Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 270 (1) (2011) 59–74
www.elsevier.com/locate/entcs

Measurements and Confluence in Quantum Lambda Calculi With Explicit Qubits

Alejandro Díaz-Caro[a,1], Pablo Arrighi[b,2], Manuel Gadella[c,3] and Jonathan Grattage[b,4]

[a] *Departamento de Ciencias de la Computación, Universidad Nacional de Rosario, Argentina*

[b] *Laboratoire d'Informatique de Grenoble, Université de Grenoble, France*

[c] *Departamento de Física Teórica, Atómica y Óptica, Universidad de Valladolid, Spain*

Abstract

This paper demonstrates how to add a measurement operator to quantum λ-calculi. A proof of the consistency of the semantics is given through a proof of confluence presented in a sufficiently general way to allow this technique to be used for other languages. The method described here may be applied to probabilistic rewrite systems in general, and to add measurement to more complex languages such as *QML* [5] or *Lineal* [2][3], which is the subject of further research.

Keywords: Quantum lambda calculus, Measurement, Confluence, Probabilistic rewrite system

1 Introduction

In the quest to develop quantum programming languages, quantum extensions of functional languages provide a promising route, hence the explosion of works on quantum lambda calculi and quantum functional languages [3][5][9][10]. The current language proposals can be split into two categories. In the first category, qubits are manipulated as pointers towards a quantum memory [7][9], thus the syntax does not provide an explicit description of the qubits. It does, however, together with a linear type system, give a convenient and coherent way to handle operations on qubits. A drawback is that the semantics of quantum operations cannot be given intrinsically in the syntax, as this would require the actual state of the quantum memory to be

[1] Email: alejandro.diaz-caro@imag.fr
[2] Email: pablo.arrighi@imag.fr
[3] Email: gadella@fta.uva.es
[4] Email: jonathan.grattage@ens-lyon.fr

doi:10.1016/j.entcs.2011.01.006

known. In the second category of languages [3][5][10] the description of the qubits is part of the programming language, and no type system is required. An advantage here is that the entire semantics can be expressed simply as a rewrite system between terms of the language. This turns into a weakness regarding measurements, because the inherently probabilistic nature of measurement makes it difficult to express as part of a rewrite system. In fact, neither category of languages allow this feature. [3][10]

The case of Altenkirch and Grattage's *QML* [5] is not so clear-cut, but it does illustrate this difficulty. *QML* includes measurements with an operational semantics given in terms of quantum circuits. However, the corresponding algebraic theory [1] stands only for a pure quantum subset of the language, with classical-control and measurement omitted.

Van Tonder's λ_q [10] is a higher-order untyped lambda calculus which includes quantum properties. This calculus carries a history track to keep the necessary information to invert reductions, to ensure that the global computation process is unitary. It is closely related to linear logic, with the syntax being a fragment of the one introduced by Wadler [11], extended with constants to represent quantum entities such as qubits and gates. Linearity concepts are used to distinguish definite terms from arbitrary superposition terms. These syntactic markers constitute the main difference with Arrighi and Dowek's *Lineal* [2][3], which is more permissive. As mentioned previously, measurement is not included in these two proposals.

The work presented here shows how to add measurement to a quantum lambda calculus with explicit qubits in an elegant manner. This is done with full details for the λ_q-calculus, with a proof that confluence, and hence the consistency of the operational semantics, is preserved by this extension. Although this calculus does not need a proof of confluence in the original setting, due to the fixed reduction strategy, this proof is necessary in the presence of measurement. Furthermore, it is non-trivial and has the novelty of showing the confluence in a probabilistic setting with the branching produced by the measurement. The methods illustrated here are general, and applying these techniques to *QML* and *Lineal* is in progress.

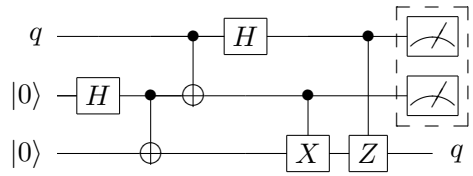
In contrast to measurement in classical mechanics, which gives the value of a given observable with an associated error, measurements in quantum mechanics have an intrinsically probabilistic character. That is, a quantum measurement can give, *a priori*, a certain number of results, each one with some finite probability. Moreover, the state of the system after the measurement is changed in an irreversible manner by the act of measurement. This unintuitive behaviour is of acute importance in quantum information processing.

Measurement is a key property in many quantum information processing tasks, such as quantum cryptography, superdense coding, and in quantum search algorithms. Not having measurements can lead to misinterpretations. Consider as an example the quantum teleportation algorithm with deferred measurement [10] as defined in Fig. 1. Here it is unclear if Alice and Bob can be physically separated, as all the channels used are quantum channels. An obvious question arises: why use

this algorithm if there is a quantum channel between Alice and Bob? Measuring the final state will result in the original logical-qubit having been transferred to Bob. The problem is not one of correctness, but of interpretation.

Secondly, understanding measurement is essential to avoid misinterpreting quantum computation as a whole (e.g. why quantum computation does not lead straightforwardly to an exponential jump in complexity). This work takes the view that in order to understand the possibilities and limitations of quantum computation, measurement needs be formalised in an elegant manner. Note that the projective measurement discussed in this paper is not the only possibility for a quantum measurement, but it is one of the simplest. In addition, any quantum measurement can be reproduced by the action of a unitary mapping and a projective measurement.

$$
\begin{array}{l}
\mathsf{teleport}\ q \to \mathbf{let}\ (e_1, e_2) = \mathsf{epr}\ \mathbf{in} \\
\qquad\qquad \mathbf{let}\ (q', y') = \mathsf{alice}\ (q, e_1)\ \mathbf{in} \\
\qquad\qquad\quad \mathsf{bob}\ (q', y', e_2) \\
\text{where} \\
\mathsf{alice}\ (q, e_1) \to \mathbf{let}\ (q', y') = \mathit{cnot}\ (q, e_1)\ \mathbf{in} \\
\qquad\qquad\qquad ((H\ q'), y') \\
\mathsf{bob}\ (q', y', e_2) \to \mathbf{let}\ (y'', e_2') = cX\ (y', e_2)\ \mathbf{in} \\
\qquad\qquad\qquad\quad \mathbf{let}\ (q'', e_2'') = cZ\ (q', e_2')\ \mathbf{in} \\
\qquad\qquad\qquad\qquad (q'', y'', e_2'') \\
\mathsf{epr} \equiv \mathit{cnot}\ ((H\ 0), 0)
\end{array}
$$



Circuit for the quantum teleportation algorithm with deferred measurement

Fig. 1. Teleportation algorithm in non-extended λ_q.

In the second section of this paper, the process of adding measurement is shown with full details for van Tonder's λ_q. The section concludes with an implementation of the teleportation algorithm in extended λ_q. Section 3 discusses and proves confluence for extended λ_q. Finally, section 4 closes with details of ongoing and future work.

2 Adding measurement

Adding a measurement operator to a quantum lambda calculus can be achieved with only small changes to the grammar. In this section we show how to change the syntax, add well-formedness rules for terms, and give the operational semantics.

2.1 Syntax

To account for measurements, the grammar of λ_q must be extended with a family of measurement operators M_I, which measure the qubits indicated by the set I. In addition, it is necessary to make the syntax for qubits precise, because their "shape" is needed by the measurement operator. This is achieved in a manner following on from *Lineal* [3] and *QML* [5]. Regarding van Tonder's original syntax, the only significant change is to split "constants" into qubit-constants, measurement-constants and gate-constants. The extended syntax is shown in Figure 2 and the added rules of well-formedness are given in Figure 3.

A term is a pre-term produced by the syntax in Figure 2 which follows the rules for well-formedness given by van Tonder [10] plus the rules in Figure 3. Amongst these rules note that M and *Gate* state that M_I and c_U are simply constant symbols. *Zero* and *One* force $|0\rangle$ and $|1\rangle$ respectively to be non-linear terms. *Tensor* and *!Tensor* allow tensorial products between qubits to be written. Although terms like $(c_U\ q) \otimes q$ are not allowed, they are a contraction for $c_{U\otimes I}\ (q \otimes q)$. *Superposition* provides a way of writing qubits in superpositions, and *Simplification* allows subterms with the scalar factor 0 to be removed.

Note that a term with a pattern $!q \otimes q$ is not well-formed, but there is always an equivalent term which can express this in a well-formed way. For example, the term $!\,|0\rangle \otimes (\alpha!\,|0\rangle + \beta!\,|1\rangle)$ is not well-formed, however, it is equivalent to $\alpha(!\,|0\rangle \otimes !\,|0\rangle) + \beta(!\,|0\rangle \otimes !\,|1\rangle)$ which is well-formed.

$t ::=$		*Pre-terms:*
	x	*variable*
	$(\lambda x.t)$	*abstraction*
	$(t\ t)$	*application*
	$!t$	*nonlinear term*
	$(\lambda !x.t)$	*nonlinear abstraction*
	c_U	*gate-constant*
	q	*qubit-constant*
	M_I	*measurement-constant*
$q ::=$		*Qubit-constants:*
	$\|0\rangle \mid \|1\rangle$	*base qubit*
	$(q \otimes q)$	*tensorial product*
	$(q + q)$	*superposition*
	$\alpha(q)$	*scalar product*
$c_U ::=$		*Gate-constants:*
	$H \mid cnot \mid X \mid Z \mid \ldots$	

Fig. 2. Syntax for extended λ_q.

$$\frac{I \subset \mathbb{N}}{\vdash M_I}\ \text{M} \qquad \frac{}{\vdash c_U}\ \text{Gate}$$

$$\frac{}{\vdash ! \left|0\right\rangle}\ \text{Zero} \qquad \frac{}{\vdash ! \left|1\right\rangle}\ \text{One}$$

$$\frac{\Gamma \vdash q_1 \qquad \Delta \vdash q_2}{\Gamma, \Delta \vdash q_1 \otimes q_2}\ \text{Tensor} \qquad \frac{\Gamma \vdash !q_1 \qquad \Delta \vdash !q_2}{\Gamma, \Delta \vdash !q_1 \otimes !q_2}\ \text{!Tensor}$$

$$\frac{\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1 \qquad \alpha_i \in \mathbb{C}, i = 0 \ldots 2^n - 1}{\vdash \alpha_0(! \left|0\right\rangle \otimes \cdots \otimes ! \left|0\right\rangle) + \cdots + \alpha_{2^n-1}(! \left|1\right\rangle \otimes \cdots \otimes ! \left|1\right\rangle)}\ \text{Superposition}$$

$$\frac{\alpha_r = 0, r \in \{0, \ldots, 2^n - 1\} \qquad \Gamma \vdash \sum_{i=0}^{2^n-1} \alpha_i q_i}{\Gamma \vdash \sum_{\substack{i=0 \\ i \neq r}}^{2^n-1} \alpha_i q_i}\ \text{Simplification}$$

Fig. 3. Rules for well-formedness added to λ_q.

Note 1 *The usual* **let** *construction will be used as a useful shorthand, defined as:*

$$\textbf{let } x = t \textbf{ in } u \textit{ gives } (\lambda x.u)\ t$$

$$\textbf{let } !x = t \textbf{ in } u \textit{ gives } (\lambda !x.u)\ t$$

It is interesting to note that a cloning machine such as $\lambda x.(\textbf{let } y = x \textbf{ in } y \otimes y)$ *is syntactic-sugar for* $\lambda x.((\lambda y.y \otimes y)\ x)$*, which is forbidden by the well-formedness rules since* y *is linear (it cannot appear twice), and moreover there is no way to tensor variables: they can only be qubit-constants.*

let *can also be used over lists, as per van Tonder's* (x, y)*, but they are written here as a tensor product. For example, the term* **let** $x \otimes y = M_{\{1,2\}}\ (q_1 \otimes q_2)$ **in** t *is the same as* **let** $(x, y) = (M_{\{1\}}\ q_1, M_{\{1\}}\ q_2)$ **in** t*. Additionally, note that* $x \otimes y$ *is used following van Tonder's* (x, y)*; it is an overloading of the operator* $\otimes$*, denoting both the tensor product between qubits and also list constructors.*

2.2 Operational Semantics

Measurement in quantum systems is an inherently probabilistic operation. Following Di Pierro *et al.* [4], where a probabilistic rewrite system is defined over a λ-calculus, the operational semantics for measurement in extended λ_q is defined as

follows:

$$\text{M: } \frac{q = \sum_{u=0}^{2^m-1} \alpha_u q^{(u)}}{\mathcal{H}; (M_I\ q) \to_{p_w} \sum_{u \in C(w,m,I)} \frac{\alpha_u}{\sqrt{p_w}} q^{(u)}} \quad \forall i \in I, 1 \leq i \leq m$$

where

- $w = 0, \ldots, 2^{|I|} - 1$.
- $q^{(u)} = !q_1^{(u)} \otimes !q_2^{(u)} \otimes \cdots \otimes !q_m^{(u)}$ with $!q_k^{(u)} = ! \left|0\right\rangle$ or $! \left|1\right\rangle$ for $k = 1 \ldots m$.
- $C(w, m, I)$ is the set of binary words of length m such that they coincide with w on the letters of index I.
- $p_w = \sum_{u \in C(w,m,I)} |\alpha_u|^2$.
- The notation $t \to_p t'$ means that t goes to t' with probability p.

It is instructive to look at an example of this rule in action:

Example 2.1 Let $m = 5$, $I = \{2, 3, 5\}$ and $q = \sum_{u=0}^{2^5-1} \alpha_u (\bigotimes_{k=1}^{5} !q_k^{(u)})$ with $!q_k^{(u)} = ! \left|x\right\rangle$ and x is the k^{th} bit in the binary representation of u. According to the previous rule, $(M_I\ q)$ will generate $2^{|I|} = 8$ different outputs (corresponding to the different possible values of the qubits 2, 3 and 5, which are measured). Take as an example the output $w = 2$ (its 3-bit binary representation is 010). Hence, $C(2, 5, I) = \{4, 6, 20, 22\}$ which are the numbers u between 0 and $2^5 - 1$ whose binary representation is of the form $x01y0$ (so they coincide with w, if we compare the bits 2, 3 and 5 of u with the bits 1, 2 and 3 of w). Then, the final term is:

$$\frac{\alpha_4}{\sqrt{p_2}} q^{(4)} + \frac{\alpha_6}{\sqrt{p_2}} q^{(6)} + \frac{\alpha_{20}}{\sqrt{p_2}} q^{(20)} + \frac{\alpha_{22}}{\sqrt{p_2}} q^{(22)}$$

where

$$q^{(4)} = ! \left|0\right\rangle \otimes ! \left|0\right\rangle \otimes ! \left|1\right\rangle \otimes ! \left|0\right\rangle \otimes ! \left|0\right\rangle$$

$$q^{(6)} = ! \left|0\right\rangle \otimes ! \left|0\right\rangle \otimes ! \left|1\right\rangle \otimes ! \left|1\right\rangle \otimes ! \left|0\right\rangle$$

$$q^{(20)} = ! \left|1\right\rangle \otimes ! \left|0\right\rangle \otimes ! \left|1\right\rangle \otimes ! \left|0\right\rangle \otimes ! \left|0\right\rangle$$

$$q^{(22)} = ! \left|1\right\rangle \otimes ! \left|0\right\rangle \otimes ! \left|1\right\rangle \otimes ! \left|1\right\rangle \otimes ! \left|0\right\rangle$$

$$p_2 = \sum_{u \in C(2,5,I)} |\alpha_u|^2 = |\alpha_4|^2 + |\alpha_6|^2 + |\alpha_{20}|^2 + |\alpha_{22}|^2$$

which represents the following quantum state:

$$\frac{1}{\sqrt{p_2}} (\alpha_4 \left|00100\right\rangle + \alpha_6 \left|00110\right\rangle + \alpha_{20} \left|10100\right\rangle + \alpha_{22} \left|10110\right\rangle)$$

2.3 Conditional statements

Measurement as a feature is only useful if the result of the measurement can be used to determine the future evolution of the program. Hence a conditional statement similar to that given in QML is needed. However, in contrast to QML's **if** statements [5], only base-qubits are allowed in the condition. This is all that is required, as the *if* structure is only needed to provide a way to read the output of measurements. Conditional statements are realised by adding the following to the syntax:

$$\textbf{if } t_1 \textbf{ then } t_2 \textbf{ else } t_3$$

and the operational semantic is given by:

$$\frac{}{\textbf{if } !\,|0\rangle \textbf{ then } t_1 \textbf{ else } t_2 \to_1 t_1} \text{ IF-}|0\rangle$$

$$\frac{}{\textbf{if } !\,|1\rangle \textbf{ then } t_1 \textbf{ else } t_2 \to_1 t_2} \text{ IF-}|1\rangle$$

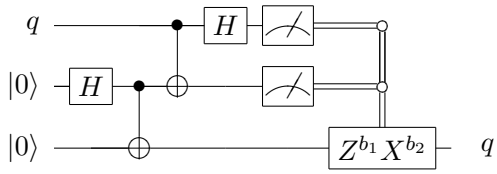
Note that as the condition may be not be a base-qubit, it is not guaranteed that the whole term will reduce.

This addition is required, as without such an *if* statement such as this being added to the language, this extension to measurements would have been equivalent to a simple extension from unitary constants to quantum operation constants.

2.4 Example: Teleportation algorithm

With the rules developed so far, the teleportation algorithm can be rewritten as shown in Fig. 4.

$\mathsf{teleport}\ q \to_1 \textbf{ let } x \otimes y = \mathsf{epr} \textbf{ in}$
$\qquad \textbf{let } b_1 \otimes b_2 = M_{\{1,2\}}\ \mathsf{alice}\ q\ x \textbf{ in}$
$\qquad\quad \mathsf{bob}\ b_1\ b_2\ y$
where
$\mathsf{alice}\ q\ x \to_1 \textbf{ let } r \otimes w = \mathit{cnot}\ q \otimes x \textbf{ in}$
$\qquad ((H\ r) \otimes w)$
$\mathsf{bob}\ b_1\ b_2\ y \to_1 \mathsf{zed}\ b_1\ (\mathsf{ex}\ b_2\ y)$
$\mathsf{ex}\ b\ x \to_1 \textbf{ if } b \textbf{ then } (X\ y) \textbf{ else } y$
$\mathsf{zed}\ b\ x \to_1 \textbf{ if } b \textbf{ then } (Z\ x) \textbf{ else } x$
$\mathsf{epr} \equiv \mathit{cnot}\ ((H\ !\,|0\rangle) \otimes !\,|0\rangle)$



Circuit for the original quantum teleportation algorithm

Fig. 4. Teleportation algorithm in extended λ_q

3 Confluence

When defining a language, a grammar must also be provided (how to construct terms), and a semantics (how these terms compute). The semantics can be denotational (terms are mapped to elements of a semantic domain, each corresponding to

what is computed by the term) or operational (terms are mapped into other terms, with each transition corresponding to a computational step). Clearly it must be proved that the semantics provided is unambiguous and consistent. For example, the semantics will usually induce an equational theory upon terms (via equality in the semantics domain or by equating two terms if one reduces to the other), and it is important that this theory should not equate all terms.

In λ_q a consistent equational theory is given. However, adding measurement does not correspond to a simple system for equational reasoning. It is not possible to proceed by replacing terms by equal terms according to any equational theory, since measurement is a probabilistic operation, and each reduction instance could produce different terms that are impossible to reconcile in the system. In the presence of an operational semantics, a usual method of proving the consistency result is to provide a proof of confluence. This property states that the order in which the transition rules are applied does not matter to the end result, thus removing any ambiguity. In this section it is shown how such a study of confluence can still be carried through, even in the presence of probabilities. As λ_q provides a fixed reduction strategy, proving confluence in the original language is trivial, because there is only one possible reduction at each step. However, this is not the case in the presence of measurement, where proving confluence is non-trivial.

3.1 Definitions and lemmas

Whilst the above-mentioned probabilistic reductions are an elegant and concise way to present the operational semantics, the study of confluence is not immediate in this setting. For confluence, it is necessary to prove that if any term t can reduce to u and to v, then there exists a w such that $u \to w \wedge v \to w$. However, in a probabilistic calculus it could be that $t \to_p u$ and $t \to_q v$, where p and q represent the probability of the respective reduction occurring, and there is no w that both u and v could reduce to. For example, given $M_{\{1\}}$, a measurement operator in the computational basis, it follows that $M_{\{1\}}\ (\alpha\,|0\rangle + \beta\,|1\rangle) \to_{|\alpha|^2} |0\rangle$ and $M_{\{1\}}\ (\alpha\,|0\rangle + \beta\,|1\rangle) \to_{|\beta|^2} |1\rangle$. However, there is no w such that $|0\rangle \to_p w$ and $|1\rangle \to_q w$.

A naïve way to deal with this would be to assume that if there is some normal form that can be reached with a certain probability, then by following any path it must to be possible to reach the same normal form with the same probability. However, this definition is not rigorous, and not applicable to terms without a normal form. Hence, it does not allow the development of a formal proof of confluence.

Probabilistic transitions need to be abstracted out in order to allow only one possible normal form for each term, and to deal with terms without normal form. With this aim, the following definition gives a notion of confluence for probabilistic calculi:

Definition 3.1 A term ensemble $\{\langle t_i, \alpha_i\rangle\}$ is defined as a collection of terms t_i, each with an associated probability α_i, such that $\sum_i \alpha_i = 1$.

Note that given a term t, it may be considered as a term ensemble $\{\langle t, 1\rangle\}$.

Example 3.2 Consider the term ensemble $\{\langle t_1, \frac{1}{2}\rangle, \langle t_1, \frac{1}{4}\rangle, \langle t_2, \frac{1}{4}\rangle\}$, where the term t_1 appears twice. By summing the probabilities of any equivalent terms, this ensemble can be identified with the more compact ensemble $\{\langle t_1, \frac{3}{4}\rangle, \langle t_2, \frac{1}{4}\rangle\}$.

Remark 3.3 Throughout this paper the symbol $=$ will be used for both α-equivalences and equalities. When referring to a set, *i.e.* where each element appears once, it is considered to be modulo α-equivalence.

The appropriate steps such that $\{\langle t, \alpha\rangle, \langle t, \gamma\rangle\}$ is identified with $\{\langle t, \alpha+\gamma\rangle\}$ need to be taken. Definition 3.4 formalises this equivalence:

Definition 3.4 Let first be a function that takes a term ensemble and returns a set defined by

$$\mathsf{first}(\{\langle t_i, \alpha_i\rangle\}) = \{t_i\}$$

As the co-domain is a set, it allows only one instance of each element.

Let $\mathsf{sumprob}$ be a function that takes a term and a term ensemble and returns the sum of the probabilities associated to each instance of the term in the ensemble:

$$\mathsf{sumprob}(s, \{\langle t_i, \alpha_i\rangle\}) = \sum_{j\in\{i|t_i=s\}} \alpha_j$$

Finally, let min be a function that takes a term ensemble and returns a term ensemble defined by

$$\mathsf{min}(\tau) = \bigcup_{t\,\in\,\mathsf{first}\ \tau} \{\langle t, \mathsf{sumprob}(t,\tau)\rangle\}$$

A term ensemble ω_1 is thus said to be *equivalent* to a term ensemble ω_2, $\omega_1 \equiv \omega_2$, *iff* $\mathsf{min}(\omega_1) = \mathsf{min}(\omega_2)$.

Note that the definition of min is correct, as $\sum_{t\,\in\,\mathsf{first}\ \tau} \mathsf{sumprob}(t,\tau)$ trivially sums to 1.

A deterministic transition rule between term ensembles can also be defined:

Definition 3.5 If X is a probabilistic rewrite system over terms, let $Det(X)$ be the deterministic rewrite system over term ensembles written $\overset{X}{\twoheadrightarrow}$ and defined as

$$\{\langle t_i, \alpha_i\rangle\} \overset{X}{\twoheadrightarrow} \{\langle t'_{i_j}, \alpha_i\gamma_{i_j}\rangle\} \text{ \textit{iff}, for each } i,\ t_i \overset{X}{\rightarrow}_{\gamma_{i_j}} t'_{i_j} \wedge \sum_j \gamma_{i_j} = 1.$$

where all the reductions between single terms are produced by following any rule in X, or none.

Lemma 3.6 *Given a probabilistic rewrite system P, then $Det(P)$ preserves ensembles.*

Proof. Let $\{\langle t_i, \alpha_i\rangle\}$ and $\{\langle t'_{i_j}, \alpha_i\gamma_{i_j}\rangle\}$ be term ensembles such that $\{\langle t_i, \alpha_i\rangle\} \overset{P}{\twoheadrightarrow} \{\langle t'_{i_j}, \alpha_i\gamma_{i_j}\rangle\}$. Then, by definition 3.5, $\forall i \ \sum_j \gamma_{i_j} = 1$.

Hence, $\sum_{i,j} \alpha_i\gamma_{i_j} = \sum_i \alpha_i \sum_j \gamma_{i_j} = \sum_i \alpha_i = 1$. □

Using these concepts, (strong) confluence for a probabilistic rewrite system can be expressed as show in definition 3.7.

Definition 3.7 Let R be a probabilistic rewrite system. R is said to be confluent if, for each term ensemble τ such that $\tau \overset{R}{\twoheadrightarrow}_* \mu \wedge \tau \overset{R}{\twoheadrightarrow}_* \nu$, there exist equivalent term ensembles ω_1 and ω_2 such that $\mu \overset{R}{\twoheadrightarrow}_* \omega_1 \wedge \nu \overset{R}{\twoheadrightarrow}_* \omega_2$. R is said to be *strongly* confluent if, for each term ensemble τ, such that $\tau \overset{R}{\twoheadrightarrow} \mu \wedge \tau \overset{R}{\twoheadrightarrow} \nu$, there exist equivalent term ensembles ω_1 and ω_2 such that $\mu \overset{R}{\twoheadrightarrow} \omega_1 \wedge \nu \overset{R}{\twoheadrightarrow} \omega_2$.

Note that strong confluence of R implies the confluence of R, and also that the confluence of R implies the strong confluence of R^*. It is possible to extend the Hindley-Rosen lemma [6][8] to these notions of confluence, as follows:

Proposition 3.8 *Let R and U be strongly confluent probabilistic rewrite systems. If R and U strongly commute, that is if for each term ensemble τ such that $\tau \overset{R}{\twoheadrightarrow} \mu \wedge \tau \overset{U}{\twoheadrightarrow} \nu$, there exist equivalent term ensembles ω_1 and ω_2 such that $\mu \overset{U}{\twoheadrightarrow} \omega_1 \wedge \nu \overset{R}{\twoheadrightarrow} \omega_2$, therefore $R \cup U$ is strongly confluent.*

Theorem 3.9 allows the remaining proofs to be simplified, by showing that it is enough to prove strong confluence (commutation) for a single-term term ensemble.

Theorem 3.9 *Let S and T be probabilistic rewrite systems such that:*

$$\forall t \quad \left.\begin{array}{l} \{\langle t, 1\rangle\} \overset{S}{\twoheadrightarrow} \mu_1 \\ \{\langle t, 1\rangle\} \overset{T}{\twoheadrightarrow} \nu_1 \end{array}\right\} \Rightarrow \exists\ \omega_1 \equiv \omega_2 \ s.t. \ \left\{\begin{array}{l} \mu_1 \overset{T}{\twoheadrightarrow} \omega_1 \\ \nu_1 \overset{S}{\twoheadrightarrow} \omega_2 \end{array}\right.$$

Then $\forall \tau$, μ and ν such that $\tau \overset{S}{\twoheadrightarrow} \mu$ and $\tau \overset{T}{\twoheadrightarrow} \nu$, there exist equivalent ω_1 and ω_2 such that $\mu \overset{T}{\twoheadrightarrow} \omega_1$ and $\nu \overset{S}{\twoheadrightarrow} \omega_2$.

Proof. Let $\tau = \{\langle t_i, \alpha_i\rangle\}$, $\mu = \{\langle u_{i_j}, \alpha_i\delta_{i_j}\rangle\}$ and $\nu = \{\langle v_{i_k}, \alpha_i\varphi_{i_k}\rangle\}$ such that $\tau \overset{S}{\twoheadrightarrow} \mu$ and $\tau \overset{T}{\twoheadrightarrow} \nu$, *i.e.* for each i:

$$t_i \overset{S}{\rightarrow}_{\delta_{i_j}} u_{i_j} \ \wedge \ \sum_j \delta_{i_j} = 1 \text{ and } t_i \overset{T}{\rightarrow}_{\varphi_{i_k}} v_{i_k} \ \wedge \ \sum_k \varphi_{i_k} = 1 \tag{1}$$

Consider the single term term-ensembles $\tau_i = \{\langle t_i, 1\rangle\}$, and the term ensembles $\mu_i = \{\langle u_{i_j}, \delta_{i_j}\rangle\}$ and $\nu = \{\langle v_{i_k}, \varphi_{i_k}\rangle\}$. By equation (1), for each i, $\tau_i \overset{S}{\twoheadrightarrow} \mu_i$ and

$\tau_i \overset{T}{\twoheadrightarrow} \nu_i$. By our hypothesis, for each i there exist equivalent term ensembles $\omega_1^i = \{\langle w_{1j_l}^i, \delta_{i_j}\sigma_{j_l}^i\rangle\}$ and $\omega_2^i = \{\langle w_{2k_s}^i, \varphi_{i_k}\gamma_{k_s}^i\rangle\}$ such that $\mu_i \overset{T}{\twoheadrightarrow} \omega_1^i$ and $\nu_i \overset{S}{\twoheadrightarrow} \omega_2^i$.

By taking $\omega_1 = \{\langle w_{1j_l}^i, \alpha_i\delta_{i_j}\sigma_{j_l}^i\rangle\}$ and $\omega_2 = \{\langle w_{2k_s}^i, \alpha_i\varphi_{i_k}\gamma_{k_s}^i\rangle\}$, it follows that $\mu \overset{T}{\twoheadrightarrow} \omega_1$ and $\nu \overset{S}{\twoheadrightarrow} \omega_2$. As $\forall i\ \omega_1^i \equiv \omega_2^i$, it is trivially the case that $\omega_1 \equiv \omega_2$. □

Lemma 3.10 guarantees that equivalence between term ensembles is a congruence by adding identical context to each term in both of the ensembles:

Lemma 3.10 *Given two equivalent term ensembles $\omega_1 = \{\langle t_i, \alpha_i\rangle\}$ and $\omega_2 = \{\langle s_j, \gamma_j\rangle\}$ and any context C, the term ensembles $\tau_1 = \{\langle C[t_i/x], \alpha_i\rangle\}$ and $\tau_2 = \{\langle C[s_j/x], \gamma_j\rangle\}$ are also equivalent.*

Proof. $\omega_1 \equiv \omega_2 \Rightarrow \mathsf{min}(\omega_1) = \mathsf{min}(\omega_2)$, defined as equal to $\{\langle w_k, \delta_k\rangle\}$, then

$$\begin{aligned}
\mathsf{min}(\tau_1) &= \bigcup_{t \in \mathsf{first}(\tau_1)} \{\langle t, \mathsf{sumprob}(t, \tau_1)\rangle\} \\
&= \mathsf{min}\left(\bigcup_{t \in \mathsf{first}(\omega_1)} \{\langle C[t/x], \mathsf{sumprob}(t, \omega_1)\rangle\}\right) \\
&= \mathsf{min}\left(\{\langle C[w_k/x], \delta_k\rangle\}\right) \\
&= \mathsf{min}\left(\bigcup_{t \in \mathsf{first}(\omega_2)} \{\langle C[t/x], \mathsf{sumprob}(t, \omega_2)\rangle\}\right) \\
&= \bigcup_{t \in \mathsf{first}(\tau_2)} \{\langle t, \mathsf{sumprob}(t, \tau_2)\rangle\} \\
&= \mathsf{min}(\tau_2)
\end{aligned}$$

and hence $\tau_1 \equiv \tau_2$. □

3.2 Strong confluence for $\{(M), IF\text{-}|0\rangle, IF\text{-}|1\rangle\}$

The strong confluence of the added rules is formally expressed and proved by theorem 3.11.

Theorem 3.11 *The probabilistic reduction rules system $T = \{(M), IF\text{-}|0\rangle, IF\text{-}|1\rangle\}$ is strongly confluent.*

Proof. Given term ensembles $\tau = \{\langle t, 1\rangle\}$, μ and ν, where $\mu \neq \nu$, and such that $\tau \overset{T}{\twoheadrightarrow} \mu$ and $\tau \overset{T}{\twoheadrightarrow} \nu$, then by proving there exist equivalent term ensembles ω_1 and ω_2 such that $\mu \overset{T}{\twoheadrightarrow} \omega_1$ and $\nu \overset{T}{\twoheadrightarrow} \omega_2$, theorem 3.9 shows that this system is strongly confluent.

This result is proved here using structural induction over t.

(i) $t = x \mid c_U \mid q \mid M_I \mid !t' \ \Rightarrow\ \nexists\ \mu \neq \nu$. Note that there is no rule in T that can reduce t in this case, and hence only Id is applicable, producing $\mu = \tau$. Therefore there cannot exist any $\nu \neq \mu$.

(ii) $\nu = \tau$. Hence $\omega_1 = \omega_2 = \mu$.

(iii) $t = \lambda x.t'$.

Let $\mu = \{\langle \lambda x.u_i, \alpha_i \rangle\}$ where $(t' \xrightarrow{T}_{\alpha_i} u_i)$, with $\sum_i \alpha_i = 1$, and let $\nu = \{\langle \lambda x.v_j, \gamma_j \rangle\}$ where $(t' \xrightarrow{T}_{\gamma_j} v_j)$ with $\sum_j \gamma_j = 1$.

By induction, there exist equivalent term ensembles $\omega_1' = \{\langle w^1_{i_k}, \alpha_i\beta_{i_k} \rangle\}$ and $\omega_2' = \{\langle w^2_{j_h}, \gamma_j\sigma_{j_h} \rangle\}$ such that $u_i \xrightarrow{T}_{\beta_{i_k}} w^1_{i_k}$ and $v_j \xrightarrow{T}_{\sigma_{j_h}} w^2_{j_h}$.
Hence $\omega_1 = \{\langle \lambda x.w^1_{i_k}, \alpha_i\beta_{i_k} \rangle\}$ and $\omega_2 = \{\langle \lambda x.w^2_{j_h}, \gamma_j\sigma_{j_h} \rangle\}$ can be taken, which are equivalent by lemma 3.10.

(iv) $t = \lambda !x.t'$, analogous to case (iii).

(v) $t = (t_1\ t_2)$. Consider the following cases:

(a) Let $\mu = \{\langle (t_1\ u_i), \alpha_i \rangle\}$ where $t_2 \xrightarrow{T}_{\alpha_i} u_i$, with $\sum_i \alpha_i = 1$, and

let $\nu = \{\langle (t_1\ v_j), \gamma_j \rangle\}$ where $t_2 \xrightarrow{T}_{\gamma_j} v_j$, with $\sum_j \gamma_j = 1$.

This case is analogous to case (iii), as lemma 3.10 is applicable.

(b) Let $\mu = \{\langle (u_i\ t_2), \alpha_i \rangle\}$ and $\nu = \{\langle (v_j\ t_2), \gamma_j \rangle\}$. This follows case (a).

(c) Let $\mu = \{\langle (u_i\ t_2), \alpha_i \rangle\}$ and $\nu = \{\langle (t_1\ v_j), \gamma_j \rangle\}$, then take $\omega_1 = \omega_2 = \{\langle (u_i\ v_j), \alpha_i\gamma_j \rangle\}$

(d) Let $t = (M_I\ q)$, $\mu = \{\langle q_i, \alpha_i \rangle\}$ where $(M_I\ q) \xrightarrow{T}_{\alpha_j} q_j$, with $\sum_i \alpha_i = 1$. This follows case (ii). $\omega_1 = \omega_2 = \mu$.

(vi) $t =$ **if** t_1 **then** t_2 **else** t_3. Consider the following cases:

(a) Let $\mu = \{\langle$ **if** t_1 **then** t_2 **else** $u_i, \alpha_i \rangle\}$ where $t_3 \xrightarrow{T}_{\alpha_i} u_i$, with $\sum_i \alpha_i = 1$ and

let $\nu = \{\langle$ **if** t_1 **then** v_j **else** $t_3, \gamma_j \rangle\}$ where $t_2 \xrightarrow{T}_{\gamma_j} v_j$, with $\sum_j \gamma_j = 1$.

This is analogous to (v.c). In fact, any combination that implies that μ and ν are obtained by the reduction of t_1, t_2 or t_3, is analogous to one of the subcases of case (v).

(b) Let $\mu = \{\langle t_2, 1 \rangle\}$, and let $\nu = \{\langle$ **if** t_1 **then** t_2 **else** $v_j, \gamma_j \rangle\}$ where $t_3 \xrightarrow{T}_{\gamma_j} v_j$, with $\sum_j \gamma_j = 1$. Then take $\omega_1 = \omega_2 = \mu$. (Analogous if $t_2 \xrightarrow{T}_{\gamma_j} v_j$ and $\mu = \{\langle t_3, 1 \rangle\}$).

(c) Let $\mu = \{\langle t_2, 1 \rangle\}$, and let $\nu = \{\langle$ **if** t_1 **then** v_j **else** $t_3, \gamma_j \rangle\}$ where $t_2 \xrightarrow{T}_{\gamma_j} v_j$, with $\sum_j \gamma_j = 1$. Then take $\omega_1 = \omega_2 = \{\langle v_j, \gamma_j \rangle\}$. (Analogous for t_3).

□

3.3 Preserving confluence

Before formalising the confluence for the whole calculus, some key examples are presented:

- *Cloning* arguments: $(\lambda x.(x\ x))\ (M_{\{1\}}\ (\frac{1}{\sqrt{2}}!\,|0\rangle + \frac{1}{\sqrt{2}}!\,|1\rangle))$

The problem here is that if copying a measurement is allowed, this may give different results for each measurement. However, by measuring first and then applying the abstraction, both measurements are the same. In λ_q, these kinds of terms are disallowed by the well-formedness rules [10]; a linear argument can appear only once in the body of a function.

- *Copying* arguments: $(\lambda!x.(x\ x))\ (M_{\{1\}}\ (\frac{1}{\sqrt{2}}!\,|0\rangle + \frac{1}{\sqrt{2}}!\,|1\rangle))$
  When the argument is linear, there is no rule in the operational semantics of λ_q that allows the application of a non-linear abstraction to a linear term. Hence, $M_{\{1\}}$ must apply first, producing a non-linear output (either $!\,|0\rangle$ or $!\,|1\rangle$).
- *Promoting* arguments: $(\lambda!x.(x\ x))\ !(M_{\{1\}}\ (\frac{1}{\sqrt{2}}!\,|0\rangle + \frac{1}{\sqrt{2}}!\,|1\rangle))$
  In this case copying the measurement operation twice is allowed, and this is the only applicable reduction strategy because $!t$ terms are values in λ_q.

In light of the above statements, a formal proof of confluence for the entire system is required.

Lemma 3.12 ensures that, under some hypotheses, measurement is independent of context:

Lemma 3.12 *Let x be a variable and let t be a linear term with only one linear instance of x. If $m \overset{(M)}{\to}_p v$, then $t[m/x] \overset{(M)}{\to}_p t[v/x]$.*

Proof. Structural induction over t

(i) Let t be such that $x \notin \mathcal{V}(t) \Rightarrow\ t[m/x] = t = t[v/x]$.

(ii) Let $t = x$. $x[m/x] = m \overset{(M)}{\to}_p v = x[v/x]$.

(iii) Let $t = \lambda y.t'$. By induction $\lambda y.t'[m/x] \overset{(M)}{\to}_p \lambda y.t'[v/x]$.

(iv) Let $t = \lambda!y.t'$. Analogous to case (iii).

(v) Let $t = (t_1\ t_2)$, with $x \in \mathcal{V}(t_1)$. Then $(t_1\ t_2)[m/x] = (t_1[m/x]\ t_2)$ and by induction, $(t_1[m/x]\ t_2) \overset{(M)}{\to}_p (t_1[v/x]\ t_2)$, which is equal to $(t_1\ t_2)[v/x]$.

(vi) Let $t = (t_1\ t_2)$, with $x \in \mathcal{V}(t_2)$. Analogous to case (v).

(vii) Let $t =$ **if** t_1 **then** t_2 **else** t_3. Analogous to case (v).

□

Next, it is proved that the original reduction rules system from λ_q and the new rules for measurements strongly commute. This is suggestive of the confluence of the whole system

Theorem 3.13 *The probabilistic reduction rules systems $S = \{(APP_1), (APP_2), (\beta), (!\beta_1), (!\beta_2), (U)\}$ and $T = \{(M), IF\text{-}|0\rangle, IF\text{-}|1\rangle\}$ strongly commute.*

Proof. If it is proved that given term ensembles $\tau = \{\langle t, 1\rangle\}$, μ and ν, $\mu \neq \nu$, such that $\tau \overset{S}{\twoheadrightarrow} \mu$ and $\tau \overset{T}{\twoheadrightarrow} \nu$, then this implies that there exist equivalent term ensembles ω_1 and ω_2 such that $\mu \overset{T}{\twoheadrightarrow} \omega_1$ and $\nu \overset{S}{\twoheadrightarrow} \omega_2$, then S and T verify the hypotheses for

theorem 3.9, which proves strong commutation between them.

This result is proved here using structural induction over t.

(i) $t = x \mid c_U \mid q \mid M_I \mid !t \;\Rightarrow\; \nexists\, \mu \neq \nu$. Note that there is no rule in T nor S that can reduce t in this case, hence only Id is applicable, producing $\mu = \tau$. Therefore there cannot exists any $\nu \neq \mu$.

(ii) $\nu = \tau$. Hence $\omega_1 = \omega_2 = \mu$. (Analogous for $\mu = \tau$).

(iii) $t = \lambda x.t'$, $\mu = \{\langle \lambda x.u, 1\rangle\}$ and $\nu = \{\langle \lambda x.v_j, \gamma_j\rangle\}$ such that $\tau \overset{S}{\twoheadrightarrow} \mu$ and $\tau \overset{T}{\twoheadrightarrow} \nu$. By induction, there exist equivalent $\omega_1' = \{\langle w_s^1, \delta_s\rangle\}$ and $\omega_2' = \{\langle w_j^2, \sigma_j\rangle\}$ such that $\{\langle u, 1\rangle\} \overset{T}{\twoheadrightarrow} \omega_1'$ and $\{\langle v_j, \gamma_j\rangle\} \overset{S}{\twoheadrightarrow} \omega_2'$. Then take $\omega_1 = \{\langle \lambda x.w_s^1, \delta_s\rangle\}$ and $\omega_2 = \{\langle \lambda x.w_j^2, \sigma_j\rangle\}$ which are equivalent by lemma 3.10.

(iv) $t = \lambda !x.t'$. Analogous to case (iii).

(v) $t = (t_1\ t_2)$. Consider the following cases:

- (a) $\mu = \{\langle (t_1\ u), 1\rangle\}$ and $\nu = \{\langle (t_1\ v_j), \gamma_j\rangle\}$. Analogous to case (iii); note that lemma 3.10 also holds in this case.
- (b) $\mu = \{\langle (u\ t_2), 1\rangle\}$ and $\nu = \{\langle (v_j\ t_2), \gamma_j\rangle\}$. Analogous to subcase (a).
- (c) $\mu = \{\langle (u\ t_2), 1\rangle\}$ and $\nu = \{\langle (t_1\ v_j), \gamma_j\rangle\}$. Take $\omega_1 = \omega_2 = \{\langle (u\, v_j), \gamma_j\rangle\}$ (Similarly if $t_2 \to_1 u$ and $t_1 \to_{\gamma_j} v_j$).
- (d) $t = (c_U\ q)$ and $\mu = \{\langle q', 1\rangle\}$, This follows case (ii). Note that if instead of $t_2 = q$ an expression like $t_2 = (M_I\ q)$ is given, it is subcase (b) which applies, where $u = t_1 = c_U$.
- (e) $t = (M_I\ q)$ and $\mu = \tau$. This follows the analogous to case (ii).
- (f) $t_1 = \lambda x.t'$, $\mu = \{\langle t'[t_2/x], 1\rangle\}$, $\nu = \{\langle (\lambda x.t'\ v_j), \gamma_j\rangle\}$. By lemma 3.12, $\omega_1 = \omega_2 = \{\langle t'[v_j/x], \gamma_j\rangle\}$ can be taken. Note that if $t_1 = \lambda !x.t'$, with the same μ, then t_2 must be non-linear due to the well-formedness rules and hence in this situation it is the subcase (d).
- (g) $t_1 =$ **if** t_1' **then** t_2' **else** t_3'. Then μ has to be obtained by the reduction of t_1', t_2', t_3' or t_2, hence, it is analogous to previous cases. Note that if, for instance, $\nu = \{\langle (t_2'\ t_2), 1\rangle\}$ and suppose that μ is obtained by the reduction of t_3' (it cannot be the application of the **if** statement to t_2 because there is not any rule that performs such a reduction) then $\omega_1 = \omega_2 = \nu$.

(vi) $t =$ **if** t_1 **then** t_2 **else** t_3. Consider the following cases:

- (a) Let $\mu = \{\langle$ **if** t_1 **then** t_2 **else** $u, 1\rangle\}$ where $t_3 \overset{S}{\to}_1 u$ and let $\nu = \{\langle$ **if** t_1 **then** v_j **else** $t_3, \gamma_j\rangle\}$ where $t_2 \overset{T}{\to}_{\gamma_j} v_j$ and $\sum_j \gamma_j = 1$. Analogous to (v.c). In fact, any combinations that implies that μ and ν are obtained by reduction of t_1, t_2, or t_3, is analogous to one of the subcases of case (v).
- (b) Let $\mu = \{\langle$ **if** t_1 **then** t_2 **else** $u, 1\rangle\}$ where $t_3 \overset{S}{\to}_1 u$ and $\nu = \{\langle t_2, 1\rangle\}$, then take $\omega_1 = \omega_2 = \nu$. Analogous if $t_2 \overset{S}{\to}_1 u$ and $\nu = \{\langle t_3, 1\rangle\}$.
- (c) Let $\mu = \{\langle$ **if** t_1 **then** u **else** $t_3, 1\rangle\}$ where $t_2 \overset{S}{\to}_1 u$ and $\nu = \{\langle t_2, 1\rangle\}$, then take $\omega_1 = \omega_2 = \{\langle u, 1\rangle\}$. Similarly if $t_3 \overset{S}{\to}_1 u$ and $\nu = \{\langle t_3, 1\rangle\}$.

□

It has been shown that T and S strongly commute, and hence T^* and S^* strongly commute. Moreover, T is confluent, and hence T^* is strongly confluent.

Now, supposing S is confluent, it follows that S^* is strongly confluent. Proposition 3.8 entails that $S^* \cup T^*$ is strongly confluent, and therefore that $S \cup T$ is confluent. Therefore, the extension of van Tonder's calculus presented here preserves confluence.

4 Conclusions

This paper extends the quantum lambda calculus λ_q, defined by van Tonder, with a family of measurement operations M_I, which measure the qubits indicated by the set I, and an *if* structure which allows reading of the output of these measurements. By defining the notion of ensembles of terms, and extending the rewrite system to a deterministic system between term ensembles, a proof of confluence for this extended calculus is presented. The extended calculus is therefore confluent, and retains the simplicity of van Tonder's original calculus.

The proof of confluence follows a method which can be applied to other calculi that make use of probabilistic transition rules. For example, this method could be applied to both *Lineal* and to *QML*, and this is the subject of ongoing research.

The addition of a measurement operation to λ_q, which preserves confluence, is a significant development. This allows a more natural expression of quantum algorithms that intrinsically make use of measurement, such as quantum teleportation, superdense coding, and quantum search algorithms. Moreover, having an operational semantic for measurements gives a way for understanding the behaviour of this quantum procedure, and this is a possible topic for future work.

Acknowledgement

A. Díaz-Caro would like to thank Pablo E. Martínez López for useful comments and helpful suggestions on an early draft of this paper, and the CAPP (QCG) group at the Laboratoire d'Informatique de Grenoble for their hospitality. The authors would also like to thank Simon Perdrix for fruitful discussions.

References

[1] Altenkirch, T., J. J. Grattage, J. K. Vizzotto and A. Sabry, *An algebra of pure quantum programming*, Electronic Notes in Theoretical Computer Science **170** (2007), pp. 23–47.

[2] Arrighi, P. and G. Dowek, *A computational definition of the notion of vectorial space*, Electronic Notes in Theoretical Computer Science **117** (2005), pp. 249–261.

[3] Arrighi, P. and G. Dowek, *Linear-algebraic lambda-calculus: higher-order, encodings and confluence*, in: B. Buchberger, editor, *Term Rewriting and Applications, 19th International Conference, RTA-08*, To appear in LNCS (2008), eprint available at `arXiv:quant-ph/0612199`.

[4] Di Pierro, A., C. Hankin and H. Wiklicky, *Probabilistic λ-calculus and quantitative program analysis*, Journal of Logic and Computation **15** (2005), pp. 159–179.

[5] Grattage, J. and T. Altenkirch, *A functional quantum programming language*, in: *LICS '05: Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science* (2005), pp. 249–258.

[6] Hindley, J. R., "The Church-Rosser property and a result in combinatory logic," Ph.D. thesis, University of Newcastle-upon-Tyne (1964).

[7] Prost, F., *Taming non-compositionality using new binders*, in: S. G. Akl, C. S. Calude, M. J. Dinneen, G. Rozenberg and T. Wareham, editors, *UC*, Lecture Notes in Computer Science **4618** (2007), pp. 150–162.

[8] Rosen, B. K., *Tree-manipulating systems and Church-Rosser theorems*, Journal of the ACM **20** (1973), pp. 160–187.

[9] Selinger, P. and B. Valiron, *A lambda calculus for quantum computation with classical control*, Mathematical Structures in Computer Science **16** (2006), pp. 527–552.

[10] van Tonder, A., *A lambda calculus for quantum computation*, SIAM Journal on Computing **33** (2004), pp. 1109–1135.

[11] Wadler, P., *A syntax for linear logic*, in: S. D. Brookes, M. G. Main, A. Melton, M. W. Mislove and D. A. Schmidt, editors, *Proceedings of the 9th International Conference on Mathematical Foundations of Programming Semantics* (1994), pp. 513–529.