thought of as *extended quantum circuits*.

Proof nets as devised in [6] are a graphical presentation of linear lambda terms (i.e. linear logic proofs) whose principal concern is reduction of terms (i.e. cut-elimination). Proof nets are "extended quantum circuits" in the following sense: (some) wires in proof nets can be naturally identified with those in quantum circuits; and at the same time higher-order computation is naturally accommodated using a linear type system ($A \multimap B \equiv A^{\perp} \parr B$). This view is hence a quantum version of the one in [22]. See §3.5 for further discussion.

Once a quantum lambda term is presented as a proof net, the *geometry of interaction (GoI)* interpretation [7]—especially its concrete presentation as *token machines* [14]—gives a concrete and operational interpretation of the term as a state transition system. This is a main advantage of the current "proof net and GoI" approach compared to the categorical one taken in [11, 16]: in the latter models tend to be abstract and huge.

A main disadvantage, however, is that it is harder to interpret extra features in a calculus. Such desired features include recursion and accommodation of duplicable classical data by the ! modality; these are all present e.g. in [11]. In fact, in the preceding work [3, 4] of the current approach, even measurements are excluded from the calculi. Hence important (and basic) examples like quantum teleportation cannot be expressed in their calculi.

Contributions. In the current work we present a comprehensive framework for programming and interpreting higher-order quantum computation based on a linear lambda calculus, proof nets and GoI interpretation. More specifically:

- We introduce MLLqm, a linear lambda calculus with quantum primitives (including measurement, unlike [3, 4]).
- We define a notion of *proof net*, into which terms of MLLqm are translated. For accommodating measurements we follow the idea of *(additive) slices* (see e.g. [8]). We also define the reduction of proof nets and prove that it is strongly normalizing.
- We define *token machine semantics* of MLLqm proof nets and prove that it is *sound*, i.e., is invariant under reduction of proof nets. Here we have multiple tokens in a token machine (this is as in [4]); the slices are suitably handled following the token machine semantics in [13] for additives.

Our framework attains a balance between *expressivity* and *concreteness of models* that we find pleasant. On the one hand, the calculus MLLqm is reasonably expressive: it does include all the quantum operations (preparation, unitary transformation, and most importantly, measurement) and is capable of expressing examples like quantum teleportation, which is not possible in the earlier work [3, 4] of the same proof net approach. Moreover, our framework can naturally express higher-order procedures that

are essential e.g. in formalizing *quantum pseudo-telepathy games* in quantum game theory. The latter are attracting attention as a useful presentation of quantum nonlocality (see e.g. [9]). On the other hand, while the languages in [11, 16, 21] are much more expressive—they include duplicable classical data (by the ! modality) and/or recursion—their models given in [11, 16] rely on abstract categorical constructions and it is not trivial to describe them in concrete terms. In contrast, our token machine semantics for MLLqm is given explicitly by a transition system.

The current work shares the same interest as [2], in the sense that both aim at pictorial formalisms for operational structures in quantum computation. We follow the linear logic tradition; an advantage is explicit correspondence with a term calculus. In contrast, [2] employs string diagrams for monoidal categories (more specifically compact closed categories with biproducts). The two approaches are not unrelated: there is a body of literature studying monoidal categories as models of linear logic. See [17] for a survey.

Organization of the Paper. After introducing the calculus MLLqm in §2, in §3 we define MLLqm *proof nets* and translate terms into proof nets. As usual, proof nets are defined to be *proof structures* satisfying a certain correctness criterion. We also define reduction (i.e. cut-elimination) of proof nets. In §4 we give GoI semantics to MLLqm proof nets, in the form of token machines. Our main result is soundness of the GoI semantics, i.e. that it is invariant under reduction of proof nets. Quantum teleportation will exemplify these constructions.

Most of the proofs are deferred to the extended version [24]. Familiarity to linear logic techniques like proof nets and token machine semantics is helpful in reading this paper. Our favorite reference is [20].

2 Syntax of Quantum Lambda Calculus MLLqm

We introduce a typed calculus MLLqm. It is a term calculus based on linear logic—specifically *multiplicative linear logic (MLL)* that has connectives $\otimes$, $\parr$ and $(\cdot)^\perp$. It is further augmented with quantum primitives that are rich enough to express any *quantum operation*. The latter notion is roughly for "what we can do to quantum states" and can be represented as a combination of *preparation*, *unitary transformation* and *measurement*. See [18, Chap. 8] for more details. The name MLLqm stands for "MLL for quantum computation with measurements."

Definition 2.1 (Types of MLLqm). Types *of* MLLqm *are defined by the following BNF:*

$$A, B ::= \mathsf{qbit} \mid \mathsf{qbit}^\perp \mid A \otimes B \mid A \parr B \,.$$

The syntactic equality shall be denoted by $\equiv$*. As is customary in linear logic, we syntactically identify types according to the following rules:* $(A \otimes B)^\perp \equiv A^\perp \parr B^\perp$, $(A \parr B)^\perp \equiv A^\perp \otimes B^\perp$, *and* $(A^\perp)^\perp \equiv A$. *We write* $A \multimap B$ *for* $A^\perp \parr B$ *and* $A^{\otimes n}$ *for* $(\cdots(A \otimes A) \otimes A)\cdots) \otimes A$ *(here* $\otimes$ *occurs* $n-1$ *times).*

Definition 2.2 (Terms of MLLqm). Terms *of* MLLqm *are defined by:*

$$\begin{aligned} M, N, L \;::=\;& x \mid \lambda x^A.M \mid MN \mid \langle M, N\rangle \mid \lambda\langle x^A, y^B\rangle.M \\ & \mid \mathsf{new}_{|\varphi\rangle} \mid \mathsf{U} \mid \mathsf{if\ meas}\ M\ \mathsf{then}\ N\ \mathsf{else}\ L \;\;. \end{aligned}$$

Here x *is an element of a fixed countable set* $\mathbf{Var}$ *of* variables. $\mathsf{new}_{|\varphi\rangle}$ *is a constant for each normalized vector* $|\varphi\rangle$ *in* $\mathbb{C}^2$ *and designates preparation of a qubit.* U *is a constant*

for each 2^n-dimension unitary matrix, where $n \in \mathbb{N}$. Measurements meas *occur only in conditionals. Note that in variable binders λx^A and $\lambda\langle x^A, y^B\rangle$, variables x, y come with explicit type labels. This is to ensure Lem. 2.5.*

Remark 2.3. *The constructor* if meas M then N else L *is intended for "classical control": operationally, the qubit represented by M is* actually measured *before going on to evaluate N or L.*

This is not to be confused with "quantum control." In quantum circuits, it is well-known that any measurement can be postponed to the end of a circuit (the principle of deferred measurement*, [18, §4.4]). This is possible by use of* controlled operations *like* CNOT *[18, §4.3]. We shall stick to classical control because, in the current higher-order setting, it is not clear how to simulate classical control by quantum control, or how to systematically construct quantum controlled operations.*

Definition 2.4 (Typing rules of MLLqm**).** *Typing rules of* MLLqm *are shown below. A* context Γ *in a type judgment is a set $\{x_1 : A_1, \ldots, x_n : A_n\}$ of variables and their types. We write its* domain $\{x_1, \ldots, x_n\}$ *as $|\Gamma|$. The juxtaposition Γ, Δ of contexts denotes their union and we assume $|\Gamma| \cap |\Delta| = \emptyset$.*

$$\frac{}{x : A \vdash x : A}\ \mathsf{ax} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x^A.M : A \multimap B}\ \multimap I_1 \qquad \frac{\Gamma, x : A, y : B \vdash M : C}{\Gamma \vdash \lambda\langle x^A, y^B\rangle.M : A \otimes B \multimap C}\ \multimap I_2$$

$$\frac{\Gamma \vdash M : A \multimap B \quad \Delta \vdash N : A}{\Gamma, \Delta \vdash MN : B}\ \multimap E \qquad \frac{\Gamma \vdash M : A \quad \Delta \vdash N : B}{\Gamma, \Delta \vdash \langle M, N\rangle : A \otimes B}\ \otimes I$$

$$\frac{}{\vdash \mathsf{new}_{|\varphi\rangle} : \mathsf{qbit}}\ \mathsf{new} \qquad \frac{}{\vdash \mathsf{U} : \mathsf{qbit}^{\otimes n} \multimap \mathsf{qbit}^{\otimes n}}\ \mathsf{U}_n$$

$$\frac{\Gamma \vdash M : \mathsf{qbit} \quad \Delta \vdash N : A \quad \Delta \vdash L : A}{\Gamma, \Delta \vdash \mathsf{if\ meas}\ M\ \mathsf{then}\ N\ \mathsf{else}\ L : A}\ \mathsf{meas}$$

The rule $\multimap I_2$ replaces the usual $\otimes E$ rule that is problematic in the current linear setting. The following will enable inductive translation of terms into proof nets.

Lemma 2.5. *A derivable type judgment $\Gamma \vdash M : A$ has a unique derivation.* □

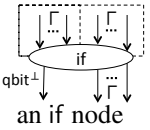
3 MLL Proof Nets with Quantum Nodes

In this section we introduce the notion of proof nets tailored for the calculus MLLqm. It is based on MLL proof nets [6] (see also [20]) and has additional nodes that correspond to quantum primitives (preparation, unitary transformation and measurement). Among them, (conditionals based on) measurements are the most challenging to model; we follow the idea of *additive slices* that are successfully utilized e.g. in [15].

As usual, we start with the notion of *proof structures* as graphs consisting of certain nodes. Then *proof nets* are defined to be those proof structures which comply with a *correctness criterion* (like Danos & Regnier's in [5]). We define translation of MLLqm terms into proof structures, which we prove to be proof nets. Moreover, we define reduction of proof structures, which we think of as one operational semantics of MLLqm terms. It is shown that proof nets are reduced to proof nets, and that reduction of proof nets is strongly normalizing (SN). Note that recursion is not in MLLqm.

3.1 MLLqm Proof Structures

In addition to the usual nodes in MLL proof nets, we introduce three kinds of nodes for quantum computation: new (preparation of a single qubit), U (unitary transformations/gates), and if (conditionals according to measurement of a qubit). An if node is as shown on the right. It is like a *box* in standard proof nets.



An if node will appear in a proof structure in the form where the two dashed boxes on its top are filled with "internal" proof structures. Such a combination of an if node and two (internal) proof structures shall be called a meas *node*. Overall, in MLLqm proof structures we allow the following seven kinds of nodes (Fig. 1).

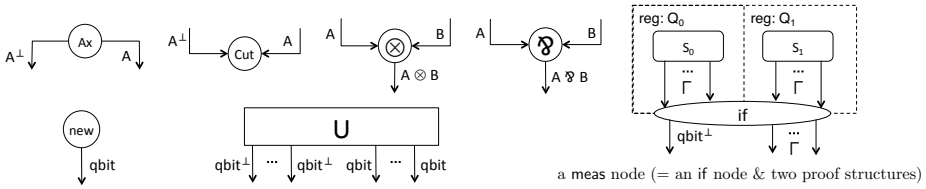


Fig. 1. Nodes of MLLqm proof structures

Note that nodes and proof structures are defined by mutual induction: in a proof structure there is a meas node, in whose dashed boxes there are other internal proof structures, and so on. We will make this precise in Def. 3.1. In Fig. 1, a unitary gate node for a 2^n-dimension unitary matrix U has n-many qbit edges and n-many $\mathsf{qbit}^{\perp}$ edges. Γ denotes a finite sequence of types. In a meas node, the $\mathsf{qbit}^{\perp}$-typed edge sticking out to the down-left is called a *query edge*.

As usual, incoming edges of a node are called *premises* and outgoing edges are called *conclusions*. A proof structure is roughly a graph that consists of nodes in Fig. 1, and

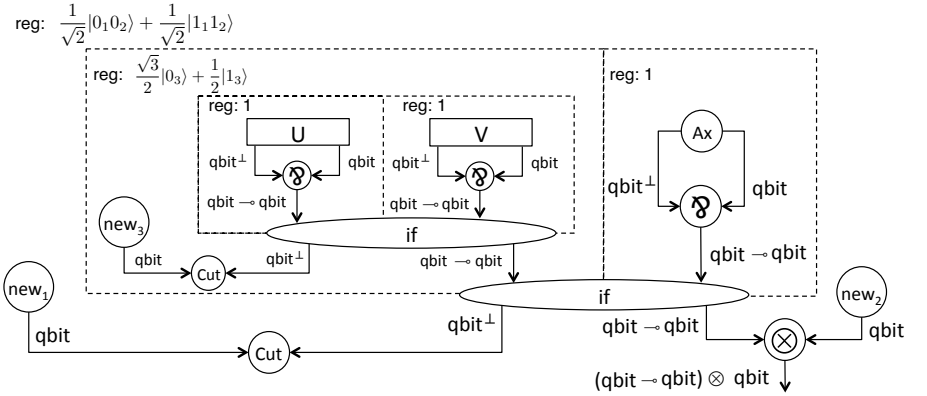


Fig. 2. An example of proof structure

is augmented with a quantum state called a *quantum register*, whose functionality we shall explain by an example.

See Fig. 2. The outermost proof structure (we say it is of *level* 0) has two new nodes, a cut node, a $\otimes$ node and a meas node. Its quantum register is a state of a 2-qubit system; each qubit corresponds to a certain new node and the correspondence is designated by indices. Therefore our intention is that each proof structure has a quantum register whose size is the number of new nodes, and that the proof structure explicitly carries the content of the quantum register. Such pairing of computational structure (proof structures here) and quantum registers is inspired by the operational semantics of [21], where a term of a calculus and a quantum state together form a quantum closure.

Definition 3.1 (MLLqm **proof structure).** *Let $\mathcal{S}$ be a directed finite graph consisting of nodes in Fig. 1; Q be a quantum register of length $n \in \mathbb{N}$ (that is, a normalized vector in $\mathbb{C}^{2^n}$); k be the number of* new *nodes in $\mathcal{S}$; and l be a bijection $\{$the* new *nodes in $\mathcal{S}\} \xrightarrow{\cong} \{1, 2, \ldots, k\}$. A triple $(\mathcal{S}, Q, l)$ satisfying*

- *each edge in $\mathcal{S}$ is well-typed;*
- *no incoming edge in $\mathcal{S}$ is dangling; and*
- $n = k$

is called a proof structure. *The types on the dangling outgoing edges in $\mathcal{S}$ are called the* conclusions *of $\mathcal{S}$.*

*Let $(\mathcal{S}_0, Q_0, l_0)$ and $(\mathcal{S}_1, Q_1, l_1)$ be proof structures with the same conclusions, say Γ. We call a triple $\big($*if *node, $(\mathcal{S}_0, Q_0, l_0), (\mathcal{S}_1, Q_1, l_1)\big)$ a* meas node *and regard it as a node with conclusions* $\mathsf{qbit}^{\perp}$, Γ. *Each of the proof structures $(\mathcal{S}_0, Q_0, l_0)$ and $(\mathcal{S}_1, Q_1, l_1)$ is called a* branch *of the* meas *node.*

The outermost proof structure is said to be of level 0 *and the branches of a* meas *node of level n are said to be* of level $n + 1$.

We emphasize again that the above definitions of proof structures and meas nodes are mutually inductive. We allow meas nodes nested only finitely many times. The bijection l in a proof structure $(\mathcal{S}, Q, l)$ gives indices to new nodes and designates correspondences between new nodes and qubits in a quantum register Q.

For example, in Fig. 2 the unitary gate nodes U and V belong to level 2. The quantum state that corresponds to the node new_3 is in the level-1 register. Note that it is invisible from level 0.

Finally we define *slices* for MLLqm proof structures, like usual additive slices. We will employ this notion later in §4.

Definition 3.2 (Slicing and slices). *Let $\mathcal{N} = (\mathcal{S}, Q, l)$ be an* MLLqm *proof structure. A* slicing *is a function $b : \{$all* if *nodes in $\mathcal{S}$ (of any level)$\} \to \{0, 1\}$. Abusing notation, a* slice *$b(\mathcal{N})$ is a graph obtained by deleting the unselected branch of each* if *node according to the slicing b, i.e. if $b(v) = 0$ delete the branch on the right and if $b(v) = 1$ delete the branch on the left for each* if *node v. Note that a slice is not a proof structure.*

3.2 Reduction of MLLqm Proof Structures

We now introduce reduction rules for MLLqm proof structures. Following the Curry-Howard intuition that normalization of a proof is computation, a reduction step is thought of as a step in quantum computation.

Definition 3.3 (Reduction rules of MLLqm proof structures). Reduction rules *are shown in Fig. 3. The first two are standard in MLL proof nets; the latter three are new. In the unitary gate rule, the unitary matrix* $\mathsf{U}^{j_1,\ldots,j_m}$ *acts on* $j_1,\ldots,j_m$*-th qubits in the same way as* U *does, and leaves other qubits unchanged. The last two rules occur probabilistically, where the resulting quantum registers* $|\varphi_0'\rangle$, $|\varphi_1'\rangle$ *and probabilities* $\sum_j |\alpha_j|^2$, $\sum_j |\beta_j|^2$ *defined in the obvious way. Explicitly:*

$$\begin{aligned} |\varphi_0\rangle &= \textstyle\sum_j \alpha_j \left(|\psi_j^0\rangle \otimes |0\rangle \otimes |\chi_j^0\rangle\right) , \quad |\varphi_0'\rangle = \sum_j \frac{\alpha_j}{\sqrt{\sum_k |\alpha_k|^2}} \left(|\psi_j^0\rangle \otimes |\chi_j^0\rangle\right) , \\ |\varphi_1\rangle &= \textstyle\sum_j \beta_j \left(|\psi_j^1\rangle \otimes |1\rangle \otimes |\chi_j^1\rangle\right) , \quad |\varphi_1'\rangle = \sum_j \frac{\beta_j}{\sqrt{\sum_k |\beta_k|^2}} \left(|\psi_j^1\rangle \otimes |\chi_j^1\rangle\right) , \end{aligned} \tag{1}$$

where $|\psi_j^b\rangle$ *of length* $m-1$ *and* m *is the index of the* new *node that is measured. The other rules occur with probability* 1*. In* meas *rules, the indexing function* l *is suitably updated too.*

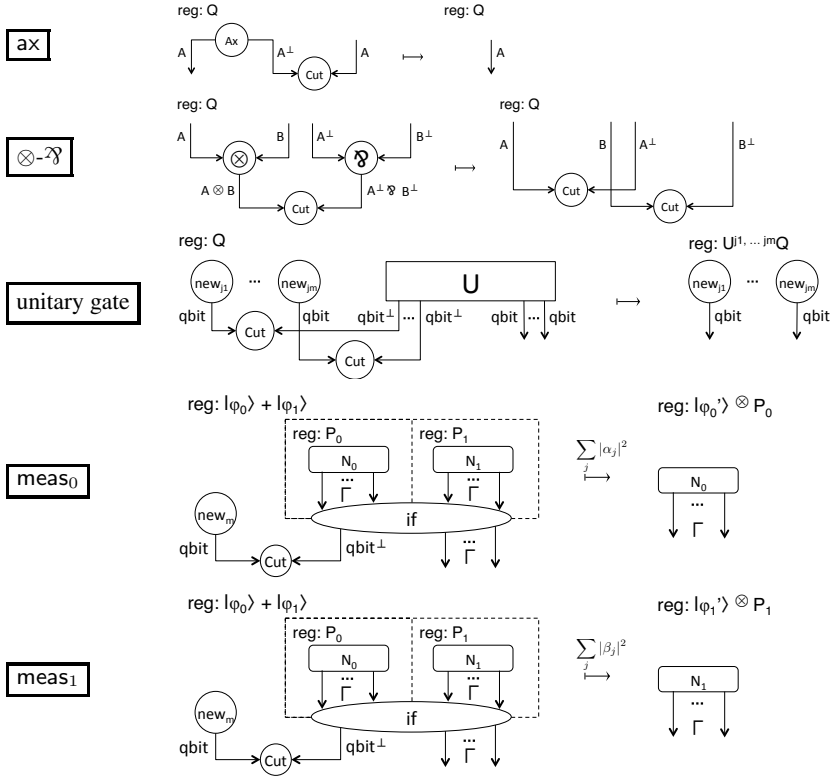
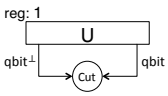


Fig. 3. Reduction rules of MLLqm proof structures

3.3 MLLqm Proof Nets and the Correctness Criterion

Our view of MLLqm proof structures is that they are "extended quantum circuits" that allow formalization of higher-order quantum computation.



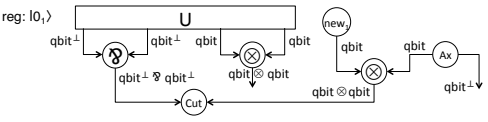
As usual with proof structures, however, Def. 3.1 does not exclude proof structures that carries no computational contents—to put it technically, those which have cut nodes that cannot be eliminated. This is mainly due to vicious "feedback loops," as seen in the proof structure on the right. We exclude such feedback loops by imposing a *correctness criterion* that is similar to Danos and Regnier's "connected and acyclic" one [5]. Then *proof nets* are proof structures that comply with the correctness criterion.

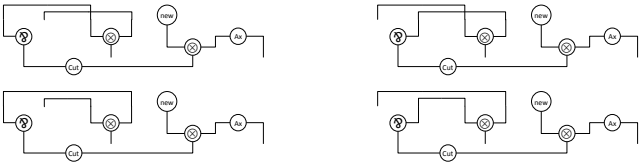
In the current quantum setting the challenge is to devise a graph-theoretic correctness condition for unitary gate nodes. We follow the idea in [4].

Definition 3.4 (Correctness graphs with quantum nodes). *Let $\mathcal{N} = (\mathcal{S}, Q, l)$ be a proof structure. A* correctness graph *of $\mathcal{N}$ is an undirected graph obtained by applying the following operations to $\mathcal{S}$.*

- *Ignore directions of all edges.*
- *For each $\parr$ node, choose one of the two premises and disconnect the other.*
- *For each unitary gate node, choose an arbitrary bijective correspondence between the sets of* $\mathsf{qbit}^{\perp}$ *edges and* qbit *edges. Remove the node and connect each correspondent pair of edges.*
- *For each* meas *node, ignore its branches.*

Here is an example. The correctness graphs for the proof structure on the right are the four undirected graphs below. There are two choices for the $\parr$ node and two for the unitary gate node.





Definition 3.5 (MLLqm **proof nets).** *A correctness graph is said to* satisfy the correctness criterion *if it is acyclic and connected.*

A proof structure $\mathcal{N}$ is called a proof net *if each of its correctness graphs satisfies the correctness criterion and every branch in it is a proof net.*

Lemma 3.6. *If a proof net $\mathcal{N}$ reduces to another proof structure $\mathcal{N}'$ (according to the rules in Def. 3.3), then $\mathcal{N}'$ is also a proof net.* □

3.4 Translation of MLLqm Terms into Proof Nets

We assign a proof structure $[\![\Gamma \vdash M : A]\!]$ to each derivable type judgment $\Gamma \vdash M : A$. This turns out to satisfy the correctness criterion. Lem. 2.5 allows for the definition of $[\![\Gamma \vdash M : A]\!]$ by induction on derivation.

Definition 3.7 (Translation of terms into proof nets). *For each derivable type judgment $\Gamma \vdash M : A$, a proof structure $[\![\Gamma \vdash M : A]\!]$ is defined inductively as in Fig. 4–5.*

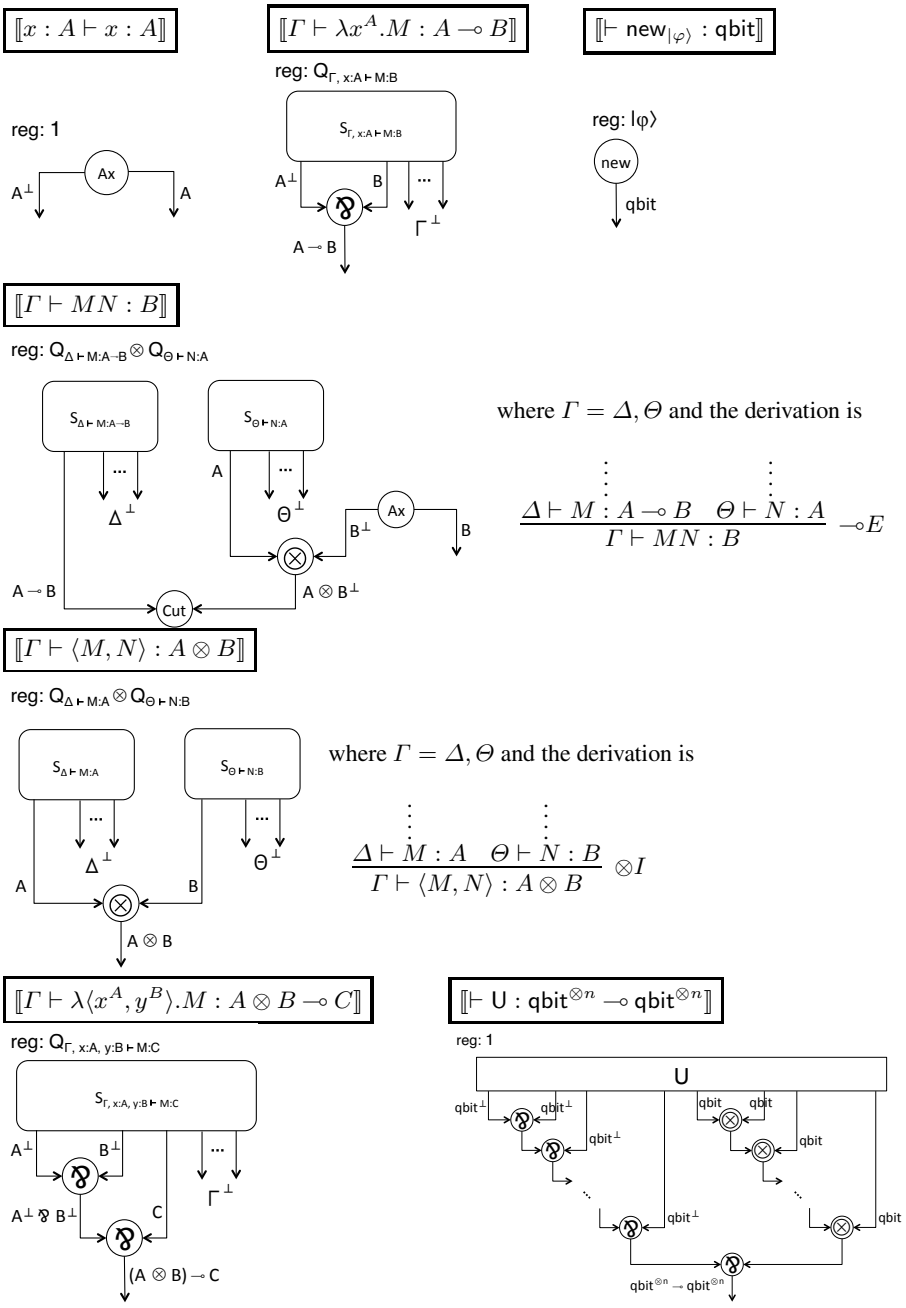


Fig. 4. Proof net translation of MLLqm terms—part I

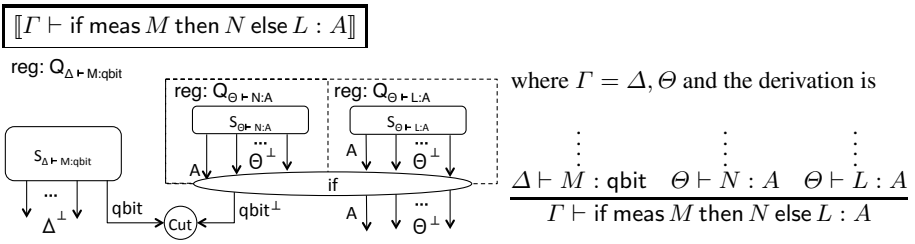


Fig. 5. Proof net translation of MLLqm terms—part II

Here we let $[\![\Gamma \vdash M : A]\!] = (\mathcal{S}_{\Gamma\vdash M:A}, Q_{\Gamma\vdash M:A}, l_{\Gamma\vdash M:A})$*; and* Γ *denotes a sequence* $A_1, A_2, \ldots, A_n$ *of types. In each case, the types* A_j *in the context* Γ *of* $\Gamma \vdash M : A$ *appear as their dual* ${A_j}^{\perp}$ *in the conclusions of* $\mathcal{S}_{\Gamma\vdash M:A}$*.*

The indexing l *between* new *nodes and quantum registers are merged in the obvious way, in the cases of* $[\![\Gamma \vdash \langle M, N\rangle : A \otimes B]\!]$ *and* $[\![\Gamma \vdash MN : B]\!]$*.*

Lemma 3.8. *For any derivable type judgment* $\Gamma \vdash M : A$*, the proof structure* $[\![\Gamma \vdash M : A]\!]$ *is a proof net.* □

Hence, regarding MLLqm proof structures as a rewriting system for quantum computation, it is sufficient to consider solely proof nets. This rewriting system exhibits the following pleasant properties (Thm. 3.9–3.10).

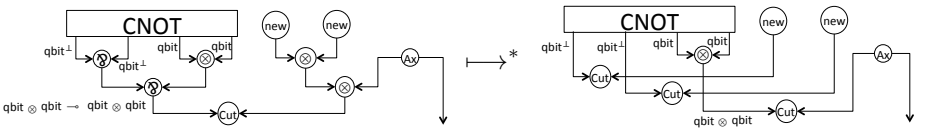
Theorem 3.9 (Termination of reduction). *The reduction of* MLLqm *proof nets is terminating.* □

Regarding reduction of proof nets as cut elimination, it is natural to expect all the cut nodes to disappear after reduction terminates. This is unfortunately not the case and we have the following restricted result (Thm. 3.10). The condition in Thm. 3.10 corresponds to the condition that a term of MLLqm is *closed*, i.e. has no free variable. Intuitively, it states that a proof net "executes all computation steps" if the whole input is given.

Theorem 3.10 (Strong normalization). *Let* $\mathcal{N} = (\mathcal{S}, Q, l)$ *be an* MLLqm *proof net. If no type containing* $\mathsf{qbit}^{\perp}$ *occurs in the conclusions of* $\mathcal{S}$*, then every maximal sequence of reductions from* $\mathcal{N}$ *reaches a proof net that contains no cut nodes, no unitary gate nodes, or no* if *nodes.* □

Remark 3.11. *For MLL proof nets, one of the purposes to introduce correctness criteria in [5, 6] is to* characterize *those proof structures which arise from some proof in sequent calculus. Therefore the converse of Lem. 3.8—so-called* sequentialization*—is also proved in [6]. It allows (re)construction of sequent calculus proofs from proof nets.*

However, sequentialization fails for MLLqm*. Consider the following reduction; the original proof net is the translation of the term* $\mathsf{CNOT}\langle \mathsf{new}_{|0\rangle}, \mathsf{new}_{|0\rangle}\rangle$*.*



After two $\otimes$-$\parr$ *reductions we do not yet get rid of the* CNOT *node; it is easily seen that there is no* MLLqm *term that gives rise to the resulting proof net.*

This is a phenomenon that reflects the nonlocal *character of* MLLqm*; and ultimately the nonlocality of quantum entanglement is to blame.*

Sequentialization fails in general. Those proof nets which are sequentializable include: the net $[\![\Gamma \vdash M : A]\!]$ *(trivially); and the normal form of the net* $[\![\Gamma \vdash M : A]\!]$ *for a closed term* M*. The latter is because Thm. 3.10 says that in that case the normal form is merely an MLL proof net with* new *nodes.*

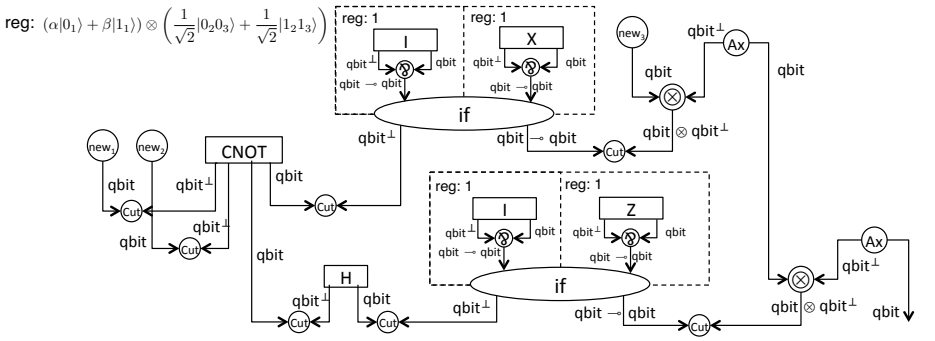


Fig. 6. Quantum teleportation (after some reductions irrelevant to the quantum part)

3.5 Examples and Discussion

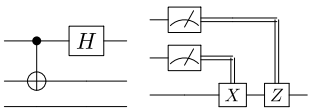
As syntax sugar we write $\langle x_1, x_2, x_3\rangle \equiv \langle x_1, \langle x_2, x_3\rangle\rangle$ and $\lambda\langle x_1^{A_1}, x_2^{A_2}, x_3^{A_3}\rangle.M \equiv \lambda\langle x_1^{A_1}, y^{A_2 \otimes A_3}\rangle.((\lambda\langle x_2^{A_2}, x_3^{A_3}\rangle.M)y)$, where y is a fresh variable. Let

$$B :\equiv \lambda\langle x^{\mathsf{qbit}}, y^{\mathsf{qbit}}, z^{\mathsf{qbit}}\rangle.\Big((\lambda\langle v^{\mathsf{qbit}}, w^{\mathsf{qbit}}\rangle.\langle \mathsf{H}\, v, w, z\rangle)(\mathsf{CNOT}\langle x, y\rangle)\Big)\ ,$$

$$C :\equiv \lambda\langle s^{\mathsf{qbit}}, t^{\mathsf{qbit}}, u^{\mathsf{qbit}}\rangle.(\mathsf{if\ meas}\ s\ \mathsf{then\ Z\ else\ I})\big((\mathsf{if\ meas}\ t\ \mathsf{then\ X\ else\ I})\, u\big)\ ,\ \text{and}$$

$$\beta_{00} :\equiv \mathsf{CNOT}\,\langle \mathsf{H}\,\mathsf{new}_{|0\rangle}, \mathsf{new}_{|0\rangle}\rangle$$

where H is the Hadamard gate, CNOT is the controlled not gate, I is the identity matrix, and Z and X are the Pauli matrices. The term β_{00} denotes one of the Bell state; and the terms B and C represent the quantum circuits on the right, respectively. Quantum teleportation of one qubit $\alpha|0\rangle + \beta|1\rangle$ (where $\alpha, \beta \in \mathbb{C}$) is then described as a MLLqm term $T :\equiv \big(\lambda x^{\mathsf{qbit}}.C(B\langle x, \beta_{00}\rangle)\big)\ \mathsf{new}_{\alpha|0\rangle+\beta|1\rangle}$.



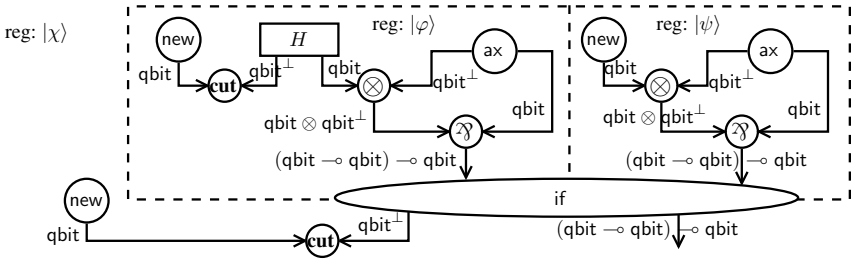
The term T is closed and has the type qbit. Its proof net translation $[\![\vdash T : \mathsf{qbit}]\!]$, after some reductions that are irrelevant to the quantum part, is shown in Fig. 6.

It is not hard to notice the similarity between the proof net in Fig. 6 and the presentation by a quantum circuit. In general, when we translate a first-order MLLqm term the resulting proof net looks quite much like a quantum circuit. Notice that the term T is indeed first-order.

It is when higher-order functions are involved that our linear logic based approach shows its real advantage. For example, the proof net in the figure below receives a transformation $\mathcal{E}$ of a qubit into a qubit as an input; and feeds $\mathcal{E}$ with either $H|\varphi\rangle$ or $|\psi\rangle$, according to the outcome of the measurement of $|\chi\rangle$. (It is straightforward to write down an MLLqm term that gives rise to this proof net. Explicitly, the term is: $\mathsf{if}\ \mathsf{meas}\ \mathsf{new}_{|\chi\rangle}\ \mathsf{then}\ (\lambda f^{\mathsf{qbit}\multimap\mathsf{qbit}}.f\ (\mathsf{H}\ \mathsf{new}_{|\phi\rangle}))\ \mathsf{else}\ (\lambda f^{\mathsf{qbit}\multimap\mathsf{qbit}}.f\ \mathsf{new}_{|\psi\rangle})$.) This is a "quantum circuit with a hole," so to speak; our current MLLqm framework can express, execute and reason about such procedures in a structural manner.

4 Token Machine Semantics for MLLqm Proof Nets

In this section we go on to introduce token machine semantics for MLLqm proof nets and prove its soundness, that is, the semantics is invariant under reduction of proof nets.



Token machines are one presentation of Girard's *geometry of interaction* [7]. Unlike the original presentation by C^*-algebras, token machines as devised in [14] are (concrete) automata and carry a strong operational flavor. For more details see [20].

The MLLqm token machines are different from the usual MLL ones in that it employs multiple tokens. Intuitively one token corresponds to one qubit; and they are required to synchronize when they go beyond a unitary gate node. This is one way how quantum entanglement (hence nonlocality) can be taken care of in token machine semantics. Use of multiple tokens is already in [4] where the style is called *wave-style token machine*. Multiple tokens inevitably results in *nondeterminism* in small-step behaviors of machines (which token moves first?). We prove confluence of small-step behaviors, and also uniqueness of big-step behaviors as its consequence. This is like in [4].

In the current work we go beyond [4] and interpret measurements too. For that purpose we rely on the ideas developed in linear logic towards accommodating additive connectives: namely *(additive) slicing* of proof nets, and *weights* in token machines. See e.g. [8, 13].

4.1 Tokens

We start with usual definitions. We follow [13] most closely. The presentation in [20] is essentially the same.

Definition 4.1 (Context). *A* context *is defined by the following BNF:*

$$C ::= [\,] \mid C \otimes A \mid A \otimes C \mid C \parr A \mid A \parr C \ ,$$

where A is a type of MLLqm*. Note that every context has exactly one* hole $[\,]$*. The type obtained by substituting a type A for the hole in a context C is denoted by $C[A]$. A context C is called a* context for *A if the type A is obtained by substituting some type B for the hole $[\,]$, i.e. $A \equiv C[B]$. The negation $C^\perp$ of a context C is defined in a natural way, e.g. $(\mathsf{qbit} \otimes [\,])^\perp := \mathsf{qbit}^\perp \parr [\,]$.*

Definition 4.2 (Token). *Given a proof net $\mathcal{N} = (\mathcal{S}, Q, l)$, a* token *is a 4-tuple (A, C, D, ζ) where*

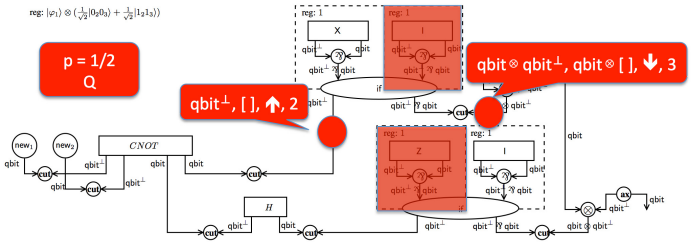
- *A is an edge of $\mathcal{S}$ (we abuse notations and identify an edge and the type occurrence A assigned to it; no confusion is likely),*
- *C is a context for A,*
- *D is a* direction*, that is an element of $\{\Uparrow, \Downarrow\}$, and*
- *$\zeta \in \mathbb{N}$.*

Intuitively, a token is a particle moving around the given proof net. The type occurrence A of a token indicates on which edge the token is. The context C designates which base type in A the token is concerned about. An example is $A \equiv \mathsf{qbit}^\perp \parr \mathsf{qbit}$ and $C \equiv [\,] \parr \mathsf{qbit}$; token machine semantics is defined in such a way that a token's context determines which edge to take when the token hits a fork, namely a $\parr$ node. The direction D of a token specifies whether it is going up or down along the edge.

Finally, the natural number ζ is a feature that is not in usual MLL proof nets: it records to which qubit of a quantum register the token corresponds. When a token is deployed the initial value of ζ is 0, meaning that the token does not yet know which qubit it corresponds to. When it hits a new node new_j, its index j is recorded in ζ.

4.2 The Token Machine $\mathcal{T}_\mathcal{N}$

Our goal is to construct a transition system (called a *token machine*) $\mathcal{T}_\mathcal{N}$ for a given MLLqm proof net $\mathcal{N}$. As an example, one state of the token machine is depicted below.



A state of $\mathcal{T}_\mathcal{N}$ is roughly the data that specifies the tokens in the proof net $\mathcal{N}$ (how many of them, their locations, their contexts, etc.).

In the current setting of MLLqm a state carries much more data, in fact. For example it has a slicing, which is depicted by hatching the unselected branches in the above figure. It may feel strange that the selection of branches are specified even before the relevant qubits are measured: a *probability*—that is also carried by a state of a token machine ($p = 1/2$ in the above figure)—represents the likelihood of the slicing actually taken. The formal definition is as follows.

Definition 4.3 (State). *Given a proof net $\mathcal{N} = (\mathcal{S}, Q_{\mathcal{N}}, l)$, a* state *of the token machine $\mathcal{T}_{\mathcal{N}}$ is a 5-tuple $(Q, p, b, T_{\mathsf{pr}}, T_{\mathsf{ms}})$ where*

- *Q is a quantum register,*
- *p is a probability, i.e. a real number satisfying $0 \leq p \leq 1$,*
- *b is a slicing,*
- *T_{pr} is a finite set of tokens (called* principal tokens*),*
- *T_{ms} is another finite set of tokens (called* measurement tokens*).*

A quantum register Q of a state is related to $Q_{\mathcal{N}}$ (that of the proof net) but not necessarily the same—this will be clarified by definitions below of the transition relation and the initial states of $\mathcal{T}_{\mathcal{N}}$.

We go on to define the transition structure $\to_{\mathcal{N}}$ of $\mathcal{T}_{\mathcal{N}}$ (Def. 4.4). We note that transitions $\to_{\mathcal{N}}$ form a binary relation between states—without any labels or probabilities assigned to transitions. Hence $\mathcal{T}_{\mathcal{N}}$ is simply a Kripke frame. We shall refer to the transitions $\to_{\mathcal{N}}$ in $\mathcal{T}_{\mathcal{N}}$ also as the *small-step semantics* of $\mathcal{T}_{\mathcal{N}}$.

The rules in Def. 4.4 are fairly complicated so their intuitions are stated first. The rules mainly describe how token(s) "move around the net." Almost every rule moves only one token. An exception is the U-Apply rule: it makes tokens "synchronized" and moves them at once. The if-Meas rule deletes one measurement token. The U-Apply and if-Meas rules also act on the quantum register and the probability of a state, reflecting the quantum effects of the corresponding operations. A slicing b is left untouched by transitions.

Definition 4.4 (Transition $\to_{\mathcal{N}}$ of the token machine $\mathcal{T}_{\mathcal{N}}$). *The* transition relation *$\to_{\mathcal{N}}$ between states of the token machine $\mathcal{T}_{\mathcal{N}}$ is defined by the rules as in Fig. 7–8. Each rule except the* U*-Apply and* if*-Meas rules is divided into two rules, one for principal tokens and the other for measurement tokens.*

For each rule, we informally depict the intended movement of token(s) too.

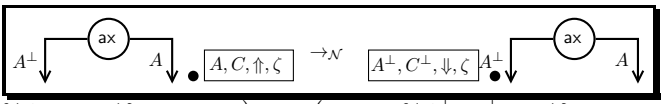
Hatching over a branch means the branch is not selected by the slicing.

Lemma 4.5 (One-step confluence). *Let $\mathcal{N} = (\mathcal{S}, Q, l)$ be an* MLLqm *proof net. The transition relation $\to_{\mathcal{N}}$ of its token machine $\mathcal{T}_{\mathcal{N}}$ is one-step confluent. That is, if both $s \to_{\mathcal{N}} s_1$ and $s \to_{\mathcal{N}} s_2$ hold, then either $s_1 = s_2$ or there exists a state s' such that $s_1 \to_{\mathcal{N}} s'$ and $s_2 \to_{\mathcal{N}} s'$.* □

4.3 Big-Step Semantics of $\mathcal{T}_{\mathcal{N}}$

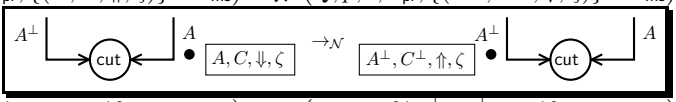
We identify the "computational content" of a proof net $\mathcal{N}$ to be the *big-step semantics* of the token machine $\mathcal{T}_{\mathcal{N}}$ that is defined below. The big-step semantics is intuitively the correspondence between an *initial state* $s \in I_{\mathcal{N}}$ and a *final state* $s' \in F_{\mathcal{N}}$, such

ax



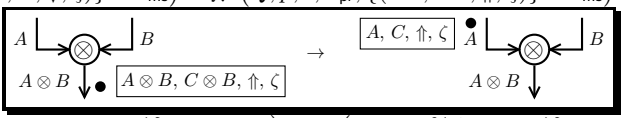
$(Q, p, b, \{(A, C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A^\perp, C^\perp, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A, C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A^\perp, C^\perp, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}})$

cut



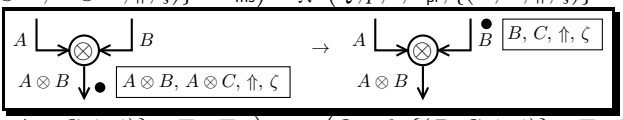
$(Q, p, b, \{(A, C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A^\perp, C^\perp, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A, C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A^\perp, C^\perp, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}})$

⊗-UpLeft



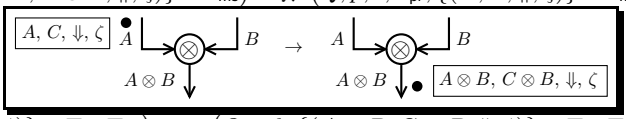
$(Q, p, b, \{(A \otimes B, C \otimes B, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A, C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A \otimes B, C \otimes B, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A, C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}})$

⊗-UpRight



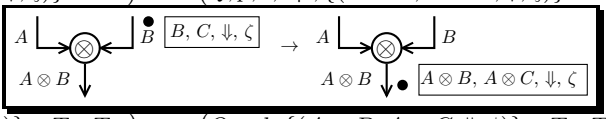
$(Q, p, b, \{(A \otimes B, A \otimes C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(B, C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A \otimes B, A \otimes C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(B, C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}})$

⊗-DownLeft



$(Q, p, b, \{(A, C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A \otimes B, C \otimes B, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A, C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A \otimes B, C \otimes B, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}})$

⊗-DownRight



$(Q, p, b, \{(B, C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A \otimes B, A \otimes C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(B, C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A \otimes B, A \otimes C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}})$

⅋-UpLeft (pictures for the ⅋ rules are similar to the ⊗ rules)

$(Q, p, b, \{(A \parr B, C \parr B, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A, C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A \parr B, C \parr B, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A, C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}})$

⅋-UpRight

$(Q, p, b, \{(A \parr B, A \parr C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(B, C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A \parr B, A \parr C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(B, C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}})$

⅋-DownLeft

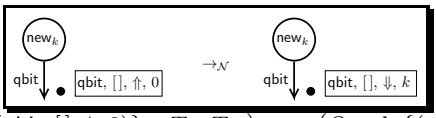
$(Q, p, b, \{(A, C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A \parr B, C \parr B, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A, C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A \parr B, C \parr B, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}})$

⅋-DownRight

$(Q, p, b, \{(B, C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A \parr B, A \parr C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(B, C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A \parr B, A \parr C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}})$

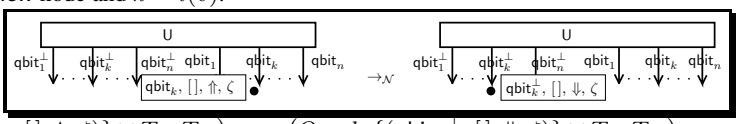
Fig. 7. Transition rules for $\mathcal{T}_{\mathcal{N}}$—part I

new



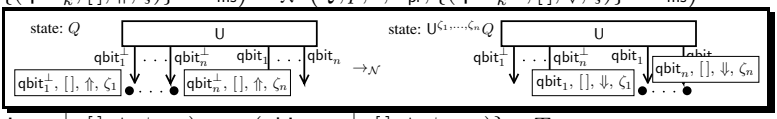
$(Q, p, b, \{(\mathsf{qbit}, [\,], \Uparrow, 0)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(\mathsf{qbit}, [\,], \Downarrow, k)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(\mathsf{qbit}, [\,], \Uparrow, 0)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(\mathsf{qbit}, [\,], \Downarrow, k)\} \cup T_{\mathsf{ms}})$
where v is the new node and $k = l(v)$.

U-Through



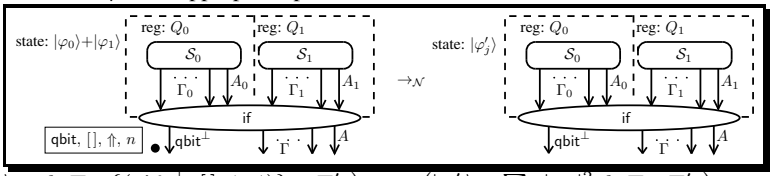
$(Q, p, b, \{(\mathsf{qbit}_k, [\,], \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(\mathsf{qbit}_k{}^{\perp}, [\,], \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(\mathsf{qbit}_k, [\,], \Uparrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(\mathsf{qbit}_k{}^{\perp}, [\,], \Downarrow, \zeta)\} \cup T_{\mathsf{ms}})$

U-Apply



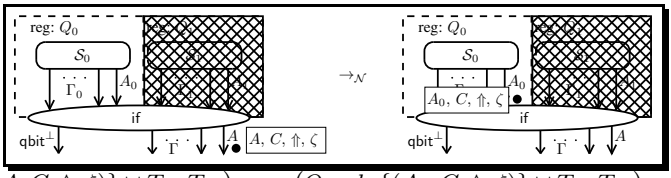
$(Q, p, b, \{(\mathsf{qbit}_{\rho(1)}{}^{\perp}, [\,], \Uparrow, \zeta_{\rho(1)}), \ldots, (\mathsf{qbit}_{\rho(m)}{}^{\perp}, [\,], \Uparrow, \zeta_{\rho(m)})\} \cup T_{\mathsf{pr}},$
$\{(\mathsf{qbit}_{\rho(m+1)}{}^{\perp}, [\,], \Uparrow, \zeta_{\rho(m+1)}), \ldots, (\mathsf{qbit}_{\rho(n)}{}^{\perp}, [\,], \Uparrow, \zeta_{\rho(n)})\} \cup T_{\mathsf{ms}})$
$\to_{\mathcal{N}} (\mathsf{U}^{\zeta_1,\ldots,\zeta_n}Q, p, b, \{(\mathsf{qbit}_{\rho(1)}, [\,], \Downarrow, \zeta_{\rho(1)}), \ldots, (\mathsf{qbit}_{\rho(m)}, [\,], \Downarrow, \zeta_{\rho(m)})\} \cup T_{\mathsf{pr}},$
$\{(\mathsf{qbit}_{\rho(m+1)}, [\,], \Downarrow, \zeta_{\rho(m+1)}), \ldots, (\mathsf{qbit}_{\rho(n)}, [\,], \Downarrow, \zeta_{\rho(n)})\} \cup T_{\mathsf{ms}})$
where $0 \leq m \leq n$ and ρ is an appropriate permutation.

if-Meas



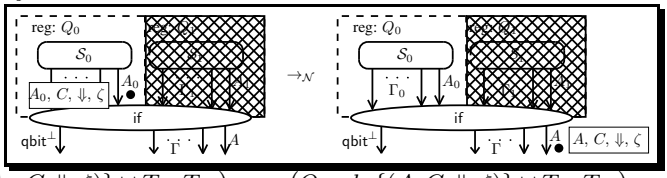
$(|\varphi_0\rangle + |\varphi_1\rangle, p, b, T_{\mathsf{pr}}, \{(\mathsf{qbit}^{\perp}, [\,], \Uparrow, \zeta)\} \cup T'_{\mathsf{ms}}) \to_{\mathcal{N}} (|\varphi'_0\rangle, p \sum_j |\alpha_j|^2, b, T_{\mathsf{pr}}, T'_{\mathsf{ms}})$
when $b(v) = 0$ where v is the if node.
$(|\varphi_0\rangle + |\varphi_1\rangle, p, b, T_{\mathsf{pr}}, \{(\mathsf{qbit}^{\perp}, [\,], \Uparrow, \zeta)\} \cup T'_{\mathsf{ms}}) \to_{\mathcal{N}} (|\varphi'_1\rangle, p \sum_j |\beta_j|^2, b, T_{\mathsf{pr}}, T'_{\mathsf{ms}})$
when $b(v) = 1$ where v is the if node.

if-In



$(Q, p, b, \{(A, C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A_j, C, \Uparrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A, C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A_j, C, \Uparrow, \zeta)\} \cup T_{\mathsf{ms}})$
when $b(v) = j$ where v is the if node.

if-Out



$(Q, p, b, \{(A_j, C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, \{(A, C, \Downarrow, \zeta)\} \cup T_{\mathsf{pr}}, T_{\mathsf{ms}})$
$(Q, p, b, T_{\mathsf{pr}}, \{(A_j, C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}}) \to_{\mathcal{N}} (Q, p, b, T_{\mathsf{pr}}, \{(A, C, \Downarrow, \zeta)\} \cup T_{\mathsf{ms}})$

Fig. 8. Transition rules for $\mathcal{T}_{\mathcal{N}}$—part II

that s reaches s' via a succession of $\to_{\mathcal{N}}$. By confluence of $\to_{\mathcal{N}}$ (Lem. 4.5) such s' is shown to be unique if it exists (Prop. 4.12); hence the big-step semantics is given as a partial function $I_{\mathcal{N}} \rightharpoonup F_{\mathcal{N}}$. Later in §4.4 we will show *soundness*, that is, the big-step semantics is invariant under the reduction of proof nets (as defined in §3), modulo certain "quantum effects."

We start with singling out some states of $\mathcal{T}_{\mathcal{N}}$ as *initial* and *final*.

Notation 4.6 (Q_b^v)**.** *Let $\mathcal{N} = (\mathcal{S}, Q_{\mathcal{N}}, l)$ be an* MLLqm *proof net, b be a slicing of $\mathcal{N}$, and v be an* if *node in $\mathcal{S}$. By Q_b^v we denote the quantum register associated with the branch designated by b.*

Hence Q_b^v is a quantum register inside a dashed box attached to the if node v.

Definition 4.7 (Initial states). *Let $\mathcal{N} = (\mathcal{S}, Q_{\mathcal{N}}, l)$ be an* MLLqm *proof net. A state $s = (Q, p, b, T_{\mathsf{pr}}, T_{\mathsf{ms}})$ of $\mathcal{T}_{\mathcal{N}}$ is said to be* initial *if:*

- $Q = Q_{\mathcal{N}} \otimes \left(\bigotimes_{v \in V} Q_b^v \right)$ *where V is the set of all* if *nodes in the slice $b(\mathcal{N})$ (of any level; recall Def. 3.2).*
- *A token (A, C, D, ζ) belongs to T_{pr} if and only if*
  - *A is a conclusion edge of level 0 (recall that we denote an edge by its type occurrence);*
  - *$C[\mathsf{qbit}] \equiv A$; $D = \Uparrow$; and $\zeta = 0$.*
- *A token (A, C, D, ζ) belongs to T_{ms} if and only if*
  - *$A \equiv \mathsf{qbit}^{\perp}$, a query edge (one sticking left-down from an* if *node) in a branch remaining in the slice $b(\mathcal{N})$;*
  - *$C \equiv [\,]$; $D = \Downarrow$; and $\zeta = 0$.*

The set of initial states is denoted by $I_{\mathcal{N}}$.

In an initial state, every principal token is at one of the conclusion edges (of level 0), waiting to go up. Measurement tokens are at query edges of any level (but only those which are in the slice $b(\mathcal{N})$). The quantum register Q keeps track not only of the level-0 register $Q_{\mathcal{N}}$ but also of "internal" registers (again which are in the slice $b(\mathcal{N})$).

Definition 4.8 (Final states). *Let $\mathcal{N} = (\mathcal{S}, Q_{\mathcal{N}}, l)$ be an* MLLqm *proof net. A state $s = (Q, p, b, T_{\mathsf{pr}}, T_{\mathsf{ms}})$ of $\mathcal{T}_{\mathcal{N}}$ is said to be* final *if:*

- *each principal token $(A, C, D, \zeta) \in T_{\mathsf{pr}}$ satisfies*
  - *A is a conclusion edge;*
  - *$C[\mathsf{qbit}] = A$; and $D = \Downarrow$.*
- *$T_{\mathsf{ms}} = \emptyset$.*

Therefore in a final state, all the principal tokens are back at conclusion edges, and all the measurement tokens are gone. Recall that the if-Meas transition in Def. 4.4 deletes a measurement token.

Definition 4.9 (Token machine). *The* token machine *for an* MLLqm *proof net $\mathcal{N}$ is the 4-tuple $\mathcal{T}_{\mathcal{N}} = (S_{\mathcal{N}}, I_{\mathcal{N}}, F_{\mathcal{N}}, \to_{\mathcal{N}})$ where $S_{\mathcal{N}}$ is the set of states (Def. 4.3), $I_{\mathcal{N}}$ and $F_{\mathcal{N}}$ are the sets of initial and final states (Def. 4.7–4.8), and $\to_{\mathcal{N}} \subseteq S_{\mathcal{N}} \times S_{\mathcal{N}}$ is the (small-step) transition relation (Def. 4.4).*

In what follows, the transitive closure of $\to_{\mathcal{N}}$ is denoted by $\to_{\mathcal{N}}^{+}$.

Definition 4.10 (Big-step semantics). *Let $\mathcal{N}$ be an* MLLqm *proof net. The* big-step semantics *of the token machine $\mathcal{T}_{\mathcal{N}}$, denoted by $[\![\mathcal{N}]\!]$, is the partial function $[\![\mathcal{N}]\!] : I_{\mathcal{N}} \rightharpoonup F_{\mathcal{N}}$ defined by $[\![\mathcal{N}]\!](s) := \begin{cases} s' \in F_{\mathcal{N}} & \text{if } s \to_{\mathcal{N}}^{+} s'; \\ \bot & \text{otherwise.} \end{cases}$*

Prop. 4.12 below exhibits the legitimacy of this definition (as a partial function). It is not *total* but *partial* in general: partiality arises when the conclusion contains a $\mathsf{qbit}^{\perp}$. For the proof nets translated from *closed* MLLqm terms, it is always total (Cor. 4.16).

Lemma 4.11 (Termination of transition). *Let $\mathcal{N} = (\mathcal{S}, Q, l)$ be an* MLLqm *proof net. There is no infinite sequence of small-step transitions $\to_{\mathcal{N}}$ in $\mathcal{T}_{\mathcal{N}}$.* □

Proposition 4.12 (Unique final state). *Let $\mathcal{N} = (\mathcal{S}, Q, l)$ be an* MLLqm *proof net. If $s \to_{\mathcal{N}}^{+} s_0$ and $s \to_{\mathcal{N}}^{+} s_1$ with $s_0, s_1 \in F_{\mathcal{N}}$, then $s_0 = s_1$.* □

4.4 Soundness of the Token Machine Semantics

Soundness of the big-step semantics—that it is invariant under the reduction of proof nets—holds only modulo certain quantum effects. The latter are formalized as follows, as suitable transformations of token machine states.

Definition 4.13 ($\overline{\mathsf{U}}$). *Let $\mathcal{N} = (\mathcal{S}, Q_{\mathcal{N}}, l)$ be an* MLLqm *proof net. Assume that there is a unitary gate node U in $\mathcal{N}$ for which the unitary gate reduction rule in Fig. 3 can be applied, resulting in the proof net $\mathcal{N}'$. In this case, we define a function $\overline{\mathsf{U}} : S_{\mathcal{N}} \to S_{\mathcal{N}'}$ by $\overline{\mathsf{U}}(Q, p, b, T_{\mathsf{pr}}, T_{\mathsf{ms}}) := (\mathsf{U}^{j_1,\dots,j_m} Q, p, b, T_{\mathsf{pr}}, T_{\mathsf{ms}})$.*

Definition 4.14 ($\overline{\mathsf{meas}}$). *Let $\mathcal{N} = (\mathcal{S}, Q_{\mathcal{N}}, l)$ be an* MLLqm *proof net. Assume that there is an* if *node v in $\mathcal{N}$ to which the* meas0 *and* meas1 *rules in Fig. 3 are applicable, resulting in nets $\mathcal{N}_0$ and $\mathcal{N}_1$, respectively.*

First we define functions $\overline{\mathsf{meas}}^{v}_{|0\rangle} : I_{\mathcal{N}} \to I_{\mathcal{N}_0}$ and $\overline{\mathsf{meas}}^{v}_{|1\rangle} : I_{\mathcal{N}} \to I_{\mathcal{N}_1}$, by

$$\overline{\mathsf{meas}}^{v}_{|0\rangle}\big(|\varphi_0\rangle + |\varphi_1\rangle, p, b, T_{\mathsf{pr}}, \{(qbit^{\perp}, [\,], \Downarrow, \zeta)\} \cup T_{\mathsf{ms}}\big) := \big(|\varphi_0'\rangle, p \textstyle\sum_j |\alpha_j|^2, b_0, T_{\mathsf{pr}}, T_{\mathsf{ms}}\big),$$
$$\overline{\mathsf{meas}}^{v}_{|1\rangle}\big(|\varphi_0\rangle + |\varphi_1\rangle, p, b, T_{\mathsf{pr}}, \{(qbit^{\perp}, [\,], \Downarrow, \zeta)\} \cup T_{\mathsf{ms}}\big) := \big(|\varphi_1'\rangle, p \textstyle\sum_j |\beta_j|^2, b_1, T_{\mathsf{pr}}, T_{\mathsf{ms}}\big),$$

where b_j is defined by $b_j(u) := b(u)$ on every if *node u in the proof net $\mathcal{N}_j$ ($j \in \{0, 1\}$). Here the token $(qbit^{\perp}, [\,], \Downarrow, \zeta)$ in the definition is on the query edge of v, and $|\varphi_0\rangle, |\varphi_0'\rangle, |\varphi_1\rangle, |\varphi_1'\rangle$ are registers as in (1) in §3.2.*

Finally we define a function $\overline{\mathsf{meas}}^{v} : I_{\mathcal{N}} \to I_{\mathcal{N}_0} + I_{\mathcal{N}_1}$ by ($+$ denotes disjoint union)

$$\overline{\mathsf{meas}}^{v}(s) := \begin{cases} \overline{\mathsf{meas}}^{v}_{|0\rangle}(s) & \text{if } b(v) = 0, \\ \overline{\mathsf{meas}}^{v}_{|1\rangle}(s) & \text{if } b(v) = 1, \end{cases} \quad \text{where } s = (|\varphi\rangle, p, b, T_{\mathsf{pr}}, T_{\mathsf{ms}}).$$

Intuitively, the function $\overline{\mathsf{meas}}^{v}$ "deletes" the if node v together with relevant entries in the slicing b. A quantum register and a probability are updated too, in an obvious manner.

Using these state transformations our main result is stated as follows.

Theorem 4.15 (Soundness). *Let $\mathcal{N} \mapsto \mathcal{N}'$ be a reduction of* MLLqm *proof nets. Then,*

1. *$[\![\mathcal{N}]\!] = [\![\mathcal{N}']\!]$ if the reduction is by the* ax-cut *or the $\otimes$-$\parr$ rule.*
2. *$[\![\mathcal{N}]\!] = [\![\mathcal{N}']\!] \circ \overline{\mathsf{U}}$ if the reduction is by the unitary gate rule, where U is the corresponding unitary matrix.*

3. $[\![\mathcal{N}]\!] \simeq ([\![\mathcal{N}_0]\!] + [\![\mathcal{N}_1]\!]) \circ \overline{\mathsf{meas}}^v$ *if the reduction is by one of the* meas *rules. In this case there must be another reduction possible due to the other* meas *rule, and we denote the resulting two proof nets by* $\mathcal{N}_0$ *and* $\mathcal{N}_1$ *(*$\mathcal{N}'$ *is one of these). The function* $[\![\mathcal{N}_0]\!] + [\![\mathcal{N}_1]\!]$ *means case-distinction (recall the type* $I_{\mathcal{N}} \to I_{\mathcal{N}_0} + I_{\mathcal{N}_1}$ *of* $\overline{\mathsf{meas}}^v$*). Here the equivalence* $\simeq$ *is a natural identification of final states of* $\mathcal{T}_{\mathcal{N}}, \mathcal{T}_{\mathcal{N}_0}$ *and* $\mathcal{T}_{\mathcal{N}_1}$*. That is,* $F \simeq G \overset{\text{def.}}{\iff} \forall x.F(x) \sim G(x)$ *and*
$s \sim s' \quad \overset{\text{def.}}{\iff} \quad s = s'$ *disregarding slicings.*

Pictorially, the statements 2. *and* 3. *say the following diagrams commute:*

$$\begin{array}{ccc} I_{\mathcal{N}} & \xrightarrow{[\![\mathcal{N}]\!]} & F_{\mathcal{N}} \\ {\scriptstyle \overline{\mathsf{U}}}\downarrow & & \| \\ I_{\mathcal{N}'} & \xrightarrow{[\![\mathcal{N}']\!]} & F_{\mathcal{N}'} \end{array} \qquad \begin{array}{ccc} I_{\mathcal{N}} & \xrightarrow{[\![\mathcal{N}]\!]} & F_{\mathcal{N}} \\ {\scriptstyle \overline{\mathsf{meas}}^v}\downarrow & & \downarrow{\scriptstyle \sim} \\ I_{\mathcal{N}_0} + I_{\mathcal{N}_1} & \xrightarrow{[\![\mathcal{N}_0]\!]+[\![\mathcal{N}_1]\!]} & F_{\mathcal{N}_0} + F_{\mathcal{N}_1} \end{array} \ . \qquad \square$$

Thm. 4.15 together with Thm. 3.10 yield the following corollary (Cor. 4.16). This corollary implies that the computation of a closed term ends *with a result*.

Corollary 4.16. *Let* $\mathcal{N}$ *be a proof net with no* $\mathsf{qbit}^{\perp}$ *in its conclusions. Then the big-step semantics* $[\![\mathcal{N}]\!]$ *is total.* $\square$

4.5 Example

As a concrete example we briefly look at the token machine for the proof net for quantum teleportation (Fig. 6); we shall demonstrate that the qubit $\alpha|0_1\rangle + \beta|1_1\rangle$ ("stored" in the node new_1) is transmitted correctly.

The initial states of our interests are the following four:

$$\big(Q, 1, b_{ij},\ \{\ (\mathsf{qbit}, [\,], \Uparrow, 0)\ \},\ \{\ (\mathsf{qbit}_x{}^{\perp}, [\,], \Downarrow, 0), (\mathsf{qbit}_z{}^{\perp}, [\,], \Downarrow, 0)\ \}\big)\ ,$$

where Q is the quantum register $(\alpha|0_1\rangle + \beta|1_1\rangle) \otimes \left(\frac{1}{\sqrt{2}}|0_2 0_3\rangle + \frac{1}{\sqrt{2}}|1_2 1_3\rangle\right)$ and $i, j \in \{0, 1\}$. Each initial state (with a different slicing b_{ij}) corresponds to possible outcomes of the two measurements. Note that each has the probability 1.

It is straightforward to see that each of the four initial states is led to the final state $\big(\alpha|0\rangle + \beta|1\rangle, 1/4, b_{ij}, \{(\mathsf{qbit}, [\,], \Downarrow, 3)\}, \emptyset\big)$, with the qubit $\alpha|0\rangle + \beta|1\rangle$ assigned to the node new_3. The probabilities ($1/4$ each) add up to 1 with the four initial states together, a fact which witnesses that the original qubit is successfully transmitted with the probability 1.

5 Conclusions and Future Work

We introduced the notion of MLLqm proof net. It is the first one that accommodates measurements as proof structures, and has suitable features for expressing higher-order computation thus going beyond quantum circuits.

The GoI semantics with measurements in this paper is also the first one, which was mentioned in [4] as one of future work. The ideas of using a form of "weakening" to capture measurements (qubits are deleted) and that states of a token machine carry probabilities are new and clean, while the overall structure of the machine follows the usual notion of slice used in linear logic.

As future work, one direction is to accommodate duplicable data, namely the bit type. Although linear logic has a standard tool—the ! modality—to handle such data, there are subtle problems coming from the no-cloning property, nonlocality, etc. Another is to accommodate recursion. We expect to be able to adapt the techniques developed in [14] and [12].

Acknowledgments. Thanks are due to Kentaro Honda, Tristan Roussel, and Alexis Saurin for useful discussions. A.Y. and I.H. are supported by Grants-in-Aid for Young Scientists (A) No. 24680001, and by Aihara Innovative Mathematical Modeling Project, FIRST Program, JSPS/CSTP. C.F. is supported by the ANR project ANR-2010-BLAN-021301 LOGOI.

References

1. 19th IEEE Symposium on Logic in Computer Science (LICS 2004), Turku, Finland, July 14-17. Proceedings. IEEE Computer Society (2004)
2. Abramsky, S., Coecke, B.: A categorical semantics of quantum protocols. In: LICS [1], pp. 415–425
3. Dal Lago, U., Faggian, C.: On multiplicative linear logic, modality and quantum circuits. In: Jacobs, B., Selinger, P., Spitters, B. (eds.) QPL. EPTCS, vol. 95, pp. 55–66 (2011)
4. Dal Lago, U., Zorzi, M.: Wave-style token machines and quantum lambda calculi (2013)
5. Danos, V., Regnier, L.: The structure of multiplicatives. Arch. for Math. Logic 28(3), 181–203 (1989)
6. Girard, J.Y.: Linear logic. Theor. Comput. Sci. 50, 1–102 (1987)
7. Girard, J.Y.: Geometry of interaction 1: Interpretation of system F. Logic Colloquium 88 (1989)
8. Girard, J.Y.: Proof-nets: The parallel syntax for proof-theory. In: Logic and Algebra, pp. 97–124. Marcel Dekker (1996)
9. Gisin, N., Methot, A.A., Scarani, V.: Pseudo-telepathy: input cardinality and Bell-type inequalities. International Journal of Quantum Information 5(4), 525–534 (2007)
10. Green, A.S., Lumsdaine, P.L., Ross, N.J., Selinger, P., Valiron, B.: Quipper: a scalable quantum programming language. In: Boehm, H.J., Flanagan, C. (eds.) PLDI, pp. 333–342. ACM (2013)
11. Hasuo, I., Hoshino, N.: Semantics of higher-order quantum computation via geometry of interaction. In: LICS, pp. 237–246. IEEE Computer Society (2011)
12. Hoshino, N.: A modified GoI interpretation for a linear functional programming language and its adequacy. In: Hofmann, M. (ed.) FOSSACS 2011. LNCS, vol. 6604, pp. 320–334. Springer, Heidelberg (2011)
13. Laurent, O.: A token machine for full geometry of interaction. In: Abramsky, S. (ed.) TLCA 2001. LNCS, vol. 2044, pp. 283–297. Springer, Heidelberg (2001)
14. Mackie, I.: The geometry of interaction machine. In: POPL, pp. 198–208 (1995)
15. Mairson, H.G., Terui, K.: On the computational complexity of cut-elimination in linear logic. In: Blundo, C., Laneve, C. (eds.) ICTCS 2003. LNCS, vol. 2841, pp. 23–36. Springer, Heidelberg (2003)
16. Malherbe, O., Scott, P., Selinger, P.: Presheaf models of quantum computation: An outline. In: Coecke, B., Ong, L., Panangaden, P. (eds.) Computation, Logic, Games and Quantum Foundations. LNCS, vol. 7860, pp. 178–194. Springer, Heidelberg (2013)

17. Melliès, P.A.: Categorical semantics of linear logic. Panoramas et Synthèses, ch. 1, vol. 27, pp. 15–215. Société Mathématique de France (2009)
18. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge Univ. Press (2000)
19. Ömer, B.: Quantum programming in QCL. Master's thesis, Institute of Information Systems, Technical University of Vienna (2000)
20. Pinto, J.S.: Implantation Parallèle avec la Logique Linéaire (Applications des Réseaux d'Interaction et de la Géométrie de l'Interaction). Ph.D. thesis, École Polytechnique, Main text in English (2001)
21. Selinger, P., Valiron, B.: Quantum lambda calculus. In: Gay, S., Mackie, I. (eds.) Semantic Techniques in Quantum Computation, pp. 135–172. Cambridge Univ. Press (2009)
22. Terui, K.: Proof nets and boolean circuits. In: LICS [1], pp. 182–191
23. van Tonder, A.: A lambda calculus for quantum computation. SIAM J. Comput. 33(5), 1109–1135 (2004)
24. Yoshimizu, A., Hasuo, I., Faggian, C., Dal Lago, U.: Measurements in proof nets as higher-order quantum circuits. Extended version with proofs (2014), www-mmm.is.s.u-tokyo.ac.jp/~ayoshimizu