Relations among Quantum Processes: Bisimilarity and Congruence

MARIE LALIRE
Leibniz Laboratory - Grenoble, France

Received 25 January 2005; Revised 23 November 2005

Full formal descriptions of algorithms making use of quantum principles must take into account both quantum and classical computing components, as well as communications between these components. Moreover, to model concurrent and distributed quantum computations and quantum communication protocols, communications over quantum channels which move qubits physically from one place to another must also be taken into account.

Inspired by classical process algebras, which provide a framework for modeling cooperating computations, a process algebraic notation is defined. This notation provides a homogeneous style to formal descriptions of concurrent and distributed computations comprising both quantum and classical parts. Based upon an operational semantics which makes sure that quantum objects, operations and communications operate according to the postulates of quantum mechanics, an equivalence is defined among process states considered as having the same behavior. This equivalence is a probabilistic branching bisimulation. From this relation, an equivalence on processes is defined. However, it is not a congruence because it is not preserved by parallel composition.

1. Introduction

The number of quantum programming languages is growing rapidly. These languages can be classified in three families: imperative, functional, and parallel and distributed. Among imperative programming languages, there are QCL (Ömer 2000), designed by Ömer, which aims at simulating quantum programs, and qGCL (Zuliani 2001) by Zuliani which allows the construction by refinement of proved correct quantum programs. QPL (Selinger 2004) is a functional language designed by Selinger with a denotational semantics. Several quantum λ-calculi have also been developed: for example (van Tonder 2003) by Van Tonder, based on a simplified linear λ-calculus and (Arrighi and Dowek 2004) by Arrighi and Dowek, which is a “linear-algebraic λ-calculus”. Gay and Nagarajan have developed CQP, a language to describe communicating quantum processes (Nagarajan and Gay 2004). This language is based on π-calculus. An important point in their work is the definition of a type system, and the proof that the operational semantics preserves typing.

Cooperation between quantum and classical computations is inherent in quantum algorithmics. Teleportation of a qubit state from Alice to Bob (Bennett et al. 1993) is a good example of this cooperation. Indeed, Alice carries out a measurement, the result of which
(two bits) is sent to Bob, and Bob uses this classical result to determine which quantum transformation he must apply. Moreover, initial preparation of quantum states and measurement of quantum results are two essential forms of interactions between the classical and quantum parts of computations which a language must be able to express. Process algebras are a good candidate for such a language since they provide a framework for modeling cooperating computations. In addition, they have well defined semantics and permit the transformation of programs as well as the formal study and analysis of their properties. Their semantics give rise to an equivalence relation on processes. Bisimilarity is an adequate equivalence relation to deal with communicating processes since it relates processes that can execute the same flows of actions while having the same branching structure.

This paper presents first the main points of the definition and semantics of a Quantum Process Algebra (QPAlg). Then, a probabilistic branching bisimilarity is defined among process states (section 3), this relation is proved to be an equivalence. As an example, in section 4, the application of the Hadamard unitary transformation is proved bisimilar with its simulation with measurement only, based on state transfer. Finally, in section 5, an equivalence relation among processes is defined. This relation is preserved by all the operators of QPAlg except parallel composition.

2. Definition of QPAlg

The process algebra QPAlg is based upon process algebras such as CCS (Milner 1989) and Lotos (Bolognesi and Brinksma 1987). The key aspects of QPAlg are developed in this section. The precise syntax and the main inference rules of the semantics are given in appendix A. For more details and more examples, see (Lalire and Jorrand 2004).

2.1. Variables

For the purpose of this paper, we consider two types of variables, one classical: \( \mathbb{Nat} \), for variables taking natural values, and one quantum: \( \mathbb{Qubit} \) for variables standing for qubits. An extended version of the process algebra would of course also include quantum registers and other types of variables.

In classical process algebras with value passing (Milner 1989; Bolognesi and Brinksma 1987; Roscoe 1998), variables are instantiated when communications between processes occur and cannot be modified after their instantiation. As a consequence, it is not necessary to store their values. In fact, when a variable is instantiated, all its occurrences are replaced by the value received.

Here, quantum variables represent physical qubits. Applying a transformation to a variable which represents a qubit modifies the state of that qubit. This means that values of variables are modified. For that reason, a process state must keep track of both variable names and variable states, this is achieved thanks to the context (cf. section 2.5).

Variables are declared using the following syntax: \( \{ x_1 : t_1, \ldots, x_n : t_n . P \} \) where
Relations among Quantum Processes: Bisimilarity and Congruence

$x_1, \ldots, x_n$ is a list of variables, $t_1, \ldots, t_n$ are their types, and $P$ is a process which can make use of these classical and quantum variables.

2.2. Expressions

The quantum expressions are quantum variables or tensor product of quantum variables, denoted $x_1 \otimes \cdots \otimes x_n$.

The classical expressions are usual classical expressions, and application of an admissible transformation to a quantum expression. Admissible transformations are also called general quantum measurements, it includes unitary transformations. For more details, see (Nielsen and Chuang 2000).

Let $A$ be a set of predefined admissible transformations. The application of the admissible transformation $A = \{A_{\tau_1}, \ldots, A_{\tau_m}\} \in A$, of dimension $n$, to the register of qubits $x_1 \otimes \cdots \otimes x_n$ is denoted $A[x_1 \otimes \cdots \otimes x_n]$.

The quantum memory is stored in the context in the form $q = \rho$ where $q$ is the list of quantum variable names and $\rho$, a density matrix representing their quantum state (cf. section 2.5). If the classical result of $A[x_1 \otimes \cdots \otimes x_n]$ is $\tau_i$, then $T_{A_{\tau_i}}$ is the super-operator which must be applied to the density matrix $\rho$, to describe the evolution of the quantum memory $q = \rho$.

$$T_{A_{\tau_i}} : \rho \mapsto \Pi^\dagger (A_{\tau_i} \otimes I^{\otimes k}) \Pi \rho (A_{\tau_i}^\dagger \otimes I^{\otimes k}) \Pi$$

where

- $\Pi$ is the permutation matrix which places the $x_i$’s at the head of $q$,
- $k = \text{size}(q) - n$,
- $I^{\otimes k} = I \otimes \cdots \otimes I$, where $I$ is the identity matrix on $\mathbb{C}^2$

Since the admissible transformation $A_{\tau_i}$ may be applied to qubits which are anywhere within the list $q$, a permutation $\Pi$ must be applied first. This permutation moves the $x_i$’s so that they are placed at the head of $q$ in the order specified by $x_1, \ldots, x_n$. Then $A_{\tau_i}$ can be applied to the first $n$ elements and $I$ to the remainder. Finally, the last operation is the inverse of the permutation $\Pi$ so that at the end, the arrangement of the elements in $\rho$ is consistent with the order of the elements in $q$.

$A[x_1 \otimes \cdots \otimes x_n]$ is not probabilistic but its evaluation produces a result with a probabilistic value. This value is stored in the context which becomes probabilistic (cf. section 2.5).

In the examples of this paper, the set $A$ of admissible transformations is:

$$A = \{H, \text{CNot}, I, \sigma_X, \sigma_Y, \sigma_Z, M_{\text{std},1}, M_{\text{std},2}, X, Z \otimes X\}$$

$H$ is Hadamard transformation, $\text{CNot}$ is ”controlled not”, $I$ is the identity, and $\sigma_X, \sigma_Y, \sigma_Z$ are Pauli operators. $M_{\text{std},1}$ and $M_{\text{std},2}$ correspond to measurement in the standard basis of respectively one and two qubits. $X$ and $Z \otimes X$ are the admissible transformations corresponding to the measurements with the Pauli observables $X$ and $Z \otimes X$. 
2.3. Basic actions

The basic actions of QPA órgão are classical expressions and communications. A classical expression is interesting as a basic action if it has side effects, as it is the case of the application of an admissible transformation.

There are several kinds of communications, depending on the type of the expression sent and the type of the receiving variable. The different kinds of communications are: classical to classical communications, classical to quantum communications for initializing qubits, and quantum to quantum communications for allowing the description of quantum communication protocols. Communication gates are not typed but we can imagine a subsequent version of QPA órgão where communication gates would be declared with a fixed type like the other variables.

Emission of an expression \( e \) from a gate \( g \) is denoted \( g ! e \), reception in a variable \( x \) is denoted \( g ? x \). In the operational semantics of parallel composition (rules 15 to 20 of the semantics given in appendix A.2), the combination of the rules for emission and reception defines communication. In a classical to quantum communication (rule 17), the qubit is initialized in the basis state \( |v \rangle \langle v| \), where \( v \) is the classical value sent (in this case, \( v \) must be 0 or 1). In a quantum to quantum communication (rule 18), the name of the sent qubit is replaced in the context by the name of the receiving qubit.

2.4. Composition operators

To create a process from basic actions, the prefix operator “. ” is used: if \( \alpha \) is an action and \( P \), a process, \( \alpha . P \) is a new process which performs \( \alpha \) first, then behaves as \( P \).

The predefined process \( nil \) cannot perform any transition.

The operators of the process algebra are: parallel composition (\( P \parallel Q \)), nondeterministic choice (\( P + Q \)), probabilistic choice (\( P \oplus_p Q \)), conditional choice (\( \{ c_1 \rightarrow P_1, \ldots, c_n \rightarrow P_n \} \)) and restriction (\( P \mid L \)). The process \( \{ c_1 \rightarrow P_1, \ldots, c_n \rightarrow P_n \} \), where \( c_i \) is a condition and \( P_i \) a process, evolves as a process chosen nondeterministically among the processes \( P_j \) such that \( c_j \) is true. Restriction is useful for disallowing the use of some gates (the gates listed in \( L \)), thus forcing internal communication within process \( P \). Communication can occur between two parallel processes whenever a value emission in one of them and a value reception in the other one use the same gate name.

The process \( P \oplus_p Q \) behaves like \( P \) with probability \( p \) and like \( Q \) with probability \( 1 - p \). As explained in (Cazorla et al. 2001) and (Cazorla et al. 2003) and shown in the example of figure 1, if a process contains both a probabilistic and a nondeterministic choice, then the probabilistic choice must always be solved first. Otherwise, a probabilistic transition labeled with a probability \( p \) does not mean that this transition will be executed with probability \( p \).

To solve probabilistic choices before nondeterministic ones, the notion of stability for processes is defined.

**Definition 1. Probabilistic stability** is defined by induction:

1. \( nil, \alpha . P, \{ x_1 : t_1, \ldots, x_n : t_n \ . P \} \) are stable.
2. \( P \mid L, \{ x_1 : t_1, \ldots, x_n : t_n \} \bullet P[a_1, \ldots, a_n] \) are stable if and only if \( P \) is stable.
3 \( P + Q, P \parallel Q \) are stable if and only if \( P \) and \( Q \) are stable.

4 \( \{ c_1 \rightarrow P_1 \ldots c_n \rightarrow P_n \} \) is stable if and only if for all \( i \), \( P_i \) is stable.

\( P \) stable is denoted \( P \downarrow \).

In the examples, another operator on processes is used: \( ; \), for sequential composition. \( P ; Q \) behaves like \( Q \) once \( P \) has terminated. This requires the introduction of a predefined process \( \text{end} \), for signaling successful termination. The operator \( ; \) can be simulated with \( \parallel \): \( P ; Q \) behaves as \( (P \parallel \delta?Q)\{\delta\} \) where \( \delta \) is a fresh gate name and with \( \text{end} \defeq \delta!\text{nil} \).

2.5. Contexts and process states

In the inference rules which describe the semantics of processes, the states of processes are process terms \( P \) together with contexts \( C \), of the form \( P/C \). The main purpose of a context is to maintain the quantum state, stored as \( q = \rho \) where \( q \) is a sequence of quantum variable names and \( \rho \) a density matrix representing their current quantum state. In order to treat classical variables in a similar way, modifications of classical variables are allowed. So, for the same reason as in the case of quantum variables, classical values are stored in the context. Storing and retrieving classical values is represented by functions \( f : \text{names} \rightarrow \text{values} \). The context keeps track of the embedding of variable scopes. To keep track of parallel composition, this is done via a ”cactus stack” structure of sets of variables, called the environment stack \( \{s\} \), which stores variable scopes and types. The set of all the variables in \( s \) is denoted \( \var{\text{var}}(s) \), ”\( .\)” adds an element on top of a stack, and ”\( |\)” concatenates two stacks.

**Definition 2.** A **context** is a tuple \(< s, q = \rho, f >\), where:

- \( s \) is the environment stack;
- \( q \) is a sequence of quantum variable names;
- \( \rho \) is a density matrix representing the quantum state of the variables in \( q \);
- \( f \) is the function which associates values with classical variables.

The evaluation of an admissible transformation (rule 2 of the semantics) produces a probabilistic result. This requires the introduction of a probabilistic composition operator for contexts. This operator is denoted \( \oplus_p \); the state \( P/C_1 \oplus_p C_2 \) is \( P/C_1 \) with probability \( p \) and \( P/C_2 \) with probability \( 1 - p \). In general, a context is either of the form \(< s, q = \rho, f >\), or of the form \( \oplus_p, < s_i, q_i = \rho_i, f_i > \) where the \( p_i \)'s are probabilities adding to 1.
As in the case of probabilities introduced by the operator $\oplus_p$, so as to guarantee that probabilistic choice is always solved first, the notion of probabilistic stability for contexts is introduced: a context $C$ is probabilistically stable, which is denoted $C \downarrow$, if it is of the form $<s,q = \rho,f>$. If the context of a process state is not stable, a probabilistic transition must be performed first (rule 10 of the semantics).

2.6. Example: teleportation

The teleportation protocol (Bennett et al. 1993) transfers the state of a qubit in a place $A$ into a qubit in a place $B$ with only two classical bits sent from place $A$ to place $B$:

*Once upon a time, there were two friends, Alice and Bob who had to separate and live away from each other. Before leaving, each one took a qubit of the same EPR pair. Then Bob went very far away, to a place that Alice did not know. Later on, someone gave Alice a mysterious qubit in a state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, with a mission to forward this state to Bob. Alice could neither meet Bob and give him the qubit, nor clone it and broadcast copies everywhere, nor obtain information about $\alpha$ and $\beta$. Nevertheless, Alice succeeded thanks to the EPR pair and the teleportation protocol.*

This protocol is described with QPAlg in program 1.

The inference rules can be used to show that this protocol results in Bob’s $z$ qubit having the state initially possessed by the $x$ qubit of Alice, with only two classical bits sent from Alice to Bob.

3. Probabilistic branching bisimilarity

The operational semantics associates a process graph with a process state. A process graph is a set of transitions between process states, and an initial state (Fokkink 2000). The transitions are action transitions: $S \xrightarrow{a} T$ where $S,T$ are states, and $a$ is an action (possibly the internal action $\tau$), or probabilistic transitions: $S \xrightarrow{p} T$, where $p$ is a probability.

In this section, an equivalence relation on process states is defined: probabilistic branching bisimilarity, which identifies states when they are associated with process graphs having the same branching structure. The bisimilarity defined here is probabilistic because of the probabilistic transitions introduced by quantum measurement and by the operator of probabilistic choice. The choice of a branching bisimilarity comes from the fact that it abstracts from silent transitions (contrary to strong bisimilarity), but is finer than any other equivalence taking into account silent steps (van Glabbeek 1993).

This definition is inspired from the definitions in (Fokkink 2000) and (Andova 1999).

3.1. Preliminary definitions and notations

Process states The set of all possible process states is denoted $S$. Let $S,T \in S$, then $S$ can be written $P/C_P$ and $T$, $Q/C_Q$ where $P,Q$ are process terms and $C_P$, $C_Q$ contexts (possibly probabilistic).
Program 1 Implementation of the teleportation protocol

\begin{verbatim}
BuildEPR \( \equiv \) \[ x : \text{Qubit}, y : \text{Qubit} \]  
    (\( g_1 ? x . g_2 ? y . H[x] . \text{CNot}[x, y] . \text{end} \)) \[ \parallel \]  
    (\( g_1 ! 0 . g_2 ! 0 . \text{end} \)) \( \setminus \{g_1, g_2\} \)

Alice \( \equiv \) \[ x : \text{Qubit}, y : \text{Qubit} \]  
    \( \text{CNot}[x, y] . H[x].g !M_{std,2}[x, y] . \text{end} \)

Bob \( \equiv \) \[ z : \text{Qubit} \]  
    \[ k : \text{Nat} . \]  
    \( g ? k . \)  
    \[ k = 0 \rightarrow I[z] . \text{end} , \]  
    \[ k = 1 \rightarrow \sigma_X[z] . \text{end} , \]  
    \[ k = 2 \rightarrow \sigma_Z[z] . \text{end} , \]  
    \[ k = 3 \rightarrow \sigma_Y[z] . \text{end} \]  

Teleport \( \equiv \) \[ \psi : \text{Qubit} \]  
    \[ a : \text{Qubit}, b : \text{Qubit} . \]  
    BuildEPR[a, b] ;  
    (\( \text{Alice}[\psi, a] \parallel \text{Bob}[b] \)) \( \setminus \{g\} \)
\end{verbatim}

Assuming that \( S = P/C_P \) and \( C_P =< s, q = \rho, f > \), if \( x \) is an initialized qubit in \( S \), i.e. \((x, \text{Qubit}) \in s\) and \( x \in q \), then \( \rho^S_x \) is the state of \( x \) and this state can be obtained with a trace out operation on \( \rho \):

\[ \rho^S_x = Tr_{(x)/q}(\rho) \]

**Silent transitions** The transitions considered as silent are of course internal transitions \( (\rightarrow) \) but also probabilistic transitions. The reason is that we want, for example, the states \( S_1 \) and \( S_2 \) described in figure 2 to be equivalent.

![Figure 2. Equivalent states](image)

Silent transitions will be denoted \( \rightsquigarrow \). \( \rightsquigarrow^* \) stands for a sequence (possibly empty) of silent transitions, and \( \rightsquigarrow^{0..1} \) stands for zero or one silent transition.
Function $\mu$. Let $\mathcal{R}$ be an equivalence on process states, $S$ be a process state and $\bar{S}$, its equivalence class with respect to $\mathcal{R}$. If $M$ is a set of process states, then $S \triangleright M$ means that there exists a sequence of transitions remaining in $M \cup \bar{S}$, from $S$ to a state in $M$.

A function $\mu_\mathcal{R} : S \times \mathcal{P}(S) \rightarrow [0, 1]$ is defined: $\mu_\mathcal{R} : (S, M) \mapsto p$, where $p$ is the probability to reach a state in the set $M$ from a state $S$ without leaving $\bar{S} \cup M$.

It should be noted that, for this function to yield a probability, nondeterminism must be eliminated in a way which allows the computation of $\mu_\mathcal{R}$. Here, nondeterminism is treated as equiprobability, but this is just a convention for the definition of $\mu_\mathcal{R}$. For example, this does not imply the equivalence of the two process states $S_1$ and $S_2$ described in figure 3.

The function $\mu_\mathcal{R}$ is defined as follows:

- if $S \in M$ then $\mu_\mathcal{R}(S, M) = 1$
- else if $\exists T \in M \cup \bar{S}$ such that $S \xrightarrow{p} T \triangleright M$ then let $E_S = \{ R \in M \cup \bar{S} | S \xrightarrow{p_R} R \triangleright M \}$ in
  
  \[
  \mu_\mathcal{R}(S, M) = \sum_{R \in E_S} p_R \mu_\mathcal{R}(R, M)
  \]
- else if $\exists T \in M \cup \bar{S}$ such that $S \xrightarrow{a} T \triangleright M$ then let $E_S = \{ R \in M \cup \bar{S} | S \xrightarrow{a_R} R \triangleright M \}$ in
  
  \[
  \mu_\mathcal{R}(S, M) = \frac{1}{|E_S|} \sum_{R \in E_S} \mu_\mathcal{R}(R, M)
  \]
- else $\mu_\mathcal{R}(S, M) = 0$

3.2. Probabilistic branching bisimulation

To define bisimilarity, the first step is the definition of a relation of bisimulation on process states. Contrary to the usual definitions of bisimulation, here, a bisimulation has to be an equivalence relation, because of the last point of the definition which ensures that the probability to reach an equivalence class is constant on each equivalence class.

Definition An equivalence relation is a probabilistic branching bisimulation if and only if:
Relations among Quantum Processes: Bisimilarity and Congruence

— if $S$ and $T$ are equivalent and if an action $a$ can be performed from $S$, then the same action can be performed from $T$, possibly after several silent transitions;
— the reached states ($S'$ and $T''$) are equivalent;
— the action $a$ must occur with the same probability in both cases.

The probability to perform an action is the sum for all the branches leading to the action, of the product of the probabilities of each branch. It is calculated thanks to the function $\mu$, defined in the previous section.

In the following, $x$, $y$ denote variables and $v$ denotes a classical value.

Let $\mathcal{B}$ be an equivalence relation on process states. $\mathcal{B}$ is a probabilistic branching bisimulation if and only if it satisfies:

— **Value sending**
  
  if $SBT$ and $S \xrightarrow{g \downarrow v} S'$ then $\exists T', T''$ such that
  
  $(T \xrightarrow{\sim} T' \xrightarrow{g \downarrow v} T'' \land SBT' \land S'BT'')$

— **Qubit sending**

  if $SBT$ and $S \xrightarrow{g \downarrow x} S'$ then $\exists T', T''$ such that
  
  $(T \xrightarrow{\sim} T' \xrightarrow{g \downarrow y} T'' \land \rho^S_x = \rho^T_{y'} \land SBT' \land S'BT'')$

— **Value reception**

  if $SBT$ and $S \xrightarrow{g ? v} S'$ then $\exists T', T''$ such that
  
  $(T \xrightarrow{\sim} T' \xrightarrow{g ? v} T'' \land SBT' \land S'BT'')$

— **Qubit reception**

  if $SBT$ and $S \xrightarrow{g ? x} S'$ then $\exists T', T''$ such that
  
  $(T \xrightarrow{\sim} T' \xrightarrow{g ? y} T'' \land \rho^S_x = \rho^T_{y'} \land SBT' \land S'BT'')$

— **Silent transition**

  if $SBT$ and $S \xrightarrow{\sim} S'$ then $\exists T', T''$ such that
  
  $(T \xrightarrow{\sim} T' \xrightarrow{\sim} 0.1 T'' \land SBT' \land S'BT'')$

— **Probabilities**

  if $SBT$ then $\mu_{SB}(S, M) = \mu_{SB}(T, M), \forall M \in S/B$

**Bisimulation and recursion** Because of recursion in a process definition, the computation of $\mu_{SB}$ in the associated process graph can lead to a linear system of equations. As a consequence, it must be proved that in this case, $\mu_{SB}$ is well-defined and that the system has a unique solution.

Let $S$ be a process state and $M$ be a set of process states, the computation of $\mu_{SB}(T, M)$ for all $T$ in $S$ leads to a linear system of equations where the $\mu_{SB}(T, M)$ are the unknowns:

$$X = AX + B$$

The $i^{th}$ row in this system can be written: $x_i = a_{i1}x_1 + \cdots + a_{in}x_n + b_i$.

The coefficients $a_{ij}$ and $b_i$ are either probabilities or average coefficients in the case
of nondeterminism. As a consequence: \(0 \leq a_{ij} \leq 1\) and \(0 \leq b_i \leq 1\). Moreover, in the definition of \(\mu_B\), every state in the set \(E_S\) is such that there exists a path from that state to the set \(M\). Therefore, the system of equations obtained can be transformed into a system such that \(b_i > 0, \forall i \in [1, n]\). From now on, we consider that the system verifies this property.

Another property of the system is: \(0 \leq \sum_j a_{ij} + b_i \leq 1\), thus \(0 \leq \sum_j a_{ij} < 1\).

To prove that the system has a unique solution, it is sufficient to prove \(\|A\| < 1\) and use the fixpoint theorem. The norms for matrices and vectors are:

\[
\|A\| = \sup_{\|X\| = 1} \|AX\| \quad \|X\| = \max_i |x_i|
\]

We obtain:

\[
\|AX\| = \max_i \left| \sum_j a_{ij} x_i \right| \leq \max_i \sum_j (|a_{ij}| |x_i|)
\]

and then \(\|A\| \leq \max_i \sum_j |a_{ij}| < 1\)

\(\|A\| < 1\) implies that the function \(f : X \mapsto AX + B\) is strictly contracting, so from the fixpoint theorem, we infer that the equation \(X = AX + B\) has a unique solution. Moreover, as \(f([0, 1]) \subseteq [0, 1]\), this solution belongs to \([0, 1]\).

As a consequence, the function \(\mu_B\) is well-defined even in the case of recursive processes.

3.3. **Probabilistic branching bisimilarity**

**Definition 3.** Two process states \(S\) and \(T\) are **bisimilar**, denoted \(S \leftrightarroweq T\) if and only if there exists a probabilistic branching bisimulation \(B\) such that \(SBT\).

**Proposition 1.** Probabilistic branching bisimilarity is an equivalence relation.

**Proof.**

**Symmetry.** \(S \leftrightarroweq T\) implies that there exists a bisimulation \(B\) such that \(SBT\). Since, \(B\) is an equivalence relation, \(TBS\) and then \(T \leftrightarroweq S\).

**Reflexivity.** The identity relation \(I\) is an equivalence relation which verifies all the points of the definition of the probabilistic branching bisimulation. \(\forall S \in S, S \leftrightarroweq S\), so \(S \leftrightarroweq S\).

**Transitivity.** \(S \leftrightarroweq T\) and \(T \leftrightarroweq V\), does \(S \leftrightarroweq V\)? To prove it, we have to find a bisimulation \(B\) such that \(SBT\).

\(S \leftrightarroweq T\) and \(T \leftrightarroweq V\), so there exist two bisimulations \(B_1\) and \(B_2\) such that \(SB_1V\) and \(VB_2T\) (denoted \(SB_1VB_2T\)).

Let \(Eq\) give the equivalence closure of a relation and \(\circ\) compose two relations: \(S(B_1 \circ B_2)V\) if and only if there exists \(V\) such that \(SB_1VB_2T\). \(B = Eq(B_1 \circ B_2)\) is an equivalence relation such that \(SBT\), we prove that it is a bisimulation.
Relations among Quantum Processes: Bisimilarity and Congruence

We develop only the points concerning value sending and probabilities of the definition of a bisimulation, the other points are similar to those developed.

**Value sending.** $\text{SBT}$ and $S \xrightarrow{a} S'$ $(a \notin \{v\})$, let’s prove: $\exists T', T''$ such that $(T \xrightarrow{a} T' \xrightarrow{a} T'' \wedge \text{SBT'} \wedge \text{SBT''})$.

Case $S = T$: $T' = S$ and $T'' = S'$.

Case $\exists V \mid \text{SB}_1 V \text{B}_2 T$: Since $\text{SB}_1 V$, there exist $V', V''$ such that $V \xrightarrow{*} V' \xrightarrow{a} V''$ and $\text{SB}_1 V'$ and $\text{SB}_1 V''$.

If $V = V'$, then $\text{VB}_2 T$ implies there exist $T', T''$ such that $(T \xrightarrow{a} T' \xrightarrow{a} T'' \wedge \text{VB}_2 T' \wedge V'' \text{B}_2 T'')$.

$\text{SB}_1 V$ and $\text{VB}_2 T'$, so $\text{SBT'}$, $\text{SB}_1 V''$ and $V'' \text{B}_2 T''$, so $\text{SBT''}$.

Otherwise $V \xrightarrow{a} V_1 \xrightarrow{a} \cdots \xrightarrow{a} V_n \xrightarrow{a} V'' \xrightarrow{a} V'$. By applying the point on silent transition of the definition of a bisimulation to the relation $\text{B}_2$ with the successive $V_i$, we obtain: $\exists T', T''$ such that $(T \xrightarrow{a} T' \xrightarrow{a} T'' \wedge \text{SBT'} \wedge \text{SBT''})$.

Case $\exists V \mid \text{SB}_2 V \text{B}_1 T$: idem previous case.

Case $\exists V \mid \text{SB}_1 V \text{B}_2 T$ with $S(\text{B}_1 \circ \text{B}_2) V$ or $S(\text{B}_2 \circ \text{B}_1) V$: by induction on the sequence of relations.

**Probabilities.** Let $\{C_i\}_{i \in I}$, $\{D_j\}_{j \in J}$ and $\{M_k\}_{k \in K}$ be the sets of equivalence classes of respectively $\text{B}_1$, $\text{B}_2$ and $\text{B}$.

For all $k$, $l$ in $K$, we prove that for all $S, T \in M_k$, $\mu_B(S, M_l) = \mu_B(T, M_l)$.

Firstly, we need to know the relations between the $M_k$’s and the $D_j$’s and $C_i$’s.

If $S, T \in C_i$, then $S \in M_k$ implies $T \in M_k$, because $\text{SB}_1 T \text{B}_2 T$. As a consequence $\forall i \in I$, $\forall k \in K$, $C_i \subseteq M_k$ or $C_i \cap M_k = \emptyset$. Since the equivalence classes of an equivalence relation form a partition of the space considered, $M_k = \bigcup_{i \in I_k} C_i$.

Similarly for $\text{B}_2$: $M_k = \bigcup_{j \in J_k} D_j$. Moreover, for all $S$, $T$ in $M_k$, there exists a path $V_1, \ldots, V_p$ such that $\text{SB}_1 V_1 \text{B}_2 V_2 \text{B}_1 \ldots \text{B}_2 V_p \text{B}_1 T$.

$$M_k = \bigcup_{i \in I_k} C_i = \bigcup_{j \in J_k} D_j \quad M_l = \bigcup_{i \in I_l} C_i = \bigcup_{j \in J_l} D_j$$

Now, let’s compute $\mu_B$ as a function of $\mu_{B_1}$ (cf. the example at the end of this section). Let $S \in M_k$, there exists $n$ such that $S \in C_n$, $\mu_B(S, M_l)$ is the probability to reach $M_l$ from $S$ without leaving $M_k \cup M_l$, in other words, it is the sum of the probabilities to reach each $C_j$ for $j \in I_l$ without leaving $\bigcup_{i \in I_k \cup J_l} C_i$.

Since $B_1$ is a bisimulation, the probability to reach $C_j$ from $C_i$ is constant for all state in $C_i$, this probability will be denoted $\mu_{B_1}(C_i, C_j)$. 


\[ \mu_B(S, M_l) = \sum_{j \in I_l} \mu_B(S, C_j) = \sum_{j \in I_l} \tilde{\mu}_B(C_n, C_j) \]

with \( \tilde{\mu}_B \) defined by:

\[ \begin{align*}
\tilde{\mu}_B(C_i, C_i) &= 1 \\
\tilde{\mu}_B(C_i, C_j) &= \sum_{m \in E_j} \mu_B(C_i, C_m) \mu_B(C_m, C_j)
\end{align*} \]

where \( E_j = \{ m \mid m \in I_k, \text{ and } C_m \succ C_j \} \).

This shows that the function \( S \mapsto \mu_B(S, M_l) \) is constant on \( C_i \), for all \( i \in I_k \).

Similarly, this function is also constant on \( D_j \), for all \( j \) in \( J_k \).

With these properties:

- if \( S, T \in C_n \), then \( \mu_B(S, M_l) = \mu_B(T, M_l) \)
- if \( S, T \in D_m \), \( \mu_B(S, M_l) = \mu_B(T, M_l) \)
- else there exists \( \{ V_1, \ldots, V_p \} \) such that \( \forall i \in [1, p], V_i \subseteq M_k \), and \( SB_1V_1B_2V_2B_1 \ldots B_2V_pB_1T \).

\[ \begin{align*}
\mu_B(S, M_l) &= \mu_B(V_1, M_l), \\
\mu_B(V_1, M_l) &= \mu_B(V_2, M_l), \\
\vdots \\
\mu_B(V_p, M_l) &= \mu_B(T, M_l)
\end{align*} \]

As a consequence, \( \mu_B(S, M_l) = \mu_B(T, M_l) \).

Example of computation of \( \mu_R \) as a function of \( \mu_{R_1} \). Figure 4 presents the equivalence classes on a process graph of two relations: \( R \) which equivalence classes are \( M_l \) and \( M_k \) and \( R_1 \) with the \( C_i \)'s as equivalence classes. The arrows represent transitions of the process graph.

![Figure 4. Example of equivalence classes](image-url)
Let $S \in C_1$.

\[
\mu_R(S, M_l) = \mu_R(S, C_4) + \mu_R(S, C_5)
\]

\[
\mu_R(S, C_4) = \mu_R(C_1, C_4) = \mu_R_1(C_1, C_4) + \mu_R_1(C_1, C_2)\mu_R(C_2, C_4)
\]

\[
= \mu_R_1(C_1, C_4) + \mu_R_1(C_1, C_2)\mu_R_1(C_2, C_4)
\]

\[
\mu_R(S, C_5) = \mu_R_1(C_1, C_2)\mu_R_1(C_2, C_5)
\]

We obtain:

\[
\mu_R(S, M_l) = \mu_R_1(C_1, C_4) + \mu_R_1(C_1, C_2)(\mu_R_1(C_2, C_4) + \mu_R_1(C_2, C_5))
\]

4. Example: "$H \leftrightarrow$ its measurement-based simulation?"

Quantum measurement is universal for quantum computation (Nielsen 2001). This means that every unitary transformation can be simulated using measurements only. We are interesting in proving with QPAlg that a unitary transformation and its measurement-based simulation behave the same way. We consider the case of the Hadamard transformation $H$.

4.1. Measurement-based simulation of $H$

There exist several models of quantum computation via measurements only. We use the model based on state transfer defined by Perdrix in (Perdrix 2004).

The gate network in figure 5 describes a step of simulation of $H$. This simulation needs an auxiliary qubit initialized in the state $|0\rangle$, it consists in two measurements: one on two qubits with observable $Z \otimes X$ and the other on one qubit with observable $X$. $X$ and $Z$ are Pauli observables. This step simulates $H$ up to a Pauli operator $\sigma$. If $\sigma = I$, $H$ has been simulated, otherwise, a correction must be applied to the result. This correction consists in simulating $\sigma$ (Pauli operators are their proper inverses) in the same way as $H$ has been simulated. The full simulation of $H$ is given by the automaton on figure 6. The states $\text{SimulHad}$, $\sigma_1$, $\sigma_2$ and $\sigma_3$ represent a step of simulation of $H$ and of the Pauli operators.

![Figure 5. A step of simulation of $H$](image-url)
4.2. Modelling with QPAlg

Unitary transformation of Hadamard. The process \textbf{Had} applies the Hadamard transformation on a qubit received from gate \textit{g} and returns this qubit through gate \textit{h}.

\[
\text{Had} \defeq [ x : \text{Qubit}, g ?x .H[x].h !x \cdot \text{nil} ]
\]

Simulation of Hadamard As for the process \textbf{Had}, the process \textbf{SimulHad} (program 2) begins by receiving a qubit \textit{x} from gate \textit{g}. Then the Hadamard transformation is simulated in the way described by the automaton shown in figure 6. At the end, the result is in \textit{y} (and not \textit{x}) which is sent through gate \textit{h}. \textbf{σ} is a process simulating a Pauli transformation, the Pauli transformation simulated is specified by the parameter \textit{i} in \textbf{σ}[\textit{i}, \textit{y}]. The process \textbf{σ} is not given, it is organized in a way similar to \textbf{SimulHad}.

**Program 2** Process for the simulation of \textit{H}

\[
\text{SimulHad} \defeq [ y : \text{Qubit}, i : \text{Nat}, x : \text{Qubit}, m, n : \text{Nat} .
\begin{align*}
  & g ?x . \quad \text{// initialization of \textit{x}} \\
  & (p ?y . \text{end} || p !0 . \text{end}) \{p\} ; \\
  & (q_1 !Z \otimes X [x,y],q_2 !X[x] . \text{end} || q_1 ?m . q_2 ?n . \text{end}) \{q_1,q_2\} ; \\
  & \{ \\
  & \quad (m = 1 \land n = 1) \rightarrow r !0 . \text{end}, \\
  & \quad (m = 1 \land n = -1) \rightarrow r !1 . \text{end}, \quad \text{// initialization of \textit{i}} \\
  & \quad (m = -1 \land n = 1) \rightarrow r !3 . \text{end}, \quad \text{// depending on} \\
  & \quad (m = -1 \land n = -1) \rightarrow r !2 . \text{end} \quad \text{// measurement results} \\
  & \} || r ?i . \text{end}) \{r\} \\
  & \quad : \textbf{σ}[i,y] ; \quad \text{// correction} \\
  & h !y . \text{nil} \quad \text{// result sending}
\end{align*}
\]

The operational semantics associates with the processes \textbf{Had} and \textbf{SimulHad} (program 2) in empty contexts the process graphs described in figure 7.
4.3. Bisimilarity

To prove that the application of the Hadamard transformation and its measurement-based simulation have the same behavior, we prove that their modellings in QPAlg are bisimilar, i.e. that there exists a bisimulation between \textbf{Had} and \textbf{SimulHad} in empty contexts.

The process graphs of \textbf{Had} and \textbf{SimulHad} in empty contexts are given in figure 7. $C_1$, $C_2$, $C_3$ and $C_4$ represents equivalence classes of an equivalence relation $\mathcal{R}$ on process states. This relation is a bisimulation, the main points of this proof are:

— from each process state in $C_1$ the transition $g \ ?x$ can be performed, possibly after several silent transitions

— from each process state in $C_2$ the transition $h \ !x$ or $h \ !y$ can be performed, possibly after several silent transitions

— when $h \ !x$ and $h \ !y$ are performed, the states of $x$ and $y$ are the same

— the probability to reach $C_3$ from each state in $C_2$ must be the same, which corresponds for any state $S$ in $C_2$, to $\mu_{\mathcal{R}}(S, C_3) = 1$

As regards the last point, the computation of the function $\mu_{\mathcal{R}}$ on each state in $C_2$ leads to the system of equations in figure 8. We obtain $\forall i \in \{0, \ldots, 3\}$, $p_i = 1$, and then $q_0 = q_1 = q_2 = 1$. From any state in the class $C_2$, the probability to reach a state in the class $C_3$ is 1.

To conclude, there exists a bisimulation relation between the modelling of the application of Hadamard and the modelling of its measurement-based simulation.
There are two possibilities to make it a congruence, either the operators on processes are extended to operators on process states or a congruence on processes is defined from the bisimilarity relation on process states.

We explore the second possibility. A relation $\sim$ on processes is defined: if $P$ and $Q$ are two processes, $P \sim Q \iff \forall C, P/C \equiv Q/C$.

Nonetheless, $\sim$ is not a congruence, as shown by the two following examples:

- $H[x].g \mathbin{\leftarrow} Q . \text{nil} \sim g \mathbin{\leftarrow} Q . \text{nil}$
  
- $H[x].g \mathbin{\leftarrow} Q . \text{nil} + h \mathbin{\leftarrow} Q . \text{nil} \not\sim g \mathbin{\leftarrow} Q . \text{nil} + h \mathbin{\leftarrow} Q . \text{nil}$

To overcome these problems, probabilistic rooted branching bisimulation and probabilistic rooted branching similarity are defined.

**Definition 4.** Let $B$ be an equivalence relation. $B$ is a probabilistic rooted branching bisimulation if and only if:

- if $SBT$ and $S \xrightarrow{a} S'$ (or $g \mathbin{\leftarrow} Q$), then $T \xrightarrow{a} T'$ with $S' \equiv T'$
- if $SBT$ and $S \xrightarrow{g} S'$, then $T \xrightarrow{g} T'$ with $\rho_x^S = \rho_x^T$ and $S' \equiv T'$
- if $SBT$ and $S \xrightarrow{?y} S'$, then $T \xrightarrow{?y} T'$ with $\rho_y^S = \rho_y^T$ and $S' \equiv T'$
- if $SBT$ and $S \xrightarrow{p} S'$, then $T \xrightarrow{q} T'$ with $S'BDT'$
- if $SBT$ then $\mu_\equiv(S, M) = \mu_\equiv(T, M)$, $\forall M \in S/ \equiv$

**Definition 5.** Two process states $S$ and $T$ are probabilistic rooted branching similar, denoted $S \equiv r T$, if and only if there is a probabilistic branching bisimulation $B$ such that $SBT$. 

---

**Figure 8. Computation of the probabilities**

\[ p_0 = 1 \]
\[ \forall i \in \{1, 2, 3\} \quad p_i = \frac{1}{2} p_0 + \frac{1}{2} p_1 + \frac{1}{2} p_2 + \frac{1}{2} p_3 \]
\[ q_1 = \frac{1}{2} p_0 + \frac{1}{2} p_1 \]
\[ q_2 = \frac{1}{2} p_2 + \frac{1}{2} p_3 \]
\[ q_0 = \frac{1}{2} q_1 + \frac{1}{2} q_2 \]
Relations among Quantum Processes: Bisimilarity and Congruence

Proposition 2. \( \equiv_r \) is an equivalence relation.

Proof. From the fact that \( \equiv \) is an equivalence relation. \( \Diamond \)

Note. For all \( S, T \) in \( S \equiv_r T \) implies \( S \equiv T \)

Definition 6. Let \( P \) and \( Q \) be two processes.

\[ P \sim Q \iff \forall C, P/C \equiv_r Q/C \]

Proposition 3. \( \sim \) is an equivalence relation and is preserved by variable declaration, action prefix, nondeterministic choice, probabilistic choice, conditional choice and restriction.

Proof. Since \( \equiv_r \) is an equivalence relation, it is easy to see that \( \sim \) is also an equivalence relation.

Let \( P, Q \) be processes such that \( P \sim Q \), that is to say \( \forall C, P/C \equiv_r Q/C \).

Action prefix. Let \( a \) be an action. The question is: are \( a.P/C \) and \( a.Q/C \) probabilistically rooted branching bisimilar for all context \( C \)?

If \( C \) is probabilistically stable: \( a.P/C \rightarrow\rightarrow a.P/C' \) and \( a.Q/C \rightarrow\rightarrow a.Q/C' \). \( P/C' \equiv_r Q/C' \). \( P/C' \equiv_r Q/C' \) implies \( P/C' \sim Q/C' \). \( B = \{ (a.P/C, a.Q/C) \} \) is a probabilistic rooted branching bisimulation.

If \( C = \oplus_i C_i \): for all \( i \), \( a.P/C \rightarrow\rightarrow a.P/C_i \) and \( a.Q/C \rightarrow\rightarrow a.Q/C_i \). From the previous case, \( \forall i: a.P/C_i \equiv_r a.Q/C_i \), we deduce \( a.P/C \equiv_r a.Q/C \).

Nondeterminism. Let \( R \) be a process. The question is: are \( P + R/C \) and \( Q + R/C \) probabilistically rooted branching bisimilar, for all context \( C \)?

If \( C \) is probabilistically stable: if \( P + R/C \rightarrow\rightarrow P'/C_P \) then, as \( P/C \equiv_r Q/C \), there exists \( Q'/C_Q \) such that \( Q + R/C \rightarrow\rightarrow Q'/C_Q \) and \( P'/C_P \equiv_r Q'/C_Q \). Otherwise \( P + R/C \rightarrow\rightarrow R'/C'' \) then, \( Q + R/C \rightarrow\rightarrow R'/C'' \) and \( R'/C'' \equiv_r R'/C'' \).

If \( C \) is not probabilistically stable, we reduce the problem to the previous point after a probabilistic transition.

The other points are similar to those developed. \( \Diamond \)

We conclude that \( \sim \) is preserved by all operators except \( \| \), as shown by the following example: \( (g \parallel 2.h \parallel 3.nil \| g ?x .h \?x .nil)\{g, h\} \sim (h \parallel 3.nil \| h ?x .nil)\{h\} \), nonetheless, \( (g \parallel 2.h \parallel 3.nil \| g ?x .h \?x .nil)\{g, h\} \parallel k !x .nil \not\sim (h \parallel 3.nil \| h ?x .nil)\{h\} \parallel k !x .nil \). The left process can send 2 or 3 through gate \( k \) whereas the right process can only send 3.

This problem could be overcome by restricting processes in parallel not to use the same variable names. This is done in CQP (Nagarajan and Gay 2004) and can be justified by the fact that variables cannot be at two places at the same time. However, because of entanglement in the quantum state, this does not solve the whole problem:

\[
M_{std,1}[x].nil \sim I[x],(0.nil \oplus 0.5 \ 1.nil)
\]
\[
M_{std,1}[x].nil \parallel g !y .nil \not\sim (I[x],(0.nil \oplus 0.5 \ 1.nil)) \parallel g !y .nil
\]

In a configuration where the state of \( x \) and \( y \) is \( \frac{1}{\sqrt{2}}((00) + (11)) \) (EPR state), the left
process can send the qubit $y$ in the mixed state $\{0.5 : |0\rangle, 0.5 : |1\rangle\}$ through gate $g$ if $M_{std,x}$ has not been applied, or in the states $|0\rangle$ or $|1\rangle$ if the measurement has been applied, whereas the right process can only send $y$ in the state $\{0.5 : |0\rangle, 0.5 : |1\rangle\}$.

6. Conclusion

This paper has presented a process algebra for quantum programming which can describe both classical and quantum programming, and their cooperation. This language has an operational semantics, one of its peculiarities is the introduction of probabilistic transitions, due to quantum measurement and to the operator of probabilistic choice.

Then a semantical equivalence relation on process states has been defined. This equivalence is a bisimulation which identifies processes associated with process graphs having the same branching structure. From this bisimulation, an equivalence relation on processes has been defined. This relation is preserved by all the operators of the process algebra except parallel composition. This is a first step toward the verification of quantum cryptographic protocols.

Several extensions are possible. As already mentioned, we could define a congruence on process states from the bisimulation defined here by extending the operators on processes to operators on process states. Another possible extension is the definition of a type system to verify statically properties such as the no-cloning theorem (quantum variables cannot be copied).

References

Appendix A. The quantum process algebra

A.1. Syntax

Expressions
\[ qexp ::= qvar \otimes qexp | qvar \]
\[ nexp ::= nfact + nexp | nfact - nexp | nfact \]
\[ nfact ::= nterm \times nfact | nterm \div nfact | nterm \]
\[ nterm ::= nvar | nval | transf_admissible \{ qexp \} | -nterm \]

Actions
\[ communication ::= gate \! variable | gate \! nexp \]
\[ action ::= communication | nexp \]

Processes
\[ type ::= \text{Nat} | \text{Qubit} \]
\[ variable\_decl ::= variable : type \{, variable : type \}^* \]
\[ process ::= nil \]
\[ \]
\[ name \{[ variable\_list ]\} \]
A.2. Main inference rules of the semantics

The semantics is specified with inference rules which give the evolution of the states of processes. There are five kinds of transitions:

— transitions for evaluating expressions: \( \rightarrow_v \) and \( \rightarrow_e \)
— action transitions: \( \rightarrow_{\alpha} \) where \( \alpha \) is \( g ! x \) or \( g ? x \);
— silent transitions: \( \tau \rightarrow \), for internal transition;
— probabilistic transitions: \( \rightarrow_p \), where \( p \) is a probability.

In the following, \( P, Q, P', Q', P_i \) and \( P'_i \) are processes, \( C, C' \) and \( C_i \) are contexts, \( \alpha \) is an action, \( g \) is a communication gate, \( v \) is a value, \( x \) is a variable, and \( c_j \) is a condition.

Expressions

\[
\frac{x < s, q = \rho, f > \rightarrow_v f(x) < s, q = \rho, f >}{x \in \text{dom}(f)} \quad (1)
\]

\[
\frac{A[x_1 \otimes \ldots \otimes x_n]/C \rightarrow_v y/\oplus_{p_i} C_i}{(2)}
\]

avec

— \( A = \{A_{\tau_1}, \ldots, A_{\tau_n}\} \), admissible transformation
— \( y\), fresh variable name
— \( C = \langle e, s, q = \rho, f > \)
— \( C_i = \langle \{y : \text{Nat}\} \cup e, s, q = \rho_i, f \cup \{y \mapsto \tau_i\} > \)
— \( x_1, \ldots, x_n \in q \)
— \( p_i = \text{Tr}(T_{A_{\tau_i}}(\rho)) \) and \( \rho_i = \frac{1}{p_i} T_{A_{\tau_i}}(\rho) \)

Evaluation contexts \( E[\_] \) is an evaluation context of an expression, it is an expression in which a sub-expression has been replaced by \( [\_] \). Similarly, \( F[\_] \) is an evaluation context of a process.

\[
\frac{e/C \rightarrow_v e'/C'}{E[e]/C \rightarrow_v E[e']/C'} \quad (3)
\]
Relations among Quantum Processes: Bisimilarity and Congruence

\[
\dfrac{e/C \rightarrow e'/C'}{F[e]/C \rightarrow F[e']/C'} \quad (4)
\]

Action Prefix

Communication.

\[
\dfrac{g !v .P/C \xrightarrow{g \cdot !v} P/C}{v \in \mathbb{N}} \quad (5)
\]

\[
\dfrac{g !x .P/C \xrightarrow{g \cdot !x} P/C}{C = \langle s, q = \rho, f \rangle \text{ and } x \in q} \quad (6)
\]

For all \( v \in \mathbb{N} \):

\[
\dfrac{g ?x .P/C \xrightarrow{g \cdot ?x} P/C'}{C = \langle s, q = \rho, f \rangle \text{ and } (x, \text{Nat}) \in s} \quad (7)
\]

with \( C = \langle s, q = \rho, f \rangle, C' = \langle s, q = \rho, f \cup \{x \mapsto v\} \rangle \text{ and } (x, \text{Qubit}) \in s \)

For all density matrix \( \nu \) of dimension 2:

\[
\dfrac{g ?x .P/C \xrightarrow{g \cdot ?x} P/C'}{C = \langle s, q = \rho, f \rangle, C' = \langle s, x.q = \nu \otimes \rho, f \rangle} \quad (8)
\]

with

\[
\langle C = \langle s, q = \rho, f \rangle, C' = \langle s, x.q = \nu \otimes \rho, f \rangle \rangle \
\langle x, \text{Qubit} \rangle \in s \text{ and } x \notin q \]

Expression.

\[
\dfrac{v.P/C \xrightarrow{\tau} P/C}{v \in \mathbb{N}} \quad (9)
\]

Probabilities

\[
\dfrac{P/\bigoplus_{p_j} C_j \xrightarrow{p_i} P/C_i}{\sum_j p_j = 1} \quad (10)
\]

\[
\dfrac{\bigoplus_{p_j} P_j/C \xrightarrow{p_i} P/C_i}{\sum_j p_j = 1} \quad (11)
\]

Non-deterministic choice

\[
\dfrac{P/C \rightarrow P'/C'}{P + Q/C \rightarrow P'/C'} \quad P \downarrow, Q \downarrow \quad (12)
\]

where \( \rightarrow \) represents any transition.

\[
\dfrac{P/C \rightarrow_p P'/C'}{P + Q/C \rightarrow_p P'/C' + Q/C'} \quad Q \downarrow \quad (13)
\]
\[
P/C \rightarrow_p P'/C' \quad Q/C \rightarrow_q Q'/C''
\]
\[
P + Q/C \rightarrow_{pq} P' + Q'/C
\]

**Parallel composition** In the rules for parallel composition, \(C, C_P\) and \(C_Q\) are defined as:

- \(C = < (s_P \parallel s_Q).s, q = \rho, f >\)
- \(C_P = < s_P | s, q = \rho, f >\)
- \(C_Q = < s_Q | s, q = \rho, f >\)

In the definition of \(C\), the operator \(\parallel\) permits to build a cactus stack (see paragraph 2.1). In the cactus stack \((s_P \parallel s_Q).s\) of the process \(P \parallel Q\), the names in \(s\) correspond to variables shared by \(P\) and \(Q\) whereas the names in \(s_P\) (resp. \(s_Q\)) correspond to variables declared in \(P\) (resp. \(Q\)).

\[
P/C_P \xrightarrow{\alpha} P'/C' P \parallel Q/C \xrightarrow{\alpha} \parallel P' \parallel Q/C' P, Q \downarrow
\]

where

- If \(C'_P = < s', q' = \rho', f' >\) then \(C' = < (s'_P \parallel s_Q).s, q' = \rho', f' >\) with \(s'_P\) such that \(s' = s'_P | s\) (\(P\) can neither add to nor remove variables from \(s\))
- If \(C'_P = \boxplus_{p_i} s_i, q_i = \rho_i', f_i' >\) then \(C' = \boxplus_{p_i} (s_{p_i} \parallel s_Q).s, q_i = \rho_i', f_i' >\) with \(s_{p_i}\) such that \(s_i = s_{p_i} | s\)

\[
P/C_P \xrightarrow{g \uparrow v} P'/C' P \parallel Q/C_Q \xrightarrow{g \downarrow v} \parallel Q'/C' P, Q \downarrow
\]

where \(v \in \mathbb{N} \), \(C'_Q = < s', q' = \rho', f' >\), and \(C' = < (s_P \parallel s_Q).s, q = \rho, f' >\)

\[
P/C_P \xrightarrow{g \uparrow v} P'/C' P \parallel Q/C_Q \xrightarrow{g \downarrow v} \parallel Q'/C' P, Q \downarrow
\]

where

- \((x, \text{Qubit}) \in s \cup s_Q, x \notin q, v \in \{0, 1\}\)
- \(C' = < (s_P \parallel s_Q).s, x.q = |v\rangle\langle v| \otimes \rho, f >\)

\[
P/C_P \xrightarrow{g \uparrow x} P'/C' P \parallel Q/C_Q \xrightarrow{g \downarrow y} \parallel Q'/C' P, Q \downarrow
\]

where

- \((x, \text{Qubit}) \in s \cup s_P, x \in q\)
- \((y, \text{Qubit}) \in s \cup s_Q, y \notin q\)
- \(C' = < (s_P \parallel s_Q).s\backslash \{x\}, q[x \leftarrow y] = \rho, f >\)

\[
P/C_P \rightarrow_p P'/C' P \parallel Q/C \rightarrow_p P' \parallel Q/C P, Q \downarrow
\]
\[
\frac{P/C_P \longrightarrow_P P'/C'_P \quad Q/C_Q \longrightarrow_Q Q'/C'_Q}{P \parallel Q/C \longrightarrow_{pq} P'' \parallel Q'/C'}
\]