Theoretical Computer Science 411 (2010) 377–409

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Quantum implicit computational complexity☆

Ugo Dal Lago [a,*], Andrea Masini [b], Margherita Zorzi [b]

[a] *Dipartimento di Scienze dell'Informazione, Università di Bologna, Italy*

[b] *Dipartimento di Informatica, Università di Verona, Italy*

ARTICLE INFO

Article history:
Received 7 October 2008
Received in revised form 5 June 2009
Accepted 15 July 2009
Communicated by P.-L. Curien

Keywords:
Quantum computation
Implicit computational complexity
Lambda calculus

ABSTRACT

We introduce a quantum lambda calculus inspired by Lafont's Soft Linear Logic and capturing the polynomial quantum complexity classes EQP, BQP and ZQP. The calculus is based on the "classical control and quantum data" paradigm. This is the first example of a formal system capturing quantum complexity classes in the spirit of implicit computational complexity — it is machine-free and no explicit bound (e.g., polynomials) appears in its syntax.

1. Introduction

This paper is about quantum computation and implicit computational complexity. More precisely, a lambda calculus is defined and proved to capture some (polynomial time) quantum complexity classes. The language under consideration is not built up from any notion of polynomials or from any concrete machine. To the authors' knowledge, this is the first example of an implicit characterization of some classes of problems coming from quantum complexity theory. A brief introduction to these two research areas is now in order.

Quantum computation. Quantum Computation (QC) [7,10–12,18,14,23] is nowadays one of the most promising computation paradigms between those going beyond classical computation (e.g. biological computation, nanocomputing, etc.). An extraordinary research effort is being put on the task of proving quantum computation to be both feasible in practice and worthwhile from a theoretical point of view. Since its very birth [13,10], in particular, one of the main motivations for studying computational applications of quantum mechanics is the potentiality of exploiting parallelism to reduce the computational complexity of some (classically) hard problems. Indeed, some of these hopes have materialized. For example, factoring of natural numbers has been shown to be solvable in polynomial time by quantum hardware [27,28]. However, quantum algorithmics is still in its early days (especially when compared with classical algorithmics), and the number of genuinely quantum algorithmic techniques can be counted on the fingers of one hand. One obstacle against progress in this field is the lack of a universally accepted formalism able to express quantum algorithms naturally, i.e. a programming language fitted for quantum computation. But even more fundamentally, similar problems affect quantum complexity theory: there is no general agreement on what should be *the* quantum computational model, i.e., the computational model on top of which one define the complexity of any computational problem (classically, this rôle is played by random access machines). Summing up, quantum computation is a very promising paradigm with many potentially interesting

☆ The authors are partially supported by PRIN project "CONCERTO". The first author is partially supported by the FIRB grant RBIN04M8S8, "Intern. Inst. for Applicable Math".

* Corresponding author. Tel.: +39 0512094991.
E-mail addresses: dallago@cs.unibo.it (U. Dal Lago), andrea.masini@univr.it (A. Masini), margherita.zorzi@univr.it (M. Zorzi).

0304-3975/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2009.07.045

complexity theoretic properties; its foundations, however, have not stabilized yet. A paper by Bernstein and Vazirani [7] can be considered as the most widely accepted reference for quantum computational complexity. There, quantum Turing machines are the computational model over which complexity classes are defined. A quantum Turing machine is defined similarly to a (classical) Turing machine. However, in classical Turing machines, any configuration is a pair $C = (q, t)$, where q is a state from a finite set and t is the content of the machine's tape, while in the quantum case, configurations are elements of an Hilbert space over the space of pairs we have just described. Apparently, this makes the computational power of quantum Turing machines higher than the one of their classical counterparts, since each base vector in the superposition can evolve independently (provided the overall evolution stays reversible). When computation stops (and defining this is not at all easy, see [7] for possible solutions), the result of the computation is obtained by quantum measurement, applied to the current configuration. As a consequence, the outcome of quantum computation is not uniquely determined, since quantum measurement is inherently probabilistic. Once a computational model is fixed, defining complexity classes over it is relatively simple. But even if we focus on problems decidable in polynomial time, one could define three distinct complexity classes, since different constraints can be imposed on success and error probabilities:

- EQP, if we impose the success probability to be 1 on all input instances.
- BQP, if we impose the success probability to be strictly greater than $1/2$ on all input instances.
- ZQP, if we impose the success probability to be strictly greater than $1/2$ and the error probability to be 0.

Implicit computational complexity. The aim of implicit computational complexity (ICC) [6,16] is giving machine-free, mathematical-logic-based characterizations of complexity classes, with particular emphasis on small complexity classes like the one of polynomial time computable functions. Many characterizations of polynomial time functions based on model theory, recursion theory and proof theory have appeared in the last twenty years [8,6,16,22]. Potential applications of implicit computational complexity lie in the areas of programming language theory (because controlling resource usage is crucial when programs are run in constrained environments) and complexity theory (since traditional, combinatorial techniques have so far failed in solving open problems about separation between complexity classes).

Linear logic. Linear logic [15] has been introduced by Jean-Yves Girard twenty years ago. It is both a decomposition and a refinement of intuitionistic logic. As such, it sheds some light on the dynamics of normalization. In particular, the copying phenomenon is put in evidence by way of modalities. Linear Logic has leveraged research in many branches of programming language theory, including functional and logic programming.

Quantum computation, ICC and linear logic. Controlling copying (and erasing) as made possible by Linear Logic is essential in both quantum computation and implicit computational complexity, for different reasons.

Classically, copying the value of a bit is always possible (think at boolean circuits). In quantum computation, on the other hand, copying a (quantum) bit is not possible in general: this is the so-called *non-cloning property*. Moreover, erasing a bit corresponds to an implicit measurement, which is often restricted to happen at the end of the computation. One technique enforcing these properties comes from linear logic: linearity corresponds to the impossibility of copying and erasing arguments during reduction. Moreover, the syntax of linear logic makes it simple to keep everything under control even if copying is permitted. Two different calculi, designed starting from the above ideas, can be found in the literature [31,26]. In both cases, only data are quantum, while control remains classical. Some developments on the same ideas can be found in a recent paper by the authors [9], which introduces a calculus called Q.

On the other hand, the possibility of copying subproofs (in a wild way) during cut-elimination is the only reason why cut-elimination is in general a computationally heavy process, for if copying is not allowed (like in plain, multiplicative linear logic), normalization can be done in polynomial time for very simple reasons: every cut-elimination step makes the underlying proof strictly smaller (and, as a consequence, cut-elimination can be performed in a linear number of steps).

Ten years ago, Jean-Yves Girard wrote [16]: *"We are seeking a* $\langle\langle$*logic of polytime*$\rangle\rangle$*. Not yet one more axiomatization, but an intrinsically polytime system."*, where the expressive power of the system was given by the computational complexity of the cut elimination procedure. Girard's main breakthrough was to understand that the problem of exponential blowup (in time and space) of cut elimination is essentially caused by *structural rules* (in particular contraction, responsible, during the cut elimination process, of duplications). In order to solve the problem Girard proposed a *light* version of linear logic [16], where duplication is controlled by restricting exponential rules; this way he was able to master the expressive power (in the Curry-Howard sense) of the logical system.

This idea has been subsequently simplified and extended into one of the most promising branches of ICC. Many distinct lambda calculi and logical systems characterizing complexity classes being inspired by linear logic have appeared since then. Some of them descend from Asperti's *light affine logic* [1,2,4], others from Lafont's *soft linear logic* [17]. Indeed, this is one of the more fertile and vital areas of implicit computational complexity. Linear lambda-calculi corresponding (in the Curry-Howard sense) to light affine logic [30] or to soft linear logic [3] both enjoy the following remarkable property: polynomial bounds on normalization time hold in the *absence* of types, too. In other words, types are not necessary to get polytime soundness.

One of the main motivations for studying QC is the potential speed-up in execution time induced by quantum superposition. Shor [27,28] surprised the scientific community, by showing (roughly speaking) that an ideal quantum computer could factorize an integer in polytime. The hope is that quantum machines could properly go beyond (from a

computational complexity point of view[1]) classical computing machines. In this perspective the research area of quantum abstract complexity theory has been recently developed (see, e.g., [7,19]), with a particular emphasis on polynomial time.

A quite natural question could be: *is it possible to investigate quantum complexity by means of ICC?* This paper is the very first answer to the question.

This paper. In this paper, we give an implicit characterization of polytime quantum complexity classes by way of a calculus called SQ. SQ is a quantum lambda calculus based on Lafont's soft linear logic. Like in many other proposals for quantum calculi and programming languages [9,23–26], control is classical and only data live in a quantum world. The correspondence with quantum complexity classes is an extensional correspondence, proved by showing that:

- on one side, any term in the language can be evaluated in polynomial time (where the underlying polynomial depends on the *box depth* of the considered term);
- on the other side, any problem P decidable in polynomial time (in a quantum sense) can be represented in the language, i.e., there exists a term M which decides P.

This is much in the style of ICC, where results of this kind abound. To the authors' knowledge, however, this is the first example of an implicit characterization of quantum polytime decision problems.

Terms and configurations of SQ form proper subclasses of the ones of Q [9], an untyped lambda calculus with classical control and quantum data previously introduced by the authors. This implies that results like standardization and confluence do not need to be proved. Based on suggestions by the anonymous referees, however, the authors decided to keep this paper self-contained by giving proofs for the above results in two appendices.

The results in this paper are not unexpected, since soft linear logic is sound and complete wrt polynomial time in the classical sense [17]. This does not mean the correspondence with quantum polynomial time is straightforward. In particular, showing polytime completeness requires a relatively non-standard technique (see Section 9).

On explicit measurements. We conclude the introduction with an important remark. The syntax of SQ does not allow to classically observe the content of the quantum register. More specifically, the language of terms does not include any measurement operator which, applied to a quantum variable, had the effect of observing the value of the corresponding qubit (this in contrast to, e.g., Selinger and Valiron's proposal [26]). We agree with researchers that put in foreground the need of a measurement operator in any respectable quantum programming language, but the concern in our proposal is quantum computation theory, not programming languages; SQ is a quantum lambda calculus that morally lives in the same world as the one of quantum Turing machines and quantum circuit families. Our choice is not a novelty, it is fully motivated by some relevant literature on quantum computing [7,18], where *measurements are assumed to take place only at the end of computation*.

2. Preliminaries

Before introducing the syntax of SQ, some preliminary concepts are necessary. As always when talking about quantum computation, we cannot be exhaustive. We refer the reader to [18] for a comprehensive survey on the topic; see also Appendix A for further notions about Hilbert spaces.

A *quantum variable set (qvs)* is any finite set $\mathcal{V}$ of quantum variables[2] (ranged over by variables like p, r and q).

Given a set $\mathcal{S}$, the Hilbert space with computational basis $\mathcal{B}(\mathcal{S}) = \{|s\rangle \mid s \in \mathcal{S}\}$ will be denoted as $\mathcal{H}(\mathcal{S})$ (see Appendix A for the relevant definitions). We will here work with slightly nonstandard Hilbert spaces. In particular, their computational basis consists of the set of all $|f\rangle$ where f is a map from a fixed qvs $\mathcal{V}$ to $\{0, 1\}$; this Hilbert space is denoted as $\mathcal{H}(\{0, 1\}^{\mathcal{V}})$ or, with a slight abuse of notation, simply as $\mathcal{H}(\mathcal{V})$. If $\mathcal{V}$ is a qvs, a *quantum register* is a normalized vector in $\mathcal{H}(\mathcal{V})$.

Usually, the computational basis of a (finite dimensional) Hilbert space is rather $\{|v\rangle \mid v \in \{0, 1\}^n\}$; this way we obtain the 2^n-dimensional Hilbert space $\mathcal{H}(\{0, 1\}^n)$.

In this paper we will show that SQ is sound and complete with respect to *polynomial time quantum Turing machines* as defined by Bernstein and Vazirani [7]. In particular, in order to show the "perfect" equivalence of the proposed calculus with polynomial time quantum Turing machines, we need to restrict our attention to a subclass of unitary operators, the so-called computable operators (see, e.g., the paper of Nishimura and Ozawa [20] on the perfect equivalence between quantum circuit families and quantum Turing machines).

In the following, we will write 1^n for the sequence $11 \ldots 1$ (n times).

Definition 1. A real number $x \in \mathbb{R}$ is *polytime computable* (x is in $\mathbf{P}\mathbb{R}$) iff there is a deterministic Turing machine which on input 1^n computes a binary representation of an integer $m \in \mathbb{Z}$ such that $|m/2^n - x| \leq 1/2^n$. A complex number $z = x + iy$ is *polytime computable* (z is in $\mathbf{P}\mathbb{C}$) iff $x, y \in \mathbf{P}\mathbb{R}$. A normalized vector ϕ in any Hilbert space $\mathcal{H}(\mathcal{S})$ is *computable* if the range of ϕ (a function from $\mathcal{S}$ to complex numbers) is $\mathbf{P}\mathbb{C}$. A unitary operator $\mathbf{U} : \mathcal{H}(\mathcal{S}) \to \mathcal{H}(\mathcal{S})$ is computable if for every computable normalized vector ϕ of $\mathcal{H}(\mathcal{S})$, $\mathbf{U}(\phi)$ is computable.

[1] The expectation is that the class of quantum feasible problems strictly contains the class of classical feasible problems.

[2] We point out that a quantum variable is not a quantum object, quantum variables will be used to give names to qubits.

Let $u \in \mathcal{H}(\{0,1\}^n)$ be the quantum register $u = \alpha_1|0\ldots0\rangle + \cdots + \alpha_{2^n}|1\ldots1\rangle$ and let $\langle q_1, \ldots, q_n\rangle$ be a sequence of names; $u^{\langle q_1,\ldots,q_n\rangle}$ is the quantum register in $\mathcal{H}(\{q_1, \ldots, q_n\})$ defined by $u^{\langle q_1,\ldots,q_n\rangle} = \alpha_1|q_1 \mapsto 0, \ldots, q_n \mapsto 0\rangle + \cdots + \alpha_{2^n}|q_1 \mapsto 1, \ldots, q_n \mapsto 1\rangle$.

Let $\mathbf{U} : \mathcal{H}(\{q_1, \ldots, q_n\}) \to \mathcal{H}(\{q_1, \ldots, q_n\})$ be an computable operator and let $\langle q_1, \ldots, q_n\rangle$ be any sequence of distinguished names in $\mathcal{V}$. Considering the bijection between $\{0,1\}^n$ and $\{0,1\}^{\{q_1,\ldots,q_n\}}$, $\mathbf{U}$ and $\langle q_1, \ldots, q_n\rangle$ induce an operator $\mathbf{U}_{\langle q_1,\ldots,q_n\rangle} : \mathcal{H}(\{q_1, \ldots, q_n\}) \to \mathcal{H}(\{q_1, \ldots, q_n\})$ defined as follows: if $|f\rangle = |q_{j_1} \mapsto b_{j_1}, \ldots, q_{j_n} \mapsto b_{j_n}\rangle$ is an element of the orthonormal basis of $\mathcal{H}(\{q_1, \ldots, q_n\})$, then

$$\mathbf{U}_{\langle q_1,\ldots,q_n\rangle}|f\rangle \stackrel{def}{=} (\mathbf{U}|b_1, \ldots, b_n\rangle)^{\langle q_1,\ldots,q_n\rangle}$$

where $q_{j_i} \mapsto b_{j_i}$ means that we associate the element b_{j_i} of the basis to the qubit named q_{j_i}.

Let $\mathcal{W} = \{q_1, \ldots, q_k\} \subseteq \mathcal{V}$. We naturally extend (by suitable standard isomorphisms) the unitary operator $\mathbf{U}_{\langle q_{j_1},\ldots,q_{j_k}\rangle} : \mathcal{H}(\mathcal{W}) \to \mathcal{H}(\mathcal{W})$ to the unitary operator $\mathbf{U}_{\langle\langle q_1,\ldots,q_k\rangle\rangle} : \mathcal{H}(\mathcal{V}) \to \mathcal{H}(\mathcal{V})$ that acts as the identity on variables not in $\mathcal{W}$ and as $\mathbf{U}_{\langle q_1,\ldots,q_k\rangle}$ on variables in $\mathcal{W}$.

3. Syntax

The syntax of SQ is very similar to the one of Q, itself introduced by the authors [9]. Here, we aim at giving a self-contained introduction to the calculus.

3.1. The language of terms

Let us associate to each computable unitary operator $\mathbf{U}$ on the Hilbert space $\mathcal{H}(\{0,1\}^n)$, a symbol U. The set of the *term expressions*, or *terms* for short, is defined by the following grammar:

$$\begin{array}{lll}
x & ::= x_0, x_1, \ldots & \textit{classical variables} \\
r & ::= r_0, r_1, \ldots & \textit{quantum variables} \\
\pi & ::= x \mid \langle x_1, \ldots, x_n\rangle & \textit{linear patterns (where } n \geq 2\textit{)} \\
\psi & ::= \pi \mid !x & \textit{patterns} \\
B & ::= 0 \mid 1 & \textit{boolean constants} \\
U & ::= U_0, U_1, \ldots & \textit{unitary operators} \\
C & ::= B \mid U & \textit{constants} \\
M & ::= x \mid r \mid !N \mid C \mid \mathtt{new}(N) \mid M_1M_2 \mid & \\
 & \quad \langle M_1, \ldots, M_n\rangle \mid \lambda\psi.N & \textit{terms (where } n \geq 2\textit{)}
\end{array}$$

In the following, capital letters such as M, N and P (possibly indexed), denote terms. Observe that, by definition, only classical variables can be bound, i.e., abstractions can be made on patterns including classical variables. We work modulo variable renaming; in other words, *terms are equivalence classes modulo α-conversion*. With $M_1 = M_2$ we denote that the terms (equivalence classes) M_1 and M_2 are α-equivalent. Substitution up to α-equivalence is defined in the usual way.

Since we are working modulo α-conversion, we are authorized to use the so called Barendregt Convention on Variables (shortly, BCV) [5]: in each mathematical context (a term, a definition, a proof...) the names chosen for bound variables will always differ from those of the free ones.

Let us denote with $\mathbf{Q}(M_1, \ldots, M_k)$ the set of quantum variables occurring in $M_1, \ldots, M_k$. For every term M and for every classical variable x the number of free occurrences $\mathsf{NFO}(x, M)$ of x in M is defined as follows, by induction on M:

$$\begin{aligned}
\mathsf{NFO}(x, x) &= 1 \\
\mathsf{NFO}(x, y) &= \mathsf{NFO}(x, r) = \mathsf{NFO}(x, C) = 0 \\
\mathsf{NFO}(x, !M) &= \mathsf{NFO}(x, \mathtt{new}(M)) = \mathsf{NFO}(x, M) \\
\mathsf{NFO}(x, \lambda y.M) &= \mathsf{NFO}(x, \lambda !y.M) = \mathsf{NFO}(x, M) \\
\mathsf{NFO}(x, \lambda\langle x_1, \ldots, x_n\rangle.M) &= \mathsf{NFO}(x, M) \\
\mathsf{NFO}(x, MN) &= \mathsf{NFO}(x, M) + \mathsf{NFO}(x, N) \\
\mathsf{NFO}(x, \langle M_1, \ldots, M_n\rangle) &= \sum_1^n \mathsf{NFO}(x, M_i)
\end{aligned}$$

3.2. Judgements and well-formed terms

Judgements are defined from various notions of environments, taking into account the way the variables are used:

- A *classical environment* is a (possibly empty) set (denoted by Δ, possibly indexed) of classical variables. With $!\Delta$ we denote the set $!x_1, \ldots, !x_n$ whenever Δ is $x_1, \ldots, x_n$. Analogously, with $\#\Delta$, we denote the environment $\#x_1, \ldots, \#x_n$

$$\frac{}{!\Delta \vdash C}\ \mathsf{const} \qquad \frac{}{!\Delta, r \vdash r}\ \mathsf{qvar} \qquad \frac{}{!\Delta, x \vdash x}\ \mathsf{cvar}$$

$$\frac{}{!\Delta, \#x \vdash x}\ \mathsf{der}_1 \qquad \frac{}{!\Delta, !x \vdash x}\ \mathsf{der}_2$$

$$\frac{\Psi_1, \#\Delta_1 \vdash M_1 \quad \Psi_2, \#\Delta_2 \vdash M_2}{\Psi_1, \Psi_2, \#\Delta_1 \cup \#\Delta_2 \vdash M_1 M_2}\ \mathsf{app}$$

$$\frac{\Psi_1, \#\Delta_1 \vdash M_1 \cdots \Psi_k, \#\Delta_k \vdash M_k}{\Psi_1, \ldots, \Psi_k, \#\Delta_1 \cup \#\Delta_2 \cup \cdots \cup \#\Delta_k \vdash \langle M_1, \ldots, M_k \rangle}\ \mathsf{tens}$$

$$\frac{\Delta_1 \vdash M}{!\Delta_2, !\Delta_1 \vdash !M}\ \mathsf{prom} \qquad \frac{\Gamma \vdash M}{\Gamma \vdash \mathtt{new}(M)}\ \mathsf{new} \qquad \frac{\Gamma, x_1, \ldots, x_n \vdash M}{\Gamma \vdash \lambda\langle x_1, \ldots, x_n \rangle.M}\ \mathsf{abs}_1$$

$$\frac{\Gamma, x \vdash M}{\Gamma \vdash \lambda x.M}\ \mathsf{abs}_2 \qquad \frac{\Gamma, \#x \vdash M}{\Gamma \vdash \lambda !x.M}\ \mathsf{abs}_3 \qquad \frac{\Gamma, !x \vdash M}{\Gamma \vdash \lambda !x.M}\ \mathsf{abs}_4$$

Fig. 1. Well forming rules.

whenever Δ is $x_1, \ldots, x_n$. If Δ is empty, then $!\Delta$ and $\#\Delta$ are empty. Notice that if Δ is a non-empty classical environment, both $\#\Delta$ and $!\Delta$ are *not* classical environments.

- A *quantum environment* is a (possibly empty) set (denoted by Θ, possibly indexed) of quantum variables.
- A *linear environment* is a (possibly empty) set (denoted by Λ, possibly indexed) Δ, Θ of classic and quantum variables.
- A *non-contractible environment* is a (possibly empty) set (denoted by Ψ, possibly indexed) Λ, $!\Delta$ where each variable name occurs at most once.
- An *environment* (denoted by Γ, eventually indexed) is a (possibly empty) set Ψ, $\#\Delta$ where each variable name occurs at most once.
- A *judgment* is an expression $\Gamma \vdash M$, where Γ is an environment and M is a term.
- If $\Gamma_1, \ldots, \Gamma_n$ are (not necessarily pairwise distinct) environments, $\Gamma_1 \cup \cdots \cup \Gamma_n$ denotes the environment obtained by means of the standard set-union of $\Gamma_1, \ldots, \Gamma_n$.

In all the above definitions, we are implicitly assuming that the same (quantum or classical) variable name cannot appear more than once in an environment, e.g. x, $!y$, $\#z$ is a correct environment, while x, $!x$ is not. Given an environment Γ, $\mathtt{var}(\Gamma)$ denotes the set of variable names in Γ.

We say that a judgement $\Gamma \vdash M$ is *well formed* (*notation:* $\triangleright\, \Gamma \vdash M$) if it is derivable by means of the *well forming rules* in Fig. 1. If d is a derivation of the well formed judgement $\Gamma \vdash M$, we write $d \triangleright\ \Gamma \vdash M$. If $\Gamma \vdash M$ is *well formed* we say that M *is well formed with respect to the environment* Γ, or, simply, that M *is well formed*.

The rôle of the underlying context in well formed judgements can be explained as follows. If $\Gamma, x \vdash M$ is well formed, then x appears free *exactly* once in M and, moreover, the only free occurrence of x does not lie in the scope of any ! construct. On the other hand, if $\Gamma, \#x \vdash M$ is well formed, then x appears free *at least* once in M and every free occurrence of x does not lie in the scope of any ! construct. Finally, if $\Gamma, !x \vdash M$ is well formed, then x appears *at most* once in M.

Proposition 1. *If* $\triangleright\, \mathbf{Q}(M) \vdash M$ *then all the classical variables in* M *are bound.*

Proof. The following, stronger, statement can be proved by structural induction on d: if $d \triangleright\ \Gamma \vdash M$ then all the free variables of M appears in Γ. □

SQ is a language inspired by Lafont's soft linear logic [17]: the classical fragment of SQ is very similar (essentially equivalent) to the language of terms of Baillot and Mogbil's soft lambda calculus [3], where the authors show how soft lambda terms can be typed with formulas of soft linear logic. But many interesting properties hold for soft lambda terms even in the absence of types, i.e., the structure of untyped terms is itself sufficient to enforce those properties. This includes soundness and completeness wrt polynomial time. This is the main reason why we decided to present SQ as an untyped language.

4. Computations

Computations are defined by means of configurations. A *preconfiguration* is a triple $[\mathcal{Q}, \mathcal{QV}, M]$ where:

- M is a term;
- $\mathcal{QV}$ is a finite quantum variable set such that $\mathbf{Q}(M) \subseteq \mathcal{QV}$;
- $\mathcal{Q} \in \mathcal{H}(\mathcal{QV})$.

β-reductions

$$[\mathcal{Q}, \mathcal{QV}, (\lambda x.M)N] \to_{\mathsf{l}.\beta} [\mathcal{Q}, \mathcal{QV}, M\{N/x\}]$$
$$[\mathcal{Q}, \mathcal{QV}, (\lambda\langle x_1, \ldots, x_n\rangle.M)\langle r_1, \ldots, r_n\rangle] \to_{\mathsf{q}.\beta} [\mathcal{Q}, \mathcal{QV}, M\{r_1/x_1, \ldots, r_n/x_n\}]$$
$$[\mathcal{Q}, \mathcal{QV}, (\lambda !x.M)!N] \to_{\mathsf{c}.\beta} [\mathcal{Q}, \mathcal{QV}, M\{N/x\}]$$

Unitary transform of quantum register

$$[\mathcal{Q}, \mathcal{QV}, U\langle r_{i_1}, \ldots, r_{i_n}\rangle] \to_{\mathsf{Uq}} [\mathbf{U}_{\langle\langle r_{i_1}, \ldots, r_{i_n}\rangle\rangle} Q, \mathcal{QV}, \langle r_{i_1}, \ldots, r_{i_n}\rangle]$$

Creation of a new qubit and quantum variable

$$[\mathcal{Q}, \mathcal{QV}, \mathtt{new}(c)] \to_{\mathsf{new}} [\mathcal{Q} \otimes |r \mapsto c\rangle, \mathcal{QV} \cup \{r\}, r] \qquad (r \text{ is fresh})$$

Commutative reductions

$$[\mathcal{Q}, \mathcal{QV}, L((\lambda\pi.M)N)] \to_{\mathsf{l.cm}} [\mathcal{Q}, \mathcal{QV}, (\lambda\pi.LM)N]$$

$$[\mathcal{Q}, \mathcal{QV}, ((\lambda\pi.M)N)L] \to_{\mathsf{r.cm}} [\mathcal{Q}, \mathcal{QV}, (\lambda\pi.ML)N]$$

Context closure

$$\frac{[\mathcal{Q}, \mathcal{QV}, M_i] \to_\alpha [\mathcal{R}, \mathcal{RV}, N_i]}{[\mathcal{Q}, \mathcal{QV}, \langle M_1, \ldots, M_i, \ldots, M_k\rangle] \to_\alpha [\mathcal{R}, \mathcal{RV}, \langle M_1, \ldots, N_i, \ldots, M_k\rangle]}\ \mathsf{t}_i$$

$$\frac{[\mathcal{Q}, \mathcal{QV}, N] \to_\alpha [\mathcal{R}, \mathcal{RV}, L]}{[\mathcal{Q}, \mathcal{QV}, MN] \to_\alpha [\mathcal{R}, \mathcal{RV}, ML]}\ \mathsf{r.a} \qquad \frac{[\mathcal{Q}, \mathcal{QV}, M] \to_\alpha [\mathcal{R}, \mathcal{RV}, N]}{[\mathcal{Q}, \mathcal{QV}, ML] \to_\alpha [\mathcal{R}, \mathcal{RV}, NL]}\ \mathsf{l.a}$$

$$\frac{[\mathcal{Q}, \mathcal{QV}, M] \to_\alpha [\mathcal{R}, \mathcal{RV}, N]}{[\mathcal{Q}, \mathcal{QV}, \mathtt{new}(M)] \to_\alpha [\mathcal{R}, \mathcal{RV}, \mathtt{new}(N)]}\ \mathsf{in.new}$$

$$\frac{[\mathcal{Q}, \mathcal{QV}, M] \to_\alpha [\mathcal{R}, \mathcal{RV}, N]}{[\mathcal{Q}, \mathcal{QV}, (\lambda !x.M)] \to_\alpha [\mathcal{R}, \mathcal{RV}, (\lambda !x.N)]}\ \mathsf{in}.\lambda_1$$

$$\frac{[\mathcal{Q}, \mathcal{QV}, M] \to_\alpha [\mathcal{R}, \mathcal{RV}, N]}{[\mathcal{Q}, \mathcal{QV}, (\lambda\pi.M)] \to_\alpha [\mathcal{R}, \mathcal{RV}, (\lambda\pi.N)]}\ \mathsf{in}.\lambda_2$$

Fig. 2. Reduction rules.

Let $\theta : \mathcal{QV}_1 \to \mathcal{QV}_2$ be a bijective function from a (nonempty) finite set of quantum variables $\mathcal{QV}_1$ to another set of quantum variables $\mathcal{QV}_2$. Then we can extend θ to any term whose quantum variables are included in $\mathcal{QV}_1$: $\theta(M)$ will be identical to M, except on quantum variables, which are changed according to θ itself. Observe that $\mathbf{Q}(\theta(M)) \subseteq \mathcal{QV}_2$. Similarly, θ can be extended to a function from $\mathcal{H}(\mathcal{QV}_1)$ to $\mathcal{H}(\mathcal{QV}_2)$ in the obvious way.

Definition 2 (*Configurations*). Two preconfigurations $[\mathcal{Q}_1, \mathcal{QV}_1, M_1]$ and $[\mathcal{Q}_2, \mathcal{QV}_2, M_2]$ are equivalent iff there is a bijection $\theta : \mathcal{QV}_1 \to \mathcal{QV}_2$ such that $\mathcal{Q}_2 = \theta(\mathcal{Q}_1)$ and $M_2 = \theta(M_1)$. If a preconfiguration C_1 is equivalent to C_2, then we will write $C_1 \equiv C_2$. The relation $\equiv$ is an equivalence relation. A *configuration* is an equivalence class of preconfigurations modulo the relation $\equiv$. Let $\mathcal{C}$ be the set of configurations.

Remark 1. The way configurations have been defined, namely quotienting preconfigurations over $\equiv$, is very reminiscent of usual α-conversion in lambda-terms. Indeed, it is enlightening to think at quantum variables as bound variables (or, even better, as pointers to the quantum register) rather than free variables.

Let $\mathscr{L} = \{\mathsf{Uq}, \mathsf{new}, \mathsf{l}.\beta, \mathsf{q}.\beta, \mathsf{c}.\beta, \mathsf{l.cm}, \mathsf{r.cm}\}$. The set $\mathscr{L}$ will be ranged over by α, β. For each $\alpha \in \mathscr{L}$, we can define a reduction relation $\to_\alpha \subseteq \mathcal{C} \times \mathcal{C}$ by means of the rules in Fig. 2. Please, notice the presence of the two commutative reduction rules l.cm and r.cm. The rôle of l.cm and r.cm is related to the standardization result presented in Section 5. Roughly speaking, commutative rules prevent quantum reduction to block classical reduction. For any subset $\mathscr{S}$ of $\mathscr{L}$, we can construct a relation $\to_{\mathscr{S}}$ by just taking the union over $\alpha \in \mathscr{S}$ of $\to_\alpha$. In particular, $\to$ will denote $\to_{\mathscr{L}}$. The usual notation for the transitive and reflexive closures will be used. In particular, $\overset{*}{\to}$ will denote the transitive and reflexive closure of $\to$. Notice we have defined $\to$ by closing reduction rules under any context except the ones in the form $!M$. So $\to$ is not a strategy.

4.1. Subject reduction

Even if SQ is not typed, we have a strong notion of well formation for terms. As we will see the well forming rules are strong enough to guarantee termination of computations with polynomial bounds (see Section 8). It is necessary to introduce a suitable notion of *well formed configuration* and, moreover, to show that well formed configurations are closed under reduction.

Definition 3. A configuration $[\mathcal{Q}, \mathcal{QV}, M]$ is said to be *well-formed* iff there is a context Γ such that $\Gamma \vdash M$ is well-formed.

The following is the main result of this section:

Theorem 1 (*Well Formation Closure*). *If C is a* well-formed *configuration and $C \xrightarrow{*} D$ then D is well formed.*

The proof is a consequence (provable by induction) of the following stronger result that, with a little abuse of language (the calculus is untyped), we call *subject-reduction theorem*.

Theorem 2 (*Subject Reduction*). *If $\triangleright \Lambda, !\Delta_1, \#\Delta_2 \vdash M_1$ and $[\mathcal{Q}_1, \mathcal{QV}_1, M_1] \to [\mathcal{Q}_2, \mathcal{QV}_2, M_2]$ then there are environments Δ_3, Δ_4 such that $\Delta_1 = \Delta_3, \Delta_4$ and $\triangleright \Lambda, !\Delta_3, \#\Delta_4 \cup \#\Delta_2, \mathcal{QV}_2 - \mathcal{QV}_1 \vdash M_2$. Moreover, $\mathcal{QV}_2 - \mathcal{QV}_1 = \mathbf{Q}(M_2) - \mathbf{Q}(M_1)$.*

Proof. In order to prove the theorem we need a number of intermediate results (the proofs are very easy and long inductions). We first need to show that certain rules are admissible. The following lemmas can be easily proved by induction on the structure of derivations:

- If $\triangleright \Gamma \vdash M$ and x is a fresh variable then $\triangleright \Gamma, !x \vdash M$;
- If $\triangleright \Gamma, x \vdash M$, then $\triangleright \Gamma, !x \vdash M$;
- If $\triangleright \Gamma, x \vdash M$, then $\triangleright \Gamma, \#x \vdash M$.

As a consequence the following rules are admissible:

$$\frac{\Gamma \vdash M}{\Gamma, !\Delta \vdash M}\ \mathsf{weak} \qquad \frac{\Gamma, \Delta \vdash M}{\Gamma, !\Delta \vdash M}\ \mathsf{der}_3 \qquad \frac{\Gamma, \Delta \vdash M}{\Gamma, \#\Delta \vdash M}\ \mathsf{der}_4$$

with the proviso that in rule weak, each x in Δ is a fresh variable. As always, proving subject reduction requires some substitution lemmas. In this case, we need four distinct substitution lemmas, which can all be proved by long inductions:

Linear case. If $\triangleright \Psi_1, \#\Delta_1, x \vdash P$ and $\triangleright \Psi_2, \#\Delta_2 \vdash N$, with $\mathsf{var}(\Psi_1) \cap \mathsf{var}(\Psi_2) = \emptyset$, then $\triangleright \Psi_1, \Psi_2, \#\Delta_1 \cup \#\Delta_2 \vdash P\{N/x\}$.
Contraction case. If $\triangleright \Gamma, \#x \vdash P$ and $\triangleright \Delta \vdash N$ and $\mathsf{var}(\Gamma) \cap \mathsf{var}(\Delta) = \emptyset$ then $\triangleright \Gamma, \#\Delta \vdash P\{N/x\}$.
Bang case. If $\triangleright \Gamma, !x \vdash P$ and $\triangleright \Delta \vdash N$ and $\mathsf{var}(\Gamma) \cap \mathsf{var}(\Delta) = \emptyset$ then $\triangleright \Gamma, !\Delta \vdash P\{N/x\}$.
Quantum case. If $\triangleright \Gamma, x_1, \ldots, x_n \vdash P$, $\triangleright !\Delta, r_1, \ldots, r_n \vdash \langle r_1, \ldots, r_n\rangle$ and $r_1, \ldots, r_n \notin \mathsf{var}(\Gamma)$ then $\triangleright \Gamma, !\Delta, r_1, \ldots, r_n \vdash P\{r_1/x_1, \ldots, r_n/x_n\}$

Observe how the hypotheses of the four cases are different. As an example, when a term N is substituted for a variable x appearing more once in P, the (free) variables in N must be "contractible" and, moreover, N cannot contain any quantum variable. On the other hand, if x appears exactly once in P, no constraint must be imposed on variables appearing in N.
Let Γ be an environment, a partial function m with domain Γ is called an *mfun* (for Γ) if:

1. if α (occurring in Γ) is either a classical variable or a quantum variable or has the shape $\#x$, then $m(\alpha) = \alpha$;
2. if $!x$ occurs in Γ then $m(!x)$ is either $!x$ or $\#x$.

It is immediate to observe that if $\Gamma = \alpha_1, \ldots, \alpha_n$ is an environment and m is an *mfun*, then $m[\Gamma] = m(\alpha_1), \ldots, m(\alpha_n)$ is an environment; We are now in a position to prove Theorem 2. We prove it in the following equivalent formulation: if $d \triangleright \Gamma \vdash M$ and $[\mathcal{Q}_1, \mathcal{QV}_1, M_1] \to [\mathcal{Q}_2, \mathcal{QV}_2, M_2]$ then there is an *mfun* m for Γ such that $\triangleright m[\Gamma], \mathcal{QV}_2 - \mathcal{QV}_1 \vdash M_2$. The proof is by induction on the height of d and by cases on the last rule r of d. There are several cases, we will show only some of them.

- r is app:

$$\frac{\begin{array}{cc} \begin{array}{c} d_1 \\ \vdots \\ \end{array} & \begin{array}{c} d_2 \\ \vdots \\ \end{array} \\ \Psi_1, \#\Delta_1 \vdash P_1 & \Psi_2, \#\Delta_2 \vdash P_2 \end{array}}{\Psi_1, \Psi_2, \#\Delta_1 \cup \#\Delta_2 \vdash P_1P_2}\ \mathsf{app}$$

  and the reduction rule is

$$\frac{[\mathcal{Q}, \mathcal{QV}, P_1] \to_\alpha [\mathcal{R}, \mathcal{RV}, P_3]}{[\mathcal{Q}, \mathcal{QV}, P_1P_2] \to_\alpha [\mathcal{R}, \mathcal{RV}, P_3P_2]}\ \mathsf{l.a.}$$

Applying the induction hypothesis to d_1 there is an *mfun* m and a derivation d_3 such that:

$$\dfrac{\begin{array}{cc} \overset{d_3}{\vdots} & \overset{d_2}{\vdots} \\ m[\Psi_1], \#\Delta_1, \mathcal{RV} - \mathcal{QV} \vdash P_3 & \Psi_2, \#\Delta_2 \vdash P_2 \end{array}}{m[\Psi_1], \Psi_2, \mathcal{RV} - \mathcal{QV}, \#\Delta_1 \cup \#\Delta_2 \vdash P_3 P_2}\ \mathsf{app}.$$

Please note that if $!y$ occur in Ψ_1 then $\#y$ cannot occur neither in $\#\Delta_1$ nor in $\#\Delta_2$, therefore even if $m(!y) = \#y$, the rule app is applied correctly.

- r is app:

$$\dfrac{\dfrac{\overset{d_1}{\vdots}}{\Gamma, \#x \vdash P}}{\Gamma \vdash \lambda !x.P}\ \mathsf{abs}_2 \qquad \dfrac{\overset{d_2}{\vdots}}{\Delta_2 \vdash N}\ \mathsf{prom}}{\Gamma, !\Delta_3, !\Delta_2 \vdash (\lambda !x.P)(!N)}\ \mathsf{app}$$

and the reduction rule is:

$$[\mathcal{Q}, \mathcal{QV}, (\lambda !x.P)!N] \to_{\mathrm{c}.\beta} [\mathcal{Q}, \mathcal{QV}, P\{N/x\}]$$

we have the thesis by means of the substitution lemmas above:

$$\dfrac{\overset{d_3}{\vdots}}{\Gamma, \#\Delta_2 \vdash P\{N/x\}}\ \mathsf{weak}$$

$$\Gamma, !\Delta_3, \#\Delta_2 \vdash P\{N/x\}$$

where d_3 is the derivation obtained by applying the contraction case to d_1 and d_2.

- r is new:

$$\dfrac{!\Delta \vdash c}{!\Delta \vdash \mathtt{new}(c)}\ \mathsf{new}$$

and the reduction rule is:

$$[\mathcal{Q}, \mathcal{QV}, \mathtt{new}(c)] \to [\mathcal{Q} \otimes |p \mapsto c\rangle, \mathcal{QV} \cup \{p\}, p]$$

The thesis is obtained by means of qvar:

$$\dfrac{}{!\Delta, p \vdash p}\ \mathsf{qvar}$$

The other cases can be handled in the same way. □

In the rest of the paper we will restrict our attention to well-formed configurations, that we continue to call simply configurations to ease reading. We conclude this Section with the notions of *computation* and *normal form*.

Definition 4 (*Normal Forms*). A configuration $C \equiv [\mathcal{Q}, \mathcal{QV}, M]$ is said to be in *normal form* iff there is no D such that $C \to D$. Let us denote with NF the set of configurations in normal form.

We define a computation as a suitable *finite* sequence of configurations:

Definition 5 (*Computations*). If C_1 is any configuration, a *computation* of length $m \in \mathbb{N}$ starting with C_1 is a sequence of configurations $\{C_i\}_{1 \le i \le m}$ such that for all $1 \le i < m$, $C_i \to C_{i+1}$ and $C_m \in \mathsf{NF}$.

As we will see, the limitation to finite sequences of computations is perfectly reasonable. Indeed, we will prove (as a byproduct of polytime soundness, Section 8) that SQ is strongly normalizing.

In the concrete realization of quantum algorithms, the initial quantum register is empty (it will be created during the computation). With this hypothesis, configurations in a computation can be proved to have a certain regular shape:

Proposition 2. *Let $\{[\mathcal{Q}_i, \mathcal{QV}_i, M_i]\}_{1 \le i \le m}$ be a computation, such that $\mathcal{QV}_1 = \emptyset$. Then for every $i \le m$ we have $\mathcal{QV}_i = \mathbf{Q}(M_i)$.*

Proof. The statement can be proved by induction on m, using in particular Theorem 2. Indeed, if $m > 1$ (the base case is trivial):

$$\begin{aligned} \mathcal{QV}_m &= (\mathcal{QV}_m - \mathcal{QV}_{m-1}) \cup \mathcal{QV}_{m-1} = (\mathbf{Q}(M_m) - \mathbf{Q}(M_{m-1})) \cup \mathcal{QV}_{m-1} \\ &= (\mathbf{Q}(M_m) - \mathbf{Q}(M_{m-1})) \cup \mathbf{Q}(M_{m-1}) = \mathbf{Q}(M_m) \end{aligned}$$

This concludes the proof. □

In the rest of the paper, $[\mathcal{Q}, M]$ denotes the configuration $[\mathcal{Q}, \mathbf{Q}(M), M]$.

4.2. Confluence

In [9] we have shown that Q enjoys confluence. It is immediate to observe that SQ is a subcalculus of Q (each well-formed configuration of SQ is also a well-formed configuration of Q, and moreover SQ and Q have the same reduction rules). The just stated subject reduction theorem is therefore sufficient to ensure that *confluence* still holds for SQ (as for simply typed λ-calculus, where confluence is a direct consequence of confluence of pure λ-calculus).

Theorem 3 (*Confluence*). *Let C, D, E be configurations with $C \xrightarrow{*} D$ and $C \xrightarrow{*} E$. Then, there is a configuration F with $D \xrightarrow{*} F$ and $E \xrightarrow{*} F$.*

Moreover, as a consequence of having adopted the so-called *surface reduction*, (i.e. it is not possible to reduce inside a subterm in the form $!M$) it is not possible to erase a diverging term (see also [29]). Therefore it is possible to show that:

Theorem 4. *A configuration C is strongly normalizing iff C is weakly normalizing.*

In any case such a result will be superseded by Theorem 6: in Section 8, we prove that *any* configuration is in fact strongly normalizing.

The proofs of Theorems 3 and 4 can be found in Appendix B.

5. Standardizing computations

Another interesting property, that SQ inherits from Q [9] is *quantum standardization*. In this section we will recall its definition and the main ingredients needed to prove it (a detailed proof can be found in Appendix C). The idea underlying quantum standardization is simple: for each computation there is an equivalent *standard* computation that:

- first performs **classical reductions** (namely reductions not involving neither the quantum register nor quantum variables). We could think that this phase is responsible for the construction of a λ-term (abstractly) representing a quantum circuit;
- secondly, **builds the quantum register** (by means of the new reductions);
- and finally, execute **quantum reductions** (as if the quantum circuit abstractly built in the first phase were applied to the quantum register built in phase two).

First of all, we define precisely what we mean by *standard computation*. We distinguish three subsets of $\mathscr{L}$, namely $\mathscr{Q} = \{\mathsf{Uq}, \mathsf{q}.\beta\}$, $n\mathscr{C} = \mathscr{Q} \cup \{\mathsf{new}\}$, and $\mathscr{C} = \mathscr{L} - n\mathscr{C}$. Let $C \to_\alpha D$ and let M be the relevant redex in C; if $\alpha \in \mathscr{Q}$ the redex M is called *quantum*, if $\alpha \in \mathscr{C}$ the redex M is called *classical*.

Definition 6. A configuration C is called *non-classical* if $\alpha \in n\mathscr{C}$ whenever $C \to_\alpha D$. Let NCL be the set of non classical configurations. A configuration C is called *essentially quantum* if $\alpha \in \mathscr{Q}$ whenever $C \to_\alpha D$. Let EQT be the set of essentially quantum configurations.

We define the notion of *standard* computation, that we call CNQ. A CNQ computation is a computation such that any new reduction is always performed after any classical reduction and any quantum reduction is always performed after any new reduction:

Definition 7. A CNQ computation is a computation $\{C_i\}_{1 \le i \le m}$ such that

1. for every $1 < i < m$, if $C_{i-1} \to_{n\mathscr{C}} C_i$ then $C_i \to_{n\mathscr{C}} C_{i+1}$;
2. for every $1 < i < m$, if $C_{i-1} \to_{\mathscr{Q}} C_i$ then $C_i \to_{\mathscr{Q}} C_{i+1}$.

Quantum standardization takes the form of the following theorem, which guarantees the existence of an equivalent CNQ computation for every computation:

Theorem 5 (*Quantum Standardization*). *For every computation $\{C_i\}_{1 \le i \le m}$ there is a* CNQ *computation $\{D_i\}_{1 \le i \le n}$ such that $C_1 \equiv D_1$ and $C_m \equiv D_n$.*

The proof of Theorem 5 proceeds by first showing that NCL is closed under $\to_{\mathscr{Q}}$ and that EQT is closed under $\to_{\mathsf{new}}$. Detailed proofs of Theorem 5 and related lemmas can be found in Appendix C.

6. Encoding classical data structures

Classically, SQ has the expressive power of soft linear logic. The aim of this Section is to illustrate some encodings of usual data structures (natural numbers, binary strings and lists). Notice that some of the encodings we are going to present are non-standard: they are not the usual Church-style encodings, which are not necessarily available in a restricted setting like the one we are considering here. The results in this section will be essential in Section 9.

In order to simplify the treatment we will consider reduction between terms rather than between configurations. If $[\mathcal{Q}, \mathcal{QV}, M] \to_\alpha [\mathcal{R}, \mathcal{RV}, N]$, then we will simply write $M \to_\alpha N$. This is sensible, since N only depends on M (and does not depend on $\mathcal{Q}$ nor on $\mathcal{QV}$).

6.1. Natural numbers

We need to stay as abstract as possible here: there will be many distinct terms representing the same natural number. Given a natural number $n \in \mathbb{N}$ and a term M, the class of *n-banged forms of M* is defined by induction on n:

- The only 0-banged form of M is M itself;
- If N is a n-banged form of M, any term $!L$ where $L \to^*_{\mathcal{N}} N$ is an $n+1$-banged form of M.

Let $!^n M$ denotes $\underbrace{!(!\ldots(!}_{n \text{ times}} M)\ldots)$. Please notice that $!^n M$ is an n-banged form of M for every $n \in \mathbb{N}$ and for every term M.

Given natural numbers $n, m \in \mathbb{N}$, a term M is said to *n-represent* the natural number m iff for every n-banged form L of N

$$ML \to_{\mathcal{N}} \lambda z.\underbrace{N(N(N(\ldots(N}_{m \text{ times}} z)\ldots))).$$

A term M is said to *(n,k)-represent* a function $f : \mathbb{N} \to \mathbb{N}$ iff for every natural number $m \in \mathbb{N}$, for every term N which 1-represents m, and for every n-banged form L of N

$$ML \to^*_{\mathcal{N}} P$$

where P k-represents $f(m)$.

For every natural number $m \in \mathbb{N}$, let $\lceil m \rceil$ be the term

$$\lambda!x.\lambda y.\underbrace{x(x(x(\ldots(x}_{m \text{ times}} y)\ldots))).$$

For every m, $\lceil m \rceil$ 1-represents the natural number m.

For every natural number $m \in \mathbb{N}$ and every positive natural number $n \in \mathbb{N}$, let $\lceil m \rceil^n$ be the term defined by induction on n:

$$\lceil m \rceil^0 = \lceil m \rceil;$$
$$\lceil m \rceil^{n+1} = \lambda!x.\lceil m \rceil^n x.$$

For every n, m, $\lceil m \rceil^n$ can be proved to $n+1$-represent the natural number m.

Lemma 1. *Let $id : \mathbb{N} \to \mathbb{N}$ be the identity function. For every natural number n, there is a term M^n_{id} which $(n,1)$-represents id. Moreover, for every $m \in \mathbb{N}$ and for every term N, $M^n_{id}!^{n+m}N \to^*_{\mathcal{N}} !^m N$.*

Proof. By induction on n:

- $M^0_{id} = \lambda x.x$. Indeed, for every N 1-representing $m \in \mathbb{N}$ and for every 0-banged form L of N:

$$M^0_{id}L = M^0_{id}N = (\lambda x.x)N \to_{\mathcal{N}} N.$$

- $M^{n+1}_{id} = \lambda!x.M^n_{id}x$. Indeed, for every N 1-representing $m \in \mathbb{N}$ and for every $n+1$-banged form L of N:

$$M^{n+1}_{id}L = M^{n+1}_{id}!P = (\lambda!x.M^n_{id}x)!P \to_{\mathcal{N}} M^n_{id}P \to^*_{\mathcal{N}} M^n_{id}Q \to^*_{\mathcal{N}} R$$

where Q is an n-banged form of N and R 1-represents m.

This concludes the proof. □

SOQ can compute any polynomial, in a strong sense:

Proposition 3. *For any polynomial with natural coefficients $p : \mathbb{N} \to \mathbb{N}$ of degree n, there is a term M_p that $(2n+1, 2n+1)$-represents p.*

Proof. Any polynomial can be written as an *Horner's polynomial*, which is either:

- The constant polynomial $x \mapsto k$, where $k \in \mathbb{N}$ does not depend on x.
- Or the polynomial $x \mapsto k + x \cdot p(x)$, where $k \in \mathbb{N}$ does not depend on x and $p : \mathbb{N} \to \mathbb{N}$ is itself an Horner's polynomial.

So, proving that the thesis holds for Horner's polynomials suffices. We go by induction, following the above recursion schema:

- Any constant polynomial $p : \mathbb{N} \to \mathbb{N}$ in the form $x \mapsto k$ is $(1,1)$-representable. Just take. $M_p = \lambda!x.\lceil k \rceil$. Indeed:

$$M_p!N \to \lceil k \rceil.$$

- Suppose $r : \mathbb{N} \to \mathbb{N}$ is a polynomial of degree n which can be $(2n+1, 2n+1)$-represented by M_r. Suppose $k \in \mathbb{N}$ and let $p : \mathbb{N} \to \mathbb{N}$ be the polynomial $x \mapsto k + x \cdot r(x)$. Consider the term

$$M_p = \lambda!x.\lambda!y.\lambda z.(\lceil k \rceil^{2n+2} y)((M_{id}^{2n+2} x)((\lambda!w.\lambda!u.!(M_r w u))xy)z)$$

Let now N be a term 1-representing a natural number i, L be any term, $!P$ be any $(2n+3)$-banged form of N and $!Q$ be any $(2n+3)$-banged form of L. Then

$$\begin{aligned}
M_p!P!Q &\to^*_{\mathcal{M}} \lambda z.(\lceil k \rceil^{2n+2} Q)((M_{id}^{2n+2} P)((\lambda!w.\lambda!u.!(M_r w u))PQ)z) \\
&\to^*_{\mathcal{M}} \lambda z.(\lceil k \rceil^{2n+2} Q)((M_{id}^{2n+2} P)((\lambda!w.\lambda!u.!(M_r w u))!R!S)z) \\
&\to^*_{\mathcal{M}} \lambda z.(\lceil k \rceil^{2n+2} !S)((M_{id}^{2n+2} !R)!(M_r RS)z) \\
&\to^*_{\mathcal{M}} \lambda z.(\lambda z.\underbrace{L(L(\ldots(L}_{k \text{ times}} z)\ldots))) \\
&\qquad (V!(M_r RS)z) \\
&\to^*_{\mathcal{M}} \lambda z.(\lambda z.\underbrace{L(L(\ldots(L}_{k \text{ times}} z)\ldots))) \\
&\qquad (\lambda z.\underbrace{(M_r RS)(\ldots((M_r RS)}_{i \text{ times}} z)\ldots))) \\
&\to^*_{\mathcal{M}} \lambda z.(\lambda z.\underbrace{L(L(\ldots(L}_{k \text{ times}} z)\ldots))) \\
&\qquad (\lambda z.\underbrace{(M_r TU)(\ldots((M_r TU)}_{i \text{ times}} z)\ldots))) \\
&\to^*_{\mathcal{M}} \lambda z.(\lambda z.\underbrace{L(L(\ldots(L}_{k+ir(i) \text{ times}} z)\ldots)))z \\
&\to_{\mathcal{M}} (\lambda z.\underbrace{L(L(\ldots(L}_{k+ir(i) \text{ times}} z)\ldots)))
\end{aligned}$$

where V 1-represents i, $!R$ is a $(2n+2)$-banged form of N, $!S$ is a $(2n+2)$-banged form of L, T is a $(2n+1)$-banged form of N and U is a $(2n+1)$-banged form of L. This concludes the proof. □

6.2. Strings

Other than natural numbers, we are interested in representing strings in an arbitrary (finite) alphabet. Given any string $s = b_1 \ldots b_n \in \Sigma^*$ (where Σ is a finite alphabet), the term $\lceil s \rceil^{\Sigma}$ is the following:

$$\lambda!x_{a_1}.\ldots.\lambda!x_{a_m}.\lambda!y.\lambda z.yx_{b_1}(yx_{b_2}(yx_{b_3}(\ldots(yx_{b_n}z)\ldots))).$$

where $\Sigma = \{a_1, \ldots, a_m\}$. Consider the term

$$\mathsf{strtonat}_{\Sigma} = \lambda x.\lambda!y.\lambda z.x\underbrace{!!(\lambda w.w)\ldots!!(\lambda w.w)}_{m \text{ times}}!(\lambda!w.\lambda r.yr)z.$$

As can be easily shown, $\mathsf{strtonat}_{\Sigma}\lceil b_1 \ldots b_n \rceil^{\Sigma}$ rewrites to a term N 1-representing n:

$$\begin{aligned}
\mathsf{strtonat}_{\Sigma}\lceil b_1 \ldots b_n \rceil^{\Sigma}!L &\to^*_{\mathcal{M}} \lambda z.\lceil b_1 \ldots b_n \rceil^{\Sigma} \underbrace{!!(\lambda w.w)\ldots!!(\lambda w.w)}_{m \text{ times}}!(\lambda!w.\lambda r.Lr)z \\
&\to^*_{\mathcal{M}} \lambda z.(\lambda!w.\lambda r.Lr)!(\lambda w.w)((\lambda!w.\lambda r.Lr)!(\lambda w.w)((\lambda!w.\lambda r.Lr)!(\lambda w.w) \\
&\qquad (\ldots((\lambda!w.\lambda r.Lr)!(\lambda w.w)z)\ldots))) \\
&\to^*_{\mathcal{M}} (\lambda z.\underbrace{L(L(\ldots(L}_{n \text{ times}} z)\ldots)))
\end{aligned}$$

6.3. Lists

Lists are the obvious generalization of strings where an infinite number of constructors is needed. Given a sequence $M_1, \ldots, M_n$ of terms (with no free variable in common), we can build a term $[M_1, \ldots, M_n]$ encoding the sequence as follows, by induction on n:

$$\begin{aligned} [] &= \lambda!x.\lambda!y.y; \\ [M, M_1, \ldots, M_n] &= \lambda!x.\lambda!y.xM[M_1, \ldots, M_n]. \end{aligned}$$

This way we can construct and destruct lists in a principled way: the terms cons and sel can be built as follows:

$$\begin{aligned} \mathsf{cons} &= \lambda z.\lambda w.\lambda!x.\lambda!y.xzw; \\ \mathsf{sel} &= \lambda x.\lambda y.\lambda z.xyz. \end{aligned}$$

They behave as follows on lists:

$$\begin{aligned} \mathsf{cons}M[M_1, \ldots, M_n] &\to^*_{\mathscr{N}} [M, M_1, \ldots, M_n]; \\ \mathsf{sel}[]!N!L &\to^*_{\mathscr{N}} L; \\ \mathsf{sel}[M, M_1, \ldots, M_n]!N!L &\to^*_{\mathscr{N}} NM[M_1, \ldots, M_n]. \end{aligned}$$

By exploiting cons and sel, we can build more advanced constructors and destructors: for every natural number n there are the terms append_n and $\mathsf{extract}_n$ behaving as follows:

$$\begin{aligned} \mathsf{append}_n[N_1, \ldots, N_m]M_1, \ldots, M_n &\to^*_{\mathscr{N}} [M_1, \ldots, M_n, N_1, \ldots, N_m]; \\ \forall m \leq n.\mathsf{extract}_n M[N_1, \ldots, N_m] &\to^*_{\mathscr{N}} M[]N_m N_{m-1} \ldots N_1; \\ \forall m \geq n.\mathsf{extract}_n M[N_1, \ldots N_m] &\to^*_{\mathscr{N}} M[N_{n+1} \ldots N_m]N_n N_{n-1} \ldots N_1. \end{aligned}$$

Terms append_n can be built by induction on n:

$$\begin{aligned} \mathsf{append}_0 &= \lambda x.x; \\ \mathsf{append}_{n+1} &= \lambda x.\lambda y_1.\ldots.\lambda y_{n+1}.\mathsf{cons}\ y_{n+1}(\mathsf{append}_n x y_1 \ldots y_n). \end{aligned}$$

Similarly, terms $\mathsf{extract}_n$ can be built inductively:

$$\begin{aligned} \mathsf{extract}_0 &= \lambda x.\lambda y.xy; \\ \mathsf{extract}_{n+1} &= \lambda x.\lambda y.(\mathsf{sel}y!(\lambda z.\lambda w.\lambda v.\mathsf{extract}_n v w z)!(\lambda z.z[]))x. \end{aligned}$$

Indeed:

$$\begin{aligned} \forall m.\mathsf{extract}_0 M[N_1, \ldots, N_m] &\to^*_{\mathscr{N}} M[N_1, \ldots, N_m]; \\ \forall n.\mathsf{extract}_{n+1} M[] &\to^*_{\mathscr{N}} M[]; \\ \forall m < n.\mathsf{extract}_{n+1} M[N, N_1 \ldots N_m] &\to^*_{\mathscr{N}} \mathsf{extract}_n M[N_1, \ldots, N_m]N \\ &\to^*_{\mathscr{N}} M[]N_m \ldots N_1 N; \\ \forall m \geq n.\mathsf{extract}_{n+1} M[N, N_1 \ldots N_m] &\to^*_{\mathscr{N}} \mathsf{extract}_n M[N_1, \ldots, N_m]N \\ &\to^*_{\mathscr{N}} M[N_{n+1} \ldots N_m]N_n \ldots N_1 N; \end{aligned}$$

7. Representing decision problems

We now need to understand how to represent subsets of $\{0, 1\}^*$ in SQ. Some preliminary definitions are needed.

A term M *outputs the binary string* $s \in \{0, 1\}^*$ *with probability* p *on input* N iff there is $m \geq |s|$ such that

$$[1, \emptyset, MN] \xrightarrow{*} [\mathcal{Q}, \{q_1, \ldots, q_m\}, [q_1, \ldots, q_m]]$$

and the probability of observing s when projecting $\mathcal{Q}$ into the subspace $\mathcal{H}(\{q_{|s|+1}, \ldots, q_m\})$ is precisely p.

Given $n \in \mathbb{N}$, two binary strings $s, r \in \{0, 1\}^k$ and a probability $p \in [0, 1]$, a term M is said to (n, s, r, p)-*decide a language* $L \subseteq \{0, 1\}^*$ iff the following two conditions hold:

- M outputs the binary string s with probability at least p on input $!^n\lceil t \rceil^{\{0,1\}}$ whenever $t \in L$;
- M outputs the binary string r with probability at least p on input $!^n\lceil t \rceil^{\{0,1\}}$ whenever $t \notin L$.

With the same hypothesis, M is said to be *error-free (with respect to* (n, s, r)*)* iff for every binary string t, the following two conditions hold:

- If M outputs s with positive probability on input $!^n\lceil t \rceil^{\{0,1\}}$, then M outputs r with null probability on the same input;
- Dually, if M outputs r with positive probability on input $!^n\lceil t \rceil^{\{0,1\}}$, then M outputs s with null probability on the same input.

Definition 8. Three classes of languages in the alphabet $\{0, 1\}$ are defined below:

1. ESQ is the class of languages which can be $(n, s, r, 1)$-decided by a term M of SQ;
2. BSQ is the class of languages which can be (n, s, r, p)-decided by a term M of SQ, where $p > \frac{1}{2}$;
3. ZSQ is the the class of languages which can be (n, s, r, p)-decided by an error-free (wrt (n, s, r)) term M of SQ, where $p > \frac{1}{2}$.

The purpose of the following two sections is precisely proving that ESQ, BSQ and ZSQ coincide with the quantum complexity classes EQP, BQP ad ZQP, respectively.

8. Polytime soundness

In this Section we assume that *all the involved terms are well formed*.

Following the approach proposed by Girard in [16] and subsequently developed in [1,17,3] we show that SQ is intrinsically a polytime calculus. This allows to show that decision problems which can be represented in SQ lie in certain polytime (quantum) complexity classes.

We start with some definitions. We distinguish two particular subsets of $\mathscr{L}$, namely $\mathscr{K} = \{\text{r.cm}, \text{l.cm}\}$ and $\mathscr{N} = \mathscr{L} - \mathscr{K}$. Reduction rules in $\mathscr{K}$ are the so-called *commuting* rules. The *size* of a term is defined in a standard way as:

$$\begin{aligned} |x| = |r| = |C| &= 1 \\ |!N| &= |N| + 1 \\ |\mathsf{new}(P)| &= |P| + 1 \\ |PQ| &= |P| + |Q| + 1 \\ |\langle M_1, \ldots, M_k\rangle| &= |M_1| + \cdots + |M_k| + 1 \\ |\lambda x.N| = |\lambda !x.N| = |\lambda\langle x_1, \ldots, x_k\rangle.N| &= |N| + 1 \end{aligned}$$

A term cannot contain more occurrences (of a variable) than its size:

Lemma 2. *For every term M and for every variable x,* $\mathsf{NFO}(x, M) \leq |M|$.

Proof. By induction on M. □

Commuting reduction steps do not alter the size of the underlying term. But on the other hand, $\rightarrow_{\mathscr{K}}$ is strongly normalizing:

Lemma 3. *If $M \overset{n}{\rightarrow}_{\mathscr{K}} N$, then* (i) $|M| = |N|$; (ii) $n \leq |M|^2$.

Proof. (i) By induction on the derivation of $M \rightarrow_{\mathscr{K}} N$. Observe that $|L((\lambda\pi.R)S)| = |(\lambda\pi.LR)S|$ and $|((\lambda\pi.R)S)L| = |(\lambda\pi.RL)S|$; these are base cases in which $L((\lambda\pi.R)S \rightarrow_{\text{l.cm}} (\lambda\pi.LR)S$ or $((\lambda\pi.R)S)L \rightarrow_{\text{r.cm}} (\lambda\pi.RL)S$. We have context closures as inductive steps. For example, let M be LP and let be

$$\frac{P \rightarrow_{\mathscr{K}} Q}{LP \rightarrow_{\mathscr{K}} LQ}\ \text{l.a}$$

the last rule in the derivation. By induction hypothesis we have $|P| = |Q|$, and $|M| = |LP| = |L|+|P|+1 = |L|+|Q|+1 = |LQ|$. The other cases are very similar to the previous one.

(ii) Define the abstraction size $|M|_\lambda$ of M as the sum over all subterms of M in the form $\lambda\pi.L$, of $|L|$. Clearly $|M|_\lambda \leq |M|^2$. Moreover, $n \leq |M|_\lambda$ because

$$\begin{aligned} |L((\lambda\pi.R)S)|_\lambda &< |(\lambda\pi.LR)S|_\lambda \\ |((\lambda\pi.R)S)L|_\lambda &< |(\lambda\pi.RL)S|_\lambda \end{aligned}$$

In other words, $|M|_\lambda$ always increases along commuting reduction.

This concludes the proof. □

In order to prove polytime soundness of the calculus it is useful to assign to each term M three quantities $\mathsf{B}(M)$, $\mathsf{D}(M)$ and $\mathsf{W}(M)$. $\mathsf{B}(M)$ is simply the maximum nesting depth of ! operators inside M. $\mathsf{D}(M)$ is the maximum number of occurrences of a bound variable in M. Finally, $\mathsf{W}(M)$ is the *weight* of M, namely a quantity that we will use to show polytime soundness.

More formally:

Definition 9 (*Box-Depth, Duplicability-Factor, Weights*). 1. The *box-depth* $\mathsf{B}(M)$ of M (the maximum between the number of !-terms nesting in M) is defined as

$$\begin{aligned}\mathsf{B}(x) = \mathsf{B}(r) = \mathsf{B}(C) &= 0\\ \mathsf{B}(!N) &= \mathsf{B}(N)+1\\ \mathsf{B}(\mathsf{new}(N)) &= \mathsf{B}(N)\\ \mathsf{B}(PQ) &= \max\{\mathsf{B}(P),\mathsf{B}(Q)\}\\ \mathsf{B}(\langle M_1,\ldots,M_k\rangle) &= \max\{\mathsf{B}(M_1),\ldots,\mathsf{B}(M_k)\}\\ \mathsf{B}(\lambda x.N) = \mathsf{B}(\lambda !x.N) = \mathsf{B}(\lambda\langle x_1,\ldots,x_k\rangle.N) &= \mathsf{B}(N);\end{aligned}$$

2. the *duplicability-factor* $\mathsf{D}(M)$ of M (the maximum between the number of occurrence of variables bound by a λ) is defined as

$$\begin{aligned}\mathsf{D}(x) = \mathsf{D}(r) = \mathsf{D}(C) &= 1\\ \mathsf{D}(!N) &= \mathsf{D}(N)\\ \mathsf{D}(\mathsf{new}(N)) &= \mathsf{D}(N)\\ \mathsf{D}(PQ) &= \max\{\mathsf{D}(P),\mathsf{D}(Q)\}\\ \mathsf{D}(\langle M_1,\ldots,M_k\rangle) &= \max\{\mathsf{D}(M_1),\ldots,\mathsf{D}(M_k)\}\\ \mathsf{D}(\lambda x.N) = \mathsf{D}(\lambda !x.N) &= \max\{\mathsf{D}(N),\mathsf{NFO}(x,N)\}\\ \mathsf{D}(\lambda\langle x_1,\ldots,x_k\rangle.N) &= \max\{\mathsf{D}(N),\mathsf{NFO}(x_1,N),\ldots,\mathsf{NFO}(x_k,N)\};\end{aligned}$$

3. the *n-weight* $\mathsf{W}_n(M)$ of M (the weight of a term with respect to n) is defined as

$$\begin{aligned}\mathsf{W}_n(x) = \mathsf{W}_n(r) = \mathsf{W}_n(C) &= 1\\ \mathsf{W}_n(!N) &= n\cdot\mathsf{W}_n(N)+1\\ \mathsf{W}_n(\mathsf{new}(N)) &= \mathsf{W}_n(N)+1\\ \mathsf{W}_n(PQ) &= \mathsf{W}_n(P)+\mathsf{W}_n(Q)+1\\ \mathsf{W}_n(\langle M_1,\ldots,M_k\rangle) &= \mathsf{W}_n(M_1)+\cdots+\mathsf{W}_n(M_k)+1\\ \mathsf{W}_n(\lambda x.N) = \mathsf{W}_n(\lambda !x.N) = \mathsf{W}_n(\lambda\langle x_1,\ldots,x_k\rangle.N) &= \mathsf{W}_n(N)+1;\end{aligned}$$

4. the *weight* of a term M is defined as $\mathsf{W}(M) = \mathsf{W}_{\mathsf{D}(M)}(M)$.

The strategy we will use to prove polytime soundness consists in proving three intermediate results:

- First of all, by means of several intermediate results, we will show that $\mathsf{W}(M)$ is an upper bound of $|M|$.
- Second, $\mathsf{W}(M)$ is shown to strictly decrease at any noncommutative reduction step and it is shown not to increase at any commutative reduction step.
- Lastly, $\mathsf{W}(M)$ is shown to be bounded from above by a polynomial $p(|M|)$, where the exponent of p depends on $\mathsf{B}(M)$ (but not on $|M|$). This implies, by Lemma 3, that the number of reduction steps from M to its normal form is polynomially related to $\mathsf{W}(M)$.

The three results above together imply polytime soundness.

Before formally stating and proving them, however, we need to show a few auxiliary lemmas about $\mathsf{B}(\cdot)$, $\mathsf{D}(\cdot)$, $\mathsf{W}(\cdot)$ and $\mathsf{NFO}(\cdot,\cdot)$ and how these quantities evolve during reduction. The duplicability-factor of a term cannot be bigger than the size of the term itself:

Lemma 4. *For every term M, $\mathsf{D}(M) \le |M|$.*

Proof. By induction on M:

- M is a variable or a constant or a quantum variable; then $\mathsf{D}(M) = 1 = |M|$.
- M is of the form $\lambda x.N$. Then:

$$\begin{aligned}\mathsf{D}(\lambda x.N) &= \max\{\mathsf{D}(N),\mathsf{NFO}(x,N)\}\\ &\overset{IH}{\le} \max\{|N|,\mathsf{NFO}(x,N)\}\\ &\le \max\{|N|,|N|\}\\ &= |N| \le |N|+1 = |M|.\end{aligned}$$

- M is of the form $\lambda !x.N$ or $\lambda\pi.N$: very similar to the previous case.

- M is PQ. Then:

$$\begin{aligned}\mathsf{D}(PQ) &= \max\{\mathsf{D}(P), \mathsf{D}(Q)\} \overset{IH}{\leq} \max\{|P|, |Q|\} \\ &\leq \max\{|P| + |Q| + 1, |P| + |Q| + 1\} \\ &= |P| + |Q| + 1 = |PQ|.\end{aligned}$$

- M is $\mathsf{new}(N)$:

$$\mathsf{D}(\mathsf{new}(N)) = \mathsf{D}(N) \overset{IH}{\leq} |N| < |N| + 1 = |M|.$$

- M is $!N$, then

$$\mathsf{D}(!N) = \mathsf{D}(N) \overset{IH}{\leq} |N| < |N| + 1 = |M|.$$

- M is $\langle N_1, \ldots, N_k\rangle$ and for all N_i, $\quad i = 1 \ldots k$, we have $\mathsf{D}(N_i) \leq |N_i|$ by induction hypothesis; then

$$\begin{aligned}\mathsf{D}(M) &= \max\{\mathsf{D}(N_1), \ldots, \mathsf{D}(N_k)\} \overset{IH}{\leq} \max\{|N_1|, \ldots, |N_k|\} \\ &< |N_1| + \cdots + |N_k| + 1 = |M|.\end{aligned}$$

This concludes the proof. □

The number of free occurrences of a variable cannot increase too much during reduction:

Lemma 5. *If $P \to_{\mathscr{L}} Q$ then $\max\{\mathsf{NFO}(x, P), \mathsf{D}(P)\} \geq \mathsf{NFO}(x, Q)$.*

Proof. The proof proceeds by proving the following facts:

1. if $\triangleright \Gamma, x \vdash P$ and $P \to_{\mathscr{L}} Q$ then $\mathsf{NFO}(x, P) \geq \mathsf{NFO}(x, Q)$;
2. if $\triangleright \Gamma, \#x \vdash P$ and $P \to_{\mathscr{L}} Q$ then $\mathsf{NFO}(x, P) \geq \mathsf{NFO}(x, Q)$;
3. if $\triangleright \Gamma, !x \vdash P$ and $P \to_{\mathscr{L}} Q$ then $\max\{\mathsf{NFO}(x, P), \mathsf{D}(P)\} \geq \mathsf{NFO}(x, Q)$.

The lemma is therefore a trivial consequence of the above facts. The proofs of 1., 2. and 3. are simple inductions on the derivation of $P \to_{\mathscr{L}} Q$. We will show here only some interesting cases.

1. We distinguish two cases:
   - If the last rule is a base rule, we have several sub-cases. If the reduction rule is $(\lambda !y.L)!M \to_{\mathrm{c}.\beta} L\{M/y\}$, please observe that $\mathsf{NFO}(x, !M) = 0$ and conclude. If the reduction rule is $(\lambda y.L)M \to_{\mathrm{l}.\beta} L\{M/y\}$, we have only two possibilities: either $\mathsf{NFO}(x, M) = 0$ and $\mathsf{NFO}(x, L) = 1$ or $\mathsf{NFO}(x, M) = 1$ and $\mathsf{NFO}(x, L) = 0$; in both cases the conclusion is immediate. The other sub-cases are easier.
   - If the last reduction rule is a context closure rules, the result follows easily by applying the induction hypothesis. For example, if the closure rule is
$$\frac{M \to N}{ML \to NL}$$
   we have two sub-cases: either $\mathsf{NFO}(x, P_1) = 0$ and $\mathsf{NFO}(x, P_2) = 1$ or $\mathsf{NFO}(x, P_1) = 1$ and $\mathsf{NFO}(x, P_2) = 0$. In the first sub-case the thesis follow immediately. In the second sub-case the result follows by applying the induction hypothesis $\mathsf{NFO}(x, M) \geq \mathsf{NFO}(x, L)$.
2. We distinguish two cases:
   - If the last rule is a base rule, we have several sub-cases If the reduction rule is $(\lambda !y.L)!M \to_{\mathrm{c}.\beta} L\{M/y\}$, please observe that $\mathsf{NFO}(x, !M) = 0$ and conclude. If the reduction rule is $(\lambda y.L)M \to_{\mathrm{l}.\beta} L\{M/y\}$, simply observe that y must occur exactly once in L and therefore $\mathsf{NFO}(x, (\lambda y.L)M) = \mathsf{NFO}(x, M) + \mathsf{NFO}(x, L) = \mathsf{NFO}(x, L\{M/y\})$. All the other base cases can be easily proved.
   - If the last reduction rule is a context closure rule, the result follows easily by applying the induction hypothesis. For example if the reduction rule is
$$\frac{M \to N}{ML \to NL}$$
   we have two sub-cases: (i) $\mathsf{NFO}(x, M) = 0$; in this case the thesis follow immediately; (ii) $\mathsf{NFO}(x, M) \neq 0$; the result follows by applying the induction hypothesis $\mathsf{NFO}(x, M) \geq \mathsf{NFO}(x, L)$.
3. The proof remains simple but it slightly more delicate, because we must consider the phenomenon of duplication.

- If the last rule is a base rule: we have several cases. If the reduction rule is $(\lambda!y.L)!M \to_{c.\beta} L\{M/y\}$, please observe that differently from the previous facts, we have two possibilities: if $\mathsf{NFO}(x, !M) = 0$ we conclude; otherwise, if $\mathsf{NFO}(x, !M) \neq 0$ we must have that $\mathsf{NFO}(x, !M) = 1$ and $\mathsf{NFO}(x, L) = 0$. Consequently

$$\begin{aligned}\max\{\mathsf{NFO}(x,P),\mathsf{D}(P)\} &= \mathsf{D}(P) = \max\{\mathsf{D}(\lambda!y.L)),\mathsf{D}(!M)\}\\ &\geq \mathsf{D}(\lambda!y.L) \geq \max\{D(L),\mathsf{NFO}(y,L)\}\\ &\geq \mathsf{NFO}(y,L)\\ &= \mathsf{NFO}(x,L) + \mathsf{NFO}(y,L)\cdot\mathsf{NFO}(x,M)\\ &= \mathsf{NFO}(x,L\{M/y\}).\end{aligned}$$

  If the reduction rule is $(\lambda y.L)M \to_{l.\beta} L\{M/y\}$, simply observe that y must occur exactly once in L and therefore $\mathsf{NFO}(x,(\lambda y.L)M) = \mathsf{NFO}(x,M) + \mathsf{NFO}(x,L) = \mathsf{NFO}(x,L\{M/y\})$.
  All the other base case are easily proved.
- if the last reduction rule is a context closure rules, the result follows easily by applying the induction hypothesis. For example if the reduction rule is

$$\frac{M \to N}{ML \to NL}$$

  we have two cases: (i) $\mathsf{NFO}(x,M) = 0$; in this case the thesis follows immediately; (ii) $\mathsf{NFO}(x,M) = 1$ and $\mathsf{NFO}(x,L) = 0$ the result follows by applying the induction hypothesis $\max\{\mathsf{NFO}(x,M),\mathsf{D}(M)\} \geq \mathsf{NFO}(x,N)$:

$$\begin{aligned}\max\{\mathsf{NFO}(x,ML),\mathsf{D}(ML)\} &= \max\{\mathsf{NFO}(x,M)+\mathsf{NFO}(x,L),\mathsf{D}(M),\mathsf{D}(L)\}\\ &= \max\{\mathsf{NFO}(x,M),\mathsf{D}(M),\mathsf{D}(L)\}\\ &\geq \max\{\mathsf{NFO}(x,M),\mathsf{D}(M)\}\\ &\geq \mathsf{NFO}(x,N) = \mathsf{NFO}(x,NL).\end{aligned}$$

This concludes the proof. □

The duplicability factor of a term obtained by substitution is bounded by (the maximum of) the duplicability factors of the involved terms:

Lemma 6. *For all terms P and Q, $\mathsf{D}(P\{Q/x\}) \leq \max\{\mathsf{D}(P),\mathsf{D}(Q)\}$.*

Proof. By induction on the term P. □

It is now possible to show that $\mathsf{D}(\cdot)$ does not increase during reduction:

Lemma 7. (i) *If $M \to_{\mathscr{K}} N$ then $\mathsf{D}(M) = \mathsf{D}(N)$;*
(ii) *If $M \to_{\mathscr{N}} N$ then $\mathsf{D}(M) \geq \mathsf{D}(N)$.*

Proof. (i) By induction on the derivation of $\to_{\mathscr{K}}$. For the base cases, if $M \to_{\mathsf{l.cm}} N$, M is of the form $L((\lambda\pi.M_1)M_2)$ and N is $(\lambda\pi.LM_1)M_2$. Observe that, thanks to *BCV* (Section 3.1) relative to l.cm, $\mathsf{NFO}(x_i, LM_1) = \mathsf{NFO}(x_i, M_1)$ for every i. We have:

$$\begin{aligned}\mathsf{D}(M) &= \max\{\mathsf{D}(L),\mathsf{D}((\lambda\pi.M_1)M_2)\}\\ &= \max\{\mathsf{D}(L),\mathsf{D}(\lambda\pi.M_1),\mathsf{D}(N)\}\\ &= \max\{\mathsf{D}(L),\mathsf{D}(M_1),\mathsf{NFO}(x_1,M_1),\ldots,\mathsf{NFO}(x_n,M_1),\mathsf{D}(M_2)\}\\ &= \max\{\mathsf{D}(L),\mathsf{D}(M_1),\mathsf{NFO}(x_i,LM_1),\ldots,\mathsf{NFO}(x_n,LM_1),\mathsf{D}(M_2)\}\\ &= \max\{\mathsf{D}(LM_1),\mathsf{NFO}(x_i,LM_1),\ldots,\mathsf{NFO}(x_n,LM_1),\mathsf{D}(M_2)\}\\ &= \max\{\mathsf{D}(\lambda\pi.LM_1),\mathsf{D}(M_2)\}\\ &= \mathsf{D}((\lambda\pi.LM_1)M_2).\end{aligned}$$

We have context closures as inductive steps. For example, let M be M_1M_2 and let

$$\frac{M_1 \to_{\mathscr{K}} M_3}{M_1M_2 \to_{\mathscr{K}} M_3M_2}\ \mathsf{l.a}$$

be the last rule instance in the derivation. Then:

$$\begin{aligned}\mathsf{D}(M) &= \max\{\mathsf{D}(M_1),\mathsf{D}(M_2)\} \overset{IH}{=} \max\{\mathsf{D}(M_3),\mathsf{D}(M_2)\}\\ &= \mathsf{D}(M_3M_2)\end{aligned}$$

The other cases are very similar to the previous one.
(ii) By induction on the derivation of $\to_{\mathscr{N}}$. We prove some cases depending on the last rule in the derivation.

- The reduction rule is
$$\frac{P_1 \to P_3}{P_1P_2 \to P_3P_2}$$
Then:
$$\begin{aligned}\mathsf{D}(M) &= \max\{\mathsf{D}(P_1), \mathsf{D}(P_2)\}\\ &\overset{IH}{\geq} \max\{\mathsf{D}(P_3), \mathsf{D}(P_2)\} = \max\{\mathsf{D}(P_3P_2)\}\end{aligned}$$
- The reduction rule is
$$\frac{P_2 \to P_3}{P_1P_2 \to P_1P_3}$$
The argument is symmetric to the previous one.
- The reduction rule is $(\lambda!x.P)!Q \to_{\mathsf{c}.\beta} P\{Q/x\}$. Then:
$$\begin{aligned}\mathsf{D}((\lambda!x.P)!Q) &= \max\{\mathsf{D}(P), \mathsf{NFO}(x, P), \mathsf{D}(Q)\}\\ &\geq \max\{\mathsf{D}(P), \mathsf{D}(Q)\} \geq \mathsf{D}(P\{Q/x\})\end{aligned}$$
where the last step is justified by Lemma 6.
- The reduction rule is $(\lambda x.P)Q \to_{\mathsf{l}.\beta} P\{Q/x\}$. Similar to the previous case.
- The reduction rule is $(\lambda\langle x_1, \ldots, x_k\rangle.P)\langle r_1, \ldots, r_k\rangle \to_{\mathsf{q}.\beta} P\{r_1/x_1, \ldots, r_k/x_k\}$. Again similar to the previous case.
- The reduction rule is $U\langle r_1, \ldots, r_k\rangle \to_{\mathsf{Uq}} \langle r_1, \ldots, r_k\rangle$; the result follows by definitions.
- The reduction rule is
$$\frac{M_i \to_\alpha N}{\langle M_1, \ldots, M_i, \ldots, M_k\rangle \to_\alpha \langle M_1, \ldots, N, \ldots, M_k\rangle}.$$
By induction hypothesis we have
$$\begin{aligned}\mathsf{D}(\langle M_1, \ldots, M_i, \ldots, M_k\rangle) &= \max\{\mathsf{D}(M_1), \ldots, \mathsf{D}(M_i), \ldots, \mathsf{D}(M_k)\}\\ &\overset{IH}{\geq} \max\{\mathsf{D}(M_1), \ldots, \mathsf{D}(N), \ldots, \mathsf{D}(M_k)\}\\ &= \mathsf{D}(\langle M_1, \ldots, N, \ldots, M_k\rangle)\end{aligned}$$
- The reduction rule is
$$\frac{P \to_\alpha Q}{\mathsf{new}(P) \to_\alpha \mathsf{new}(Q)}\ \mathsf{in.new}$$
Then:
$$\mathsf{D}(\mathsf{new}(P)) = \mathsf{D}(P) \overset{IH}{\geq} \mathsf{D}(Q) = \mathsf{D}(\mathsf{new}(Q)).$$
- The reduction rule is
$$\frac{P \to_\alpha Q}{\lambda!x.P \to_\alpha \lambda!x.Q}\ \mathsf{in}.\lambda_1$$
Then, by Lemma 5
$$\mathsf{D}(\lambda!x.P) = \max\{\mathsf{D}(P), \mathsf{NFO}(x, P)\} \geq \mathsf{NFO}(x, Q)$$
moreover by the induction hypothesis
$$\mathsf{D}(\lambda!x.P) = \max\{\mathsf{D}(P), \mathsf{NFO}(x, P)\} \geq \mathsf{D}(P) \geq \mathsf{D}(Q)$$
and therefore
$$\mathsf{D}(\lambda!x.P) = \max\{\mathsf{D}(P), \mathsf{NFO}(x, P)\} \geq \max\{\mathsf{D}(Q), \mathsf{NFO}(x, Q)\} = \mathsf{D}(\lambda!x, Q)$$

This concludes the proof. □

It is important to remark that Lemma 7 does not hold for non-well-formed terms. For example let us consider $M = \lambda!x.((\lambda!z.zz)!(xxx))$. We have $M \to N$ where $N = \lambda!x.((xxx)(xxx))$, but $\mathsf{D}(M) = 3$ and $\mathsf{D}(N) = 6$. By the way, M is well-formed in Q [9].

Lemma 8. *For every term M, $|M| \leq \mathsf{W}(M)$.*

Proof. By induction on the term M. In some cases, we will use the following fact: for all term M, for all $n, m \in \mathbb{N}$, $1 \leq m \leq n$, then $\mathsf{W}_m(M) \leq \mathsf{W}_n(M)$.

- M is a variable, a constant or a quantum variable. Then, $|M| = 1 = \mathsf{W}_0(M) = \mathsf{W}_{\mathsf{D}(M)}(M) = \mathsf{W}(M)$.
- M is $!N$. We can proceed as follows:

$$\begin{aligned}|M| &= |N| + 1 \leq \mathsf{W}(N) + 1\\ &= \mathsf{W}_{\mathsf{D}(N)}(N) + 1 = \mathsf{W}_{\mathsf{D}(N)}(!N)\\ &= \mathsf{W}_{\mathsf{D}(!N)}(!N) = \mathsf{W}_{\mathsf{D}(M)}(M) = \mathsf{W}(M).\end{aligned}$$

- M is $\mathsf{new}(N)$; then

$$\begin{aligned}|\mathsf{new}(N)| &= |N| + 1 \overset{IH}{\leq} \mathsf{W}_{\mathsf{D}(N)}(N) + 1\\ &= \mathsf{W}_{\mathsf{D}(N)}(\mathsf{new}(N)) = \mathsf{W}_{\mathsf{D}(\mathsf{new}(N))}(\mathsf{new}(N)) = \mathsf{W}(\mathsf{new}(N)).\end{aligned}$$

- M is PQ; then

$$\begin{aligned}|M| &= |P| + |Q| + 1 \leq \mathsf{W}(P) + \mathsf{W}(Q) + 1\\ &= \mathsf{W}_{\mathsf{D}(P)}(P) + \mathsf{W}_{\mathsf{D}(Q)}(Q) + 1\\ &\leq \mathsf{W}_{\max\{\mathsf{D}(P),\mathsf{D}(Q)\}}(P) + \mathsf{W}_{\max\{\mathsf{D}(P),\mathsf{D}(Q)\}}(Q) + 1\\ &= \mathsf{W}_{\mathsf{D}(PQ)}(P) + \mathsf{W}_{\mathsf{D}(PQ)}(Q) + 1\\ &= \mathsf{W}_{\mathsf{D}(PQ)}(PQ) = \mathsf{W}(PQ) = \mathsf{W}(M).\end{aligned}$$

- M is $\langle N_1, \ldots, N_k\rangle$; then

$$\begin{aligned}|\langle N_1, \ldots, N_k\rangle| &= |N_1| + \cdots + |N_k| + 1\\ &\overset{IH}{\leq} \mathsf{W}_{\mathsf{D}(N_1)}(N_1) + \cdots + \mathsf{W}_{\mathsf{D}(N_k)}(N_k) + 1\\ &\leq \mathsf{W}_{\mathsf{D}(M)}(N_1) + \cdots + \mathsf{W}_{\mathsf{D}(M)}(N_k) + 1\\ &= \mathsf{W}_{\mathsf{D}(M)}(M) = \mathsf{W}(M).\end{aligned}$$

- M is $\lambda x.N$; then

$$|M| = |N| + 1 \overset{IH}{\leq} \mathsf{W}_{\mathsf{D}(N)}(N) + 1 \leq \mathsf{W}_{\mathsf{D}(M)}(N) + 1 = \mathsf{W}(M)$$

  where the last inequality holds observing that $\mathsf{D}(M) = \max\{\mathsf{D}(N), \mathsf{NFO}(x, N)\}$, so $\mathsf{D}(N) \leq \mathsf{D}(M)$.
- M is $\lambda\pi.N$ or M is $\lambda!x.N$: as in the previous case.

This concludes the proof. □

We now need to revisit the Substitution Lemma. In particular, the weight of terms obtained as substitutions can be properly bounded by some simple expressions involving the weight of the involved terms:

Lemma 9 (*Substitution Lemma, Revisited*).

- *Linear case. If* $\triangleright \Psi_1, \#\Delta_1, x \vdash M$ *and* $\triangleright \Psi_2, \#\Delta_2 \vdash N$, *with* $\mathsf{var}(\Psi_1) \cap \mathsf{var}(\Psi_2) = \emptyset$, *then for all* $m, n \in \mathbb{N}$, $n \geq m \geq 1$, $\mathsf{W}_m(M\{N/x\}) \leq \mathsf{W}_n(M) + \mathsf{W}_n(N)$;
- *Contraction case. If* $\triangleright \Gamma, \#x \vdash M$ *and* $\triangleright \Delta \vdash N$, $\mathsf{var}(\Gamma) \cap \mathsf{var}(\Delta) = \emptyset$, *then for all* $m, n \in \mathbb{N}$, $n \geq m \geq 1$, $\mathsf{W}_m(M\{N/x\}) \leq \mathsf{W}_n(M) + \mathsf{NFO}(x, M) \cdot \mathsf{W}_n(N)$;
- *Bang case. If* $\triangleright \Gamma, !x \vdash M$ *and* $\triangleright \Delta \vdash N$, $\mathsf{var}(\Gamma) \cap \mathsf{var}(\Delta) = \emptyset$, *then for all* $m, n \in \mathbb{N}$, $n \geq m \geq 1$, $\mathsf{W}_m(M\{N/x\}) \leq \mathsf{W}_n(M) + n \cdot \mathsf{W}_n(N)$;
- *Quantum case. If* $\triangleright \Gamma, x_1, \ldots, x_k \vdash M$ *and* $\triangleright !\Delta, r_1, \ldots, r_k \vdash \langle r_1, \ldots, r_k\rangle$, $\mathsf{var}(\Gamma) \cap \mathsf{var}(!\Delta) = \emptyset$, *then for all* $m, n \in \mathbb{N}$, $n \geq m \geq 1$, $\mathsf{W}_m(M\{r_1/x_1, \ldots, r_k/x_k\}) \leq \mathsf{W}_n(M)$;

Proof. The four statements can be proved by induction on the structure of the derivation for M. We give here only some cases as examples.

- *Linear case*. For example, if M is x, with $\dfrac{}{!\Delta, x \vdash x}$. We have $\mathsf{W}_m(x\{N/x\}) = \mathsf{W}_m(N) \leq \mathsf{W}_n(x) + \mathsf{W}_n(N)$.
- *Contraction case*. For example, suppose M is PQ. The last rule in the derivation for $\triangleright \Gamma, \#x \vdash M$ must have the following shape:

$$\dfrac{\Psi_1, \#\Delta_1 \vdash P \quad \Psi_2, \#\Delta_2 \vdash Q}{\Psi_1, \Psi_2, \#\Delta_1 \cup \#\Delta_2 \vdash PQ}\,\mathsf{app}$$

Suppose that $\#x$ is both in $\#\Delta_1$ and in $\#\Delta_2$. By induction hypothesis we have $\mathsf{W}_m(P\{N/x\}) \leq \mathsf{W}_n(P) + \mathsf{NFO}(x, P) \cdot \mathsf{W}_n(N)$, and $\mathsf{W}_m(Q\{N/x\}) \leq \mathsf{W}_n(Q) + \mathsf{NFO}(x, Q) \cdot \mathsf{W}_n(N)$. Now:

$$\begin{aligned}
\mathsf{W}_m(P(Q)\{N/x\}) &= \mathsf{W}_m(P\{N/x\}Q\{N/x\}) \\
&= \mathsf{W}_m(P\{N/x\}) + \mathsf{W}_m(Q\{N/x\}) + 1 \\
&\overset{IH}{\leq} \mathsf{W}_n(P) + \mathsf{NFO}(x, P) \cdot \mathsf{W}_n(N) + \mathsf{W}_n(Q) + \mathsf{NFO}(x, Q) \cdot \mathsf{W}_n(N) + 1 \\
&= \mathsf{W}_n(P) + \mathsf{W}_n(Q) + 1 + (\mathsf{NFO}(x, P) + \mathsf{NFO}(x, Q)) \cdot \mathsf{W}_n(N) \\
&= \mathsf{W}_n(P(Q)) + \mathsf{NFO}(x, P(Q)) \cdot \mathsf{W}_n(N).
\end{aligned}$$

- *Bang case*. For Example: M is $!P$. P may have two possible derivations, by means of prom rule: either

$$\frac{\Delta \vdash P}{!\Delta, !\Delta_1, !x \vdash !P}\,\mathsf{prom}$$

or

$$\frac{\Delta, x \vdash P}{!\Delta, !\Delta_1, !x \vdash !P}\,\mathsf{prom}.$$

The only interesting case is the second. Using the linear case, we obtain

$$\begin{aligned}
\mathsf{W}_m((!P)\{N/x\}) &= \mathsf{W}_m(!(P\{N/x\})) \\
&= m \cdot \mathsf{W}_m(P\{N/x\}) + 1 \\
&\leq m \cdot (\mathsf{W}_n(P) + \mathsf{W}_n(N)) + 1 \leq n \cdot \mathsf{W}_n(P) + 1 + n \cdot \mathsf{W}_n(N) \\
&= \mathsf{W}_n(!P) + n \cdot \mathsf{W}_n(N).
\end{aligned}$$

- *Quantum case*. For example: M is PQ and for simplicity, suppose that $x_1, \ldots, x_k$ occur in P. Then

$$\begin{aligned}
\mathsf{W}_m(M\{r_1/x_1, \ldots, r_k/x_k\}) &= \mathsf{W}_m((PQ)\{r_1/x_1, \ldots, r_k/x_k\}) \\
&= \mathsf{W}_m(P\{r_1/x_1, \ldots, r_k/x_k\}Q) \\
&= \mathsf{W}_m(P\{r_1/x_1, \ldots, r_k/x_k\}) + \mathsf{W}_m(Q) + 1 \\
&\overset{IH}{\leq} \mathsf{W}_n(P) + \mathsf{W}_m(Q) + 1 \leq \mathsf{W}_n(P) + \mathsf{W}_n(Q) + 1 \\
&= \mathsf{W}_n(PQ).
\end{aligned}$$

This concludes the proof. □

The following lemma tell us that weight $\mathsf{W}(\cdot)$, as for $\mathsf{D}(\cdot)$, is monotone:

Lemma 10. (i) *If* $M \to_{\mathscr{K}} N$, *then* $\mathsf{W}(M) \geq \mathsf{W}(N)$;
(ii) *if* $M \to_{\mathscr{N}} N$, *then* $\mathsf{W}(M) > \mathsf{W}(N)$.

Proof. By means of the previous substitution lemmas it is possible to prove that for all terms M, N and for all $n, m \in \mathbb{N}$, $n \geq m \geq 1$ and $n \geq \mathsf{D}(M)$, (i) if $M \to_{\mathscr{K}} N$ then $\mathsf{W}_n(M) \geq \mathsf{W}_m(N)$, and (ii) if $M \to_{\mathscr{N}} N$ then $\mathsf{W}_n(M) > \mathsf{W}_m(N)$. The proof is by induction on the derivation of $\to_{\mathscr{L}}$. We cite only the most interesting cases.

(i) Notice that, by Lemma 7, if $M \to_{\mathscr{K}} N$, then $\mathsf{D}(M) = \mathsf{D}(N)$. The result follows by definition. Inductive steps are performed by means of context closures.
(ii) Let r be the last rule of the derivation.
- M is $(\lambda!x.P)!Q$ and the reduction rule is $(\lambda!x.P)!Q \to_{\mathrm{c}.\beta} P\{Q/x\}$. We have to distinguish two sub-cases:
  - if the derivation for M is

$$\frac{\dfrac{\Gamma, \#x \vdash P}{\Gamma \vdash \lambda!x.P} \qquad \dfrac{\Delta_1 \vdash Q}{!\Delta_1, !\Delta_2 \vdash !Q}\,\mathsf{prom}}{\Gamma, !\Delta_1, !\Delta_2 \vdash (\lambda!x.P)!Q}\,\mathsf{app}$$

  then we can exploit the contraction case of Lemma 9 as follows:

$$\begin{aligned}
\mathsf{W}_n((\lambda!x.P)!Q) &= \mathsf{W}_n(\lambda!x.P) + \mathsf{W}_n(!Q) + 1 \\
&= \mathsf{W}_n(\lambda!x.P) + n \cdot \mathsf{W}_n(Q) + 2 \\
&= \mathsf{W}_n(P) + n \cdot \mathsf{W}_n(Q) + 3 \\
&> \mathsf{W}_n(P) + n \cdot \mathsf{W}_n(Q) \\
&\geq \mathsf{W}_n(P) + \mathsf{NFO}(x, P) \cdot \mathsf{W}_n(Q) \\
&\geq \mathsf{W}_m(P\{Q/x\}).
\end{aligned}$$

– if the derivation for M is

$$\dfrac{\dfrac{\Gamma, !x \vdash P}{\Gamma \vdash \lambda !x.P} \qquad \dfrac{\Delta_1 \vdash Q}{!\Delta_1, !\Delta_2 \vdash !Q}\,\mathsf{prom}}{\Gamma, !\Delta_1, !\Delta_2 \vdash (\lambda !x.P)!Q}\,\mathsf{app}$$

then we can exploit the bang case of Lemma 9 as follows:

$$\begin{aligned}\mathsf{W}_n((\lambda !x.P)!Q) &= \mathsf{W}_n(\lambda !x.P) + \mathsf{W}_n(!Q) + 1\\ &= \mathsf{W}_n(\lambda !x.P) + n \cdot \mathsf{W}_n(Q) + 2\\ &= \mathsf{W}_n(P) + n \cdot \mathsf{W}_n(Q) + 3\\ &> \mathsf{W}_n(P) + n \cdot \mathsf{W}_n(Q)\\ &\geq \mathsf{W}_m(P\{Q/x\}).\end{aligned}$$

This concludes the proof. □

The weight $\mathsf{W}(\cdot)$ and the box-depth $\mathsf{B}(\cdot)$ are related by the following properties:

Lemma 11. *For every term M, for all positive $n \in \mathbb{N}$, $\mathsf{W}_n(M) \leq |M| \cdot n^{\mathsf{B}(M)}$*

Proof. By induction on M:

- M is a variable, a constant or a quantum variable. We have
$$\mathsf{W}_n(M) = 1 \leq 1 \cdot n^0 = |M| \cdot n^{\mathsf{B}(M)}.$$
- M is $\mathsf{new}(N)$:
$$\begin{aligned}\mathsf{W}_n(\mathsf{new}(N)) &= \mathsf{W}_n(N) + 1 \overset{IH}{\leq} |N| \cdot n^{\mathsf{B}(N)} + 1\\ &\leq |N| \cdot n^{\mathsf{B}(N)} + n^{\mathsf{B}(N)} = (|N| + 1) \cdot n^{\mathsf{B}(N)}\\ &= |M| \cdot n^{\mathsf{B}(N)} = |M| \cdot n^{\mathsf{B}(M)}.\end{aligned}$$
- M is $!N$:
$$\begin{aligned}\mathsf{W}_n(!N) &= n \cdot \mathsf{W}_n(N) + 1 \overset{IH}{\leq} n \cdot |N| \cdot n^{\mathsf{B}(N)} + 1\\ &= |N| \cdot n^{\mathsf{B}(N)+1} + 1 \leq |N| \cdot n^{\mathsf{B}(N)+1} + n^{\mathsf{B}(N)+1}\\ &= (|N| + 1) \cdot n^{\mathsf{B}(N)+1} = |M| \cdot n^{\mathsf{B}(M)}.\end{aligned}$$
- M is PQ:
$$\begin{aligned}\mathsf{W}_n(PQ) &= \mathsf{W}_n(P) + \mathsf{W}_n(Q) + 1 \overset{IH}{\leq} |P| \cdot n^{\mathsf{B}(P)} + |Q| \cdot n^{\mathsf{B}(Q)} + 1\\ &\leq |P| \cdot n^{\mathsf{B}(P(Q))} + |Q| \cdot n^{\mathsf{B}(P(Q))} + 1\\ &\leq |P| \cdot n^{\mathsf{B}(P(Q))} + |Q| \cdot n^{\mathsf{B}(P(Q))} + n^{\mathsf{B}(P(Q))}\\ &= (|P| + |Q| + 1) \cdot n^{\mathsf{B}(P(Q))} = |M| \cdot n^{\mathsf{B}(M)}.\end{aligned}$$
- M is $\langle N_1, \ldots, N_k\rangle$:
$$\begin{aligned}\mathsf{W}_n(\langle N_1, \ldots, N_k\rangle) &= \mathsf{W}_n(N_1) + \cdots + \mathsf{W}_n(N_k) + 1\\ &\overset{IH}{\leq} |N_1| \cdot n^{\mathsf{B}(N_1)} + \cdots + |N_k| \cdot n^{\mathsf{B}(N_k)} + 1\\ &\leq |N_1| \cdot n^{\mathsf{B}(M)} + \cdots + |N_k| \cdot n^{\mathsf{B}(M)} + 1\\ &\leq |N_1| \cdot n^{\mathsf{B}(M)} + \cdots + |N_k| \cdot n^{\mathsf{B}(M)} + n^{\mathsf{B}(M)} = |M| \cdot n^{\mathsf{B}(M)}.\end{aligned}$$
- M is $\lambda x.N$ or $\lambda !x.N$ or $\lambda \langle x_1, \ldots, x_k\rangle.N$:
$$\begin{aligned}\mathsf{W}_n(M) &= \mathsf{W}_n(N) + 1 \overset{IH}{\leq} |N| \cdot n^{\mathsf{B}(N)} + 1\\ &\leq |N| \cdot n^{\mathsf{B}(N)} + n^{\mathsf{B}(N)} = (|N| + 1) \cdot n^{\mathsf{B}(N)} = |M| \cdot n^{\mathsf{B}(M)}.\end{aligned}$$

This concludes the proof. □

Lemma 12. *For every term M, $\mathsf{W}(M) \leq |M|^{\mathsf{B}(M)+1}$.*

Proof. By means of Lemmas 4 and 11: $\mathsf{W}(M) = \mathsf{W}_{\mathsf{D}(M)}(M) \leq |M| \cdot \mathsf{D}(M)^{\mathsf{B}(M)} \leq |M| \cdot |M|^{\mathsf{B}(M)} = |M|^{\mathsf{B}(M)+1}$. □

We have all the technical tools to prove another crucial lemma:

Lemma 13. *If $M \xrightarrow{*} N$, then $|N| \leq |M|^{\mathsf{B}(M)+1}$.*

Proof. By means of Lemmas 8, 10 and 12: $|N| \leq \mathsf{W}(N) \leq \mathsf{W}(M) \leq |M|^{\mathsf{B}(M)+1}$. □

With all the intermediate lemmas we have just presented, proving that SQ is polystep is relatively easy:

Theorem 6 (*Bounds*). *There is a family of unary polynomials $\{p_n\}_{n\in\mathbb{N}}$ such that for any term M, for any $m \in \mathbb{N}$, if $M \xrightarrow{m} N$ (M reduces to N in m steps) then $m \leq p_{\mathsf{B}(M)}(|M|)$ and $|N| \leq p_{\mathsf{B}(M)}(|M|)$.*

Proof. We show now that the suitable polynomials are $p_n(x) = x^{3(n+1)} + 2x^{2(n+1)}$. We need some definitions. Let $\mathbf{K}$ be a finite sequence $M_0, \ldots, M_\nu$ such that $\forall i \in [1, \nu]$. $M_{i-1} \to_c M_i$. $f(\mathbf{K}) = M_0$, $l(\mathbf{K}) = M_\nu$ and $\#\mathbf{K}$ denote respectively the first element, the last element and the length of the reduction sequence $\mathbf{K}$. Let us define the weight of a sequence $\mathbf{K}$ as $\mathsf{W}(\mathbf{K}) = \mathsf{W}(f(\mathbf{K}))$. We write a computation in the form $M = M_0, \ldots, M_m = N$ as a sequence of blocks of commutative steps $\mathbf{K}_0, \ldots, \mathbf{K}_\alpha$ where $M_0 = f(\mathbf{K}_0)$ and $l(\mathbf{K}_{i-1}) \to_{\mathscr{N}} f(\mathbf{K}_i)$ for every $1 \leq i \leq \alpha$. Note that $\alpha \leq |M|^{\mathsf{B}(M)+1}$; indeed, $\mathsf{W}(\mathbf{K}_0) > \cdots > \mathsf{W}(\mathbf{K}_\alpha)$ and

$$\mathsf{W}(\mathbf{K}_0) = \mathsf{W}(f(\mathbf{K}_0)) = \mathsf{W}(M_0) \leq |M|^{\mathsf{B}(M)+1}.$$

For every $i \in [0, \nu]$

$$\#\mathbf{K}_i \leq |f(\mathbf{K}_i)|^2 \leq (\mathsf{W}(f(\mathbf{K}_i)))^2 \leq (\mathsf{W}(M_0))^2 \leq |M|^{2(\mathsf{B}(M)+1)}.$$

Finally:

$$\begin{aligned}
m &\leq \#\mathbf{K}_0 + \cdots + \#\mathbf{K}_\alpha + \alpha \\
&\leq \underbrace{|M|^{2(\mathsf{B}(M)+1)} + \cdots + |M|^{2(\mathsf{B}(M)+1)}}_{\alpha+1} + |M|^{\mathsf{B}(M)+1} \\
&\leq \underbrace{(|M|^{2(\mathsf{B}(M)+1)} + \cdots + |M|^{2(\mathsf{B}(M)+1)})}_{|M|^{\mathsf{B}(M)+1}+2} \\
&= |M|^{2(\mathsf{B}(M)+1)} \cdot (|M|^{\mathsf{B}(M)+1} + 2) = |M|^{3(\mathsf{B}(M)+1)} + 2|M|^{2(\mathsf{B}(M)+1)} \\
&= p_{\mathsf{B}(M)}(|M|).
\end{aligned}$$

Moreover,

$$\begin{aligned}
|N| &= |f(\mathbf{K}_\alpha)| \leq \mathsf{W}(f(\mathbf{K}_\alpha)) \leq \mathsf{W}(M_0) \leq |M|^{\mathsf{B}(M)+1} \\
&\leq p_{\mathsf{B}(M)}(|M|).
\end{aligned}$$

This concludes the proof. □

Corollary 1. *Every configuration is strongly normalizing.*

Here is the main result of this section:

Theorem 7 (*Polytime Soundness*). *The following inclusions hold: ESQ ⊆ EQP, BSQ ⊆ BQP and ZSQ ⊆ ZQP.*

Proof. Let us consider the first inclusion. Suppose a language $\mathcal{L}$ is in ESQ. This implies that $\mathcal{L}$ can be $(n, s, r, 1)$-decided by a term M. By the Standardization Theorem, for every $t \in \{0,1\}^*$, there is a CNQ computation $\{C_i^t\}_{1\leq i\leq n_t}$ starting at $[1, \emptyset, M!^n\lceil t\rceil^{\{0,1\}}]$. By Theorem 6, n_t is bounded by a polynomial on the length $|t|$ of t. Moreover, the size of any C_i^t (that is to say, the sum of the term in C_i^t and the number of quantum variables in the second component of C_i^t) is itself bounded by a polynomial on $|t|$. Since $\{C_i^t\}_{1\leq i\leq n_t}$ is CNQ, any classical reduction step comes before any new-reduction step, which itself comes before any quantum reduction step. As a consequence, there is a polynomial time deterministic Turing machine which, for every t, computes one configuration in $\{C_i^t\}_{i\leq n_t}$ which only contains non-classical redexes (if any). But notice that a configuration only containing non-classical redexes is nothing but a concise abstract representation of a quantum circuit, fed with boolean inputs. Moreover, all the quantum circuits produced in this way are finitely generated, i.e., they can only contain the quantum gates (i.e. unitary operators) which appears in M, since $!^n\lceil t\rceil^{\{0,1\}}$ does not contain any unitary operator and reduction does not introduce new unitary operators in the underlying term. Summing up, the first component $\mathcal{Q}$ of $C_{n_t}^t$ is simply an element of an Hilbert Space $\mathcal{H}(\{q_1, \ldots, q_m\})$ (where $[q_1, \ldots, q_m]$ is the third component of $C_{n_t}^t$) obtained by evaluating a finitely generated quantum circuit whose size is polynomially bounded on $|t|$ and whose code can be effectively computed from t in polynomial time. By the results in [20], $L \in$ EQP. The other two inclusions can be handled in the same way. □

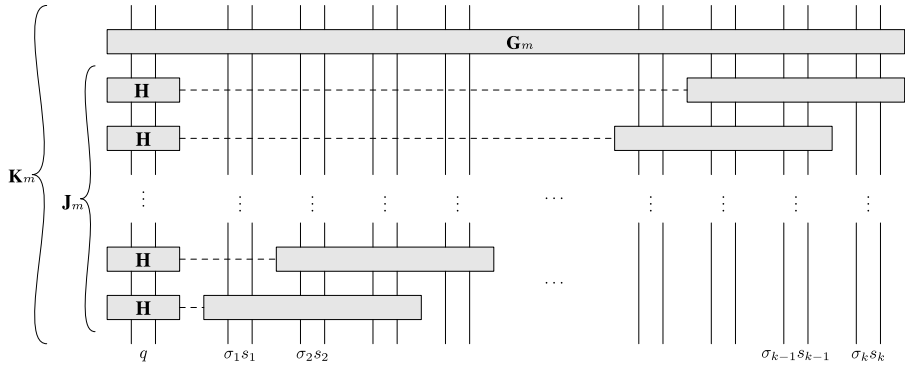


Fig. 3. The quantum circuit computing one step of the simulation.

9. Polytime completeness

In this section, we will prove the converse of Theorem 7. To do that, it is necessary to show that all problems which are computable in polynomial time by some quantum devices can be decided by a SQ term. Quantum devices upon which EQP, BQP and ZQP are defined, are either quantum Turing machines or quantum circuit families. The technique used here consists of encoding quantum Turing machines, although not in a direct way. More specifically, we will show, that the quantum circuit family corresponding (in Yao's sense [32]) to a polynomial time quantum Turing machine can be encoded into SQ. This way, we avoid dealing directly with both quantum Turing machines and classical Turing machines (the latter being an essential ingredient of the definition of a quantum circuit family).

9.1. Yao's circuits

We need to recall Yao's encoding of Quantum Turing machines into quantum circuit families [32].

From now on, we suppose to work with finite alphabets including a special symbol, called *blank* and denoted with $\square$. Moreover, each alphabet comes equipped with an injection $\sigma : \Sigma \to \{0,1\}^{\lceil \lg_2(|\Sigma|)\rceil}$. Σ^ω is the set of (bi)infinite strings on the alphabet Σ, i.e., elements of Σ^ω are functions from $\mathbb{Z}$ to Σ. $\Sigma^\#$ is a subset of Σ^ω containing strings which are different from $\square$ in finitely many positions.

Consider a Quantum Turing Machine $\mathcal{M} = (Q, \Sigma, \delta)$ working in time bounded by a polynomial $t : \mathbb{N} \to \mathbb{N}$. Configurations of $\mathcal{M}$ are elements of the Hilbert space $\mathcal{H}(Q \times \Sigma^\# \times \mathbb{Z})$. The computation of $\mathcal{M}$ on input of length n can be simulated by a quantum circuit $\mathbf{L}_{t(n)}$ built as follows:

- for each m, $\mathbf{L}_m$ has $\eta + k(\lambda + 2)$ inputs (and outputs), where $\eta = \lceil \log_2 |Q| \rceil$, $k = 2m + 1$ and $\lambda = \lceil \log_2 |\Sigma| \rceil$. The first η qubits correspond to a binary encoding q of a state in Q. The other inputs correspond to a sequence $\sigma_1 s_1, \ldots, \sigma_k s_k$ of binary strings, where each σ_i (with $|\sigma_i| = \lambda$) corresponds to the value of a cell of $\mathcal{M}$, while each s_i (with $|s_i| = 2$) encodes a value from $\{0, 1, 2, 3\}$ controlling the simulation.
- $\mathbf{L}_m$ is built up by composing m copies of a circuit $\mathbf{K}_m$, which is depicted in Fig. 3 and has $\eta + k(\lambda + 2)$ inputs (and outputs) itself.
- $\mathbf{K}_m$ is built up by composing $\mathbf{G}_m$ with $\mathbf{J}_m$. $\mathbf{G}_m$ does nothing but switching the inputs corresponding to each s_i from 1 to 2 and vice-versa.
- $\mathbf{J}_m$ can be itself decomposed into $k - 3$ instances of a circuit $\mathbf{H}$ with $\eta + 3(\lambda + 2)$ inputs, acting on different qubits as shown in Fig. 3. Notice that $\mathbf{H}$ can be assumed to be computable (in the sense of Definition 1), because $\mathcal{M}$ can be assumed to have amplitudes in $\mathbf{P}\mathbb{C}$ [19].

Theorem 8 (*Yao [32]*). *The circuit family* $\{\mathbf{L}_m\}_{m \in N}$ *simulates the Quantum Turing Machine* $\mathcal{M}$.

9.2. Encoding polytime quantum Turing machines

We now need to show that SQ is able to simulate Yao's construction. Clearly, the simulation must be uniform, i.e. there must be *a single* term M generating all the possible $\mathbf{L}_m$ where m varies over the natural numbers.

The following two propositions show how $\mathbf{G}_m$ and $\mathbf{J}_m$ can be generated uniformly by a SQ term.

Proposition 4. *For every n, there is a term* M_G^n *which uniformly generates* $\mathbf{G}_m$*, i.e. such that whenever L n-encodes the natural number m,* $M_G^n L \to_{\mathscr{K}} R_G^m$ *where* R_G^m *encodes* $\mathbf{G}_m$.

Proof. Consider the following terms:

$$M_G^n = \lambda x.\lambda y.\mathsf{extract}_\eta(\lambda z.\lambda w_1.\ldots.\lambda w_\eta.\mathsf{append}_\eta w_1 \ldots w_\eta(N_G^n xz))y$$
$$N_G^n = \lambda x.x!^n(\lambda y.\lambda z.\mathsf{extract}_{\lambda+2}((L_G y)z))(\lambda y.y)$$
$$L_G = \lambda x.\lambda y.\lambda z_1.\ldots.\lambda z_{\lambda+2}.(\lambda\langle w, q\rangle.\mathsf{append}_{\lambda+2}(xy)z_1 \ldots z_\lambda wq)(cnot\langle z_{\lambda+1}, z_{\lambda+2}\rangle)$$

For the purpose of proving the correctness of the encoding, let us define P_G^m for every $m \in \mathbb{N}$ by induction on m as follows:

$$P_G^0 = \lambda x.x$$
$$P_G^{m+1} = (\lambda y.\lambda z.(\mathsf{extract}_{\lambda+2}((L_G y)z)))P_G^m$$

First of all, observe that if L n-encodes the natural number m, then $N_G^n L \to_{\mathscr{K}} P_G^m$. Indeed, if L n-encodes m, then

$$\begin{aligned} N_G^n L &\to_{\mathscr{K}} L!^n(\lambda y.\lambda z.\mathsf{extract}_{\lambda+2}((L_G y)z))(\lambda y.y) \\ &\to_{\mathscr{K}} \underbrace{P(P(P(\ldots(P}_{m \text{ times}}(\lambda x.x))\ldots))) = P_G^m \end{aligned}$$

where $P = (\lambda y.\lambda z.(\mathsf{extract}_{\lambda+2}((L_G y)z)))$. Now, we can prove that for every $m \in \mathbb{N}$:

$$[\mathcal{Q}, \mathcal{QV}, P_G^m[q_1, \ldots, q_{m(\lambda+2)}, \ldots, q_h]] \xrightarrow{*} [\mathcal{R}, \mathcal{QV}, [q_1, \ldots, q_{m(\lambda+2)}, \ldots, q_h]]$$

where

$$\mathcal{R} = \mathbf{cnot}_{\langle\langle q_{\lambda+1}, q_{\lambda+2}\rangle\rangle}(\mathbf{cnot}_{\langle\langle q_{2\lambda+3}, q_{2\lambda+4}\rangle\rangle}(\ldots(\mathbf{cnot}_{\langle\langle q_{m(\lambda+2)-1}, q_{m(\lambda+2)}\rangle\rangle}(\mathcal{Q}))\ldots))$$

By induction on m:

- If $m = 0$, then

$$[\mathcal{Q}, \mathcal{QV}, P_G^0[q_1, \ldots, q_h]] \xrightarrow{*} [\mathcal{Q}, \mathcal{QV}, [q_1, \ldots, q_h]]$$

- Now, suppose the thesis holds for m. Then:

$$\begin{aligned} &[\mathcal{Q}, \mathcal{QV}, P_G^{m+1}[q_1, \ldots, q_{(m+1)(\lambda+2)}, \ldots, q_h]] \\ &\xrightarrow{*} [\mathcal{Q}, \mathcal{QV}, \mathsf{extract}_{\lambda+2}(L_G^n P_G^m)[q_1, \ldots, q_h]] \\ &\xrightarrow{*} [\mathcal{Q}, \mathcal{QV}, (L_G P_G^m)[q_{\lambda+3}, \ldots, q_h]q_1 \ldots q_{\lambda+2}] \\ &\xrightarrow{*} [\mathcal{Q}, \mathcal{QV}, \mathsf{append}_{\lambda+2}(P_G^m[q_{\lambda+3}, \ldots, q_h])q_1 \ldots q_{\lambda+2}] \\ &\xrightarrow{*} [\mathcal{R}, \mathcal{QV}, \mathsf{append}_{\lambda+2}[q_{\lambda+3}, \ldots, q_h]q_1 \ldots q_{\lambda+2}] \\ &\xrightarrow{*} [\mathcal{S}, \mathcal{QV}, [q_1, \ldots, q_h]] \end{aligned}$$

  where

$$\mathcal{R} = \mathbf{cnot}_{\langle\langle q_{2\lambda+3}, q_{\lambda+4}\rangle\rangle}(\mathbf{cnot}_{\langle\langle q_{3\lambda+5}, q_{3\lambda+6}\rangle\rangle}(\ldots(\mathbf{cnot}_{\langle\langle q_{m(\lambda+2)-1}, q_{m(\lambda+2)}\rangle\rangle}(\mathcal{Q}))\ldots))$$
$$\mathcal{S} = \mathbf{cnot}_{\langle\langle q_{\lambda+1}, q_{\lambda+2}\rangle\rangle}(\mathbf{cnot}_{\langle\langle q_{2\lambda+3}, q_{2\lambda+4}\rangle\rangle}(\ldots(\mathbf{cnot}_{\langle\langle q_{m(\lambda+2)-1}, q_{m(\lambda+2)}\rangle\rangle}(\mathcal{Q}))\ldots))$$

Now, if L n-encodes the natural number m, then

$$\begin{aligned} M_G^n L &\to_{\mathscr{K}} \lambda y.\mathsf{extract}_\eta(\lambda z.\lambda w_1.\ldots.\lambda w_\eta.\mathsf{append}_\eta w_1 \ldots w_\eta(N_G^n Lz))y \\ &\to_{\mathscr{K}} \lambda y.\mathsf{extract}_\eta(\lambda z.\lambda w_1.\ldots.\lambda w_\eta.\mathsf{append}_\eta w_1 \ldots w_\eta(P_G^m z))y \end{aligned}$$

which has all the properties we require for R_G^m. This concludes the proof. □

Proposition 5. *For every n, there is a term M_J^n which uniformly generates J_m, i.e. such that $M_J^n L \to_{\mathscr{K}} R_J^m$ where R_J^m encodes $\mathbf{J}_m$ whenever L n-encodes the natural number m.*

Proof. Consider the following terms:

$$M_J^n = \lambda x.x!^n(N_J)(\lambda y.y)$$
$$N_J = \lambda x.\lambda y.\mathsf{extract}_{\eta+\lambda+2}(L_J x)y$$
$$\begin{aligned} L_J = \lambda x.\lambda y.\lambda z_1.\ldots.\lambda z_\eta.\lambda w_1.\ldots.\lambda w_{\lambda+2}. \\ \mathsf{extract}_{\eta+2(\lambda+2)}(P_J w_1 \ldots w_{\lambda+2})(x(\mathsf{append}_\eta yz_1 \ldots z_\eta)) \end{aligned}$$
$$\begin{aligned} P_J = \lambda x_1.\ldots.\lambda x_{\lambda+2}.\lambda w.\lambda y_1.\ldots.\lambda y_\eta.\lambda z_1.\ldots.\lambda z_{2(\lambda+2)}.(\lambda\langle q_1.\ldots.\lambda q_{\eta+3(\lambda+2)}\rangle. \\ \mathsf{append}_{\eta+3(\lambda+2)} wq_1 \ldots q_{\eta+3(\lambda+2)})(H\langle y_1, \ldots, y_\eta, x_1, \ldots, x_{\lambda+2}, z_1, \ldots z_{2(\lambda+2)}\rangle). \end{aligned}$$

For the purpose of proving the correctness of the encoding, let us define $R_J^{n,m}$ for every $n, m \in \mathbb{N}$ by induction on m as follows:

$$R_J^0 = \lambda x.x$$
$$R_J^{m+1} = \lambda z.(\mathsf{extract}_{\eta+\lambda+2}(L_J R_J^m))z.$$

First of all, observe that if L n-encodes the natural number m, then $M_J^n L \to_{\mathscr{K}} R_G^m$. Indeed, if L n-encodes m, then

$$M_J^n L \to_{\mathscr{K}} L!^n(N_J)(\lambda y.y)$$
$$\to_{\mathscr{K}} \underbrace{N_J(N_J(N_J(\ldots(N_J}_{m \text{ times}}(\lambda x.x))\ldots))) = R_J^m.$$

Now, we can prove that for every $m \in \mathbb{N}$:

$$[\mathcal{Q}, \mathcal{QV}, R_J^m[q_1, \ldots, q_{\eta+(2m+1)(\lambda+2)}]] \xrightarrow{*} [\mathcal{R}, \mathcal{QV}, [q_1, \ldots, q_{\eta+(2m+1)(\lambda+2)}]]$$

where

$$\mathcal{R} = J_m(\mathcal{Q})$$

by induction on m:

- If $m = 0$, then

$$[\mathcal{Q}, \mathcal{QV}, R_J^0[q_1, \ldots, q_h]] \xrightarrow{*} [\mathcal{Q}, \mathcal{QV}, [q_1, \ldots, q_h]].$$

- Now, suppose the thesis holds for m. Then:

$$\begin{aligned}
&[\mathcal{Q}, \mathcal{QV}, R_J^{m+1}[q_1, \ldots, q_{(2m+3)(\lambda+2)}]] \\
\xrightarrow{*}\ &[\mathcal{Q}, \mathcal{QV}, \mathsf{extract}_{\eta+\lambda+2}(L_J R_J^m)[q_1, \ldots, q_{(2m+3)(\lambda+2)}]] \\
\xrightarrow{*}\ &[\mathcal{Q}, \mathcal{QV}, \mathsf{extract}_{\eta+2(\lambda+2)}(P_J q_{\eta+1} \ldots q_{\eta+\lambda+2}) \\
&\quad (R_J^m(\mathsf{append}_\eta[q_{\eta+(\lambda+2)+1}, \ldots, q_{(2m+3)(\lambda+2)}]q_1 \ldots q_\eta))] \\
\xrightarrow{*}\ &[\mathcal{R}, \mathcal{QV}, \mathsf{extract}_{\eta+2(\lambda+2)}(P_J^n q_{\eta+1} \ldots q_{\eta+\lambda+2}) \\
&\quad ([q_1, \ldots, q_\eta, q_{\eta+(\lambda+2)+1}, \ldots, q_{(2m+3)(\lambda+2)}]] \\
\xrightarrow{*}\ &[\mathcal{R}, \mathcal{QV}, P_J q_{\eta+1} \ldots q_{\eta+\lambda+2}[q_{\eta+3(\lambda+2)+1}, \ldots, q_{(2m+3)(\lambda+2)}]q_1 \ldots q_\eta q_{\eta+\lambda+3} \ldots q_{\eta+3(\lambda+2)}] \\
\xrightarrow{*}\ &[\mathcal{R}, \mathcal{QV}, (\lambda\langle q_1 \ldots . \lambda q_{\eta+3(\lambda+2)}\rangle. \\
&\quad (\mathsf{append}_{\eta+3(\lambda+2)}[q_{\eta+3(\lambda+1)+1}, \ldots, q_{(2m+3)(\lambda+2)}]q_1 \ldots q_{\eta+3(\lambda+2)})(H\langle q_1, \ldots, q_{\eta+3(\lambda+2)}\rangle))] \\
\xrightarrow{*}\ &[\mathcal{S}, \mathcal{QV}, (\mathsf{append}_{\eta+3(\lambda+2)}[q_{\eta+3(\lambda+1)+1}, \ldots, q_{(2m+3)(\lambda+2)}]q_1 \ldots q_{\eta+3(\lambda+2)}] \\
\xrightarrow{*}\ &[\mathcal{S}, \mathcal{QV}, [q_1, \ldots, q_{(2m+3)(\lambda+2)}]]
\end{aligned}$$

where

$$\mathcal{R} = (\mathbf{I}_{\langle q_{\eta+1}, \ldots, q_{\eta+\lambda+2}\rangle} \otimes (\mathbf{J}_m)_{\langle q_1, \ldots, q_\eta, q_\eta+\lambda+3, \ldots, q_{(2m+3)(\lambda+2)}\rangle})(\mathcal{Q})$$
$$\mathcal{S} = (\mathbf{I}_{\langle q_{\eta+3(\lambda+2)+1}, \ldots, q_{(2m+3)(\lambda+2)}\rangle} \otimes \mathbf{H}_{\langle q_1, \ldots, q_{\eta+3(\lambda+2)}\rangle})(\mathcal{R})$$

which implies

$$\mathcal{S} = ((\mathbf{J}_{m+1})_{\langle q_1, \ldots, q_{(2m+3)(\lambda+2)}\rangle})(\mathcal{Q}).$$

This concludes the proof. □

We are almost ready to state and prove a simulation theorem. Preliminary to that is a formal definition of what constitutes a faithful simulation of a quantum Turing machine by a SQ term.

Given a Hilbert's space $\mathcal{H}$, an element $\mathcal{Q}$ of $\mathcal{H}$ and a condition E defining a subspace of $\mathcal{Q}$, the probability of observing E when globally measuring $\mathcal{Q}$ is denoted as $\mathscr{P}_{\mathcal{Q}}(E)$. For example, if $\mathcal{H} = \mathcal{H}(Q \times \Sigma^\# \times \mathbb{Z})$ is the configuration space of a quantum Turing machine, E could be $state = q$, which means that the current state is $q \in Q$. As another example, if $\mathcal{H}$ is $\mathcal{H}(\mathcal{QV})$, E could be

$$q_1, \ldots, q_n = s,$$

which means that the value of the variables $q_1, \ldots, q_n$ is $s \in \{0, 1\}^n$.

Given a quantum Turing machine $\mathcal{M} = (Q, \Sigma, \delta)$, we say that a term M *simulates* the machine $\mathcal{M}$ iff there is a natural number n and an injection $\rho : Q \to \{0,1\}^{\lceil \log_2 |Q| \rceil}$ such that for every string $s \in \Sigma^*$ it holds that if $\mathcal{C}$ is the final configuration of $\mathcal{M}$ on input s, then

$$[1, \emptyset, M!^n \lceil s \rceil^{\Sigma}] \xrightarrow{*} [\mathcal{Q}, \{q_1, \ldots, q_m\}, [q_1, \ldots, q_m]]$$

where for every $q \in Q$

$$\mathscr{P}_{\mathcal{C}}(state = q) = \mathscr{P}_{\mathcal{Q}}(q_1, \ldots, q_{\lceil \log_2 |Q| \rceil} = \rho(q)).$$

Finally, we can state and prove the main result of this Section:

Theorem 9. *For every polynomial time quantum Turing Machine $\mathcal{M} = (Q, \Sigma, \delta)$ there is a term $M_{\mathcal{M}}$ such that $M_{\mathcal{M}}$ simulates the machine $\mathcal{M}$.*

Proof. The theorem follows from Propositions 4, 5 and 3. More precisely, the term $M_{\mathcal{M}}$ has the form $\lambda!x.(M^{\mathrm{circ}}_{\mathcal{M}}x)(M^{\mathrm{init}}_{\mathcal{M}}x)$ where

- $M^{\mathrm{circ}}_{\mathcal{M}}$ builds the Yao's circuit, given a string representing the input;
- $M^{\mathrm{init}}_{\mathcal{M}}$ builds a list of quantum variables to be fed to the Yao's circuit, given a string representing the input.

Now, suppose $\mathcal{M}$ works in time $p : \mathbb{N} \to \mathbb{N}$, where p is a polynomial of degree k. For every term M and for every natural number $n \in \mathbb{N}$, we define $\{M\}_n$ by induction on n:

$$\begin{aligned} \{M\}_0 &= M \\ \{M\}_{n+1} &= \lambda!x.!(\{M\}_n x) \end{aligned}$$

It is easy to prove that for every M, for every N, for every $n \in \mathbb{N}$ and for every n-banged form L of N, $\{M\}_n L \to^*_{\mathscr{N}} P$ where P is an n-banged form of MN. Now, $M^{\mathrm{circ}}_{\mathcal{M}}$ has the following form

$$\lambda!x.(N^{\mathrm{circ}}_{\mathcal{M}}x)(L^{\mathrm{circ}}_{\mathcal{M}}x)$$

where

$$\begin{aligned} N^{\mathrm{circ}}_{\mathcal{M}} &= \lambda x.M_{2p+1}(\{\mathsf{strtonat}_{\Sigma}\}_{2k+1}(M^{2k+2}_{id}x)) \\ L^{\mathrm{circ}}_{\mathcal{M}} &= \lambda x.(\{P^{\mathrm{circ}}_{\mathcal{M}}\}_{2k+1}x) \\ P^{\mathrm{circ}}_{\mathcal{M}} &= \lambda!z.\lambda y.(M^{2k+1}_{J}(M_{2p+1}(\{\mathsf{strtonat}_{\Sigma}\}_{2k+1}z)))(M^{2k+1}_{G}(M_{2p+1}(\{\mathsf{strtonat}_{\Sigma}\}_{2k+1}z)))y \end{aligned}$$

M^{2k+1}_{G} comes from Proposition 4, M^{2k+1}_{J} comes from Proposition 5 and M_{2p+1} comes from Proposition 3. Now, consider any string $s = b_1 \ldots b_n \in \Sigma^*$. First of all:

$$\begin{aligned} N^{\mathrm{circ}}_{\mathcal{M}}!^{4k+3}\lceil s \rceil^{\Sigma} &\to^*_{\mathscr{N}} M_{2p+1}(\{\mathsf{strtonat}_{\Sigma}\}_{2k+1}(M^{2k+2}_{id}!^{4k+3}\lceil s \rceil^{\Sigma})) \\ &\to^*_{\mathscr{N}} M_{2p+1}(\{\mathsf{strtonat}_{\Sigma}\}_{2k+1}!^{2k+1}\lceil s \rceil^{\Sigma}) \\ &\to^*_{\mathscr{N}} M_{2p+1}N \end{aligned}$$

where N is a $2k+1$-banged form of $\mathsf{strtonat}_{\Sigma}\lceil s \rceil^{\Sigma}$, itself a term which 1-represents the natural number n. As a consequence:

$$M_{2p+1}N \to^*_{\mathscr{N}} L$$

where L $2k+1$-represents the natural number $2p(n)+1$. Now:

$$\begin{aligned} L^{\mathrm{circ}}_{\mathcal{M}}!^{4k+2}\lceil s \rceil^{\Sigma} &\to^*_{\mathscr{N}} \{P^{\mathrm{circ}}_{\mathcal{M}}\}_{2k+1}!^{4k+3}\lceil s \rceil^{\Sigma} \\ &\to^*_{\mathscr{N}} P \end{aligned}$$

where P is a $2k+1$-banged form of $P^{\mathrm{circ}}_{\mathcal{M}}!^{2k+2}\lceil s \rceil^{\Sigma}$. So, we can conclude that $M^{\mathrm{circ}}_{\mathcal{M}}!^{4k+4}\lceil s \rceil^{\Sigma}$ rewrites to a term representing the circuit $\mathbf{L}_n$. $M^{\mathrm{init}}_{\mathcal{M}}$ can be built with similar techniques. □

Corollary 2 (*Polytime Completeness*). *The following inclusions hold: EQP $\subseteq$ ESQ, BQP $\subseteq$ BSQ and ZQP $\subseteq$ ZSQ.*

From Theorem 7 and Corollary 2, $\mathsf{EQP} = \mathsf{ESQ}$, $\mathsf{BQP} = \mathsf{BSQ}$ and $\mathsf{ZQP} = \mathsf{ZSQ}$. In other words, there is a perfect correspondence between (polynomial time) quantum complexity classes and classes of languages decidable by SQ terms.

10. Conclusions and further work

We have introduced SQ, an intrinsically polytime lambda calculus for quantum computation. More precisely, SQ captures the three standard quantum complexity classes EQP, BQP and ZQP. In doing so, we have shown that the ICC paradigm can be successfully extended to the framework of quantum computing.

We plan to extend the proposal in two different directions:

- In this paper we only consider quantum decision problems and related complexity classes (in full agreement with the literature on quantum complexity theory). Nevertheless it should be interesting to also analyze the case of quantum complexity classes for functions. A lot of work remains to be done in this direction, not only within ICC, but within quantum computation theory and complexity.
- We would also like to study a typed version of SQ. In order to retain the correspondence with quantum complexity classes, it would be probably necessary to consider polymorphic type systems.

Appendix A. Hilbert spaces

In the paper, we assumed some basic knowledge on Hilbert spaces. This section is devoted to recall the main notions. For a full account on ideas and results about Hilbert spaces, the reader is invited to consult a good mathematical textbook such as [21].

Definition 10 (*Complex Inner Product Space*). A *complex inner product space* is a vector space $\mathcal{H}$ on the field $\mathbb{C}$ equipped with a function $\langle\cdot,\cdot\rangle : \mathcal{H}\times\mathcal{H}\to\mathbb{C}$ that satisfies the following properties:

1. $\langle\phi,\psi\rangle = \langle\psi,\phi\rangle^*$;
2. $\langle\psi,\psi\rangle$ is a non-negative real number;
3. if $\langle\psi,\psi\rangle = 0$ then $\psi = \mathbf{0}$;
4. $\langle c_1\phi_1 + c_2\phi_2, \psi\rangle = c_1^*\langle\phi_1,\psi\rangle + c_2^*\langle\phi_2,\psi\rangle$;
5. $\langle\phi, c_1\psi_1 + c_2\psi_2\rangle = c_1\langle\phi,\psi_1\rangle + c_2\langle\phi,\psi_2\rangle$.

The function $\langle\cdot,\cdot\rangle$ is the inner product of $\mathcal{H}$ and induces a norm $||\cdot||_{\mathcal{H}}$ defined by $||\phi||_{\mathcal{H}} = \sqrt{\langle\phi,\phi\rangle}$.

Definition 11 (*Completeness*). Given the metric $d(\psi,\phi) = ||\psi-\phi||_{\mathcal{H}}$, an inner product space $\mathcal{H}$ is *complete* if any Cauchy sequence[3] $(\phi_n)_{n<\omega}$ is convergent.

Definition 12 (*Hilbert Space*). An *Hilbert space* $\mathcal{H}$ is a complex inner product space that is complete with respect to the distance induced by the inner product.

Proposition 6. *Any finite dimensional inner product space is a Hilbert space.*

Definition 13 (*Unitary Operators*). Let $\mathcal{H}$ be a finite dimensional Hilbert space, and let $\mathbf{U} : \mathcal{H}\to\mathcal{H}$ be a linear map. The *adjoint* of $\mathbf{U}$ is the unique linear transform $\mathbf{U}^\dagger : \mathcal{H}\to\mathcal{H}$ such that for all ϕ,ψ $\langle\mathbf{U}\phi,\psi\rangle = \langle\phi,\mathbf{U}^\dagger\psi\rangle$. If $\mathbf{U}^\dagger\mathbf{U}$ is the identity, we say that $\mathbf{U}$ is a *unitary operator*.

Definition 14 (*Hilbert Basis*). Let $\mathcal{B}$ a maximal orthonormal set in a Hilbert space $\mathcal{H}$ (whose existence is consequence of Zorn's lemma). $\mathcal{B}$ is said to be an *Hilbert basis of* $\mathcal{H}$.

Please note that the concept of a Hilbert basis is *different* from the concept of vector space basis (a maximal linearly independent set of vectors), the so-called *Hamel basis*. In fact it is possible to exhibit a space $\mathcal{H}$ with Hilbert basis $\mathcal{M}$ s.t. $\mathcal{H}$ is not finitely generated by $\mathcal{M}$ and therefore $\mathcal{M}$ is not a maximal linearly independent set of vectors (see e.g. [21], page 189). An orthonormal Hamel basis is usually called *orthonormal basis*.

In the finite dimensional case, the two concepts of a Hamel basis and a Hilbert basis coincide. This fails for the infinite dimensional cases [21].

Definition 15 (*Span*). Let $\mathcal{H}$ be an inner-product space and let $\mathcal{S}\subset\mathcal{H}$, the span of $\mathcal{S}$ is the inner product subspace of $\mathcal{H}$ defined by

$$span(S) = \left\{ \sum_{i=1}^{n} c_i s_i \mid c_i\in\mathbb{C}, s_i\in S \right\}.$$

Even if $\mathcal{H}$ is an Hilbert space, $span(S)$ is not necessarily an Hilbert space (see below).

In the paper we deal with Hilbert spaces of two sorts that, via trivial isomorphisms, correspond to the finite dimensional space $\mathbb{C}^n$ and the $\aleph_0$-dimensional space ℓ^2.

[3] $\forall\epsilon>0.\exists N>0.\forall n,m>N.d(\phi_n,\phi_m)<\epsilon$.

A.1. The Hilbert space $\mathcal{H}(\mathcal{S})$

Let $\mathcal{S}$ a set such that $|\mathcal{S}| \leq \aleph_0$ and let $\mathcal{H}(\mathcal{S})$ be the set

$$\left\{\phi \mid \phi : \mathcal{S} \to \mathbb{C}, \sum_{s\in\mathcal{S}} |\phi(s)|^2 < \infty\right\}$$

equipped with:

(i) an inner sum $+ : \mathcal{H}(\mathcal{S}) \times \mathcal{H}(\mathcal{S}) \to \mathcal{H}(\mathcal{S})$ defined by $(\phi + \psi)(s) = \phi(s) + \psi(s)$;
(ii) a multiplication by a scalar $\cdot : \mathbb{C} \times \mathcal{H}(\mathcal{S}) \to \mathcal{H}(\mathcal{S})$ defined by $(c \cdot \phi)(s) = c \cdot (\phi(s))$;
(iii) an inner product[4] $\langle\cdot,\cdot\rangle : \mathcal{H}(\mathcal{S}) \times \mathcal{H}(\mathcal{S}) \to \mathbb{C}$ defined by $\langle\phi, \psi\rangle = \sum_{s\in\mathcal{S}} \phi(s)^*\psi(s)$;

It is quite easy to show that $\mathcal{H}(\mathcal{S})$ is an Hilbert space.
We call *quantum register* any normalized vector in $\mathcal{H}(\mathcal{S})$.
The set $\mathcal{B}(\mathcal{S}) = \{|s\rangle : s \in \mathcal{S}\}$, where $|s\rangle : \mathcal{S} \to \mathbb{C}$ is defined by:

$$|s\rangle(t) = \begin{cases} 1 & \text{if } s = t \\ 0 & \text{if } s \neq t \end{cases}$$

is a Hilbert basis of $\mathcal{H}(\mathcal{S})$, usually called the *computational basis* in the literature.
It is now interesting to distinguish two cases:

(1) $\mathcal{S}$ is *finite*: in this case $\mathcal{B}(\mathcal{S})$ is also an orthonormal (Hamel) basis of $\mathcal{H}(\mathcal{S})$ and consequently $span(\mathcal{B}(\mathcal{S})) = \mathcal{H}(\mathcal{S})$. $\mathcal{H}(\mathcal{S})$ is isomorphic to $\mathbb{C}^{|\mathcal{S}|}$. With a little abuse of language, we can also say that $\mathcal{H}(\mathcal{S})$ is "generated" or "spanned" by $\mathcal{S}$.
(2) $\mathcal{S}$ is *denumerable*: in this case it is easy to show that $\mathcal{B}(\mathcal{S})$ is a Hilbert basis of $\mathcal{H}(\mathcal{S})$, but *it is not* a Hamel basis. In fact let us consider the subspace $span(\mathcal{B}(\mathcal{S}))$. We see immediately that $span(\mathcal{B}(\mathcal{S})) \subsetneq \mathcal{H}(\mathcal{S})$[5] is an inner-product infinite dimensional space with $\mathcal{B}(\mathcal{S})$ as the Hamel basis, but $span(\mathcal{B}(\mathcal{S}))$ *is not* a Hilbert space because *it is not complete* (see [21]). The careful reader immediately recognizes that $\mathcal{H}(\mathcal{S})$ is the well known fundamental Hilbert space $\ell^2(\mathcal{S})$. There are strong relationships between $span(\mathcal{B}(\mathcal{S}))$ and $\mathcal{H}(\mathcal{S})$, in fact it is possible to show (this is a standard result for ℓ^2) that $span(\mathcal{B}(\mathcal{S}))$ is a dense subspace of $\mathcal{H}(\mathcal{S})$, and that $\mathcal{H}(\mathcal{S})$ is the (unique!) *completion* of $span(\mathcal{B}(\mathcal{S}))$. This fact is important because in the main literature on quantum Turing machines, unitary transforms are usually defined on spaces like $span(\mathcal{B}(\mathcal{S}))$, but this could be problematic because $span(\mathcal{B}(\mathcal{S}))$ *is not a Hilbert space*. Anyway, this is not a real problem: it is possible to show that each unitary operator $\mathbf{U}$ in $span(\mathcal{B}(\mathcal{S}))$ has a standard extension in $\mathcal{H}(\mathcal{S})$ [7].

Appendix B. Confluence

First of all, we need to show that whenever $M \to_\alpha N$, the underlying quantum register evolves in a uniform way:

Lemma 14 (*Uniformity*). *For every M, N such that $M \to_\alpha N$, exactly one of the following conditions holds:*

1. $\alpha \neq \mathsf{new}$ *and there is a unitary transformation* $U_{M,N} : \mathcal{H}(\mathbf{Q}(M)) \to \mathcal{H}(\mathbf{Q}(M))$ *such that* $[\mathcal{Q}, \mathcal{QV}, M] \to_\alpha [\mathcal{R}, \mathcal{RV}, N]$ *iff* $[\mathcal{Q}, \mathcal{QV}, M] \in \mathcal{C}$, $\mathcal{RV} = \mathcal{QV}$ *and* $\mathcal{R} = (\mathbf{U}_{M,N} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(M)})\mathcal{Q}$.
2. $\alpha = \mathsf{new}$ *and there are a constant c and a quantum variable r such that* $[\mathcal{Q}, \mathcal{QV}, M] \to_{\mathsf{new}} [\mathcal{R}, \mathcal{RV}, N]$ *iff* $[\mathcal{Q}, \mathcal{QV}, M] \in \mathcal{C}$, $\mathcal{RV} = \mathcal{QV} \cup \{r\}$ *and* $\mathcal{R} = \mathcal{Q} \otimes |r \mapsto c\rangle$.

Proof. We go by induction on M. M cannot be a variable nor a constant nor a unitary operator nor a term $!L$. If M is an abstraction $\lambda\psi.L$, then $N = \lambda\psi.R$, $L \to_\alpha R$ and the thesis follows from the inductive hypothesis. Similarly when M is a tuple $\langle M_1, \ldots, M_k\rangle$. If $M = LQ$, then we distinguish a number of cases:

- $N = RQ$ and $L \to_\alpha R$. The thesis follows from the inductive hypothesis.
- $N = LP$ and $Q \to_\alpha P$. The thesis follows from the inductive hypothesis.
- $L = U^n$, $Q = \langle r_1, \ldots, r_n\rangle$ and $N = \langle r_1, \ldots, r_n\rangle$. Then case 1 holds. In particular, $\mathbf{Q}(M) = \{r_1, \ldots, r_n\}$ and $\mathbf{U}_{M,N} = \mathbf{U}_{\langle\langle r_1,\ldots,r_n\rangle\rangle}$.
- $L = \lambda x.S$ and $N = S\{Q/x\}$. Then case 1 holds. In particular $\mathbf{U}_{M,N} = \mathbf{I}_{\mathbf{Q}(M)}$.
- $L = \lambda\langle x_1, \ldots, x_n\rangle.S$, $Q = \langle r_1, \ldots, r_n\rangle$ and $N = S\{r_1/x_1, \ldots, r_n/x_n\}$. Then case 1 holds and $\mathbf{U}_{M,N} = \mathbf{I}_{\mathbf{Q}(M)}$.
- $L = \lambda!x.S$, $Q = !T$ and $N = S\{T/x\}$. Then case 1 holds and $\mathbf{U}_{M,N} = \mathbf{I}_{\mathbf{Q}(M)}$.
- $Q = (\lambda\pi.S)T$ and $N = (\lambda\pi.LS)T$. Then case 1 holds and $\mathbf{U}_{M,N} = \mathbf{I}_{\mathbf{Q}(M)}$.
- $L = (\lambda\pi.S)T$ and $N = (\lambda\pi.SQ)T$. Then case 1 holds and $\mathbf{U}_{M,N} = \mathbf{I}_{\mathbf{Q}(M)}$.

If $M = \mathtt{new}(c)$ then N is a quantum variable r and case 2 holds. This concludes the proof. □

[4] In order the inner product definition make sense we must prove that the sum $\sum_{s\in\mathcal{S}} \phi(s)^*\psi(s)$ converges.
[5] $span(\mathcal{B}(\mathcal{S}))$ contains all the functions of $\mathcal{H}(\mathcal{S})$ that are almost everywhere 0.

Notice that $\mathbf{U}_{M,N}$ is always the identity function when performing classical reduction.
The following technical lemma will be useful when proving confluence:

Lemma 15. *Suppose* $[\mathcal{Q}, \mathcal{QV}, M] \to_\alpha [\mathcal{R}, \mathcal{RV}, N]$.

1. *If* $[\mathcal{Q}, \mathcal{QV}, M\{L/x\}] \in \mathcal{C}$, *then*
$$[\mathcal{Q}, \mathcal{QV}, M\{L/x\}] \to_\alpha [\mathcal{R}, \mathcal{RV}, N\{L/x\}].$$
2. *If* $[\mathcal{Q}, \mathcal{QV}, M\{r_1/x_1, \ldots, r_n/x_n\}] \in \mathcal{C}$, *then*
$$[\mathcal{Q}, \mathcal{QV}, M\{r_1/x_1, \ldots, r_n/x_n\}] \to_\alpha [\mathcal{R}, \mathcal{RV}, N\{r_1/x_1, \ldots, r_n/x_n\}].$$
3. *If* $x, \Gamma \vdash L$ *and* $[\mathcal{Q}, \mathcal{QV}, L\{M/x\}] \in \mathcal{C}$, *then*
$$[\mathcal{Q}, \mathcal{QV}, L\{M/x\}] \to_\alpha [\mathcal{R}, \mathcal{RV}, L\{N/x\}].$$

Proof. Claims 1 and 2 can be proved by induction on the proof of $[\mathcal{Q}, \mathcal{QV}, M] \to_\alpha [\mathcal{R}, \mathcal{RV}, N]$. Claim 3 can be proved by induction on N. □

A property similar to one-step confluence holds in SQ. This is a consequence of having adopted the so-called *surface reduction*: it is not possible to reduce inside a subterm in the form $!M$ and, as a consequence, it is not possible to erase a diverging term. This has been already pointed out in the literature [29].

Strictly speaking, one-step confluence does not hold in SQ. For example, if $[\mathcal{Q}, \mathcal{QV}, (\lambda\pi.M)((\lambda x.N)L)] \in \mathcal{C}$, then both

$$[\mathcal{Q}, \mathcal{QV}, (\lambda\pi.M)((\lambda x.N)L)] \to_{\mathscr{N}} [\mathcal{Q}, \mathcal{QV}, (\lambda\pi.M)(N\{L/x\})]$$

and

$$\begin{aligned}[\mathcal{Q}, \mathcal{QV}, (\lambda\pi.M)((\lambda x.N)L)] &\to_{\mathscr{K}} [\mathcal{Q}, \mathcal{QV}, (\lambda x.(\lambda\pi.M)N)L] \\ &\to_{\mathscr{N}} [\mathcal{Q}, \mathcal{QV}, (\lambda\pi.M)(N\{L/x\})].\end{aligned}$$

However, this phenomenon is only due to the presence of commutative rules:

Proposition 7 (*One-Step Confluence*). *Let* C, D, E *be configurations with* $C \to_\alpha D$, $C \to_\beta E$. *Then:*

1. *If* $\alpha \in \mathscr{K}$ *and* $\beta \in \mathscr{K}$, *then either* $D = E$ *or there is* F *with* $D \to_{\mathscr{K}} F$ *and* $E \to_{\mathscr{K}} F$.
2. *If* $\alpha \in \mathscr{N}$ *and* $\beta \in \mathscr{N}$, *then either* $D = E$ *or there is* F *with* $D \to_{\mathscr{N}} F$ *and* $E \to_{\mathscr{N}} F$.
3. *If* $\alpha \in \mathscr{K}$ *and* $\beta \in \mathscr{N}$, *then either* $D \to_{\mathscr{N}} E$ *or there is* F *with* $D \to_{\mathscr{N}} F$ *and* $E \to_{\mathscr{K}} F$.

Proof. Let $C \equiv [\mathcal{Q}, \mathcal{QV}, M]$. We go by induction on M. M cannot be a variable nor a constant nor a unitary operator. If M is an abstraction $\lambda\pi.N$, then $D \equiv [\mathcal{R}, \mathcal{RV}, \lambda\pi.P]$, $E \equiv [\mathcal{S}, \mathcal{SV}, \lambda\pi.Q]$ and

$$[\mathcal{Q}, \mathcal{QV}, N] \to_\alpha [\mathcal{R}, \mathcal{RV}, P]$$
$$[\mathcal{Q}, \mathcal{QV}, N] \to_\beta [\mathcal{S}, \mathcal{SV}, Q]$$

The IH easily leads to the thesis. Similarly when $M = \lambda!x.N$. If $M = NL$, we can distinguish a number of cases depending on the last rule used to prove $C \to_\alpha D$, $C \to_\beta E$:

- $D \equiv [\mathcal{R}, \mathcal{RV}, PL]$ and $E \equiv [\mathcal{S}, \mathcal{SV}, NR]$ where $[\mathcal{Q}, \mathcal{QV}, N] \to_\alpha [\mathcal{R}, \mathcal{RV}, P]$ and $[\mathcal{Q}, \mathcal{QV}, L] \to_\beta [\mathcal{S}, \mathcal{SV}, R]$. We need to distinguish four sub-cases:
  - If $\alpha, \beta = \mathsf{new}$, then, by Lemma 14, there exist two quantum variables $s, q \notin \mathcal{QV}$ and two constants d, e such that $\mathcal{RV} = \mathcal{QV} \cup \{s\}$, $\mathcal{SV} = \mathcal{QV} \cup \{q\}$, $\mathcal{R} = \mathcal{Q} \otimes |s \mapsto d\rangle$ and $\mathcal{S} = \mathcal{Q} \otimes |q \mapsto e\rangle$. Applying 14 again, we obtain
$$D \to_{\mathsf{new}} [\mathcal{Q} \otimes |s \mapsto d\rangle \otimes |v \mapsto e\rangle, \mathcal{QV} \cup \{s, v\}, PR\{v/q\}] \equiv F$$
$$E \to_{\mathsf{new}} [\mathcal{Q} \otimes |q \mapsto e\rangle \otimes |u \mapsto d\rangle, \mathcal{QV} \cup \{q, u\}, P\{u/s\}R] \equiv G$$
    As can be easily checked, $F \equiv G$.
  - If $\alpha = \mathsf{new}$ and $\beta \neq \mathsf{new}$, then, by Lemma 14 there exists a quantum variable r and a constant c such that $\mathcal{RV} = \mathcal{QV} \cup \{r\}$, $\mathcal{R} = \mathcal{Q} \otimes |r \mapsto c\rangle$, $\mathcal{SV} = \mathcal{QV}$ and $\mathcal{S} = (\mathbf{U}_{L,R} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(L)})\mathcal{Q}$. As a consequence, applying Lemma 14 again, we obtain
$$D \to_\beta [(\mathbf{U}_{L,R} \otimes \mathbf{I}_{\mathcal{QV}\cup\{r\}-\mathbf{Q}(L)})(\mathcal{Q} \otimes |r \mapsto c\rangle), \mathcal{QV} \cup \{r\}, PR] \equiv F$$
$$E \to_{\mathsf{new}} [((\mathbf{U}_{L,R} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(L)})\mathcal{Q}) \otimes |r \mapsto c\rangle, \mathcal{QV} \cup \{r\}, PR] \equiv G$$
    As can be easily checked, $F \equiv G$.
  - If $\alpha \neq \mathsf{new}$ and $\beta = \mathsf{new}$, then we can proceed as in the previous case.

– If $\alpha, \beta \neq \mathsf{new}$, then by Lemma 14, there exist $\mathcal{SV} = \mathcal{RV} = \mathcal{QV}$, $\mathcal{R} = (\mathbf{U}_{N,P} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(N)})\mathcal{Q}$ and $\mathcal{S} = (\mathbf{U}_{L,R} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(L)})\mathcal{Q}$. Applying 14 again, we obtain

$$D \to_\beta [(\mathbf{U}_{L,R} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(L)})((\mathbf{U}_{N,P} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(N)})\mathcal{Q}), \mathcal{QV}, PR] \equiv F$$
$$E \to_\alpha [(\mathbf{U}_{N,P} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(L)})((\mathbf{U}_{L,R} \otimes \mathbf{I}_{\mathcal{QV}-\mathbf{Q}(L)})\mathcal{Q}), \mathcal{QV}, PR] \equiv G$$

As can be easily checked, $F \equiv G$.

- $D \equiv [\mathcal{R}, \mathcal{RV}, PL]$ and $E \equiv [\mathcal{S}, \mathcal{SV}, QL]$, where $[\mathcal{Q}, \mathcal{QV}, N] \to [\mathcal{R}, \mathcal{RV}, P]$ and $[\mathcal{Q}, \mathcal{QV}, N] \to [\mathcal{S}, \mathcal{SV}, Q]$. Here we can apply the inductive hypothesis.
- $D \equiv [\mathcal{R}, \mathcal{RV}, NR]$ and $E \equiv [\mathcal{S}, \mathcal{SV}, NS]$, where $[\mathcal{Q}, QV, L] \to [\mathcal{R}, \mathcal{RV}, R]$ and $[\mathcal{Q}, \mathcal{QV}, L] \to [\mathcal{S}, \mathcal{SV}, S]$. Here we can apply the inductive hypothesis as well.
- $N = (\lambda x.T)$, $D \equiv [\mathcal{Q}, \mathcal{QV}, T\{L/x\}]$, $E \equiv [\mathcal{R}, \mathcal{RV}, NR]$, where $[\mathcal{Q}, \mathcal{QV}, L] \to_\beta [\mathcal{R}, \mathcal{RV}, R]$. Clearly $[\mathcal{Q}, \mathcal{QV}, T\{L/x\}] \in \mathcal{C}$ and, by Lemma 15, $[\mathcal{Q}, \mathcal{QV}, T\{L/x\}] \to [\mathcal{R}, \mathcal{RV}, T\{R/x\}]$. Moreover, $[\mathcal{R}, \mathcal{RV}, NR] \equiv [\mathcal{R}, \mathcal{RV}, (\lambda x.T)R] \to [\mathcal{R}, \mathcal{RV}, T\{R/x\}]$
- $N = (\lambda x.T)$, $D \equiv [\mathcal{Q}, \mathcal{QV}, T\{L/x\}]$, $E \equiv [\mathcal{R}, \mathcal{RV}, (\lambda x.V)L]$, where $[\mathcal{Q}, \mathcal{QV}, T] \to_\beta [\mathcal{R}, \mathcal{RV}, V]$. Clearly $[\mathcal{Q}, \mathcal{QV}, T\{L/x\}] \in \mathcal{C}$ and, by Lemma 15, $[\mathcal{Q}, \mathcal{QV}, T\{L/x\}] \to_\beta [\mathcal{R}, \mathcal{RV}, V\{L/x\}]$. Moreover, $[\mathcal{R}, \mathcal{RV}, (\lambda x.V)L] \to_\beta [\mathcal{R}, \mathcal{RV}, V\{L/x\}]$
- $N = (\lambda !x.T)$, $L = !Z$, $D \equiv [\mathcal{Q}, \mathcal{QV}, T\{Z/x\}]$, $E \equiv [\mathcal{R}, \mathcal{RV}, (\lambda !x.V)L]$, where $[\mathcal{Q}, \mathcal{QV}, T] \to_\beta [\mathcal{R}, \mathcal{RV}, V]$. Clearly $[\mathcal{Q}, \mathcal{QV}, T\{Z/x\}] \in \mathcal{C}$ and, by Lemma 15, $[\mathcal{Q}, \mathcal{QV}, T\{Z/x\}] \to_\beta [\mathcal{R}, \mathcal{RV}, V\{Z/x\}]$. Moreover, $[\mathcal{R}, \mathcal{RV}, (\lambda x.V)!Z] \to_\beta [\mathcal{R}, \mathcal{RV}, V\{Z/x\}]$
- $N = (\lambda \langle x_1, \ldots, x_n \rangle .T)$, $L = \langle r_1, \ldots, r_n \rangle$, $D \equiv [\mathcal{Q}, \mathcal{QV}, T\{r_1/x_1, \ldots, r_n/x_n\}]$, $E \equiv [\mathcal{R}, \mathcal{RV}, (\lambda \langle x_1, \ldots, x_n \rangle .V)L]$, where $[\mathcal{Q}, \mathcal{QV}, T] \to_\beta [\mathcal{R}, \mathcal{RV}, V]$. Clearly $[\mathcal{Q}, \mathcal{QV}, T\{r_1/x_1, \ldots, r_n/x_n\}] \in \mathcal{C}$ and, by Lemma 15, $[\mathcal{Q}, \mathcal{QV}, T\{r_1/x_1, \ldots, r_n/x_n\}] \to_\beta [\mathcal{R}, \mathcal{RV}, V\{r_1/x_1, \ldots, r_n/x_n\}]$. Moreover, $[\mathcal{R}, \mathcal{RV}, (\lambda \langle x_1, \ldots, x_n \rangle .V)L] \to_\beta [\mathcal{R}, \mathcal{RV}, V\{r_1/x_1, \ldots, r_n/x_n\}]$.
- $N = (\lambda x.T)Z$, $D \equiv [\mathcal{Q}, \mathcal{QV}, (\lambda x.TL)Z]$, $E \equiv [\mathcal{Q}, \mathcal{QV}, (T\{Z/x\})L]$, $\alpha = \mathsf{r.cm}$, $\beta = \mathsf{l}.\beta$. Clearly, $[\mathcal{Q}, \mathcal{QV}, (\lambda x.TL)Z] \to_{\mathsf{l}.\beta} [\mathcal{Q}, \mathcal{QV}, (T\{Z/x\})L]$.
- $N = (\lambda \pi .T)Z$, $D \equiv [\mathcal{Q}, \mathcal{QV}, (\lambda \pi .TL)Z]$, $E \equiv [\mathcal{R}, \mathcal{RV}, ((\lambda \pi .V)Z)L]$, $\alpha = \mathsf{r.cm}$, where $[\mathcal{Q}, \mathcal{QV}, T] \to_\beta [\mathcal{R}, \mathcal{RV}, V]$. Clearly, $[\mathcal{Q}, \mathcal{QV}, (\lambda x.TL)Z] \to_{\mathsf{r.cm}} [\mathcal{R}, \mathcal{RV}, (\lambda x.VL)Z]$ and $[\mathcal{R}, \mathcal{RV}, ((\lambda \pi .V)Z)L] \to_\beta [\mathcal{R}, \mathcal{RV}, (\lambda \pi .VL)Z]$.
- $N = (\lambda \pi .T)Z$, $D \equiv [\mathcal{Q}, \mathcal{QV}, (\lambda x.TL)Z]$, $E \equiv [\mathcal{R}, \mathcal{RV}, ((\lambda \pi .T)X)L]$, $\alpha = \mathsf{r.cm}$, where $[\mathcal{Q}, \mathcal{QV}, Z] \to_\beta [\mathcal{R}, \mathcal{RV}, X]$. Clearly, $[\mathcal{Q}, \mathcal{QV}, (\lambda x.TL)Z] \to_{\mathsf{r.cm}} [\mathcal{R}, \mathcal{RV}, (\lambda x.TL)X]$ and $[\mathcal{R}, \mathcal{RV}, ((\lambda \pi .T)X)L] \to_\beta [\mathcal{R}, \mathcal{RV}, (\lambda \pi .TL)X]$.
- $N = (\lambda \pi .T)Z$, $D \equiv [\mathcal{Q}, \mathcal{QV}, (\lambda x.TL)Z]$, $E \equiv [\mathcal{R}, \mathcal{RV}, ((\lambda \pi .T)Z)R]$, $\alpha = \mathsf{r.cm}$, where $[\mathcal{Q}, \mathcal{QV}, L] \to_\beta [\mathcal{R}, \mathcal{RV}, R]$. Clearly, $[\mathcal{Q}, \mathcal{QV}, (\lambda x.TL)Z] \to_{\mathsf{r.cm}} [\mathcal{R}, \mathcal{RV}, (\lambda x.TR)Z]$ and $[\mathcal{R}, \mathcal{RV}, ((\lambda \pi .T)Z)R] \to_\beta [\mathcal{R}, \mathcal{RV}, (\lambda \pi .TR)Z]$.
- $N = (\lambda \pi .T)$, $L = (\lambda x.Z)Y$, $D \equiv [\mathcal{Q}, \mathcal{QV}, (\lambda x.NZ)Y]$, $E \equiv [\mathcal{Q}, \mathcal{QV}, N(Z\{Y/x\})]$, $\alpha = \mathsf{l.cm}$, $\beta = \mathsf{l}.\beta$. Clearly, $[\mathcal{Q}, \mathcal{QV}, (\lambda x.NZ)Y] \to_{\mathsf{l}.\beta} [\mathcal{Q}, \mathcal{QV}, N(Z\{Y/x\})]$.

M cannot be in the form $\mathtt{new}(c)$, because in that case $D \equiv E$. □

The following definition is useful when talking about reduction lengths, and takes into account both commuting and non-commuting reductions:

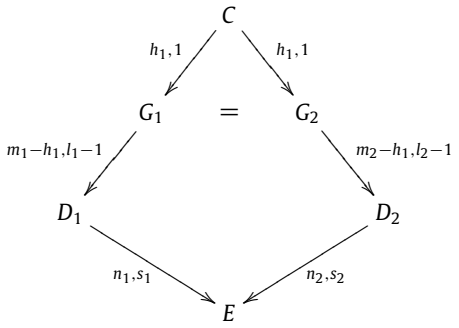
Definition 16. Let $C_1, \ldots, C_n$ be a sequence of configurations such that $C_1 \to \cdots \to C_n$. The sequence is called an *m-sequence of length n from C_1 to C_n* iff m is a natural number and there is $A \subseteq \{2, \ldots, n\}$ with $|A| = m$ and $C_{i-1} \to_{\mathcal{N}} C_i$ iff $i \in A$. If there is a m-sequence of length n from C to D, we will write $C \xrightarrow{m,n} D$ or simply $C \xrightarrow{m} D$.

This way we can generalize Proposition 7 to another one talking about reduction sequences of arbitrary length:

Proposition 8. *Let C, D_1, D_2 be configurations with $C \xrightarrow{m_1} D_1$ and $C \xrightarrow{m_2} D_2$. Then, there is a configuration E with $D_1 \xrightarrow{n_1} E$ and $D_2 \xrightarrow{n_2} E$ with $n_1 \leq m_2$, $n_2 \leq m_1$ and $n_1 + m_1 = n_2 + m_2$.*

Proof. We prove the following, stronger statement: suppose there are C, D_1, D_2, a m_1-sequence of length l_1 from C to D_1 and an m_2-sequence of length l_2 from C to D_2. Then, there is a configuration E, a n_1-sequence of length k_1 from D_1 to E and n_2-sequence of length k_2 from D_2 to E with $n_1 \leq m_2$, $n_2 \leq m_1$, $k_1 \leq l_2$, $k_2 \leq l_1$ and $n_1 + m_1 = n_2 + m_2$. We go by induction on $l_1 + l_2$. If $l_1 + l_2 = 0$, then $C \equiv D_1 \equiv D_2$, $E \equiv D_1 \equiv D_2$ and all the involved natural numbers are 0. If $l_1 = 0$, then $D_1 \equiv C$ and $E \equiv D_2$. Similarly when $l_2 = 0$. So, we can assume $l_1, l_2 > 0$. There are G_1, G_2, two integers $h_1, h_2 \leq 1$ with $C \to_\alpha G_2$ and $C \to_\beta G_2$, an $(m_1 - h_1)$-sequence of length $l_1 - 1$ from G_1 to D_1 and an $(m_2 - h_2)$-sequence of length $l_2 - 1$ from G_2 to D_2. We can distinguish three cases, depending on the outcome of Proposition 7:

- $\alpha \in \mathscr{K}, \beta \in \mathscr{K}$ with $G_1 = G_2$, or $\alpha \in \mathscr{N}, \beta \in \mathscr{N}$ with $G_1 = G_2$. By applying one time the the induction hypothesis we have the following diagram:

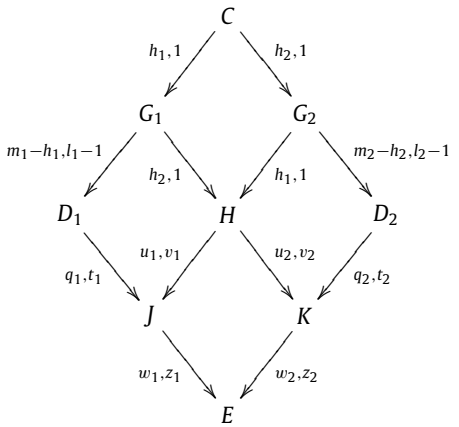


with the equations:

$$\begin{aligned} n_1 &\leq m_2 - h_1 \\ n_2 &\leq m_1 - h_1 \\ s_1 &\leq l_2 - 1 \\ s_2 &\leq l_1 - 1 \\ n_1 + (m_1 - h_1) &= n_2 + (m_2 - h_1) \end{aligned}$$

from which $n_1 \leq m_2$, $n_2 \leq m_1$, and $n_1 + m_1 = n_2 + m_2$.

- $\alpha \in \mathscr{K}, \beta \in \mathscr{K}$ with $G_1 \neq G_2$, or $\alpha \in \mathscr{N}, \beta \in \mathscr{N}$ with $G_1 \neq G_2$ and there is H with $G_1 \to_\beta H$ and $G_2 \to_\alpha H$. By applying the induction hypothesis several times, we end up with the following diagram



together with the equations:

$$\begin{array}{lll} q_1 \leq h_2 & q_2 \leq h_1 & w_1 \leq u_2 \\ t_1 \leq 1 & t_2 \leq 1 & z_1 \leq v_2 \\ u_1 \leq m_1 - h_1 & u_2 \leq m_2 - h_2 & w_2 \leq u_1 \\ v_1 \leq l_1 - 1 & v_2 \leq l_2 - 1 & z_2 \leq v_1 \end{array}$$

and

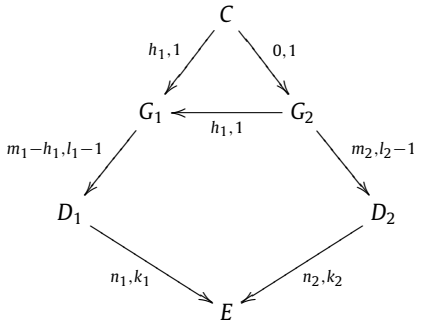
$$m_1 - h_1 + q_1 = u_1 + h_2 \qquad h_1 + u_2 = m_2 - h_2 + q_2 \qquad w_1 + u_1 = w_2 + u_2$$

from which

$$\begin{aligned} q_1 + w_1 &\leq h_2 + u_2 \leq h_2 + m_2 - h_2 = m_2 \\ t_1 + z_1 &\leq 1 + v_2 \leq 1 + l_2 - 1 = l_2 \\ q_2 + w_2 &\leq h_1 + u_1 \leq h_1 + m_1 - h_1 = m_1 \\ t_2 + z_2 &\leq 1 + v_1 \leq 1 + l_1 - 1 = l_1 \\ q_1 + w_1 + m_1 &= h_1 + h_2 + u_1 + w_1 = h_1 + h_2 + u_2 + w_2 = m_2 + w_2 + q_2 \end{aligned}$$

So we can just put $n_1 = q_1 + w_1$, $n_2 = q_2 + w_2$, $k_1 = t_1 + z_1$, $k_2 = t_2 + z_2$.

- $\alpha \in \mathscr{K}$, $\beta \in \mathscr{N}$ and there is H with $G_1 \equiv H$ and $G_2 \to_\beta H$. By applying the induction hypothesis several times, we end up with the following diagram:



  together with the equations:

$$n_1 \leq m_2$$
$$k_1 \leq l_2 - 1$$
$$n_2 \leq m_1$$
$$k_2 \leq l_1$$

  and

$$m_1 + n_1 = m_2 + n_2$$

  from which the desired equations can be easily obtained.
- The last case is similar to the previous one.

This concludes the proof. □

Theorem 3 is an immediate consequence of Proposition 8. But we are now able to prove Theorem 4.

Proof (*Theorem* 4). Strong normalization implies weak normalization. Suppose, by way of contradiction, that C is weakly normalizing but not strongly normalizing. This implies there is a configuration D in normal form and an m-sequence from C to D. Since C is not strongly normalizing, there is an infinite sequence $C \equiv C_1, C_2, C_3, \ldots$ with $C_1 \to C_2 \to C_3 \to \cdots$ From this infinite sequence, we can extract an $m+1$-sequence, due to Lemma 3. Applying Proposition 8, we get a configuration F and a 1-sequence from D to F. However, such a 1-sequence cannot exist, because D is normal. □

Appendix C. Standardization

NCL is closed under new reduction:

Lemma 16. *If* $C \in \mathsf{NCL}$ *and* $C \to_{\mathsf{new}} D$ *then* $D \in \mathsf{NCL}$.

Proof. Let C be $[\mathcal{Q}, \mathcal{QV}, M]$ and D be $[\mathcal{R}, \mathcal{RV}, N]$. The expression $\mathsf{C}[\cdot]$ will denote a term context.[6] Let $\mathtt{new}(c)$ be the reduced redex in M. Clearly, there is a context $\mathsf{C}[\cdot]$ such that $M = \mathsf{C}[\mathtt{new}(c)]$ and $N = \mathsf{C}[r]$. The proof proceeds by induction on the structure of $\mathsf{C}[\cdot]$:

- If $\mathsf{C}[\cdot] = [\cdot]$, then $N = r$ does not contain any redex.
- Clearly, $\mathsf{C}[\cdot] \neq !\mathsf{D}[\cdot]$, because reduction cannot take place under the scope of the operator !.
- If $\mathsf{C}[\cdot] = \mathtt{new}(\mathsf{D}[\cdot])$ then by IH $\mathsf{D}[r]$ cannot contain any classical redex and, hence $\mathsf{C}[r]$ cannot contain any classical redex.
- If $\mathsf{C}[\cdot] = \mathsf{D}[\cdot]L$, then by IH $\mathsf{D}[r]$ cannot contain any classical redex. Moreover, L itself cannot contain any classical redex. So, if $N = \mathsf{D}[r]L$ contain any classical redex, the redex should be N itself. But it is immediate to check that in any of these cases, $M = \mathsf{D}[\mathtt{new}(c)]L$ is a redex too (which goes against the hypothesis). For example, if $\mathsf{D}[\cdot] = \lambda x.\mathsf{E}[\cdot]$, then $M = (\lambda x.\mathsf{E}[\mathtt{new}(c)])L$ contains a classical redex (M itself).
- If $\mathsf{C}[\cdot] = L\mathsf{D}[\cdot]$, we can proceed exactly as in the previous case.
- If

$$\mathsf{C}[\cdot] = \langle N_1, \ldots, N_{k-1}, \mathsf{D}[\cdot], N_{k+1}, \ldots, N_n \rangle$$

  then, by inductive hypothesis, $\mathsf{D}[r]$ cannot contain any classical redex. Moreover, $N_1, \ldots, N_{k-1}, N_{k+1}, \ldots, N_n$ cannot contain any classical redex themselves. But this implies N cannot contain any classical redex.

[6] A term context is a term with one hole.

- The same argument can be applied to the cases $\mathsf{C}[\cdot] = \lambda\pi.\mathsf{D}[\cdot]$ and $\mathsf{C}[\cdot] = \lambda!x.\mathsf{D}[\cdot]$.

This concludes the proof. □

EQT is closed under quantum reduction:

Lemma 17. *If $C \in \mathsf{EQT}$ and $C \to_{\mathscr{Q}} D$ then $D \in \mathsf{EQT}$.*

Proof. Let C be $[\mathcal{Q}, \mathcal{QV}, M]$ and D be $[\mathcal{R}, \mathcal{RV}, N]$. Let L be the reduced redex in M. Clearly, there is a context $\mathsf{C}[\cdot]$ such that $M = \mathsf{C}[L]$ and $N = \mathsf{C}[P]$. Observe that L can be either in the form

$$(\lambda\langle x_1, \ldots, x_n\rangle.Q)\langle r_1, \ldots, r_n\rangle$$

or in the form

$$U\langle r_1, \ldots, r_n\rangle.$$

In the first case, we say that L is a *variable passing redex*, while in the second case, we say that L is a *unitary transformation redex*. The proof proceeds by induction on the structure of $\mathsf{C}[\cdot]$:

- If $\mathsf{C}[\cdot] = [\cdot]$, then:
  - If L is a unitary transformation redex, then $N = \langle r_1, \ldots, r_n\rangle$ does not contain any redex.
  - If L is a variable passing redex, then $N = Q\{r_1/x_1, \ldots, r_n/x_n\}$. But the following lemma can be easily proved by induction on R: for any term R, if R only contains quantum redexes, then $R\{r_1/x_1, \ldots, r_n/x_n\}$ only contains quantum redexes, too.
- Clearly, $\mathsf{C}[\cdot] \neq !\mathsf{D}[\cdot]$, because reduction cannot take place under the scope of the operator !.
- If $\mathsf{C}[\cdot] = \mathtt{new}(\mathsf{D}[\cdot])$ then by IH $\mathsf{D}[P]$ only contains quantum redexes. Now, observe that $\mathsf{D}[P]$ cannot be a boolean constant. Indeed, if L is a unitary transformation redex, then P contains, at least, the term $\langle r_1, \ldots, r_n\rangle$. If L is a variable passing redex, on the other hand, P contains the quantum variables $r_1, \ldots, r_n$ because the variables $x_1, \ldots, x_n$ appears exactly once in Q. Hence $\mathsf{C}[P]$ only contains quantum redexes.
- If $\mathsf{C}[\cdot] = \mathsf{D}[\cdot]R$, then by IH $\mathsf{D}[P]$ only contains quantum redexes. Moreover, R itself only contains quantum redexes. So if $N = \mathsf{D}[P]R$ contain any non-quantum redex, the redex must be N itself. Let us check that in any of these cases, $M = \mathsf{D}[L]R$ is a non-quantum redex too:
  - If N is a I.β redex, then $\mathsf{D}[\cdot] = \lambda x.\mathsf{E}[\cdot]$, and $M = (\lambda x.\mathsf{E}[L])R$ contains a classical redex (M itself).
  - If N is a c.β redex, then $\mathsf{D}[\cdot] = \lambda!x.\mathsf{E}[\cdot]$, $R = !S$ and and $M = (\lambda!x.\mathsf{E}[L])!S$ contains a classical redex.
  - If N is a I.cm redex, then $R = (\lambda\pi.S)T$ and $M = \mathsf{D}[L]R = \mathsf{D}[L](\lambda\pi.S)T$ is a I.cm redex, too.
  - If N is a r.cm redex, then $\mathsf{D}[P] = (\lambda\pi.S)T$. We have to distinguish four sub-cases:
    - If $\mathsf{D}[\cdot] = [\cdot]$, then L must be a variable passing redex and, as a consequence, $M = LR$ is a r.cm redex.
    - If $\mathsf{D}[\cdot] = [\cdot]T$, then L must be a variable passing redex and, as a consequence, LT is a r.cm redex.
    - If $\mathsf{D}[\cdot] = (\lambda\pi.\mathsf{E}[\cdot])T$, then M is $((\lambda\pi.\mathsf{E}[L])T)R$, which is a r.cm redex.
    - If $\mathsf{D}[\cdot] = (\lambda\pi.S)\mathsf{E}[\cdot]$, then M is $((\lambda\pi.S)\mathsf{E}[L])R$, which is a r.cm redex.
- If $\mathsf{C}[\cdot] = L\mathsf{D}[\cdot]$, we can proceed as in the previous case.
- If

  $$\mathsf{C}[\cdot] = \langle N_1, \ldots, N_{k-1}, \mathsf{D}[\cdot], N_{k+1}, \ldots, N_n\rangle$$

  then, by inductive hypothesis, $\mathsf{D}[P]$ cannot contain any classical redex. Moreover, $N_1, \ldots, N_{k-1}, N_{k+1}, \ldots, N_n$ cannot contain any classical redex themselves. But this implies N cannot contain any classical redex.
- The same argument can be applied to the cases $\mathsf{C}[\cdot] = \lambda\pi.\mathsf{D}[\cdot]$ and $\mathsf{C}[\cdot] = \lambda!x.\mathsf{D}[\cdot]$.

This concludes the proof. □

We conclude with the proof of Theorem of *Quantum Standardization*.

Proof (*Theorem* 5). We build a CNQ computation in three steps:

1. Let us start to reduce $D_1 \equiv C_1$ *by using* $\mathscr{C}$ reductions as much as possible. By Theorem 4 we must obtain a finite reduction sequence $D_1 \to_{\mathscr{C}} \cdots \to_{\mathscr{C}} D_k$ such that $1 \leq k$ and no $\mathscr{C}$ reductions are applicable to D_k.
2. Reduce D_k *by using* new reductions as much as possible. By Theorem 4 we must obtain a finite reduction sequence $D_k \to_{\mathsf{new}} \cdots \to_{\mathsf{new}} D_j$ s.t. $k \leq j$ and no new reductions are applicable to D_j. Note that by Lemma 16 such reduction steps cannot generate classical redexes and in particular no classical redex can appear in D_j.
3. Reduce D_j *by using* $\mathscr{Q}$ reductions as much as possible. By Theorem 4 we must obtain a finite reduction sequence $D_j \to_{\mathscr{Q}} \cdots \to_{\mathscr{Q}} D_n$ such that $j \leq n$ and no $\mathscr{Q}$ reductions are applicable to D_n. Note that by Lemma 17 such reduction steps cannot generate neither $\mathscr{C}$ redexes nor new redexes and in particular neither $\mathscr{C}$ nor new reductions are applicable to D_n. Therefore D_n *is in normal form.*

The reduction sequence $\{D_i\}_{1\leq i\leq n}$ is such that $D_0 \to_{\mathscr{C}} \cdots \to_{\mathscr{C}} D_k \to_{\mathsf{new}} \cdots \to_{\mathsf{new}} D_j \to_{\mathscr{Q}} \cdots \to_{\mathscr{Q}} D_n$ is a CNQ computation. By Theorem 3 we observe that $C_m \equiv D_n$, which implies the thesis. □

References

[1] A. Asperti, Light affine logic, in: Thirteenth Annual IEEE Symposium on Logic in Computer Science (Indianapolis, IN, 1998), IEEE Computer Soc., Los Alamitos, CA, 1998, pp. 300–308.
[2] A. Asperti, L. Roversi, Intuitionistic light affine logic (proof-nets, normalization complexity, expressive power), ACM Trans. Comput. Log. 3 (1) (2002) 1–38 (electronic).
[3] P. Baillot, V. Mogbil, Soft lambda-calculus: A language for polynomial time computation, in: Foundations of Software Science and Computation Structures, in: Lecture Notes in Comput. Sci., vol. 2987, Springer, Berlin, 2004, pp. 27–41.
[4] P. Baillot, K. Terui, A feasible algorithm for typing in elementary affine logic, in: Typed Lambda Calculi and Applications, in: Lecture Notes in Comput. Sci., vol. 3461, Springer, Berlin, 2005, pp. 55–70.
[5] H.P. Barendregt, The Lambda Calculus, revised edition, in: Studies in Logic and the Foundations of Mathematics, vol. 103, North-Holland Publishing Co., Amsterdam, 1984, Its Syntax and Semantics.
[6] S. Bellantoni, S. Cook, A new recursion-theoretic characterization of the polytime functions, Comput. Complexity 2 (2) (1992) 97–110.
[7] E. Bernstein, U. Vazirani, Quantum complexity theory, SIAM J. Comput. 26 (5) (1997) 1411–1473.
[8] S.R. Buss, Bounded Arithmetic, in: Studies in Proof Theory. Lecture Notes, vol. 3, Bibliopolis, Naples, 1986.
[9] U. Dal Lago, A. Masini, M. Zorzi, On a measurement-free quantum lambda calculus with classical control, Math. Structures Comput. Sci. 19 (2) (2009) 297–335. doi:10.1017/S096012950800741X.
[10] D. Deutsch, Quantum theory, the Church–Turing principle and the universal quantum computer, Proc. R. Soc. Lond. Ser. A A400 (1985) 97–117.
[11] D. Deutsch, Quantum computational networks, Proc. R. Soc. Lond. A 425 (1989) 73.
[12] D. Deutsch, A. Barenco, A. Ekert, Universality in quantum computation, Proc. Roy. Soc. London Ser. A 449 (1937) (1995) 669–677.
[13] R.P. Feynman, Simulating physics with computers, Internat. J. Theoret. Phys. 21 (6–7) (1981–82) 467–488. Physics of computation, Part II (Dedham, Mass., 1981).
[14] S.J. Gay, Quantum programming languages: Survey and bibliography, Math. Structures Comput. Sci. 16 (4) (2006).
[15] J.-Y. Girard, Linear logic, Theoret. Comput. Sci. 50 (1) (1987) 101.
[16] J.-Y. Girard, Light linear logic, Inform. and Comput. 143 (2) (1998) 175–204.
[17] Y. Lafont, Soft linear logic and polynomial time, Theoret. Comput. Sci. 318 (1–2) (2004) 163–180.
[18] M.A. Nielsen, I.L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, Cambridge, 2000.
[19] H. Nishimura, M. Ozawa, Computational complexity of uniform quantum circuit families and quantum turing machines, Theoret. Comput. Sci. 276 (1–2) (2002) 147–181.
[20] H. Nishimura, M. Ozawa, Perfect computational equivalence between quantum turing machines and finitely generated uniform quantum circuit families, Technical Report, 2008. arXiv:quant-ph/0511117v2.
[21] S. Roman, Advanced Linear Algebra, third edition, in: Graduate Texts in Mathematics, vol. 135, Springer, New York, 2008.
[22] H. Schwichtenberg, An arithmetic for polynomial-time computation, Theoret. Comput. Sci. 357 (1–3) (2006) 202–214.
[23] P. Selinger, A brief survey of quantum programming languages, in: Proceedings of the 7th International Symposium on Functional and Logic Programming, in: Lecture Notes in Computer Science, vol. 2998, Springer, 2004, pp. 1–6.
[24] P. Selinger, Towards a quantum programming language, Math. Structures Comput. Sci. 14 (4) (2004) 527–586.
[25] P. Selinger, Towards a semantics for higher-order quantum computation, in: Proceeding of the 2nd International Workshop on Quantum Programming Languages, in: TUCS General Publication, vol. 33, Turku Centre for Comp. Sci. General Publication, 2004, pp. 127–143.
[26] P. Selinger, B. Valiron, A lambda calculus for quantum computation with classical control, Math. Structures Comput. Sci. 16 (3) (2006) 527–552.
[27] P.W. Shor, Algorithms for quantum computation: Discrete logarithms and factoring, in: 35th Annual Symposium on Foundations of Computer Science (Santa Fe, NM, 1994), IEEE Comput. Soc. Press, Los Alamitos, CA, 1994, pp. 124–134.
[28] P.W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Rev. 41 (2) (1999) 303–332 (electronic).
[29] A. Simpson, Reduction in a linear lambda-calculus with applications to operational semantics, in: Term Rewriting and Applications, in: Lecture Notes in Comput. Sci., vol. 3467, Springer, Berlin, 2005, pp. 219–234.
[30] K. Terui, Light affine lambda calculus and polynomial time strong normalization, Arch. Math. Logic 46 (3–4) (2007) 253–280.
[31] A. van Tonder, A lambda calculus for quantum computation, SIAM J. Comput. 33 (5) (2004) 1109–1135 (electronic).
[32] A. Yao, Quantum circuit complexity, in: Proceedings of the 34th Annual Symposium on Foundations of Computer Science, Los Alamitos, California, IEEE Press, 1993, pp. 352–360.