$\pi-$Calculus with Noisy Channels

Mingsheng Ying*
State Key Laboratory of Intelligent Technology and Systems,
Department of Computer Science and Technology,
Tsinghua University, Beijing 1000815, China
Email: yingmsh@tsinghua.edu.cn

Abstract

It is assumed in the $\pi-$calculus that communication channels are always noiseless. But it is usually not the case in the mobile systems that developers are faced with in the real life. In this paper, we introduce an extension of π, called π_N, in which noisy channels may be present. A probabilistic transitional semantics of π_N is given. The notions of approximate (strong) bisimilarity and equivalence between agents in π_N are proposed, and various algebraic laws for them are established. In particular, we introduce the notion of stratified bisimulation which is suited to describe behavior equivalence between infinite probabilistic processes. Some useful techniques for reasoning about approximate bisimilarity and equivalence are developed. We also introduce a notion of reliability in order to compare different behaviors of an agent in π and π_N. It is shown that reliability is preserved by the basic combinators in π. A link between reliability and bisimulation is given. This provides us with a uniform framework in which we can reason about both correctness properties and reliability of mobile systems. Also, a potential way of combing value-passing process algebras and Shannon's information theory is pointed out.

Key Words: Process algebras, Shannon's information theory, Mobile process; Communication; Noisy channel; Probability distribution, Transitional semantics; Bisimulation; Equivalence; Reliability

Contents

*This work was partly supported by the National Foundation of Natural Sciences of China (Grant No: 60273003, 60321002, 601596321) and the Key Grant Project of Chinese Ministry of Education (Grant No: 10403)

1. Introduction

The studies of communication and concurrency in computer science began in the late of 1960's and the early of 1970's. Since then various models and theories of concurrency have been introduced, including Petri nets [47], CSP [34,35], CCS [41,42] and ACP [15]. Among them, CCS is one of the most mathematically developed. Interactive systems whose components communicate may be represented very well in CCS.

The recent development of information technology brings us new challenges. Many systems in the information world are much more complicated than interactive systems. Not only can their subsystems communicate, but they are also able to change their communication linkages and then their topological structures. Such systems are usually called mobile systems. It is well known that CCS cannot express mobility directly. Thus, it becomes a very important problem in theoretical computer science to find satisfactory models of mobile systems. Indeed, as early as in 1986, not long after the invention of CCS, Engberg and Nielsen [26] extended CCS to include mobility. In the late of 1980's, Milner, Parrow and Walker [44] successfully established a basic calculus, namely, $\pi-$calculus, of mobile processes. The $\pi-$calculus has a much greater expressiveness than CCS, and agents modelled in π may not only communicate with each other but also dynamically reconfigure their communication topology. After [44], many variants of π have been proposed; for example, Boudol [17] and Honda and Tokoro [36] introduced the asynchronous $\pi-$calculus, a subset of π in which communication is asynchronous in the sense that the actions of sending a message and receiving it do not need to occur at the same time. Also, some higher-order generalizations have been introduced; see [50], Chapters 12 and 13 for an excellent exposition.

The capacity of dynamic reconfiguration of agents in the $\pi-$calculus is realized by passing communication linkages along channels connecting the involved agents. It is worth noting that in the $\pi-$calculus there is an implicit but essential assumption:

all communication channels are noiseless. This means that what is received at the output of a channel is exactly what was sent through it. However, it is usually not the case when we consider communication in the real world, and channels are often not completely reliable.

The aim of this paper is to introduce a new version of the $\pi-$calculus, called π_N, in which noisy channels are allowed. The language of π_N is the same as that of π. The essential difference between π and π_N lies behind the syntax. Both π and π_N presuppose a set of names, each of which is used to identify a communication channel. As mentioned above, in π it is assumed that each channel is noiseless, although not explicitly stated. But in π_N we are going to deal with a more realistic situation where noise may reside in some channels. According to a basic idea from Shannon's information theory [2, 53], noise may be described in a statistic way. Thus, we associate to each pair of (channel) names x and y a probability distribution $p_x(\cdot|y)$ over the output alphabet (here it is just the set of names): for any name z, $p_x(z|y)$ indicates the probability that z is received from channel x when y is sent along it. Under this new assumption for channels, we have to reexamine the behavior of agents in π, and this enables us to present a transitional semantics of π_N. The transitional semantics of π_N is given in terms of probabilistic transition systems, and probability information arises from noise in channels. On the other hand, the $\pi-$calculus possesses a non-probabilistic structural operational semantics. The difference between π and π_N is mainly caused by the actions performed by an output agent $\overline{x}y.P$. In the $\pi-$calculus this agent will perform the transition $\overline{x}y.P \xrightarrow{\overline{x}y} P$, which means that name y is sent through channel x and the same name will be received at the output of the channel. However, in the π_N-calculus the transition performed by it would be

$$\overline{x}y.P \xrightarrow{\overline{x}z[p_x(z|y)]} P$$

This probabilistic transition indicates that although the name sent by the agent is y, the name received as output on channel x might be the name z, different from y, with the probability $p_x(z|y)$.

Various behavioral equivalences are important concepts in process algebras because they provide a formal description that one system implements another. In [44], the notions of strong bisimilarity and equivalence were generalized into the $\pi-$calculus. Furthermore, they were elaborated and refined by Sangiorgi, Walker and Boreale [16,50]. In this paper, we want to extend these notions of equivalence into the π_N-calculus. Note that an agent in π_N is represented by a probabilistic transition system. In the literature, there have been two kinds of bisimulation for probabilistic processes. The first one is the exact version of probabilistic bisimulation [39], which equates two processes whenever they perform the same actions with the same probabilities. The other is the approximate (and "more probabilistic") version: two processes may be equated by a bisimulation up to a certain degree of difference in the probabilistic transitions [19, 25, 59]. Here, we adopt an approximate version of probabilistic bisimulation, and the notion of $\lambda-$bisimulation is introduced

in π_N. In a $\lambda-$bisimulation, an action of a process should be matched by the same action of another process related to it, but the occurrence probabilities of the action in these two processes may have a difference not exceeding the given threshold λ. It is obvious that the notion of $\lambda-$bisimulation gives us a continuous spectrum of equivalence relations with parameter ranging from 0 to 1. At the top of this spectrum is the exact version of probabilistic bisimulation, and to those agents that cannot be equated by an exact probabilistic bisimulation this spectrum will assign a similarity degree less than 1.

Various algebraic laws in the $\pi-$calculus were established under the assumption that all communication channels are noiseless. These laws provide us with a basis for reasoning about systems modelled in π. One of the main purposes of the present paper is to carefully reexamine these laws in the situation of noisy channels. It is found that some of them are still valid in π_N; for example, commutativity and associativity of summation and parallel composition. However, other laws are no longer true in general, and their validity depend on the nature of noise in the involved channels; see Propositions 6(2), (3), 7(8) and (12) for example.

Behavioral equivalence are used to compare different processes. The π_N-calculus gives us a chance to compare processes from a new angle, in a sense orthogonal to that of behavioral equivalence. We imagine that the ideal behavior of a mobile system is described as an agent P in π. However, communication between its subsystems may be subject to certain disturbance from the environment so that its behavior is not always reliable. Thus, the same agent P can be used to model the real behavior of this system, but in the different framework of π_N. Now what we need to compare is the behaviors of the same agent in different environments, depicted respectively by the calculus π and π_N. A formalization of this issue leads to a notion of reliability.

It is very interesting to note that as early as in 1992 N. Francez [28] already noted the necessity of combining correctness of programs with reliability, and he pointed out:

"Sometimes, the 'very' idea of program verification, using any mathematical or logical method, is criticized. The main argument is that, when programs are actually executed on electronic computers, the prerequisites for successful application (listed in [28], pages 4 and 16) cannot be assumed to hold. Computers, being physical devices, cannot be assumed to behave reliably. In addition, standard implementations at best approximate the formal definition of semantics. Thus, no logical conclusion can be drawn about the real-life behavior of programs, no more than about any other natural phenomena, with absolute certainty."

To the author's best knowledge, however, up to now formal methods (for reasoning about correctness properties of systems) and software reliability are still studied separately. Nevertheless, the π_N-calculus gives a possibility of putting them naturally into a single picture. Suppose that we are designing a mobile system and

write down its specification as an agent S in π. Furthermore, we suppose that an ideal implementation is described also in π as agent I. With the mathematical tools provided by the $\pi-$calculus, we may prove correctness of I with respect to S by establishing a behavioral equivalence between them. However, this is still not enough to guarantee that a real physical implementation built according to I will be properly correct in the sense that it behaves completely as specified by S. The reason is that noise in communication between components of the physical implementation is entirely ignored in correctness reasoning conducted in π. Indeed, a direct and natural modelling of channel noise is not easy to find in the $\pi-$calculus although the $\pi-$calculus has proved, time and again, its ability to encode almost any phenomena, and a noisy channel could probably be modelled by a process that takes the original message as input and produces as output a nondeterministic choice of all possible channel names that could be produced as the result of noise. On the other hand, the π_N-calculus gives rise to a mechanism suited for this purpose. Using formal devices presented in this paper, we are able to establish certain reliability of I in the environment of noisy channels. Thus, by combining π and π_N it is possible to derive a useful connection between the specification and the real implementation and to verify certain correctness properties of the real implementation with respect to the specification. The situation exposed here may be visualized by the following figure:

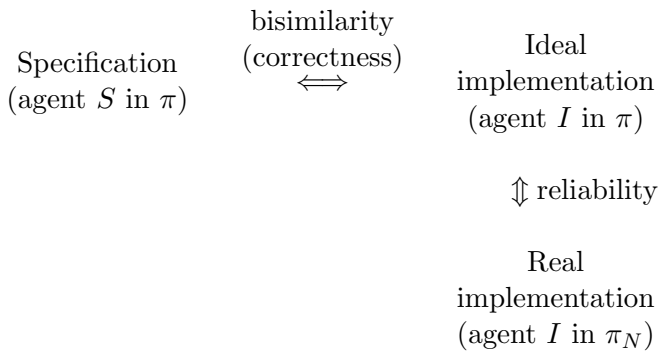

Fig.1

1.1. Overview of the paper

The present paper is an elaboration of the ideas exposed above, and the remainder of it is organized as follows. In Section 2, we recall the syntax and the structural operational semantics of π. As we mentioned above, the syntax of π_N is the same as that of π. Thus, this section also fixes the syntactic notations needed in what follows. The presentation of the operational semantics of π gives us a basis for comparing it with that of π_N.

In Section 3, we review the mathematical model of noisy channels from Shannon's

information theory. What is concerned in this paper is the simplest, memoryless channels, but a definition of more general channels with memory is also presented because it will be used in the studies of some generalizations and variants of π_N, say higher-order π_N or polyadic version of π_N (see some comments on pages 13 and 14).

In Section 4, we start our study on the π_N-calculus by presenting its structural operational semantics. This semantics is given in terms of probabilistic transition systems. Certain distinctions between the transitional rules in π and in π_N are explained. Several examples are given, and the behaviors of some agents in π_N are carefully compared with their behavior in π. Also, several lemmas on probabilistic transitions of agents in π_N are presented. They generalizes Lemmas 1 to 16 in [44, II] and Lemmas 1.4.4 and 1.4.16 in [50], and they will be widely used in establishing some key properties of bisimulation and reliability.

The notions of $\lambda-$strong bisimulation and strong bisimulation degree are introduced in Section 16. Also, various properties of them are established there. In particular, it is shown that $\lambda-$bisimilarity is preserved by the basic combinators in π (and so in π_N), and many equational laws for $\lambda-$bisimilarity are proved. What should be emphasized is that the corresponding results in the $\pi-$calculus are valid universally, but in the π_N-calculus some of them need to impose certain constraints on the nature of communication channels. It is especially interesting that π_N enjoys an expansion law for unfolding parallel composition, but it cannot be expressed in the syntax of π (and π_N). In order to present this expansion law, we have to introduce a probabilistic extension of the language of the $\pi-$calculus.

We propose the concept of $\lambda-$strong equivalence in Section 6. It is defined to be a $\lambda-$equivalence under certain substitutions of names. Also in this section, various algebraic laws for $\lambda-$equivalence are derived, and they generalized the main results of Section 4 in [44, II]. Of course, certain constraints on communication channels are necessary too in order to guarantee the validity of these results. An expansion laws with respect to strong equivalence is also obtained.

It is proved that $\lambda-$bisimilarity is preserved by some substitutions of a single name (see Proposition 16). However, in doing so a quite strict condition on relevant communication channels has to be added. In order to remove part of this condition, in Section 7 we introduce the notion of $(\lambda, \mu)-$stratified bisimulation and use it to give a generalization of Proposition 16. An example is presented to illustrate that $(\lambda, \mu)-$stratified bisimilarity is properly weaker than $\lambda-$bisimilarity. The concept of $\lambda-$bisimulation up to $\mu-$bisimilarity is proposed in Section 16, but it can be applied to derive $\lambda-$bisimilarity between certain agents merely for the case of $\mu = 1$. Now $(\lambda, \mu)-$stratified bisimilarity gives an opportunity of applying the proof technique of $\lambda-$bisimulation up to $\mu-$bisimilarity with $\mu < 1$. It should be pointed out that $(\lambda, \mu)-$stratified bisimulation still needs an elaboration, and its power is to be exploited. We believe that it will become a suitable tool in dealing with recursive constructs in π_N and other probabilistic process algebras.

Section 8 is devoted to develop the theory of reliability in the π_N-calculus.

The notions of $\lambda-$reliability bisimulation and reliability degree are introduced. It is shown that two agents bisimilar both in π and in π_N have the same reliability degree. This shows us an interesting link from reliability to bisimilarity. In addition, some algebraic properties of reliability are derived, and we prove that reliability is preserved by the basic combinators of π under certain conditions. The latter can be seen as a logical support to the modularization technique in analyzing reliability of mobile systems.

Section 9 is the concluding section, several interesting problems for further research are proposed there, and in particular, a potential way to combine value-passing process algebras with Shannon's information theory is pointed out.

1.2. Related works

To conclude the introduction, we briefly expose some related works. This will help us in clarifying the connection of this paper to the existing literature.

The basic idea of developing an extension of the $\pi-$calculus in which noisy communication channels are allowed was already proposed in the concluding section of the author's paper [59] as a problem for further studies, and the first part of the present paper is just an elaboration of this idea. But the idea of reliability of agents in π and its potential link to Shannon's information theory was not mentioned there.

Several formal models of systems with unreliable communication channels have been studied in the literature. In a series of papers [2-8], Abdulla et al. considered systems consisting of finite-state processes that communicate via unbounded FIFO channels which may nondeterministically lose messages. These systems are called lossy channel systems (LCSs), and they are able to model link protocols such as the Alternating Bit Protocol and HDLC. Various verification problems for this class of systems were solved in [2-8]. Since then, lossy channel systems have been investigated by some other authors; for example, see [11, 38, 51]. On the other hand, in [14] Berger and Honda augmented the asynchronous $\pi-$calculus with a timer and with locations, message-loss, location failure and the ability to save process state, and in [12, 13] Berger introduce π_{mlt}, the asynchronous $\pi-$calculus with timer and message failure. It is easy to see that noise in communication channels considered in the current paper is different from the unreliability of channels dealt with in [2-8, 12-14].

This paper is based on Milner, Parrow and Walker's original version [44] of the $\pi-$calculus because the author believes it is the most popular in the process algebra community. A clear and thorough comparison of π and π_N is made in Sections 4-8. This should help us in understanding the major difference between them. Of course, it is possible to introduce noisy channels into various variants of the $\pi-$calculus and its higher-order generalizations.

As pointed out above, in the structural operational semantics of π_N an agent is represented by a probabilistic transition system. The study of probabilistic process

algebras has a history of more than ten years, and there have been a considerable number of papers in the literature devoted to this topic. For example, Hansson and Jonsson [31] proposed a probabilistic extension of CCS, Seidel [52] presented a probabilistic variant of CSP, and Baeten, Bergstra and Smolka [3] introduced a probabilistic counterpart of ACP.

Some probabilistic extensions of the $\pi-$calculus have been proposed by Herescu, Palamidessi, Lu and Wei [32, 33, 40]. In [32, 33], a notion of probabilistic choice was introduced into the asynchronous $\pi-$calculus, and an operational semantics was defined, which distinguishes between probabilistic choice, made internally by the process, and nondeterministic choice, made externally by an adversary scheduler. In [40], only probabilistic choice was considered, and a so-called risk semantics was introduced. Roughly speaking, risk semantics is a probabilistic generalization of Hoare's failure semantics [21, 35]. In addition, Priami [48] presented a random extension of the $\pi-$calculus. The essential difference between probabilistic $\pi-$calculus and random $\pi-$calculus is that time in the former is treated as a continuous (and explicit) variable, but in the latter it is seen as a discrete (and implicit) variable.

What we have to emphasize is that π_N is essentially different from the probabilistic process algebras in the previous literature. A common way of accommodating probability information in various probabilistic extensions of process algebras is to add some probabilistic constructs, such as probabilistic choice, into the languages of non-probabilistic process algebras. But it may be observed that noisy channels cannot be explicitly represented in the syntax of π_N. Instead, they are indicated by an assumption behind the language. On the other hand, translations of the π_N-calculus and a probabilistic extension of the $\pi-$calculus into each other seems possible.

In order to provide some useful mathematical tools for describing approximate correctness and evolution of concurrent systems, the author [57] tried to develop topology in process algebras. In particular, he and Wirsing [61] introduced the notions of $\lambda-$bisimulation and approximate bisimilarity in CCS equipped with a metric on its set of action names. These notions were further applied by the author to probabilistic processes [59]. It was noted by Feng and Zhang [27] that the definition of $\lambda-$bisimulation given in [59] needs a slight modification because it is not preserved by summation. This paper follows [69, 61] and use $\lambda-$bisimilarity to act as an approximate behavioral equivalence between agents in π_N. It should be pointed out that the idea of approximate bisimilarity has also been proposed by some other researchers. For example, Desharnais, Gupta, Jagadeesan and Panangaden [25] formalized a notion of approximate bisimilarity by defining a metric on labelled Markov processes, and van Breugel and Worrell [19] gave a co-inductive definition of approximate bisimilarity using the Hutchinson metric on probability measures. In addition, a related idea of approximate correctness has been applied by the author to probabilistic sequential programs [60].

2. $\pi-$Calculus

For convenience of the reader, in this section we recall from [44] the syntax and the transitional semantics of the $\pi-$calculus. Another purpose of this section is to fix notations used in the sequel.

We presuppose in the $\pi-$calculus a countably infinite set $\mathbf{N}$ of names, and we shall let $u, v, w, x, y, z, ...$, with or without subscripts, act as meta-variables ranging over names. The set $\mathbf{N}$ is taken to be infinite mainly for the purpose of avoiding name capture uniformly in the theoretical development. Also, we assume a set $\mathbf{K}$ of agent identifiers, each assigned an arity, a non-negative integer. We shall use $A, B, C, ...$ to range over agent identifiers. And, we employ $P, Q, R, ...$ to serve as meta-variables of agent or process expressions. Then the syntax of π may be presented by the following BNF grammar:

$$P ::= \mathbf{0} \mid \overline{y}x.P \mid y(x).P \mid \tau.P \mid P_1 + P_2 \mid P_1|P_2 \mid (x)P \mid [x = y]P \mid A(y_1, ..., y_n)$$

where n is the arity of A, $\tau \notin \mathbf{N}$ is called a silent action, and $\mathbf{0}$ is a designated agent symbol which can do nothing and so is called inaction.

The above syntax of π is exactly the original one given in [44]. Indeed, after [44] several simplified versions of π have been introduced. For example, in the asynchronous $\pi-$calculus [17, 23], name matching is removed and summation is allowed only on input prefixes and τ's since they often cause trouble, and output prefixing is also removed since it can be encoded with asynchronous communications. In this paper, however, we still adopt the original version of the $\pi-$calculus. The main reason is that we hope to carefully compare our results with those in [44]. In particular, we keep (unrestricted) summation and name matching because they are needed in presenting the expansion law from which we can see an essential difference between π and π_N. On the other hand, we use output prefixes because with them some examples can be presented in a much simpler way.

We need to fix some syntactic notations before going forward. The input prefix $y(x)$ and restriction (x) bind the name x. The free names and bound names are defined in the standard way and their detailed definitions are omitted here. We denote the free names and bound names in P by $\mathrm{fn}(P)$ and $\mathrm{bn}(P)$, respectively, and we write $\mathrm{n}(P)$ for the names of P; that is, $\mathrm{n}(P) = \mathrm{fn}(P) \cup \mathrm{bn}(P)$.

A substitution is a function σ from $\mathbf{N}$ to itself such that $\sigma x = x$ for all but a finite number of names x in $\mathbf{N}$. Let $\widetilde{x} = \{x_1, ..., x_m\}$ and $\widetilde{y} = \{y_1, ..., y_m\}$ be two vectors of names. Suppose further that $x_1, ..., x_m$ are distinct (but $y_1, ..., y_m$ do not necessarily have to be distinct). If $\sigma(x_i) = y_i$ for each $i \leq m$ and $\sigma(x) = x$ for any $x \in \mathbf{N} - \{x_1, ..., x_m\}$, then σ is often written as $\{\widetilde{y}/\widetilde{x}\}$. For any subset X of $\mathbf{N}$, we use $\sigma\lceil X$ to denote the restriction of σ on X. More precisely, $\sigma\lceil X(x) = \sigma(x)$ if $x \in X$ and otherwise $\sigma\lceil X(x) = x$. Suppose that P is an agent and $\sigma = \{\widetilde{y}/\widetilde{x}\}$. We write $P\sigma$ or $P\{\widetilde{y}/\widetilde{x}\}$ for the resulting expression of the simultaneous substitution of y_i for all free occurrences of x_i for each $i \leq m$. Of course, in defining $P\sigma$ we need to

change some bound names to avoid name captures. This can be done in a familiar way and here we omit the details (see [44], part II, Definition 6).

The relation of alpha-convertibility between agents is defined in the standard way, and it is denoted by $\equiv_\alpha$ (see [44], part II, Definition 7).

The language of π is easily understandable from its notation for the reader who is familiar with the process algebra CCS. The output prefix $\overline{y}x.P$ sends the name x along channel y and then behaves like P. The input prefix $y(x).P$ receives any name z along channel y, putting it into all free occurrences of x, and then behaves like $P\{z/x\}$. The silent prefix $\tau.P$ performs the silent action τ and then becomes P. The summation $P_1 + P_2$ behaves like one of P_1 and P_2. The agents P_1 and P_2 in the parallel composition $P_1|P_2$ can act independently, and they may also communicate with each other. It is noteworthy that here the communication is not only a simple synchronization, as in the pure CCS; it is instead a process of name-passing: P_1 or P_2 sends a name along a channel, and the other receives this name along the same channel. The restriction $(x)P$ behaves like P except that communication with its environment are not allowed along channel x, which is private to P. The match $[x = y]P$ is an agent who acts like P whenever the names x and y are the same, and can do nothing if they are different. It is assumed that each agent identifier A of arity n has a unique defining equation of the form:

$$A(\widetilde{x}) \stackrel{def}{=} P$$

where $\widetilde{x} = \{x_1, ..., x_n\}$ is a vector of n distinct names, and $\mathrm{fn}(P) \subseteq \widetilde{x}$. Here, P is called the defining agent of $A(\widetilde{x})$. This equation decrees that for any vector $\widetilde{y}$ of names with length n, $A(\widetilde{y})$ behaves like $P\{\widetilde{y}/\widetilde{x}\}$. Following [44], to avoid pathological technical difficulties we always assume that for any agent identifier A, $\mathrm{bn}(A(\widetilde{x}))$ is finite, and if $A(\widetilde{x}) \stackrel{def}{=} P$ then $\mathrm{bn}(A(\widetilde{y})) = \mathrm{bn}(P\{\widetilde{y}/\widetilde{x}\})$ (see [44], part II, Definition 4).

An agent identifier is weakly guarded in an agent P if it always lies within some prefix sub-term $\tau.Q$ or $\overline{x}y.Q$ or $x(y).Q$ of P. An agent is said to be weakly guarded if all agent identifiers in it is weakly guarded. Agents containing no agent identifiers are called finite agents.

The above intuitive explanation of various agent expressions may be formally reformulated in the transitional semantics of π. To present this semantics, we first isolate the actions performed by agents. There are four kinds of actions in the $\pi-$calculus: the silent action τ, input actions $x(y)$, free output actions $\overline{x}y$, and bound output actions $\overline{x}(y)$. Free output actions and bound output actions will collectively be called output actions, and the silent action and free output actions will be called free actions. As we saw before, τ stands for an internal action within an agent, $x(y)$ is the action of receiving any name from the channel x and then putting it into the places held by y, $\overline{x}y$ means that an agent emits the free name y along the channel x, and $\overline{x}(y)$ means that an agent emits a private name along the channel x and y is a reference to where this private name will go. We write Act for the set of actions; that is, $Act = \{\tau, x(y), \overline{x}y, \overline{x}(y) : x, y \in \mathbf{N}\}$. Let meta-variables α,

β, ... range over actions. If $\alpha = \overline{x}y$ or $\alpha = x(y)$ or $\alpha = \overline{x}(y)$, then x is called the subject and y the object or parameter of α. We define free names and bound names in an action α by

$$\mathrm{fn}(\alpha) = \begin{cases} \emptyset, & \text{if } \alpha = \tau \\ \{x, y\}, & \text{if } \alpha = \overline{x}y \\ \{x\}, & \text{if } \alpha = x(y) \text{ or } \overline{x}y \end{cases}$$

and

$$\mathrm{bn}(\alpha) = \begin{cases} \emptyset, & \text{if } \alpha = \tau \text{ or } \overline{x}y \\ \{y\}, & \text{if } \alpha = x(y) \text{ or } \overline{x}(y) \end{cases}$$

Also, we write $\mathrm{n}(\alpha) = \mathrm{fn}(\alpha) \cup \mathrm{bn}(\alpha)$ for the names in α.

Now the structural operational semantics of the $\pi-$calculus may be given by a family of transition relations $\xrightarrow{\alpha}$ ($\alpha \in Act$), which are defined to be the smallest relations satisfying the rules of action in the following table:

TAU-ACT:

$$\frac{}{\tau.P \xrightarrow{\tau} P}$$

OUTPUT-ACT:

$$\frac{}{\overline{x}y.P \xrightarrow{\overline{x}y} P}$$

INPUT-ACT:

$$\frac{}{x(z).P \xrightarrow{x(w)} P\{w/z\}} \qquad w \notin \mathrm{fn}((z)P)$$

SUM:

$$\frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}$$

MATCH:

$$\frac{P \xrightarrow{\alpha} P'}{[x = x]P \xrightarrow{\alpha} P'}$$

IDE:

$$\frac{P\{\widetilde{y}/\widetilde{x}\} \xrightarrow{\alpha} P'}{A(\widetilde{y}) \xrightarrow{\alpha} P'} \qquad A(\widetilde{x}) \stackrel{def}{=} P$$

PAR:

$$\frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \qquad \mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$$

COM:

$$\frac{P \xrightarrow{\overline{x}y} P' \qquad Q \xrightarrow{x(z)} Q'}{P|Q \xrightarrow{\tau} P'|Q'\{y/z\}}$$

CLOSE:

$$\frac{P \xrightarrow{\overline{x}(w)} P' \qquad Q \xrightarrow{x(w)} Q'}{P|Q \xrightarrow{\tau} (w)(P'|Q')}$$

RES:

$$\frac{P \xrightarrow{\alpha} P'}{(y)P \xrightarrow{\alpha} (y)P'} \qquad y \notin \mathrm{n}(\alpha)$$

OPEN:

$$\frac{P \xrightarrow{\overline{x}y} P'}{(y)P \xrightarrow{\overline{x}(w)} P'\{w/y\}} \qquad y \neq x, w \notin \mathrm{fn}((y)P')$$

Note that a symmetric form of the rule SUM is omitted in the above table, and this is also the case for the PAR rule, the COM rule and the CLOSE rule. The rule for input agents is presented in the scheme of early instantiation (for a detailed discussion of early instantiation and later instantiation, we refer to [44], part II, Section 1.3.1).

3. Noisy Channels

A fundamental assumption in π_N is that communication channels may be noisy; that is, their inputs are subject to certain disturbances in transmission. In other

words, the communication situation is conceived as that an input is transmitted through a channel and the output is produced at the end of the channel, but the output is often not completely determined by the input. For convenience of the reader, in this section we recall briefly the notion of channel in Shannon's information theory; for more details, see [9, 53].

In the mathematical model of channels from statistic communication theory, the noisy nature is usually described by a probability distribution over the output alphabet. This distribution of course depends on the input and in addition it may depend on the internal state of the channel. Usually, communication channels are divided into two categories: discrete channels and continuous channels. It is obvious that what we consider in the π_N-calculus are discrete channels. Both the input alphabet and output alphabet are taken to be the set $\mathbf{N}$ of names. In the case of discrete channels, if the sender emits a sequence $x_1, ..., x_n$ of inputs along a channel, then the receiver will get a sequence $y_1, ..., y_n$ of the same length from the channel. Note that the output sequence $y_1, ..., y_n$ is determined by not only the input sequence $x_1, ..., x_n$ but also the internal state s of the channel at the time the inputs are applied. Formally, a channel may be defined to be a family of probability distributions

$$p_n(y_1, ..., y_n|x_1, ..., x_n; s)$$

where $x_1, ..., x_n, y_1, ..., y_n \in \mathbf{N}$, $n = 1, 2, ...$, and s is assumed from a set S called the set of states. This family of probability distributions is of course required to satisfy the following two conditions:

(1) $p_n(y_1, ..., y_n|x_1, ..., x_n; s) \geq 0$ for all $n \geq 0$, $x_1, ..., x_n, y_1, ..., y_n \in \mathbf{N}$ and s; and

(2)

$$\sum_{y_1,...,y_n \in \mathbf{N}} p_n(y_1, ..., y_n|x_1, ..., x_n; s) = 1$$

for all $n \geq 0$, $x_1, ..., x_n \in \mathbf{N}$ and $s \in S$.

The intuitive interpretation of the quantity $p_n(y_1, ..., y_n|x_1, ..., x_n; s)$ is the probability that the sequence $y_1, ..., y_n$ will be received at the output when the input sequence $x_1, ..., x_n$ is sent and the internal state of the channel at that time is s. It should be pointed out that in the π_N-calculus each channel is represented by a family of probability distributions and at the same time we need a name to give it an unambiguous identity.

It is easy to see that the channel defined above has memory; that is, the distribution of the output y_n may depend on the previous inputs $x_1, ..., x_{n-1}$ and outputs $y_1, ..., y_{n-1}$. As the first step in the study of value-passing process algebras with noisy channels, here we only deal with a special class of channels, namely memoryless channels. In the continuation of this paper we shall consider channels with memory. In particular, the factor of memory will become serious when we examine higher-order processes [49, 54, 55] where a process may be transmitted through a channel because a process usually consists of more than one symbol. Another situation where memory of channels may concern us is the polyadic variants of π_N. π_N is an extension of the original, monadic π, which is limited in that only single

names can be communicated. However, polyadic π [43] allows tuples of names to be sent. Then memory of noisy channels cannot be overlooked in a polyadic version of π_N. Indeed, even in the monadic case memory of channels can also be naturally incorporated into π_N. A possible way is to enrich the labels of transitions so that they record not only the current action but also those in the history of process performance. Such an idea is indeed borrowed from Boudol and Castellani's non-interleaving semantics [18] for CCS in which the notion of proved transition is employed.

Memoryless channels are characterized by the following two conditions:

(i) The distribution functions $p_n(y_1, ..., y_n|x_1, ..., x_n; s)$ do not depend on the internal state s, and thus they may be rewritten as $p_n(y_1, ..., y_n|x_1, ..., x_n)$; and

(ii)

$$p_n(y_1, ..., y_n|x_1, ..., x_n) = \prod_{i=1}^{n} p_1(y_i|x_i)$$

for all $n \geq 0$ and $x_1, ..., x_n, y_1, ..., y_n \in \mathbf{N}$.

Intuitively, the second condition means that in a memoryless channel successive symbols are acted on independently. If we introduce the quantities

$$p_n(y_1, ..., y_{n-k}|x_1, ..., x_n) = \sum_{y_{n-k+1}, ..., y_n \in \mathbf{N}} p_n(y_1, ..., y_n|x_1, ..., x_n) \quad (1 \leq k \leq n-1)$$

and

$$p_n(y_n|x_1, ..., x_n; y_1 ..., y_{n-1}) = \frac{p_n(y_1, ..., y_n|x_1, ..., x_n)}{p_n(y_1, ..., y_{n-k}|x_1, ..., x_n)},$$

then condition (ii) may be equivalently stated as follows:

(a) $p_n(y_n|x_1, ..., x_n; y_1, ..., y_{n-1}) = p_1(y_n|x_n)$ for all $x_1, ..., x_n, y_1, ..., y_n \in \mathbf{N}$; and

(b) $p_n(y_1, ..., y_{n-k}|x_1, ..., x_n) = p_{n-k}(y_1, ..., y_{n-k}|x_1, ..., x_{n-k})$ for all $x_1, ..., x_n, y_1,$ $..., y_{n-k} \in \mathbf{N}$ and $1 \leq k \leq n-1$.

Here, $p_n(y_1, ..., y_{n-k}|x_1, ..., x_n)$ may be explained as the probability that the first $n-k$ output symbols will be $y_1, ..., y_{n-k}$ if the input sequence is $x_1, ..., x_n$, and $p_n(y_n|x_1, ..., x_n; y_1 ..., y_{n-1})$ is the conditional probability that the nth output symbol will be y_n given that the input sequence is $x_1, ..., x_n$ and the first $n-1$ output symbols are $y_1, ..., y_{n-1}$. Condition (a) is a clear indication of the memoryless feature, and condition (b) shows the non-anticipatory behavior of the channel.

From the above condition (ii) it is clear that a memoryless channel is completely described by its channel matrix

$$[p(y|x)]_{x,y \in \mathbf{N}}$$

where $p(y|x)$ is the abbreviation of $p_1(y|x)$.

A typical example of memory channel is the binary symmetric channel. Suppose that x_0 and x_1 are in $\mathbf{N}$. In this channel, x_i is correctly transmitted with probability

p, and is incorrectly changed to x_{1-i} with probability $1-p$ $(i=0,1)$. Thus, the channel matrix is

$$\begin{pmatrix} p & 1-p & 0 \\ 1-p & p & \\ 0 & & I_{\mathbf{N}-\{x_0,x_1\}} \end{pmatrix}$$

where $I_{\mathbf{N}-\{x_0,x_1\}}$ is the identity matrix over $\mathbf{N}-\{x_0,x_1\}$.

The channel matrix of a noiseless channel is the identity matrix over $\mathbf{N}$, and it is simply given by

$$p(y|x) = \begin{cases} 1, & \text{if } y = x \\ 0, & \text{otherwise} \end{cases}$$

We shall see that the results in π_N obtained below degenerate to the corresponding ones in π when this kind of channel matrices are considered.

4. Transitional Semantics of π_N

The syntax of π_N is entirely the same as that of the $\pi-$calculus. The essential difference between π and π_N arises from an implicit assumption about communication channels. In the $\pi-$calculus, although not explicitly stated, it was supposed that every channel is noiseless. This means that in a communication along such a channel the receiver will always get exactly what the sender delivers. In the π_N, however, we assume that some noise may reside in a communication channel. This is obviously a more realistic assumption. As already pointed out in Section 3, in this paper we only consider memoryless channels. Thus, we may suppose that each name $x \in \mathbf{N}$ has a channel matrix

$$M_x = [p_x(z|y)]_{y,z\in\mathbf{N}}$$

where $p_x(z|y)$ is the probability that the receiver will get the name z at the output when the sender emits the name y along the channel x.

The structural operational semantics of π_N is also given by a family $\xrightarrow{\alpha[p]}$ $(\alpha \in Act, 0 < p \leq 1)$ of transition relations. It is different from the operational semantics of π in that what concern us here are probabilistic transition relations. We shall see shortly that randomness represented by probability values p comes completely from noise in communication channels. A probabilistic transition in the π_N is of the form

$$P \xrightarrow{\alpha[p]} Q$$

where P and Q are two processes, $\alpha \in Act$ and $0 < p \leq 1$. The intuitive meaning of this transition is that after performing action α the agent P will become Q with the probability p. It is clear that this transition has one more parameter p than the transitions in the $\pi-$calculus. For the case of $p = 1$, we shall omit it and simply write $P \xrightarrow{\alpha} Q$ for $P \xrightarrow{\alpha[p]} Q$. The family $\xrightarrow{\alpha[p]}$ $(\alpha \in Act, 0 < p \leq 1)$ of transition relations is defined by the following set of inference rules:

OUTPUT-ACT:

$$\frac{--}{\overline{x}y.P \xrightarrow{\overline{x}z[p_x(z|y)]} P} \qquad p_x(z|y) > 0$$

SUM:

$$\frac{P \xrightarrow{\alpha[p]} P'}{P + Q \xrightarrow{\alpha[p]} P'}$$

MATCH:

$$\frac{P \xrightarrow{\alpha[p]} P'}{[x = x]P \xrightarrow{\alpha[p]} P'}$$

IDE:

$$\frac{P\{\widetilde{y}/\widetilde{x}\} \xrightarrow{\alpha[p]} P'}{A(\widetilde{y}) \xrightarrow{\alpha[p]} P'} \qquad A(\widetilde{x}) \stackrel{def}{=} P$$

PAR:

$$\frac{P \xrightarrow{\alpha[p]} P'}{P|Q \xrightarrow{\alpha[p]} P'|Q} \qquad \mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$$

COM:

$$\frac{P \xrightarrow{\overline{x}y[p]} P' \qquad Q \xrightarrow{x(z)} Q'}{P|Q \xrightarrow{\tau[p]} P'|Q'\{y/z\}}$$

CLOSE:

$$\frac{P \xrightarrow{\overline{x}(w)[p]} P' \qquad Q \xrightarrow{x(w)} Q'}{P|Q \xrightarrow{\tau[p]} (w)(P'|Q')}$$

RES:

$$\frac{P \xrightarrow{\alpha[p]} P'}{(y)P \xrightarrow{\alpha[p]} (y)P'} \qquad y \notin \mathrm{n}(\alpha)$$

OPEN:

$$\frac{P \xrightarrow{\overline{x}y[p]} P'}{(y)P \xrightarrow{\overline{x}(w)[p]} P'\{w/y\}} \qquad y \neq x, w \notin \mathrm{fn}((y)P')$$

The appearances of TAU-ACT and INPUT-ACT are the same as in the $\pi-$calculus, and we omit them here. However, it should be noted that in π_N these two rules indeed have the probability parameter p, and its absence in the presentation is just because p is always 1. It is obvious that all input actions have probability $p = 1$. But it is not the case for τ actions because τ actions have another possibility rather than those given by $\tau-$prefix forms, namely, they may come from the applications of the rules COM and CLOSE, and the probability of τ actions in the conclusion transitions of these rules is inherited from the (free or bound) output actions in the premises, which may be less than 1. The symmetric forms of the rules SUM, PAR, COM and CLOSE are omitted too.

All rules except OUTPUT-ACT are easily understandable. In fact, they are just simple imitations of the corresponding rules in the $\pi-$calculus, and the only difference is that a probability parameter p is added. The OUTPUT-ACT rule deserves a careful explanation. OUTPUT-ACT represents the noisy nature of channels in the π_N-calculus. It means that the agent $\overline{x}y.P$ of output prefix form sends the name y through the channel x, but what the receiver gets at the output of this channel may not be y due to noise residing in it, and the name z will be received with the probability $p_x(z|y)$. Indeed, OUTPUT-ACT is the unique rule which all differences between π and π_N come from.

In the above OUTPUT-ACT rule, the action performed by $\overline{x}y.P$ is not $\overline{x}y$ but $\overline{x}z$, and the probability $p_x(z|y)$ that y becomes z in channel x is indicated. This may be thought of as that noise happens at the end of sending, not at the end of receiving. We call this design decision the early noise instantiation schema. There is an alternative treatment, named the late noise instantiation schema, to noise in the transitional semantics of π_N. In such a schema, we keep the OUTPUT-ACT rule from π unchanged, and thus output transitions will not carry any probability information. But the COM rule has to be modified as follows:

$$\frac{P \xrightarrow{\overline{x}y} P' \qquad Q \xrightarrow{x(w)} Q'}{P|Q \xrightarrow{\tau[p_x(z|y)]} P'|Q'\{z/w\}}$$

The CLOSE rule needs a similar modification, and the OPEN rule is taken to be the same as that in π. This is quite similar to the case where the early and late bisimulations are distinguished (see [44, II], Section 2.3). It is worth noting that the early (name) instantiation is usually preferred because it coincides with barbed bisimulation in many cases, as shown by Milner and Sangiorgi [45]. An elaboration of the late noise instantiation schema is deferred to another paper.

Probabilistic models of processes were divided by van Glabbeek et al. [30, 31] into three categories in accordance with the relationship between occurrences of actions and transition probabilities: (a) in a reactive system, for any process state, a separate probability distribution is associated with the outgoing transitions labelled by the same action, and choice among actions is nondeterministic and it is made by the environment; (b) in a generative system, all outgoing transitions are governed by a single probability distribution, regardless of the action names labelling these transitions; and (c) the stratified model is an extension of the generative one, and it allows for level-wise and nested probabilistic branching. For any agent P and for any action α, we write

$$p(P,\alpha) = \sum\{|p : P \xrightarrow{\alpha[p]} P' \text{ for some } P'|\}$$

where $\{|,|\}$ stands for the multi-set brackets, and $p(P) = \sum_\alpha p(P,\alpha)$. Then in a reactive system it is usually required that $p(P,\alpha) \in \{0,1\}$ for any agent P and action α, and in a generative system, it is required that $p(P) \in \{0,1\}$ for any agent P. Sometimes, a sub-stochastic approach is adopted, that is, the above two conditions are weakened to $p(P,\alpha) \leq 1$ for any agent P and action α, and $p(P) \leq 1$ for any agent P, respectively. However, from the SUM and PAR rules we can see it may happens in the transitional semantics of π_N that $p(P,\alpha) > 1$ for some agent P and action α. This contradicts in a sense our intuition for a probability distribution. In the literature on probabilistic processes, a standard method to overcome this objection is the normalization procedure. Concretely, we can use $\frac{p}{p(P,\alpha)}$ to replace the probability p in each transition $P \xrightarrow{\alpha[p]} P'$ of agent P with action α. Unfortunately, the normalization procedure often makes reasoning in a theory of probabilistic processes extremely complicated, for example see [23, 24, 46]. We decide not to use normalization in our transitional semantics of π_N. (Of course, we have to treat in mind transition probabilities in the normalized way.) This allows us to have a much simpler theory of behavior equivalence and reliability in π_N.

There is another assumption for probabilistic transition systems in the literature: for any agents P and P', for any $\alpha \in Act$, and for any $p_1, p_2 \in (0,1]$, if $P \xrightarrow{\alpha[p_1]} P'$ and $P \xrightarrow{\alpha[p_2]} P'$, then $p_1 = p_2$. However, from the SUM rule we can see that it is not the case in π_N. For example, if $p_x(z|y_1) = \frac{1}{2}$ and $p_x(z|y_2) = \frac{1}{3}$, then

$$\overline{x}y_1.\mathbf{0} + \overline{x}y_2.\mathbf{0} \xrightarrow{\overline{x}z[\frac{1}{2}} \mathbf{0}$$

and

$$\overline{x}y_1.\mathbf{0} + \overline{x}y_2.\mathbf{0} \stackrel{\overline{x}z[\frac{1}{3}}{\longrightarrow} \mathbf{0}$$

Some authors believe this is unreasonable according to the usual understanding about probability distributions, and a way to deal with this problem in the previous works on probabilistic processes is to modify a probabilistic transition system by adding up all possible values of transition probability with the same source and target agents and the same action (for instance, see [29, 30, 59]). More precisely, suppose that we derive the probabilistic transitions $P \stackrel{\alpha[p_i]}{\longrightarrow} P'$ from the original rules. Then the modified transition would be $P \stackrel{\alpha[\sum_i p_i]}{\longrightarrow} P'$ (sometimes, we also need to normalize the transition probabilities). Once again, this kind of modification highly complicates a theory of probabilistic processes. In this paper, we choose not to modify the probabilistic transitional semantics of π_N, and we understand the values of transition probability in a different way. We think that the probability of transition $P \stackrel{\alpha}{\longrightarrow} P'$ is one of these p_i's, but we do not exactly know what is its value, and the choice among these p_i's is made by the environment. Furthermore, the set $\{p_i : P \stackrel{\alpha[p_i]}{\longrightarrow} P'\}$ can be understood as an imprecise probability value of the transition $P \stackrel{\alpha}{\longrightarrow} P'$. Indeed, imprecise probability distributions have been widely studied in the communities of Statistics and Artificial Intelligence (for a detailed exposition, we refer to [56]).

To illustrate further these transition rules, let us consider some simple examples. For convenient comparison, these examples are all taken from [44, I]. The purpose of choosing these examples is to see how the same agent behaves in different ways when they are considered in π and π_N respectively.

Example 1. Let

$$P \stackrel{def}{=} \overline{a}5.P',$$

$$Q \stackrel{def}{=} a(x).Q'$$

and

$$R \stackrel{def}{=} (a)(P|Q).$$

The expression R describes that an agent P wishes to send a value 5 to an agent Q, along a private channel a. In the $\pi-$calculus, the unique transition of R is

$$R \stackrel{\tau}{\longrightarrow} (a)(P'|Q'\{5/x\})$$

We now treat it in the π_N-calculus. Suppose that $p_a(5|5) = 0.8$, $p_a(4|5) = p_a(6|5) = 0.1$ and $p_a(x|5) = 0$ for other names x. Then we have

$$P \stackrel{\overline{a}5[0.8]}{\longrightarrow} P',$$

$$P \stackrel{\overline{a}4[0.1]}{\longrightarrow} P'$$

and

$$P \stackrel{\overline{a}6[0.1]}{\longrightarrow} P'$$

by the OUTPUT-ACT rule. Furthermore, it follows that

$$R \stackrel{\tau[0.8]}{\longrightarrow} (a)(P'|Q'\{5/x\}),$$

$$R \stackrel{\tau[0.1]}{\longrightarrow} (a)(P'|Q'\{4/x\})$$

and

$$R \stackrel{\tau[0.1]}{\longrightarrow} (a)(P'|Q'\{6/x\})$$

from INPUT-ACT, COM and RES. So, the agent Q receives a wrong value 4 or 6 with the total probability 0.2. □

Example 2. Suppose that an agent P wishes to delegate a new agent, Q, the task of transmitting a value 5 to another agent R. In the notation of π we may write:

$$P \stackrel{def}{=} \overline{b}a.\overline{b}5.P',$$

$$Q \stackrel{def}{=} b(y).b(z).\overline{y}z.\mathbf{0}$$

and

$$R = a(x).R'.$$

Let

$$S \stackrel{def}{=} (a)(b)(P|Q|R)$$

Then in the $\pi-$calculus the agent S behaves in the following way:

$$S \stackrel{\tau}{\longrightarrow} (a)(b)(\overline{b}5.P'|b(z).\overline{a}z.\mathbf{0}|a(x).R')$$

$$\stackrel{\tau}{\longrightarrow} (a)(b)(P'|\overline{a}5.\mathbf{0}|a(x).R')$$

$$\stackrel{\tau}{\longrightarrow} (a)(b)(P'|\mathbf{0}|R'\{5/x\})$$

On the other hand, in the π_N-calculus, we assume that $p_b(a|a) = 0.95$, $p_b(c|a) = 0.05$, $p_b(5|5) = 1$ and $p_b(u|a) = p_b(v|5) = 0$ for all names $u \neq a, c$ and for all $v \neq 5$. Then one of the possible transitions of S is:

$$S \stackrel{\tau[0.05]}{\longrightarrow} (a)(b)(\overline{b}5.P'|b(z).\overline{c}z.\mathbf{0}|a(x).R')$$

$$\stackrel{\tau}{\longrightarrow} (a)(b)(P'|\overline{c}5.\mathbf{0}|a(x).R')$$

Now the value 5 cannot be sent to R because what Q receives from P is a wrong channel name, which is different from the channel used by R. So, the probability that the agent Q fails to fulfil the task is 0.05. □

Example 3. In the $\pi-$calculus, scope intrusion happens in the situation where an agent P wishes to pass a link to Q but Q already has, with another agent S, a private link of the same name as that of the link which P sends to it. More precisely, let

$$P \stackrel{def}{=} \overline{y}x.P'$$

and

$$Q \stackrel{def}{=} y(z).Q',$$

and we further suppose that P has the link x to R, and Q also has the link x to S but x is private between Q and S. Once P sends x to Q along the channel y, it will intrude the private scope of x between Q and S. To avoid this scope intrusion, the private link x must be renamed. This case can be described by the following transition:

$$P|R|(x)(Q|S) \stackrel{\tau}{\longrightarrow} P'|R|(x')(Q'\{x'/x\}\{x/z\}|S\{x'/x\}$$

where x' is a fresh name. However, in the π_N-calculus, it may be that scope intrusion does not happen in such a situation with a certain probability. For instance, suppose that $p_y(x|x) = 0.7$ and $p_y(w|x) = 0.3$, where w is a new name. Then it holds that

$$P|R|(x)(Q|S) \stackrel{\tau[0.3]}{\longrightarrow} P'|R|(x)(Q'\{w/z\}|S)$$

Here, although the link that P passes to Q is still x, we escape from scope intrusion with the probability 0.3. The reason is that noise in the channel y makes Q to receive a wrong link w, which has no name conflict with the private link between Q and S. $\square$

Example 4. We now examine scope extrusion. Let both P and Q be as in Example 3. In addition, we assume that x is a private link between P and agent R. Whenever x is passed from P to Q, P extrudes the scope of the private link x. In the $\pi-$calculus, this may be formally depicted by the following transition:

$$(x)(P|R)|Q \stackrel{\tau}{\longrightarrow} (x)(P'|R|Q'\{x/z\})$$

where x is supposed not to be free in Q. (If x is free in Q, then the transition will become a little bit more complicated:

$$(x)(P|R)|Q \stackrel{\tau}{\longrightarrow} (x')(P'\{x'/x\}|R\{x'/x\}|Q'\{x'/z\})$$

where a fresh name x' is taken to avoid name conflict.) In Example 3 we saw that scope intrusion may be absent in π_N with a small probability in certain situation where it occurs in π. However, here it will be shown that there is the possibility that scope extrusion happens in π_N but not in π. Let us consider the agent $(w)(P|R)|Q$. Note that the restriction (w) does not bind the name x that P wishes to send along the link y to Q. Then in the $\pi-$calculus we have the transition:

$$(w)(P|R)|Q \stackrel{\tau}{\longrightarrow} (w)(P'|R)|Q'\{x/z\}$$

no matter $x \in \mathrm{fn}(Q)$ or not, and there is no scope extrusion. On the other hand, in the π_N-calculus, if we suppose that $p_y(x|x) = 0.7$ and $p_y(w|x) = 0.3$, as in Example 3, then the rules OUTPUT-ACT, PAR and OPEN lead to

$$P \stackrel{\overline{y}w[0.3]}{\longrightarrow} P',$$

$$P|R \stackrel{\overline{y}w[0.3]}{\longrightarrow} P'|R$$

and

$$(w)(P|R) \stackrel{\overline{y}(w)[0.3]}{\longrightarrow} P'|R$$

So, it holds that

$$(w)(P|R)|Q \stackrel{\tau[0.3]}{\longrightarrow} (w)(P'|R|Q'\{w/z\})$$

provided $w \notin \mathrm{fn}(Q)$, or more general,

$$(w)(P|R)|Q \stackrel{\tau[0.3]}{\longrightarrow} (w')(P'\{w'/w\}|R\{w'/w\}|Q'\{w'/z\})$$

where w' is a fresh name. In this transition, scope extrusion happens with the probability 0.3. □

Noise in communication channels may make that some wrong channel names are received by the involved agents. As shown in the above examples, these wrong channel names will be used as communication linkages in the further process of communication. Thus, noise channels can even change the topological structures and dynamic configurations of mobile systems.

The remainder of this section is devoted to establish some basic properties of probabilistic transitions in π_N. These properties are very useful in examining various algebraic laws of probabilistic bisimilarity defined in the next section. We begin with a comparison between input actions in π and π_N.

Lemma 1. *$P \stackrel{x(y)}{\longrightarrow} P'$ in π if and only if $P \stackrel{x(y)}{\longrightarrow} P'$ in π_N.*

Proof: Obvious from the inference rules in the transitional semantics of π_N. □

Second, we observe the relation among free variables in the source, target and action of a probabilistic transition. Here, two new syntactic notations are needed. We use $\mathrm{rn}(P)$ to denote the set of names bound by restrictions in P, and we write $\mathrm{son}(P)$ for the set of subjects of output prefixes in P. It is necessary to assume that $\mathrm{rn}(A(\widetilde{x})) = \mathrm{rn}(P)$ and $\mathrm{son}(A(\widetilde{x})) = \mathrm{son}(P)$ whenever $A(\widetilde{x}) \stackrel{def}{=} P$. Of course, when some agent constants (i.e., identifiers) A occur in P, $\mathrm{rn}(A)$ and $\mathrm{son}(A)$ must be counted in $\mathrm{rn}(P)$ and $\mathrm{son}(P)$, respectively.

Lemma 2. *(i) If* $P \xrightarrow{\overline{x}y[p]} P'$ *or* $P \xrightarrow{x(y)} P'$ *or* $P \xrightarrow{\overline{x}(y)[p]} P'$*, then* $x \in \mathrm{fn}(P)$*.*

(ii) If $P \xrightarrow{\overline{x}y[p]} P'$*, then*

$$p \leq \sup_{u \in \mathrm{fn}(P) \cup \mathrm{rn}(P)} p_x(y|u) \tag{1}$$

(iii) If $P \xrightarrow{\alpha[p]} P'$ *and* $\alpha \neq \tau$*, then* $\mathrm{fn}(P') \subseteq \mathrm{fn}(P) \cup \mathrm{bn}(\alpha)$*.*

(iv) If $P \xrightarrow{\tau[p]} P'$*, then* $\mathrm{fn}(P') \subseteq \mathrm{fn}(P)$*, or there exist* $x \in \mathrm{son}(P)$ *and* y *satisfying Eq. (1) and* $\mathrm{fn}(P') \subseteq \mathrm{fn}(P) \cup \{y\}$*.*

Proof: It is carried out by induction on the depth of inference in the premises. As an example, we prove (iv) but only consider the case that the last rule in the derivation of $P \xrightarrow{\tau[p]} P'$ is COM. Suppose that $P = U|V$, $U \xrightarrow{\overline{x}y[p]} U'$ and $V \xrightarrow{x(z)} V'$. Then $P' = U'|V'\{y/z\}$, and

$$\mathrm{fn}(P') = \mathrm{fn}(U') \cup \mathrm{fn}(V'\{y/z\}) \subseteq \mathrm{fn}(U') \cup (\mathrm{fn}(V') - \{z\}) \cup \{y\}$$

By (iii) we know that $\mathrm{fn}(U') \subseteq \mathrm{fn}(U)$ and $\mathrm{fn}(V') \subseteq \mathrm{fn}(V) \cup \{z\}$. Thus, $\mathrm{fn}(P') \subseteq \mathrm{fn}(U|V) \cup \{y\} = \mathrm{fn}(P) \cup \{y\}$. It is clear that $x \in \mathrm{son}(U) \subseteq \mathrm{son}(P)$. Furthermore, from (ii) we have

$$p \leq \sup_{u \in \mathrm{fn}(U) \cup \mathrm{rn}(U)} p_x(y|u) \leq \sup_{u \in \mathrm{fn}(P) \cup \mathrm{rn}(P)} p_x(y|u)$$

because $\mathrm{fn}(U) \subseteq \mathrm{fn}(P)$ and $\mathrm{rn}(U) \subseteq \mathrm{rn}(P)$. $\square$

It was shown in the $\pi-$calculus that if $P \xrightarrow{\alpha} P'$ then $\mathrm{fn}(\alpha) \subseteq \mathrm{fn}(P)$ (see [44, II], Lemma 1(i)). For the case that α is an input action or a bound output action, this conclusion has a direct generalization in π_N, namely, Lemma 2(i). However, for an output action α, the thing becomes much more complicated, and we have two subcases: (i) if we are concerned with the subject of α a direct generalization still exists (see also Lemma 2(i)); and (ii) for the object of α, due to noise in the channels probabilistic information has to be accommodated; see Lemma 2(ii). Indeed, a more pertinent probabilistic generalization should be: if $P \xrightarrow{\overline{x}y[p]} P'$ then

$$p \leq \sup_{u \in \mathrm{fn}(P)} p_x(y|u)$$

But it is not true unless P contains no restrictions. For example, if $p_x(x|x) = 1$, $p_x(u|u) = 0.8$ and $p_x(y|u) = 0.2$, then

$$\frac{\overline{x}u.\mathbf{0} \xrightarrow{\overline{x}y[0.2]} \mathbf{0}}{(u)(\overline{x}u.\mathbf{0}) \xrightarrow{\overline{x}y[0.2]} (u)\mathbf{0}}$$

and

$$\sup_{z \in \mathrm{fn}((u)(\overline{x}u.\mathbf{0}))} p_x(y|z) = 0$$

The last inequality is violated. It is easy to see that such a difficulty arises mainly from the RES rule. In the agent $(u)(\overline{x}u.\mathbf{0})$, the object u in the output prefix $\overline{x}u$ is bound by (u). On the other hand, a new name y may be produced when u is emitted from the channel x because of noise in x, and y is not bound by (u). Thus, RES allows the output action $\overline{x}y$. One way to overcome this objection is to add a new parameter, except action α and probability p, in a transition relation. This parameter is used to record the original name which was emitted along the channel and produced the object of the output action under consideration. Then the RES rule may be modified in such a way: an output action is not allowed whenever the name recorded by the new parameter is bound by restriction. Nevertheless, we decide not to adopt this modification. The design decision is based on the reason that the new parameter should not be visible from the outside. It is also known in the $\pi-$calculus that $P \xrightarrow{\alpha} P'$ implies $\mathrm{fn}(P') \subseteq \mathrm{fn}(P) \cup \mathrm{bn}(\alpha)$ (see [44, II], Lemma 1(ii)). In the π_N-calculus, this conclusion also splits up into two cases, namely Lemma 2(iii) and (iv).

There is an alternative generalization of Lemma 1 in [44, II]. In order to present it, we define $\mathrm{fn}^*(P)$ to be the set of free names in P and those names produced by noise when sending free names. Formally, it is defined as follows: for each agent P,

$$\mathrm{fn}^*(P) = \bigcup_{n=0}^{\infty} \mathrm{fn}^n(P)$$

where $\mathrm{fn}^n(P)$ is recursively given by $\mathrm{fn}^0(P) = \emptyset$ for all agents P, and for any $n \geq 0$,

(i) $\mathrm{fn}^{n+1}(\mathbf{0}) = \emptyset$;

(ii) $\mathrm{fn}^{n+1}(\overline{y}x.P) = \mathrm{fn}^n(P) \cup \{y\} \cup \{z \in \mathbf{N} : p_y(z|x) > 0\}$;

(iii)

$$\mathrm{fn}^{n+1}(y(x).P) = \bigcup_{z=x \text{ or } z \text{ is a new name in } P} [(\mathrm{fn}^n(P\{z/x\}) - \{z\}) \cup \{y\}];$$

(iv) $\mathrm{fn}^{n+1}(\tau.P) = \mathrm{fn}^n(P)$;

(v) $\mathrm{fn}^{n+1}(P_1 + P_2) = \mathrm{fn}^{n+1}(P_1|P_2) = \mathrm{fn}^n(P_1) \cup \mathrm{fn}^n(P_2)$;

(vi)

$$\mathrm{fn}^{n+1}((x)P) = \bigcup_{z=x \text{ or } z \text{ is a new name in } P} (\mathrm{fn}^n(P\{z/x\}) - \{z\});$$

(vii) $\mathrm{fn}^{n+1}([x = y]P) = \mathrm{fn}^n(P) \cup \{x, y\}$; and

(viii) $\mathrm{fn}^{n+1}(A(y_1, ..., y_n)) = \mathrm{fn}^n(A) \cup \{y_1, ...y_n\}$.

Note that the presence of agent constants (i.e., identifiers) forces us to define $\mathrm{fn}^*(P)$ in a recursive way. In the defining equation of $\mathrm{fn}^{n+1}(\overline{y}x.P)$, the free name x is not explicitly added into the right-hand side. Instead, it is (implicitly) present in the set $\{z \in \mathbf{N} : p_y(z|x) > 0\}$ because it is usually reasonable to assume that $p_y(x|x) > 0$. Note that there may be infinitely many names z with $p_y(z|x) > 0$. Then from (ii) we know that $\mathrm{fn}^*(P)$ may be an infinite set for some agents P. This is different from the case of π where $\mathrm{fn}(P)$ is always finite. In clauses (iii) and (vi), a union with z ranging over x and all new names in P is made. This is just because we want to guarantee that $\mathrm{fn}^*(P)$ is invariant under $\alpha-$conversion.

The following lemma shows that Lemma 1 in [44, II] has a simple generalization in π_N provided we replace $\mathrm{fn}(\cdot)$ by $\mathrm{fn}^*(\cdot)$.

Lemma 3. *If* $P \xrightarrow{\alpha[p]} P'$, *then*
(i) $\mathrm{fn}(\alpha) \subseteq \mathrm{fn}^*(P)$; *and*
(ii) $\mathrm{fn}^*(P') \subseteq \mathrm{fn}^*(P)$ *when* α *is an output action.*

Proof: Induction on the depth of inference $P \xrightarrow{\alpha[p]} P'$. $\Box$

Obviously, Lemma 3(ii) generalizes Lemma 1(ii) of [44, II] only in the case of output actions. For other cases, computing $\mathrm{fn}^*(P')$ amounts to deal with $\mathrm{fn}^*(P\{z/y\})$, which is evaluated by the following lemma.

Lemma 4. *If* $y, z \notin \mathrm{bn}(P)$, *then*

$$\mathrm{fn}^*(P\{z/y\}) \subseteq \mathrm{fn}^*(P) \cup \{z\} \cup K(P; y, z)$$

where

$$K(P; y, z) = K_{so}(y, z) \cup \bigcup_{x \in s(y,P)} K_s(x; y, z) \cup \bigcup_{x \in o(y,P)} K_o(x; y, z)$$

$$K_{so}(y, z) = \{u : p_z(u|z) > 0\} - \{u : p_y(u|y) > 0\}$$

$$K_s(x; y, z) = \{u : p_x(u|z) > 0\} - \{u : p_x(u|y) > 0\}$$

$$K_o(x; y, z) = \{u : p_z(u|x) > 0\} - \{u : p_y(u|x) > 0\}$$

$$s(y, P) = \{x \in \mathbf{N} : \overline{x}y.P' \text{ is a subterm of } P \text{ for some } P'\}$$

and

$$o(y, P) = \{x \in \mathbf{N} : \overline{y}x.P' \text{ is a subterm of } P \text{ for some } P'\}$$

The meaning of these sets can be easily seen from their definitions. For example, $K_s(x; y, z)$ *is, roughly speaking, the names that will be received from channel* x *when*

z is sent but will not be received when y is sent, and $s(y, P)$ is the names that act as the subjects of some output prefixes in P with y as their objects.

Proof: We proceed by induction on the length of P.

Case 1. P is an output agent. We have to cope with four subcases: $P = \overline{u}v.Q$, $P = \overline{y}v.Q$, $P = \overline{u}y.Q$ or $P = \overline{y}y.Q$. We only consider the subcase of $P = \overline{y}v.Q$ as an example. Now it holds that $P\{z/y\} = \overline{z}v.Q\{z/y\}$ and

$$\mathrm{fn}^*(P\{z/y\}) = \mathrm{fn}^*(Q\{z/y\}) \cup \{z\} \cup \{w : p_z(w|v) > 0\}$$

By the induction hypothesis we have

$$\mathrm{fn}^*(Q\{z/y\}) \subseteq \mathrm{fn}^*(Q) \cup \{z\} \cup K(Q; y, z)$$

Note that $K(Q; y, z) \subseteq K(P; y, z)$ and

$$\{w : p_z(w|v) > 0\} \subseteq \{w : p_y(w|v) > 0\} \cup K_o(v; y, z)$$

$$\subseteq \{w : p_y(w|v) > 0\} \cup K(P; y, z)$$

because $v \in o(y, P)$. Then

$$\mathrm{fn}^*(P\{z/y\}) \subseteq \mathrm{fn}^*(Q) \cup \{z\} \cup \{w : p_y(w|v) > 0\} \cup K(P; y, z)$$

$$\subseteq \mathrm{fn}^*(P) \cup \{z\} \cup K(P; y, z)$$

Case 2. $P = (x)Q$. From the condition that $y, z \notin \mathrm{bn}(P)$, we have $y, z \neq x$ and $P\{z/y\} = (x)Q\{z/y\}$. Thus, the induction hypothesis leads to

$$\begin{aligned}
\mathrm{fn}^*(P\{z/y\}) &= \bigcup_{u=x \text{ or } u \text{ is a new name in } Q\{z/y\}} (\mathrm{fn}^*(Q\{y/z\}\{u/x\}) - \{u\}) \\
&= \bigcup_{u=x \text{ or } u \text{ is a new name in } Q\{z/y\}} (\mathrm{fn}^*(Q\{u/x\}\{y/z\}) - \{u\}) \\
&\subseteq \bigcup_{u=x \text{ or } u \text{ is a new name in } Q\{z/y\}} (\mathrm{fn}^*(Q\{u/x\}) \cup \{z\} \cup K(P; y, z) - \{u\} \\
&\subseteq \bigcup_{u=x \text{ or } u \text{ is a new name in } Q} (\mathrm{fn}^*(Q\{u/x\}) - \{u\}) \cup \{z\} \cup K(P; y, z) \\
&= \mathrm{fn}^*(P) \cup \{z\} \cup K(P; y, z)
\end{aligned}$$

Other cases are omitted. □

We know that $\mathrm{fn}(P\{z/y\}) \subseteq (\mathrm{fn}(P) - \{y\}) \cup \{z\}$ if $y, z \notin \mathrm{bn}(P)$. This naturally suggests us to anticipate an improvement of Lemma 4:

$$\mathrm{fn}^*(P\{z/y\}) \subseteq (\mathrm{fn}^*(P) - \{y\}) \cup \{z\} \cup K(P; y, z)$$

However, a simple example shows that such an improvement is indeed false: let $P = \overline{u}v.\mathbf{0}$, and let $p_u(v|v) = 0.7$ and $p_u(y|v) = 0.3$. Then

$$\mathrm{fn}^*(P\{z/y\}) = \mathrm{fn}^*(P) = \{u, v, y\} \not\subseteq \{u, v, z\} = (\mathrm{fn}^*(P) - \{y\}) \cup \{z\} \cup K(P; y, z)$$

We now observe the role of the objects in bound actions. As a direct generalization of Lemma 2 in [44, II], we have:

Lemma 5. *If $P \stackrel{a(y)[p]}{\longrightarrow} P'$ where $a = x$ or $a = \overline{x}$, and $z \notin \mathrm{n}(P)$, then there exists $P" \equiv_\alpha P'\{z/y\}$ such that $P \stackrel{a(z)[p]}{\longrightarrow} P"$ can be derived by an inference of no greater depth.*

Proof: Induction on the depth of inference $P \stackrel{a(y)[p]}{\longrightarrow} P'$. $\square$

Note that all of the above lemmas do not depend on the nature of noise in channels. In other words, they are valid no matter what are the involved noisy channels. However, in order to give a generalization of Lemma 3 in [44, II] we need to impose a constraint on channels. Such a constraint is formally presented as the following:

Definition 1. *Let $x \in \mathbf{N}$ and $\lambda \in (0, 1]$, and let σ be a substitution. If for any $y, z \in \mathbf{N}$ it holds that*

$$p_{x\sigma}(z\sigma|y\sigma) \geq \lambda \cdot p_x(z|y)$$

then channel x is said to be $\lambda-$compatible with σ.

Intuitively, the condition in Definition 1 means that if z will be received from channel x when y is sent, then $z\sigma$ will be received from channel $x\sigma$ when $y\sigma$ is sent, provided an error ratio λ of probability is allowed.

For the case of $\lambda = 1$, a canonical example of $\lambda-$compatibility between channel x and substitution σ is when the channel matrix of $x\sigma$ is defined by

$$p_{x\sigma}(u|y\sigma) = \sum_{z\sigma=u} p_x(z|y)$$

for all $u \in \mathbf{N}$. Another example of $1-$compatibility is given by noiseless channels. If both x and $x\sigma$ are noiseless, then x is compatible with σ. In the sequel, $1-$compatibility of x with respect to σ will be simply referred to as compatibility.

Now we need to introduce an auxiliary notation. In the next two lemmas, we have to compare two probabilities p and q. To this end, for any $p, q \in (0, 1]$, we define an implication degree of p and q by:

$$p * q = \min(1, \frac{q}{p})$$

This quantity will also play an important role in the definition of probabilistic bisimulation (see Definition 4 below).

Clearly, it holds that

$$p \cdot q \leq r \text{ if and only if } p \leq q * r$$

for any $p, q, r \in (0, 1]$. This means that product $\cdot$ and implication operator $*$ form a Galois connection. Furthermore, it is easy to derive the following useful properties of implication degree:

(1) $p * q = 1$ if and only if $p \leq q$.

(2) If $p \leq p'$ and $q' \leq q$ then $p' * q' \leq p * q$.

(3) $p \cdot (p * q) \leq q$.

(4) $(p * q) \cdot (q * r) \leq p * r$.

(5) $(p * q) \cdot (p' * q') \leq (p \cdot p') * (q \cdot q')$.

(6) It holds that

$$\inf_{i \in I}(p_i * q_i) \leq (\inf_{i \in I} p_i) * (\inf_{i \in I} q_i)$$

whenever $\inf_{i \in I} p_i > 0$, and

$$\inf_{i \in I}(p_i * q_i) \leq (\sup_{i \in I} p_i) * (\sup_{i \in I} q_i)$$

where $p, q, r, p', q', p_i, q_i \in (0, 1]$ for each $i \in I$, and I may be any index set.

By the way, we point out that the probabilistic implication operator $*$ was also used by the author in reasoning about probabilistic sequential programs [60].

The next lemma shows how a probabilistic transition of agent $P\sigma$ can be derived from a transition of P, and it generalizes Lemma 3 in [44, II].

Lemma 6. *Suppose that* $P \xrightarrow{\alpha[p]} P'$, $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P'\sigma) = \emptyset$ *and* $\sigma\lceil \mathrm{bn}(\alpha) = \mathrm{id}$ *(see page 9 for the definition of operator* $\lceil$*), and suppose that the subjects of output actions appeared in the inference for deriving* $P \xrightarrow{\alpha[p]} P'$ *are all* $\lambda-$*compatible with* σ*. Then there are* $q \in (0, 1]$ *and* $P" \equiv_\alpha P'\sigma$ *such that* $p * q \geq \lambda$*, and* $P\sigma \xrightarrow{\alpha\sigma[q]} P"$ *can be derived with an inference of no greater depth.*

Proof: Induction on the depth of inference $P \xrightarrow{\alpha[p]} P'$. We only consider the case where the last rule is OUTPUT-ACT (and the other cases are the same as in π). Now it holds that $P = \overline{x}y.Q \xrightarrow{\overline{x}z[p_x(z|y)]} Q$, and

$$P\sigma = \overline{x\sigma}y\sigma.Q\sigma \xrightarrow{\overline{x\sigma}u[p_{x\sigma}(u|y\sigma)]} Q\sigma$$

Let $u = z\sigma$. Then we have $\overline{x\sigma}u = (\overline{x}z)\sigma$, $p_{x\sigma}(u|y\sigma) \geq \lambda \cdot p_x(z|y)$ and $p_x(z|y) * p_{x\sigma}(u|y\sigma) \geq \lambda$. $\square$

We write

$$c(x,\sigma) = \inf_{y,z\in\mathbf{N} \text{ with } p_x(z|y)>0} \frac{p_{x\sigma}(z\sigma|y\sigma)}{p_x(z|y)}$$

and call it the compatibility index of x with respect to σ. Then Lemma 6 may be equivalently restated in terms of $c(x,\sigma)$ as the following:

Lemma 6'. *Suppose that* $P \xrightarrow{\alpha[p]} P'$, $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P'\sigma) = \emptyset$ *and* $\sigma\lceil\mathrm{bn}(\alpha) = \mathrm{id}$. *Then there are* $q \in (0,1]$ *and* $P" \equiv_\alpha P'\sigma$ *such that* $p * q \geq \min_x c(x,\sigma)$, *and* $P\sigma \xrightarrow{\alpha\sigma[q]} P"$ *can be derived with an inference of no greater depth, where* x *ranges over the subjects of output actions involved in inferring* $P \xrightarrow{\alpha[p]} P'$. □

Next we are going to establish a partial converse of Lemma 6. This requires us to introduce some new conditions on communication channels.

Definition 2. *Let* $z, w \in \mathbf{N}$ *and* $\lambda \in (0,1]$.

(1) If $p_u(z|v) = 0$ *for all* $u, v \neq z$, *then* z *is said to be normal.*

(2) If the following conditions (i)-(vi) are satisfied, then z *and* w *are said to be* $\lambda-$*compatible:*

(i) $\max\{p_z(w|z), p_z(z|z)\} \geq \lambda \cdot p_w(w|w)$;

(ii) $\max\{p_u(w|z), p_u(z|z)\} \geq \lambda \cdot p_u(w|w)$ *for all* $u \neq z$;

(iii) $\max\{p_z(w|v), p_z(z|v)\} \geq \lambda \cdot p_w(w|v)$ *for all* $v \neq z$;

(iv) $p_u(y|z) \geq \lambda \cdot p_u(y|w)$ *for all* $y \neq z, w$ *and for all* $u \neq z$;

(v) $p_z(y|v) \geq \lambda \cdot p_w(y|v)$ *for all* $y \neq z, w$ *and for all* $v \neq z$; *and*

(vi) $p_z(y|z) \geq \lambda \cdot p_w(y|w)$ *for all* $y \neq z, w$.

The conditions in Definition 2 put certain constraints on the channels. Intuitively, normality of z means that it is impossible to receive z from a channel u ($\neq z$) when v ($\neq z$) is emitted; in other words, z cannot be produced at the output merely due to noise in channel u. Obviously, if all $u \neq z$ are noiseless, then z is normal. The conditions (i)-(vi) for $\lambda-$compatibility between z and w become much more understandable when $\lambda = 1$. For example, (iii) indicates that if w will be received from channel w after v ($\neq z$) is sent, then either w or z will be received from channel z with no smaller probability. At this moment, these conditions are imposed mainly for the technical reason. The intuition behind them is not very clear.

We shall simply say that z and w are compatible when they are $1-$compatible. Note that all these conditions are satisfied well by noiseless channels. For instance, if channels w and z are noiseless, then $p_w(w|v) > 0$ implies $v = w$ and $p_z(w|v) = 1$, and it follows that $\max\{p_z(w|v), p_z(z|v)\} = 1 \geq p_w(w|v)$. This shows that condition (iv) holds for noiseless channels w and z.

The following lemma shows how can we infer the actions of an agent from the actions of its substitutions. It is a generalization of Lemma 4 in [44, II].

Lemma 7. *Suppose that $z \neq w$, z is normal and z and w are $\lambda-$compatible. If $P\{w/z\} \xrightarrow{\alpha[p]} P'$ where $w \notin \mathrm{fn}(P)$ and $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P, w) = \emptyset$, then there are $q \in (0, 1]$, Q and β such that $Q\{w/z\} \equiv_\alpha P'$, $\beta\{w/z\} = \alpha$, $p * q \geq \lambda$, and $P \xrightarrow{\beta[q]} Q$ may be derived by an inference of no greater depth.*

Proof: We proceed by induction on the depth of inference $P\{w/z\} \xrightarrow{\alpha[p]} P'$. Only the case that the last rule is OUTPUT-ACT needs a careful examination (and based on this case, the other cases are similar to the corresponding arguments in the $\pi-$calculus). We write σ for $\{w/z\}$ and assume that $P = \overline{u}v.U$. Then

$$P\sigma = \overline{u\sigma}v\sigma.U\sigma \xrightarrow{\overline{u\sigma}y[p_{u\sigma}(y|v\sigma)]} U\sigma$$

is the unique transition possible for $P\sigma$. This leads to $\alpha = \overline{u\sigma}y$, $p = p_{u\sigma}(y|v\sigma)$ and $P' = U\sigma$. On the other hand,

$$P \xrightarrow{\overline{u}x[p_u(x|v)]} U$$

is the unique transition possible for P. So, we have to take $Q = U$. Now it suffices to prove the following

Claim: if $p_{u\sigma}(y|v\sigma) > 0$ then there exists x such that $y = x\sigma$ and $p * p_u(x|v) \geq \lambda$.

It is easy to see that $u\sigma, v\sigma \neq z$ because we suppose that $z \neq w$. Thus, from condition (i) we know that $y \neq z$.

Subcase 1. $y = w$. Then $w\sigma = z\sigma = y$. We set $\beta = \overline{u}w$ or $\beta = \overline{u}z$. It follows that $\beta\sigma = \alpha$,

$$P \xrightarrow{\overline{u}w[p_u(w|v)]} U$$

and

$$P \xrightarrow{\overline{u}z[p_u(z|v)]} U$$

Consequently, we may take $q = \max\{p_u(w|v), p_u(z|v)\}$, and what we still need to check is that $p * q \geq \lambda$. When $u \neq z$ and $v \neq z$, we have $p = p_u(w|v)$, and $q \geq p$ is valid automatically. If $u = z$ or $v = z$, then $p * q \geq \lambda$ is guaranteed by conditions (ii) to (iv).

Subcase 2. $y \neq w$. Since $y \neq z$, we have $y\sigma = y$. Let $\beta = \overline{u}y$ and $q = p_u(y|v)$. Then $\beta\sigma = \alpha$. Furthermore, $q = p$ is always true when $u, v \neq z$, and $p * q \geq \lambda$ follows from conditions (v)-(vii) when $u = z$ or $v = z$. $\square$

By introducing a new index of compatibility, we are able to present Lemma 7 in a way similar to Lemma 6'. Let

$$l_1(z,w) = \frac{\max\{p_z(w|z), p_z(z|z)\}}{p_w(w|w)}$$

if $p_w(w|w) > 0$, and $l_1(z,w) = 1$ if $p_w(w|w) = 0$,

$$l_2(z,w) = \inf_{u \neq z} \frac{\max\{p_u(w|z), p_u(z|z)\}}{p_u(w,w)},$$

$$l_3(z,w) = \inf_{v \neq z} \frac{\max\{p_z(w|v), p_z(z|v)\}}{p_w(w|v)},$$

$$l_4(z,w) = \inf_{y \neq z,w \text{ and } u \neq z} \frac{p_u(y|z)}{p_u(y|w)},$$

$$l_5(z,w) = \inf_{y \neq z,w \text{ and } v \neq z} \frac{p_z(y|v)}{p_w(y|v)}$$

and

$$l_6(z,w) = \inf_{y \neq z,w} \frac{p_z(y|z)}{p_w(y|w)}$$

Then

$$l(z,w) = \min_{i=1}^{6} l_i(z,w)$$

is called the compatibility index of z and w.

It is easy to see that the indexes $l_1(z,w)$ to $l_6(z,w)$ exactly correspond to the conditions (i)-(vi) in Definition 2(2).

Now the above lemma can be restated as the following:

Lemma 7'. *Suppose that $z \neq w$ and z is normal. If $P\{w/z\} \xrightarrow{\alpha[p]} P'$ where $w \notin \mathrm{fn}(P)$ and $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P,w) = \emptyset$, then there are $q \in (0,1]$, Q and β such that $Q\{w/z\} \equiv_\alpha P'$, $\beta\{w/z\} = \alpha$, $p * q \geq l(z,w)$, and $P \xrightarrow{\beta[q]} Q$ may be derived by an inference of no greater depth.* $\square$

It was shown that transitions in the $\pi-$calculus are preserved well by alpha-conversion (see [44, II], Lemma 5). This is obviously a very useful tool in reasoning about the behavior of agents, and we hope to generalize it into the π_N-calculus. However, the case of π_N is much more complicated because we need to take care of the influence of renaming on the channel matrixes of bound names. To overcome this difficulty, we have to impose some conditions on channel matrixes. We distinguish free names and bound names from the beginning, namely, let $\mathbf{N} = \mathbf{FN} \cup \mathbf{BN}$, where $\mathbf{FN}$ stands for the set of free names and $\mathbf{BN}$ the set of bound names, and it is required that $\mathbf{FN} \cap \mathbf{BN} = \emptyset$. Then assume that free names (resp. bound names) in

all agents must be taken from **FN** (resp. **BN**). This way of treating free names and bound names is not new, and it follows many standard textbooks of mathematical logic. Furthermore, we need the following two hypotheses:

(H1) $p_x(z|y) = p_x(y|z) = 0$ for any $x \in \mathbf{N}$, $y \in \mathbf{FN}$ and $z \in \mathbf{BN}$; and

(H2) $p_{x_1}(z|y) = p_{x_2}(z|y)$ and $p_z(y|x_1) = p_z(y|x_2)$ for any $x_1, x_2 \in \mathbf{BN}$ and for any $y, z \in \mathbf{N}$.

Indeed, these hypotheses are very natural. (H1) indicates that free names and bound names cannot be confused by noise in channels. Roughly speaking, (H2) means that any two bound names are interchangeable. This hypothesis is reasonable because bound names are seen as dumb variables in our calculus.

It is easy to see that in essence distinguishing **FN** and **BN** and adding the hypotheses (H1) and (H2) do not decrease the expressive power of π_N.

Lemma 8. *Suppose that (H1) and (H2) are valid. Let $P \equiv_\alpha Q$.*

(i) If $P \xrightarrow{\alpha[p]} P'$ and α is a free action, then there exists $Q' \equiv_\alpha P'$ such that $Q \xrightarrow{\alpha[p]} Q'$ can be derived by an inference of no greater depth.

(ii) If $P \xrightarrow{a(y)[p]} P'$ with $a = x$ or $a = \overline{x}$, and $z \notin \mathrm{n}(Q)$, then there exists Q' such that $P'\{z/y\} \equiv_\alpha Q'$, and $Q \xrightarrow{a(z)[p]} Q'$ can be derived by an inference of no greater depth.

Proof: Induction on the depth of inference in the premises. □

The following example illustrates that the hypotheses (H1) and (H2) are necessary in Lemma 8.

Example 5. Let $p_y(x|z) = 1$, $P = (x)\overline{y}z.\mathbf{0}$ and $Q = (u)\overline{y}z.\mathbf{0}$. It is obvious that $P \equiv_\alpha Q$. On the other hand, we have $P \xrightarrow{\overline{y}(w)} \mathbf{0}$ for any w, and $Q \xrightarrow{\overline{y}z} \mathbf{0}$. So, the conclusion of Lemma 8 does not hold in this case. The reason is that (H1) is violated by $p_y(x|z) = 1$.

Suppose that $p_x(u|z) = p_y(v|z) = 1$. Let $P = (x)(\overline{x}z.\mathbf{0}|x(u).\overline{u}u.\mathbf{0})$ and $Q = (y)(\overline{y}z.\mathbf{0}|y(u).\overline{u}u.\mathbf{0})$ Then $P \equiv_\alpha Q$, and we have $P \xrightarrow{\tau} (x)(\mathbf{0}|\overline{u}u.\mathbf{0})$ and $Q \xrightarrow{\tau} (y)(\mathbf{0}|\overline{v}v.\mathbf{0})$. We see that the conclusion of Lemma 8 is also not true in this case because the first part of (H2) is violated by the channel matrixes of bound names x and y.

Suppose that $p_x(y|a) = \frac{1}{2}$, $p_x(y|b) = \frac{1}{3}$, $P = (a)\overline{x}a.\mathbf{0}$ and $Q = (b)\overline{x}b.\mathbf{0}$. Now, the second part of (H2) is violated. We have $P \equiv_\alpha Q$, but the conclusion of Lemma 8

is not true because $P \stackrel{\overline{x}y[\frac{1}{2}]}{\longrightarrow} \mathbf{0}$ and $Q \stackrel{\overline{x}y[\frac{1}{3}]}{\longrightarrow} \mathbf{0}$.

For simplicity of presentation, in what follows we uniformly assume that the hypotheses (H1) and (H2) hold although some results do not need them.

To conclude this section, we give the image-finiteness of the probabilistic transitional semantics of π_N. This property depends heavily on a certain finiteness of noise in communication channels, which is formally defined by the following:

Definition 3. *(1) A channel name x in $\mathbf{N}$ is said to be support-finite if for any $y \in \mathbf{N}$, the support set of channel x with respect to y*

$$\mathrm{supp}(x, y) = \{z \in \mathbf{N} : p_x(z|y) > 0\}$$

is a finite set.

(2) An agent P is said to be support-finite if the subject of every output prefix in P is support-finite.

Lemma 9. *Suppose that P is support-finite, and suppose that the defining agent of each agent identifier occurring in P is weakly guarded (see page 10 for the definition of a weakly guarded agent). Then (up to alpha-convertibility $\equiv_\alpha$) we have:*

(i) $\{(p, P') : P \stackrel{\tau[p]}{\longrightarrow} P'\}$ is a finite set;

(ii) $\{(x, y, p, P') : P \stackrel{\overline{x}y[p]}{\longrightarrow} P'\}$ is a finite set;

(iii) for any x there are a finite number of agents $P_1, ..., P_n$ $(n \geq 0)$ and $z \notin \mathrm{fn}(P)$ such that if $P \stackrel{x(y)}{\longrightarrow} P'$ then $P' = P_i\{y/z\}$ for some $i \leq n$; and

(iv) for any x there are a finite number of $p_1, p_2, ..., p_m$ $(m \geq 0)$, a finite number of agents $P_1, ..., P_n$ $(n \geq 0)$ and $z \notin \mathrm{fn}(P)$ such that if $P \stackrel{\overline{x}(y)[p]}{\longrightarrow} P'$ then $p = p_i$ for some $i \leq m$ and $P' = P_j\{y/z\}$ for some $j \leq n$.

Proof: (1) We first prove the conclusion for all finite agents P. This may be done simultaneously for all the items (i) to (iv) by induction on the length of P. Note that the condition of support-finiteness is used in the case of $P = \overline{x}y.P'$.

(2) We now consider agent identifier $P = A(\widetilde{y})$. If $A(\widetilde{x}) \stackrel{def}{=} Q$, then $A(\widetilde{y}) \stackrel{\alpha[p]}{\longrightarrow} P'$ if and only if $Q\{\widetilde{y}/\widetilde{x}\} \stackrel{\alpha[p]}{\longrightarrow} P'$. Let Q_0, P'_0 be the agents resulting from replacing all agent identifiers in Q and P', respectively, by $\mathbf{0}$. Then Q_0 is a finite agent. Furthermore, since each agent identifier in Q is weakly guarded, by induction on depth of inference we are able to show that if $Q\{\widetilde{y}/\widetilde{x}\} \stackrel{\alpha[p]}{\longrightarrow} P'$ then $Q_0\{\widetilde{y}/\widetilde{x}\} \stackrel{\alpha[p]}{\longrightarrow} P'_0$. Thus, from (1) it is known that the conclusion also holds for agent identifier.

(3) Finally, using (1) and (2) we may proceed by induction on the length of P to prove the proposition. □

It is obvious that the above lemma is a generalization of Lemmas 1.4.4 and 1.4.5 in [50]. But we should note that in the above lemma a condition of weak guardedness is imposed. Indeed, without this condition the above lemma is no longer valid. This will further violate the validity of Propositions 4 and 15 below. On the other hand, such a condition is unnecessary for Lemmas 1.4.4 and 1.4.5 in [50]. The reason is that in the syntax of our calculus agent identifiers are used to represent recursion, whereas in [50] replication construct is adopted for the same purpose.

5. Strong Bisimilarity

Bisimilarity is one of the most important behavioral equivalences in process algebras. Roughly speaking, it equates two processes whenever their (external) actions are identical. The notion of bisimilarity has been generalized into probabilistic process algebras in two different ways: exact probabilistic bisimilarity and approximate probabilistic bisimilarity. $\lambda-$bisimilarity defined in [57-59] and [61] is a version of approximate probabilistic bisimilarity. In this section, we extend it into the π_N-calculus. We only introduce the notion of $\lambda-$strong bisimulation here. A treatment of $\lambda-$weak bisimulations will be given in another paper.

Definition 4. *Let S be a binary relation on agents and $\lambda \in (0,1]$. Then S is called a $\lambda-$(strong) simulation if for all agents P, Q with PSQ, we have:*

(i) if $P \xrightarrow{\alpha[p]} P'$, and α is a free action, then there are agent Q', and $q \in (0,1]$ such that $Q \xrightarrow{\alpha[q]} Q'$, $P'SQ'$ and $p * q \geq \lambda$ (see page 27 for the definition of operation $*$);

(ii) if $P \xrightarrow{x(y)} P'$, and $y \notin \mathrm{n}(P,Q)$, then there is Q' such that $Q \xrightarrow{x(y)} Q'$, and $P'\{w/y\}SQ'\{w/y\}$ for all names w; and

(iii) if $P \xrightarrow{\overline{x}(y)[p]} P'$, and $y \notin \mathrm{n}(P,Q)$, then there exist Q', and $q \in (0,1]$ such that $Q \xrightarrow{\overline{x}(y)[q]} Q'$, $P'SQ'$ and $p * q \geq \lambda$.

A binary relation S on agents is called a $\lambda-$(strong) bisimulation if both S and its inverse S^{-1} are $\lambda-$simulations.

The $\lambda-$(strong) bisimilarity $\dot{\sim}_\lambda$ is defined as follows: for all agents P and Q, $P \dot{\sim}_\lambda Q$ if and only if PSQ for some $\lambda-$bisimulation S. In other words,

$$\dot{\sim}_\lambda = \bigcup \{\lambda - \text{bisimulations}\}$$

We shall call $\lambda-$simulation ($\lambda-$bisimulation, $\lambda-$bisimilarity) simply simulation (resp. bisimulation, bisimilarity) and drop the subscript λ of $\dot{\sim}_\lambda$ whenever $\lambda = 1$.

The design decision of adopting the operator $*$ in the above definition deserves a careful explanation. As pointed out in [59], the notion of $\lambda-$bisimilarity provides

us with a continuous spectrum of equivalence relations with parameter λ ranging from 0 to 1. We first consider a special case of $\lambda = 1$, which is at the top of this spectrum. From the fact that $p * q = 1$ if and only if $p \leq q$, it is easy to see that a bisimulation equates two processes whenever an action of a process may be simulated by the same action of the other process with an equal or higher probability. This is slightly different from the definition of probabilistic strong bisimulation given by Larsen and Skou [39]. They required that a bisimulation equates two agents only when these agents perform the same actions with the same probabilities. In our definition, however, a higher probability of an action is allowed to simulate the same action. At first glance, it seems that the definition by Larsen and Skou is much more reasonable. Indeed, our definition is also based on a solid intuition. It is well known that implication employed in mathematics is always the material implication, which is different from the strict implication in modal logic (or other philosophical logics). If $\Rightarrow$ is the material implication, then $\phi \Rightarrow \psi$ is true when ϕ is false or ψ is true. The counterpart of this statement in probabilistic logic (or other many-valued logics) would be that $\phi \Rightarrow \psi$ is true when the truth value of ϕ is not greater than that of ψ. On the other hand, roughly speaking, a bisimulation is usually defined in such a schema that if $P \stackrel{\alpha}{\longrightarrow} P'$ then $Q \stackrel{\alpha}{\longrightarrow} Q'$ for some Q'. In a probabilistic transition system, $P \stackrel{\alpha[p]}{\longrightarrow} P'$ may be understood as that the truth value of proposition "$P \stackrel{\alpha}{\longrightarrow} P'$" in probabilistic logic is p. Then a natural probabilistic generalization of the above schema should be that if $P \stackrel{\alpha[p]}{\longrightarrow} P'$ then $Q \stackrel{\alpha[q]}{\longrightarrow} Q'$ for some $q \geq p$ and Q'. This explains well our design decision in defining $1-$bisimulation.

Now we turn to consider the general case of $\lambda < 1$. In the definition of $\lambda-$bisimulation, the transition $P \stackrel{\alpha[p]}{\longrightarrow} P'$ is required to be simulated by a transition $Q \stackrel{\alpha[q]}{\longrightarrow} Q'$ with $p * q \geq \lambda$. It is obvious that a probability value q lower than p is possible because $\lambda < 1$. But the condition $p * q \geq \lambda$ indicates that the value of q is not allowed to be too low. Indeed, the quantity $p * q$ can be imagined as an index which measures the truth of the statement that p is not less than q. Thus, $p * q \geq \lambda$ means intuitively that the statement that p is not less that q is "quite true". Note that we use $*$ instead of the operator $\circledast$ introduced below, as what was done in [59]. The reason is that $p * q$ measures the truth of the statement that p is not less than q, whereas $p \circledast q$ measures the truth of the statement that p is equal to q. It is interesting to see that for the special case of $\lambda = 1$, if $\circledast$ was adopted then our notion of $\lambda-$bisimulation would coincide with Larsen and Skou's probabilistic strong bisimulation.

To illustrate the above definition further, let us consider some simple examples. This requires us to define the operator $\circledast$ first. For any $p, q \in (0, 1]$, a nearness (or biimplication) degree of p and q is defined by:

$$p \circledast q = \min(\frac{p}{q}, \frac{q}{p})$$

Some basic properties of $\circledast$ are displayed as follows:

(1) $p \circledast q = \min(p * q, q * p) = (p * q) \cdot (q * p)$.

(2) $p \circledast q = 1$ if and only if $p = q$.

(3) $\lim_{p\to 0}(p \circledast q) = 0$.

(4) $(p \circledast q) \cdot (p' \circledast q') \leq (p \cdot p') \circledast (q \cdot q')$.

(5) $p \circledast q = q \circledast p$.

(6) $(p \circledast q) \cdot (q \circledast r) \leq p \circledast r$.

(7) $p \leq p' \leq q' \leq q$ implies $p \circledast q \leq p' \circledast q'$.

(8) It holds that

$$\inf_{i\in I}(p_i \circledast q_i) \leq (\inf_{i\in I} p_i) \circledast (\inf_{i\in I} q_i)$$

$$\inf_{i\in I}(p_i \circledast q_i) \leq (\sup_{i\in I} p_i) \circledast (\sup_{i\in I} q_i)$$

The above property (1) indicates clearly the relationship between operators $*$ and $\circledast$. It was pointed out before that $*$ is indeed an implication operator in probabilistic logic. Then $\circledast$ should be seen as an biimplication operator in probabilistic logic.

Let $x \in \mathbf{N}$. We define a mapping $\rho_x : \mathbf{N} \times \mathbf{N} \longrightarrow (0, 1]$ as follows: for any $u, v \in \mathbf{N}$,

$$\rho_x(u, v) = \inf_{w\in\mathbf{N}}[p_x(w|u) \circledast p_x(w|v)]$$

Intuitively, the quantity $p_x(w|u) \circledast p_x(w|v)$ stands for the nearness degree of the probabilities that the same name w is received at the output of channel x when the names u and v are sent, respectively, and $\rho_x(u, v)$ is the greatest lower bound of $p_x(w|u) \circledast p_x(w|v)$ as w traverses $\mathbf{N}$. Thus, $\rho_x(u, v)$ may be seen as a similarity degree between the outcomes of emitting u and v through channel x. It is easy from the above properties of $\circledast$ to see that

(i) $\rho_x(u, u) = 1$;

(ii) $\rho_x(u, v) = \rho_x(v, u)$; and

(iii) $\rho_x(u, v) \cdot \rho_x(v, w) \leq \rho_x(u, w)$.

This further yields a metric d_x on $\mathbf{N}$ by defining

$$d_x(u, v) = -\log \rho_x(u, v)$$

for all $u, v \in \mathbf{N}$.

With these preparations, we are ready to give some simple examples of $\lambda-$bisimilarity.

Example 6. Consider the agent identifier

$$A(x, y) \stackrel{def}{=} \overline{x}y.A(x, y)$$

It sends name y repeatedly from channel x. If $u \neq v$, then $A(u, y)$ and $A(v, y)$ send the same name y via different channels u and v. For any $\lambda > 0$, it does not hold that $A(u, y) \dot{\sim}_\lambda A(v, y)$. On the other hand, we have:

$$A(x, u) \dot{\sim}_{\rho_x(u,v)} A(x, v)$$

Note that $A(x, u)$ and $A(x, v)$ send their values from the same channel x. The difference between them is that in $A(x, u)$ what is sent to channel x is u and in

$A(x,v)$ the name sent to channel x is v. Since x may be subject to noise, it might happen that the same name is received although the names sent through channel x are different. Here, the quantity $\rho_x(u,v)$ is used to represent such a possibility.

Example 7. It is known in the $\pi-$calculus that if $x \neq y$ and $u \neq v$ then

$$\overline{x}u.\mathbf{0}|y(v).\mathbf{0} \dot{\sim} \overline{x}u.y(v).\mathbf{0} + y(v).\overline{x}u.\mathbf{0} \tag{2}$$

On the other hand, we have:

$$\overline{x}u.\mathbf{0}|x(v).\mathbf{0} \dot{\not\sim} \overline{x}u.x(v).\mathbf{0} + x(v).\overline{x}u.\mathbf{0} \tag{3}$$

and

$$\overline{x}u.\mathbf{0}|y(u).\mathbf{0} \dot{\not\sim} \overline{x}u.y(u).\mathbf{0} + y(u).\overline{x}u.\mathbf{0} \tag{4}$$

Furthermore, it follows from Eq. (3) that

$$z(y).(\overline{x}u.\mathbf{0}|y(v).\mathbf{0}) \dot{\not\sim} z(y).(\overline{x}u.y(v).\mathbf{0} + y(v).\overline{x}u.\mathbf{0}) \tag{5}$$

This is a typical example demonstrating that strong bisimilarity is not preserved by input prefix. We now reconsider this example in the π_N-calculus. First, it holds that

$$\overline{x}u.\mathbf{0}|y(w).\mathbf{0} \dot{\sim}_{\rho_x(u,v)} \overline{x}v.y(w).\mathbf{0} + y(w).\overline{x}v.\mathbf{0} \tag{6}$$

provided $x \neq y$ and $w \neq v$. Indeed, for instance, the transition

$$\overline{x}u.\mathbf{0}|y(w).\mathbf{0} \stackrel{\overline{x}z[p_x(z|u)]}{\longrightarrow} \mathbf{0}|y(w).\mathbf{0}$$

is simulated by

$$\overline{x}v.y(w).\mathbf{0} + y(w).\overline{x}v.\mathbf{0} \stackrel{\overline{x}z[p_x(z|v)]}{\longrightarrow} y(w).\mathbf{0}$$

and the nearness degree of probabilities is $p_x(z|u) * p_x(z|v) \geq \rho_x(u,v)$. For the case of $u = v$, Eq. (6) degenerates to Eq. (2). As a generalization of Eqs. (2) and (4), we have:

$$\overline{x}u.\mathbf{0}|x(v).\mathbf{0} \dot{\not\sim}_\lambda \overline{x}u.x(v).\mathbf{0} + x(v).\overline{x}u.\mathbf{0} \tag{7}$$

and

$$z(y).(\overline{x}u.\mathbf{0}|y(v).\mathbf{0}) \dot{\not\sim}_\lambda z(y).(\overline{x}u.y(v).\mathbf{0} + y(v).\overline{x}u.\mathbf{0}) \tag{8}$$

for any $\lambda > 0$. It is interesting to note that in contrast Eq. (3) cannot be simply generalized into the π_N-calculus; we have instead:

$$\overline{x}u.\mathbf{0}|y(u).\mathbf{0} \dot{\sim}_{\inf_{w\neq x,y}\rho_x(u,w)} \overline{x}u.y(u).\mathbf{0} + y(u).\overline{x}u.\mathbf{0} \tag{9}$$

It is quite often that only a finite number of names in $\mathbf{N}$ are truly involved in real applications. In such a case, it is possible that $\inf_{w\neq x,y}\rho_x(u,w) > 0$, and the agents in the two sides of Eq. (3) may be bisimilar with a certain degree, although not absolutely bisimilar. $\square$

The following three propositions presents some basic properties of $\lambda-$bisimulations.

Proposition 1. *(1) The relation $\equiv_\alpha$ of alpha-convertibility on agents is a $(1-)$bisimulation.*

(2) If S is a $\lambda-$bisimulation, then S^{-1} is also a $\lambda-$bisimulation.

(3) If S_i is a $\lambda-$bisimulation for each $i \in I$, then $\bigcup_{i\in I} S_i$ is also a $\lambda-$bisimulation.

(4) $\dot{\sim}_\lambda$ is a $\lambda-$bisimulation.

Proof: (1) is a simple corollary of Lemma 8, (2) is obvious, (3) may be directly proved using Definition 4, and (4) is immediate from (3). □

From Proposition 1(4) we know that $\dot{\sim}_\lambda$ is the biggest $\lambda-$bisimulation. The following corollary shows that $\dot{\sim}$ includes alpha-convertibility $\equiv_\alpha$ and $\dot{\sim}_\lambda$ is symmetric. From Definition 4 it is obvious that if $\lambda < \mu$ then each $\mu-$bisimulation is a $\lambda-$bisimulation, and $\dot{\sim}_\mu \subseteq \dot{\sim}_\lambda$. Thus, for any $\lambda > 0$, $\dot{\sim}_\lambda$ includes $\equiv_\alpha$ and it is reflexive.

Corollary 1. *(1) If $P \equiv_\alpha Q$ then $P \dot{\sim} Q$.* □

(2) $P \dot{\sim}_\lambda Q$ implies $Q \dot{\sim}_\lambda P$.

The next proposition gives an equivalent characterization of $\lambda-$bisimilarity.

Proposition 2. *$P \dot{\sim}_\lambda Q$ if and only if*

(i) whenever $P \xrightarrow{\alpha[p]} P'$ and α is a free action, there are Q' and $q \in (0,1]$ such that $Q \xrightarrow{\alpha[q]} Q'$, $P' \dot{\sim}_\lambda Q'$ and $p * q \geq \lambda$;

(ii) whenever $P \xrightarrow{x(y)[p]} P'$ and $y \notin \mathrm{n}(P,Q)$, there are Q' and $q \in (0,1]$ such that $Q \xrightarrow{x(y)[q]} Q'$, $P'\{w/y\} \dot{\sim}_\lambda Q'\{w/y\}$ for all w and $p * q \geq \lambda$;

(iii) whenever $P \xrightarrow{\overline{x}(y)[p]} P'$ and $y \notin \mathrm{n}(P,Q)$, there are Q' and $q \in (0,1]$ such that $Q \xrightarrow{\overline{x}(y)[q]} Q'$, $P' \dot{\sim}_\lambda Q'$ and $p * q \geq \lambda$; and the symmetric forms of (i), (ii) and (iii).

Proof: Similar to Proposition 4.4 in [42]. □

The following proposition is a probabilistic version of transitivity of bisimilarity.

Proposition 3. *If $P \dot{\sim}_{\lambda_1} Q$ and $Q \dot{\sim}_{\lambda_2} R$, then $P \dot{\sim}_{\lambda_1\lambda_2} R$. Moreover, $\dot{\sim}$ is an equivalence relation.*

Proof: The idea is similar to the proof of Theorem 2(a) in [44, II]. We show that $\dot{\sim}_{\lambda_1}\dot{\sim}_{\lambda_2}$ is a $\lambda_1\lambda_2-$bisimulation. Let $P \dot{\sim}_{\lambda_1}\dot{\sim}_{\lambda_2} R$. Then $P \dot{\sim}_{\lambda_1} Q$ and $Q \dot{\sim}_{\lambda_2} R$ for some agent Q. Suppose that $P \xrightarrow{\alpha[p]} P'$ and α is a free action. Then from $P \dot{\sim}_{\lambda_1} Q$ and Proposition 2 it follows that for some Q' and q, $Q \xrightarrow{\alpha[q]} Q'$, $P' \dot{\sim}_{\lambda_1} Q'$ and $p * q \geq \lambda_1$. Furthermore, from $Q \dot{\sim}_{\lambda_2} R$ we obtain that for some R' and r, $R \xrightarrow{\alpha[r]} R'$, $Q' \dot{\sim}_{\lambda_2} R'$ and $q * r \geq \lambda_2$. Now, it holds that $P' \dot{\sim}_{\lambda_1}\dot{\sim}_{\lambda_2} R'$ and $p * r \geq (p * q) \cdot (q * r) \geq \lambda_1\lambda_2$.

For the case of input actions or bound output actions, a procedure is needed for a suitable rearrangement of names. This may be done by simply following the corresponding parts in the proof of [44, II], Theorem 2(a) and by using Lemma 2. □

The notion of bisimilarity may be described in a different way; that is, we can use a real number in the unit interval to measure bisimilarity between two agents.

Definition 5. *Let P and Q be two agents. Then the bisimilarity degree of them is defined by*

$$\mathrm{Sim}(P, Q) = \sup\{\lambda \in (0, 1] : P \dot{\sim}_\lambda Q\}$$

With the above definition, Corollary 1 and Proposition 3 may be equivalently restated as the following:

Corollary 2. *(1) If $P \equiv_\alpha Q$ then* $\mathrm{Sim}(P, Q) = 1$.
(2) $\mathrm{Sim}(P, Q) = \mathrm{Sim}(Q, P)$.
(3) $\mathrm{Sim}(P, Q) \cdot \mathrm{Sim}(Q, R) \leq \mathrm{Sim}(P, R)$. □

The quantity $\mathrm{Sim}(P, Q)$ measures (bi)similarity between two agents P and Q: the bigger $\mathrm{Sim}(P, Q)$ is, the more (bi)similar P and Q are. A much more familiar mathematical notion that can be used to measure similarity is metric. It is clear that $\mathrm{Sim}(\cdot, \cdot)$ is not a metric on agents. But a simple transformation of $\mathrm{Sim}(\cdot, \cdot)$ gives a metric. For any two agents P and Q, we define

$$d(P, Q) := -\log \mathrm{Sim}(P, Q)$$

Then from the above corollary we see that $d(\cdot, \cdot)$ is a metric on agents. Obviously, the smaller $d(P, Q)$ is, the more (bi)similar P and Q are. Furthermore, from this metric we can construct a topological structure on agents in a standard way. The convergence of agents in π_N according to this metric topology is an interesting topic for further researches (cf. [57]).

The next proposition establishes a close link between $\lambda-$bisimilarity and bisimilarity degree. Also, it demonstrates that $\lambda-$bisimilarity $\dot{\sim}_\lambda$ is left-continuous with respect to variable λ. We first need to introduce some notations. Let P be an agent and α an action. Then we define $D(P,\alpha)$ to be the set of immediate derivatives after performing action α, together with transition probabilities from P, namely,

$$D(P,\alpha) = \{(p, P') : P \xrightarrow{\alpha[p]} P'\}$$

If $P \xrightarrow{\alpha[p]} P'$ for some action α and $p \in (0,1]$, then P' is called an immediate derivatives of P. The set of all immediate derivatives of P is denoted by $D(P)$. Moreover, if there are $n \geq 0$, $p_1, ..., p_n \in (0,1]$, actions $\alpha_1, ..., \alpha_n$ and agents $P_1, ..., P_{n-1}$ such that

$$P \xrightarrow{\alpha_1[p_1]} P_1 \xrightarrow{\alpha_2[p_2]} P_2 \ \ ... \xrightarrow{\alpha_{n-1}[p_{n-1}]} P_{n-1} \xrightarrow{\alpha_n[p_n]} P'$$

then P' is called a derivative of P. We write $D^*(P)$ for the set of all derivatives of P.

Proposition 4. *Suppose that each P' in $D^*(P) \cup D^*(Q)$ is support-finite and the defining agent of each agent identifier occurring in P' is weakly guarded. Then it holds that*

$$P \dot{\sim}_{\mathrm{Sim}(P,Q)} Q$$

for any agents P and Q.

Proof: Suppose that $\{\lambda_n\}_{n=1}^{\infty}$ is an increasing sequence of real numbers in the unit interval $[0,1]$, $\lambda_n < \mathrm{Sim}(P,Q)$ for each $n = 1, 2, ...$, and $\lim_{n\to\infty} \lambda_n = \mathrm{Sim}(P,Q)$. From Definition 5 it is easy to see that $P \dot{\sim}_{\lambda_n} Q$ for any $n = 1, 2, ...$ Thus, we have $(P,Q) \in \bigcap_{n=1}^{\infty} \dot{\sim}_{\lambda_n}$.

Let $D^* = D^*(P) \cup D^*(Q)$. The key idea is to show that $(\bigcap_{n=1}^{\infty} \dot{\sim}_{\lambda_n}) \cap (D^* \times D^*)$ is a $\mathrm{Sim}(P,Q)-$bisimulation. If

$$(U,V) \in (\bigcap_{n=1}^{\infty} \dot{\sim}_{\lambda_n}) \cap (D^* \times D^*)$$

and $U \xrightarrow{\alpha[p]} U'$ where α is a free action, then for each $n = 1, 2, ...$, we have $(U,V) \in \dot{\sim}_{\lambda_n}$. Since $\dot{\sim}_{\lambda_n}$ is a λ_n-bisimulation, it implies in turn that there exist V_n and $q_n \in (0,1]$ with $V \xrightarrow{\alpha[q_n]} V_n$, $U' \dot{\sim}_{\lambda_n} V_n$ and $p * q_n \geq \lambda_n$. Note that $V \in D^*$. Then V is support-finite and the defining agent of each agent identifier in V is weakly guarded. From Lemma 9 we know that

$$\{(q_n, V_n) : n = 1, 2, ...\} \subseteq D(V,\alpha)$$

is a finite set. Hence, there are V' and $q \in (0,1]$ and an increasing sequence $\{n_i\}_{i=1}^{\infty}$ of positive integers such that $(q_{n_i}, V_{n_i}) = (q, V')$ for all $i = 1, 2, ...$ Now it follows

that

$$P \xrightarrow{\alpha[q=q_{n_i}]} V_{n_i} = V'$$

For each i, we have $(U', V') = (U', V_{n_i}) \in \dot{\sim}_{\lambda_{n_i}}$, and it follows that

$$(U', V') \in \bigcap_{i=1}^{\infty} \dot{\sim}_{\lambda_{n_i}} = \bigcap_{n=1}^{\infty} \dot{\sim}_{\lambda_n}$$

Furthermore, noting that $U', V' \in D^*$, we obtain

$$(U', V') \in (\bigcap_{n=1}^{\infty} \dot{\sim}_{\lambda_n}) \cap (D^* \times D^*)$$

On the other hand, for each i, $p * q = p * q_{n_i} \geq \lambda_{n_i}$. Noting that

$$\lim_{i \to \infty} \lambda_{n_i} = \lim_{n \to \infty} \lambda_n = \mathrm{Sim}(P, Q)$$

we assert that $p * q \geq \mathrm{Sim}(P, Q)$.

For input actions and bound output actions we have a similar argument. This leads to the conclusion that $(\bigcap_{n=1}^{\infty} \dot{\sim}_{\lambda_n}) \cap (D^* \times D^*)$ is a $\mathrm{Sim}(P, Q)-$bisimulation, and consequently we complete the proof. $\square$

Note that the preceding proposition depends heavily on Lemma 9. Thus, it does not hold whenever some involved channels are not support-finite.

We now turn to show that $\lambda-$bisimilarity is congruent for some combinators in the $\pi-$calculus. First, it is demonstrated that $\lambda-$bisimilarity is preserved by certain substitutions of a single name. To this end, some notations are needed. Let t be a probabilistic transition $P \xrightarrow{\alpha[p]} P'$. We write $\mathrm{so}(t)$ for the set of subjects of output actions involved in t. More concretely,

$$\mathrm{so}(t) = \begin{cases} \emptyset, & \text{if } \alpha \text{ is an input action, or } \alpha = \tau \text{ is obtained by using TAU-ACT} \\ \{x\}, & \text{if } \left\{ \begin{array}{c} \alpha = \overline{x}y \text{ or } \overline{x}(y), \text{ or } \alpha = \tau \text{ is obtained by using COM on output} \\ \text{action } \overline{x}y \text{ and a complementary input action, or by using CLOSE} \\ \text{on bound output action } \overline{x}(y) \text{ and a complementary input action} \end{array} \right\} \end{cases}$$

Furthermore, for any agent P, we define

$$\mathrm{so}(P) = \bigcup \{\mathrm{so}(t) : t \text{ is a possible probabilistic transition of } P\}$$

The definitions of $\mathrm{so}(P)$ and $\mathrm{son}(P)$ given in page 22 are quite different, but the following lemma indicates that they are indeed the same thing.

Lemma 10. *For any agent P, $\mathrm{son}(P) = \mathrm{so}(P)$.*

Proof. Induction on the length of P. □

Proposition 5. *Suppose that $z \neq w$, z is normal, and z and w are compatible (see Definition 2). Also, suppose that each channel $x \in \mathrm{son}(P) \cup \mathrm{son}(Q)$ is compatible with $\{w/z\}$ (see Definition 1). If $P \dot{\sim}_\lambda Q$ and $w \notin \mathrm{fn}(P,Q)$, then $P\{w/z\} \dot{\sim}_\lambda Q\{w/z\}$. Thus, for any $w \notin \mathrm{fn}(P,Q)$, we have:*

$$\mathrm{Sim}(P,Q) \leq \mathrm{Sim}(P\{w/z\}, Q\{w/z\})$$

Proof: The idea is similar to the proof of Lemma 6 in [44, II]. We set

$$S_0 = \dot{\sim}_\lambda$$

$$S_{n+1} = \{(P\{w/z\}, Q\{w/z\}) : PS_nQ, w \notin \mathrm{fn}(P,Q),$$
$$z \text{ and } w \text{ are as in the suppostion of the proposition}\}$$

for every $n \geq 0$, and $S = \bigcup_{n=0}^{\infty} S_n$.

It suffices to show that S is a $\lambda-$bisimulation. Let USV. Then US_nV for some $n \geq 0$. We first use induction on n to prove the following

Claim: If $U \xrightarrow{\alpha[p]} U'$ and α is a free action, then there exist $q \in (0,1]$ and Q' such that $V \xrightarrow{\alpha[q]} V'$, $p * q \geq \lambda$ and $U'SV'$.

For the case of $n = 0$, $S_n = \dot{\sim}_\lambda$, it is clear. We now assume that $US_{n+1}V$, i.e. $U = P\{w/z\}$ and $V = Q\{w/z\}$ where PS_nQ, and z and w are as in the assumption of the proposition. With Lemma 7 we can find some p', P' and β such that $p' \geq p$, $P'\{w/z\} \equiv_\alpha U'$, $\beta\{w/z\} = \alpha$ and $P \xrightarrow{\beta[p']} P'$. Now by the induction hypothesis we have $Q \xrightarrow{\beta[q']} Q'$ for some q' and Q' with $p' * q' \geq \lambda$ and $P'SQ'$. Furthermore, from Lemma 10 and the assumption of the proposition it follows that the subjects of output actions involved in $Q \xrightarrow{\beta[q']} Q'$ are compatible with $\{w/z\}$, and from Lemma 3 we know that there are $q \geq q'$ and $V' \equiv_\alpha Q'\{w/z\}$ with

$$V = Q\{w/z\} \xrightarrow{\alpha=\beta\{w/z\}[q]} V'$$

Then it follows that $p * q \geq p' * q' \geq \lambda$ and $U' \equiv_\alpha P'\{w/z\}SQ'\{w/z\} \equiv_\alpha V'$.

For input actions and bound output actions a similar argument enables us to complete the proof. □

The proof of congruence property of $\lambda-$bisimilarity requires the technique of bisimulation up to restriction, and it is introduced in the following definition.

Definition 6. *Let $\lambda \in (0,1]$. Then a binary relation S on agents is called a $\lambda-$(strong) simulation up to restriction if for all agents P, Q with PSQ, we have:*

(1) if $w \notin \mathrm{fn}(P,Q)$, then $P\{w/z\}SQ\{w/z\}$; and

(2.i) whenever $P \xrightarrow{\overline{x}y[p]} P'$, then there are agent Q', and $q \in (0,1]$ such that $Q \xrightarrow{\overline{x}y[q]} Q'$, $P'SQ'$ and $p * q \geq \lambda$;

(2.ii) whenever $P \xrightarrow{\tau[p]} P'$, then there are agent Q', and $q \in (0,1]$ such that $Q \xrightarrow{\tau[q]} Q'$, $p * q \geq \lambda$, and $P'SQ'$ or for some $P", Q"$ and w, $P' \equiv_\alpha (w)P"$, $Q' \equiv_\alpha (w)Q"$ and $P"SQ"$;

(2.iii) whenever $P \xrightarrow{x(y)} P'$, and $y \notin \mathrm{n}(P,Q)$, then there exists Q' such that $Q \xrightarrow{x(y)} Q'$, and $P'\{w/y\}SQ'\{w/y\}$ for all names w; and

(2.iv) whenever $P \xrightarrow{\overline{x}(y)[p]} P'$, and $y \notin \mathrm{n}(P,Q)$, then there exist Q', and $q \in (0,1]$ such that $Q \xrightarrow{\overline{x}(y)[q]} Q'$, $P'SQ'$ and $p * q \geq \lambda$.

A binary relation S on agents is called a $\lambda-$(strong) bisimulation up to restriction if both S and its inverse S^{-1} are $\lambda-$simulations up to restriction.

The following lemma guarantees validity of the technique of bisimulation up to restriction.

Lemma 11. *If S is a $\lambda-$strong bisimulation up to restriction, then $S \subseteq \dot{\sim}_\lambda$.*

Proof: Similar to Lemma 7 in [44, II]. We set $S_0 = S$,

$$S_{n+1} = \{((w)P, (w)Q) : PS_nQ, w \in \mathbf{N}\}$$

and

$$S^* = \bigcup_{n=0}^{\infty} S_n$$

Then it may be shown that S^* is a $\lambda-$bisimulation. □

Now we are able to show that under certain conditions on communication channels $\lambda-$bisimilarity is preserved by various combinators in the $\pi-$calculus.

Proposition 6. *(1) If $P \dot{\sim}_\lambda Q$, then*

$$\tau.P \dot{\sim}_\lambda \tau.Q$$

$$\overline{x}y.P \dot{\sim}_{\min(\lambda,\rho_x(y,z))} \overline{x}z.Q$$
$$[x=y]P \dot{\sim}_\lambda [x=y]Q; \text{ and}$$
$$P+R \dot{\sim}_\lambda Q+R$$

(2) Suppose that for any

$$z \in \bigcup_{R\in D^*(P)\cup D^*(Q)} \mathrm{fn}(R),$$

$$w \notin \bigcap_{R\in D^*(P)\cup D^*(Q)} \mathrm{fn}(R)$$

and

$$x \in \bigcup_{R\in D^*(P)\cup D^*(Q)} \mathrm{so}(R),$$

z is normal, z and w are compatible, and x is compatible with $\{w/z\}$ (see Definitions 1 and 2). If $P \dot{\sim}_\lambda Q$, then $P|R \dot{\sim}_\lambda Q|R$.

(3) Suppose that z is normal, and suppose that z and w are compatible, and x is compatible with $\{w/z\}$ for any

$$w \notin \bigcap_{R\in D^*(P)\cup D^*(Q)} \mathrm{n}(R)$$

and for any

$$x \in \bigcup_{R\in D^*(P)\cup D^*(Q)} \mathrm{so}(R)$$

If $P \dot{\sim}_\lambda Q$, then $(z)P \dot{\sim}_\lambda (z)Q$.

(4) Suppose that y is normal, and suppose that for any $z \notin \mathrm{fn}(P,Q,y)$ and for any $x \in \mathrm{son}(P)\cup\mathrm{son}(Q)$, y and z are compatible and x is compatible with $\{z/y\}$. If for all $w \in \mathrm{fn}(P,Q,y)$, $P\{w/y\} \dot{\sim}_\lambda Q\{w/y\}$, then

$$x(y).P \dot{\sim}_\lambda x(y).Q$$

Proof: (1) We only prove the second conclusion, and the others are similar. We show that

$$S = \{(\overline{x}y.P, \overline{x}z.Q)\} \cup \dot{\sim}_\lambda$$

is a $\min(\lambda, \rho_x(y,z))-$bisimulation. In fact, transition

$$\overline{x}y.P \xrightarrow{\overline{x}u[p_x(u|y)]} P$$

is simulated by

$$\overline{x}z.Q \stackrel{\overline{x}u[p_x(u|z)]}{\longrightarrow} Q$$

and at the same time we have

$$p_x(u|z) * p_x(u|y) \geq \rho_x(y,z) \geq \min(\lambda, \rho_x(y,z))$$

and $P \dot{\sim}_\lambda Q$, which implies $P \dot{\sim}_{\min(\lambda,\rho_x(y,z))} Q$.

(2) With Lemma 11, it suffices to show that

$$S = \{(U|R, V|R) : U \dot{\sim}_\lambda V,\ U \in D^*(P) \text{ and } V \in D^*(Q)\}$$

is a $\lambda-$bisimulation up to restriction. To verify that S satisfies condition (1) in Definition 6 we only need to show that for all $w \notin \text{fn}(U,V)$ and for all z, $U \dot{\sim}_\lambda V$, $U \in D^*(P)$ and $V \in D^*(Q)$ imply $U\{w/z\} \dot{\sim}_\lambda V\{w/z\}$. If $z \notin \text{fn}(U,V)$, then $U\{w/z\} = U$ and $V\{w/z\} = V$. It is obvious. For the case of $z \in \text{fn}(U,V)$, such an implication is warranted by Proposition 5 and the assumption on the channels. It is routine to check that S satisfies condition (2) of Definition 6.

(3) Let

$$S_0 = \dot{\sim}_\lambda \cap (D^*(P) \times D^*(Q))$$

and

$$S = \dot{\sim}_\lambda \cup \{((z)P', (z)Q') : (P', Q') \in S_0\}$$

Then it suffices to show that S is a $\lambda-$bisimulation. For any $(P', Q') \in S_0$, we only consider the case of bound output actions by $(z)P'$ and $(z)Q'$. If $(z)P' \stackrel{\overline{y}(w)[p]}{\longrightarrow} U$, and $w \notin \text{n}((z)P', (z)Q') = \text{n}(P', Q', z)$, then $z \neq y$, and $P' \stackrel{\overline{y}(z)[p]}{\longrightarrow} P$" and $U = P"\{w/z\}$ for some $P"$. Since $P' \dot{\sim}_\lambda Q'$, there must be $Q"$, and $q \in (0,1]$ such that $Q' \stackrel{\overline{y}(z)[q]}{\longrightarrow} Q"$, $P" \dot{\sim}_\lambda Q"$ and $p * q \geq \lambda$. Consequently, $(z)Q' \stackrel{\overline{y}(w)}{\longrightarrow} Q"\{w/z\}$. Note that $P", Q" \in D^*(P) \cup D^*(Q)$. With Proposition 5 we have $U \dot{\sim}_\lambda Q"\{w/z\}$ and $(U, Q"\{w/z\}) \in S$.

(4) For any $u \notin \text{n}(x(y).P, x(y).Q) = \text{n}(P, Q, x, y)$, we have $x(y).P \stackrel{x(u)}{\longrightarrow} P\{u/y\}$ and $x(y).Q \stackrel{x(u)}{\longrightarrow} Q\{u/y\}$. With Proposition 2 it suffices to show that for all names w, we have

$$P\{u/y\}\{w/u\} \dot{\sim}_\lambda Q\{u/y\}\{w/u\}$$

Noting that $u \notin \text{n}(P, Q, x, y)$, we obtain $P\{u/y\}\{w/u\} = P\{w/y\}$ and $Q\{u/y\}\{w/u\} = Q\{w/y\}$. For the case of $w \in \text{fn}(P, Q, y)$, it is already given by the assumption. On the other hand, let $w = y$ in the assumption. Then it holds that $P \dot{\sim}_\lambda Q$, and furthermore by using Proposition 5 we obtain $P\{w/y\} \dot{\sim}_\lambda Q\{w/y\}$ for all $w \notin \text{fn}(P, Q, y)$. □

In [44, II] the way of proving that $P \dot{\sim} Q$ implies $(z)P \dot{\sim} (z)Q$ is to show that $\dot{\sim}$ is a bisimulation up to restriction and then to use the construction S^* for $S = \dot{\sim}$.

However, this method does not work in the π_N-calculus because it requires us to use Proposition 5 for all names z. In turn, such a global application of Proposition 5 needs the assumption that all names z are normal. It is easy to see that all names will be noiseless and π_N will degenerate to π whenever all names are normal. Note that the way in which we prove the implication from $P \dot{\sim}_\lambda Q$ to $(z)P \dot{\sim}_\lambda (z)Q$ here is in fact a localization of Proposition 5 at the fixed name z.

The above proposition may be equivalently restated in terms of bisimilarity degree as the following:

Corollary 3. *(1) It holds that*

$$\mathrm{Sim}(P,Q) \leq \mathrm{Sim}(\tau.P, \tau.Q)$$

$$\min[\mathrm{Sim}(P,Q), \rho_x(y,z)] \leq \mathrm{Sim}(\overline{x}y.P, \overline{x}z.Q)$$

$$\mathrm{Sim}(P,Q) \leq \mathrm{Sim}([x=y]P, [x=y]Q); \text{ and}$$

$$\mathrm{Sim}(P,Q) \leq \mathrm{Sim}(P+R, Q+R)$$

(2) With the same assumption as in Proposition 6(2), we have

$$\mathrm{Sim}(P,Q) \leq \mathrm{Sim}(P|R, Q|R)$$

(3) With the same assumption as in Proposition 6(3), we have

$$\mathrm{Sim}(P,Q) \leq \mathrm{Sim}((w)P, (w)Q); \text{ and}$$

(4) With the same assumption as in Proposition 6(4), we have

$$\inf_{w \in \mathrm{fn}(P,Q,y)} \mathrm{Sim}(P\{w/y\}, Q\{w/y\}) \leq \mathrm{Sim}(x(y).P, x(y).Q)$$

Proof: Immediate from Proposition 6. □

Some basic equational laws for $(1-)$bisimilarity are presented in the next proposition. These laws are concerned with prefix, summation, matching and restriction. Some equational laws for parallel composition will be given later since their proofs require new proof techniques.

Proposition 7. *(1)* $P + \mathbf{0} \dot{\sim} P$;
(2) $P + P \dot{\sim} P$;

(3) $P_1 + P_2 \dot{\sim} P_2 + P_1$*;*
(4) $P_1 + (P_2 + P_3) \dot{\sim} (P_1 + P_2) + P_3$*;*
(5) If $A(\widetilde{x}) \stackrel{def}{=} P$*, then* $A(\widetilde{y}) \dot{\sim} P\{\widetilde{y}/\widetilde{x}\}$*;*
(6) $[x = y]P \dot{\sim} \mathbf{0}$ *if* $x \neq y$*;*
(7) $[x = x]P \dot{\sim} P$*;*
(8) $(y)P \dot{\sim} P$ *if* $y \notin \mathrm{fn}^*(P)$*;*
(9) $(y)(z)P \dot{\sim} (z)(y)P$*;*
(10) $(y)(P + Q) \dot{\sim} (y)P + (y)Q$*;*
(11) $(y)\alpha.P \dot{\sim} \alpha.(y)P$ *if* α *is not a (free) output action and* $y \notin \mathrm{n}(\alpha)$*;*
(12) $(y)\overline{x}z.P \dot{\sim} \overline{x}z.(y)P$ *if* $y \neq x$ *and* $p_x(y|z) = 0$*; and*
(13) $(y)\alpha.P \dot{\sim} \mathbf{0}$ *if* y *is the subject of* α*.*

Proof: Similar to Theorems 3-6 in [44, II]. □

Proposition 7(8) is a generalization of Theorem 6(a) in [44, II]. However, the condition in π_N to warrant $(y)P \dot{\sim} P$ is slightly stronger than that in π; that is, it is required in π_N that $y \notin \mathrm{fn}^*(P)$, but in π we only need to assume that $y \notin \mathrm{fn}(P)$. The following example will show that the condition $y \notin \mathrm{fn}(P)$ is not sufficient in the π_N-calculus. Also, Propositions 7(11) and (12) are counterparts of Theorem 6(d) in [44, II]. For the case that α is not a (free) output action, the conditions to guarantee $(y)\alpha.P \dot{\sim} \alpha.(y)P$ in π and in π_N are the same, namely, $y \notin \mathrm{n}(\alpha)$. But when $\alpha = \overline{x}z$ is an output action, such a condition has to be strengthened to $y \neq x$ and $p_x(y|z) = 0$. The necessity of this modification is explained by the following example too.

Example 8. Let $\mathbf{N} = \{x_1, x_2, x_3, ...\}$ be the set of names. Suppose that $p_{x_1}(x_1|x_2) = 0$ and

$$p_{x_1}(x_n|x_2) = \frac{1}{2^{n-1}}$$

for all $n \geq 2$. Then for any $\lambda > 0$,

$$(x_3)\overline{x_1}x_2.\mathbf{0} \dot{\sim}_\lambda \overline{x_1}x_2.\mathbf{0}$$

does not hold although $x_3 \notin \mathrm{fn}(\overline{x_1}x_2.\mathbf{0})$. In fact,

$$\overline{x_1}x_2.\mathbf{0} \stackrel{\overline{x_1}x_3[\frac{1}{4}]}{\longrightarrow} \mathbf{0}$$

cannot be simulated by any transition of $(x_3)\overline{x_1}x_2.\mathbf{0}$.

Similarly, we can see that $x_3 \notin \mathrm{n}(\overline{x_1}x_2)$ but

$$(x_3)\overline{x_1}x_2.\mathbf{0} \dot{\sim}_\lambda \overline{x_1}x_2.(x_3)\mathbf{0}$$

does not hold for all $\lambda > 0$.

In order to prove some useful equational laws for parallel composition, we need to introduce two new proof techniques, namely, bisimulation up to bisimilarity and bisimulation up to bisimilarity and restriction. They will be formally defined in Definitions 7 and 8, and their validity will be shown by Lemmas 12 and 13, respectively.

Definition 7. *A binary relation S on agents is called a $\lambda-$(strong) simulation up to $\dot{\sim}_\mu$ if for all agents P, Q with PSQ, we have:*

(i) if $P \xrightarrow{\alpha[p]} P'$, and α is a free action, then there are agent Q', and $q \in (0,1]$ such that $Q \xrightarrow{\alpha[q]} Q'$, $P' \dot{\sim}_\mu S \dot{\sim}_\mu Q'$ and $p * q \geq \lambda$;

(ii) if $P \xrightarrow{x(y)} P'$, and $y \notin \mathrm{n}(P,Q)$, then there exits Q' such that $Q \xrightarrow{x(y)} Q'$, and $P'\{w/y\} \dot{\sim}_\mu S \dot{\sim}_\mu Q'\{w/y\}$ for all names w; and

(iii) if $P \xrightarrow{\overline{x}(y)[p]} P'$, and $y \notin \mathrm{n}(P,Q)$, then there exist Q', and $q \in (0,1]$ such that $Q \xrightarrow{\overline{x}(y)[q]} Q'$, $P' \dot{\sim}_\mu S \dot{\sim}_\mu Q'$ and $p * q \geq \lambda$.

A relation S on agents is called a $\lambda-$(strong) bisimulation up to $\dot{\sim}_\mu$ if both S and its inverse S^{-1} are $\lambda-$simulations up to $\dot{\sim}_\mu$.

Lemma 12. *Let S be a $\lambda-$bisimulation up to $\dot{\sim}$. Suppose that*

(i) for any agents P, Q, U and V with $P \dot{\sim} USV \dot{\sim} Q$, there is a normal name $z \notin \mathrm{n}(P,Q,U,V)$;

(ii) each normal name z and each name w are compatible (see Definition 2); and

(iii) for any names x and w, and for any normal name z, x is compatible with $\{w/z\}$ (see Definition 1).

Then we have $S \subseteq \dot{\sim}_\lambda$.

Proof: We construct S^* in a way similar to the proof of Lemma 9 in [44, II]:

$$S^* = \bigcup_{n=0}^{\infty} S_n$$

where

$$S_0 = \dot{\sim} S \dot{\sim}, \text{ and}$$

$$S_{n+1} = \{(P\{w/z\}, Q\{w/z\}) : PS_nQ,\ z \text{ is a normal name, and } w \notin \mathrm{fn}(P,Q)\}$$

for each $n \geq 0$. Now we show that S^* is a $\lambda-$bisimulation. Suppose that PS^*Q. Then PS_nQ for some $n \geq 0$. We proceed by induction on n.

For the case of $n = 0$, we only consider the transitions with bound output actions. Since PS_0Q, it holds that $P \dot{\sim} USV \dot{\sim} Q$ for some agents U and V. If $P \xrightarrow{\overline{x}(y)[p]} P'$

and $y \notin \mathrm{n}(P,Q)$, then we can choose a normal name $z \notin \mathrm{n}(P,Q,U,V)$. From Lemma 5 it holds that $P \xrightarrow{\overline{x}(z)[p]} P'\{z/y\}$. Note that $z \notin \mathrm{n}(P,U)$ and $P \dot{\sim} U$. Then there is an agent U' such that $U \xrightarrow{\overline{x}(z)[p]} U'$ and $P'\{z/y\} \dot{\sim} U'$. Furthermore, since S is a $\lambda-$bisimulation up to $\dot{\sim}$ and $z \notin \mathrm{n}(U,V)$, there exist V' and $q \in (0,1]$ such that $V \xrightarrow{\overline{x}(z)[q]} V'$, $U' \dot{\sim} S \dot{\sim} V'$ and $p * q \geq \lambda$. Thus, we have $Q \xrightarrow{\overline{x}(z)[q]} Q$" and $V' \dot{\sim} Q$" for some agent Q". Let $Q' = Q"\{y/z\}$. Then it holds that $Q \xrightarrow{\overline{x}(y)[q]} Q'$. In addition, $P'\{z/y\} \dot{\sim} U' \dot{\sim} S \dot{\sim} V' \dot{\sim} Q"$, $P'\{z/y\} S_0 Q"$, and $P' = P'\{z/y\}\{y/z\} S_1 Q"\{y/z\} = Q'$.

In general, suppose that $PS_{n+1}Q$. Then $P = U\{w/z\}$, $Q = V\{w/z\}$ for some w, U and V with US_nV and $w \notin \mathrm{fn}(U,V)$. Now we only consider the case of free actions. If $P \xrightarrow{\alpha[p]} P'$, then from Lemma 7 we have $U \xrightarrow{\beta[p]} U'$ for some β and U' with $\beta\{w/z\} = \alpha$ and $U'\{w/z\} \equiv_\alpha P'$. With the induction hypothesis, we obtain $V \xrightarrow{\beta[q]} V'$, $U' S^* V'$ and $p * q \geq \lambda$ for some V' and $q \in (0,1]$. Furthermore, from Lemma 6 we know that $Q \xrightarrow{\alpha[q]} Q' \equiv_\alpha V'\{w/z\}$, and $P' S^* Q'$. $\square$

As a combination of Definitions 6 and 7, we introduce the notion of $\lambda-$bisimulation up to $\dot{\sim}_\mu$ and restriction.

Definition 8. *Let $\lambda, \mu \in (0,1]$. Then a binary relation S on agents is called a $\lambda-$(strong) simulation up to $\dot{\sim}_\mu$ and restriction if for all agents P, Q with PSQ, we have:*

(1) if $w \notin \mathrm{fn}(P,Q)$, then $P\{w/z\} S Q\{w/z\}$; and

(2.i) whenever $P \xrightarrow{\overline{x}y[p]} P'$, then there are agent Q', and $q \in (0,1]$ such that $Q \xrightarrow{\overline{x}y[q]} Q'$, $P' \dot{\sim}_\mu S \dot{\sim}_\mu Q'$ and $p * q \geq \lambda$;

(2.ii) whenever $P \xrightarrow{\tau[p]} P'$, then there are agent Q', and $q \in (0,1]$ such that $Q \xrightarrow{\tau[q]} Q'$, $p * q \geq \lambda$, and $P' \dot{\sim}_\mu S \dot{\sim}_\mu Q'$ or for some $P", Q"$ and w, $P' \dot{\sim}_\mu (w)P"$, $Q' \dot{\sim}_\mu (w)Q"$ and $P" S Q"$;

(2.iii) whenever $P \xrightarrow{x(y)} P'$, and $y \notin \mathrm{bn}(P,Q)$, then there exists Q' such that $Q \xrightarrow{x(y)} Q'$, and $P'\{w/y\} \dot{\sim}_\mu S \dot{\sim}_\mu Q'\{w/y\}$ for all names w; and

(2.iv) whenever $P \xrightarrow{\overline{x}(y)[p]} P'$, and $y \notin \mathrm{bn}(P,Q)$, then there exist Q', and $q \in (0,1]$ such that $Q \xrightarrow{\overline{x}(y)[q]} Q'$, $P' \dot{\sim}_\mu S \dot{\sim}_\mu Q'$ and $p * q \geq \lambda$.

A binary relation S on agents is called a $\lambda-$(strong) bisimulation up to $\dot{\sim}_\mu$ and restriction if both S and its inverse S^{-1} are $\lambda-$simulations up to restriction.

Lemma 13. *With the same assumption as in Lemma 12, if S is a $\lambda-$bisimulation up to $\dot{\sim}$ and restriction, then $S \subseteq \dot{\sim}_\lambda$.*

Proof: It is similar to the proof of Lemma 12 to prove that $S^* = \bigcup_{n=0}^{\infty} S_n$ is a $\lambda-$bisimulation, where

$$S_0 = \dot{\sim} S \dot{\sim}, \text{ and}$$

$$S_{n+1} = \dot{\sim} \{((w)P, (w)Q) : PS_nQ \text{ and } w \text{ is a name}\} \dot{\sim}$$

for all $n \geq 0$. $\square$

It should be noted that in the definition of $\lambda-$bisimulation up to $\dot{\sim}_\mu$ (and restriction), the parameter μ is allowed to be less than 1. But we only considered the case of $\mu = 1$ in Lemmas 12 and 13. This special case is sufficient to prove the following proposition. The general case will be discussed in Section 7, after the notion of $(\lambda, \mu)-$stratified bisimulation is introduced.

We are now ready to show some basic equational laws for parallel composition.

Proposition 8. *(1)* $P|\mathbf{0} \dot{\sim} P$*;*

(2) $P_1|P_2 \dot{\sim} P_2|P_1$*;*

(3) $(y)P_1|P_2 \dot{\sim} (y)(P_1|P_2)$ *if* $y \notin \text{fn}^*(P_2)$*, and* $(y)(P_1|P_2) \dot{\sim} (y)P_1|(y)P_2$ *if* $y \notin \text{fn}^*(P_1) \cap \text{fn}^*(P_2)$*; and*

(4) $(P_1|P_2)|P_3 \dot{\sim} P_1|(P_2|P_3)$.

Proof: (1) and (2) are easy and omitted here.

(3) The second part is a simple corollary of the first one. To prove the first part, it amounts to show that

$$S = \{((y)P_1|P_2, (y)(P_1|P_2)) : P_1, P_2 \text{ are agents, and } y \notin \text{fn}^*(P_2)\} \cup Id$$

is a strong bisimulation up to $\dot{\sim}$ and restriction, where Id is the identity relation on agents. The details are similar to the proof of Theorem 8(c) in [44, II]. The fact that S is a bisimulation up to $\dot{\sim}$ and restriction does not automatically imply $(y)P_1|P_2 \dot{\sim} (y)(P_1|P_2)$ whenever $y \notin \text{fn}^*(P_2)$, and Lemma 13 is needed in order to warrant such an implication. However, applying Lemma 13 requires us to find normal names z which satisfy the conditions (i), (ii) and (iii) in Lemma 12. This is completely different from the case of channels without noise. Usually, such normal names may not exist. This forces us to add some new names into the set $\mathbf{N}$ of names and expand $\mathbf{N}$. The idea of expanding the syntax (or language) of π is indeed inspired by Henkin's method of constructing models from new constants in model theory [22, Chapter 2]. Let $z^* \notin \mathbf{N}$. We set $\mathbf{N}' = \mathbf{N} \cup \{z^*\}$. For any $x, y \in \mathbf{N}$, the probability distribution $p_x(.|y)$ on $\mathbf{N}$ is extended to a probability distribution on $\mathbf{N}'$ by simply letting $p_x(z^*|y) = 0$. And we define

$$p_{z^*}(x|y) = \begin{cases} 1, & \text{if } x = y \\ 0, & \text{if } x \neq y \end{cases}$$

for all $x, y \in \mathbf{N}'$. Thus, we can develop the π_N-calculus with the name set $\mathbf{N}'$. For convenience, this expanded π_N-calculus is denoted by π'_N. Then it is routine to check the following two claims.

Claim 1: for any agents P and Q in π_N, $P \dot{\sim}_\lambda Q$ holds in π_N if and only if $P \dot{\sim}_\lambda Q$ holds in π'_N.

Claim 2: in π'_N, z^* is a normal name, z^* and each $w \in \mathbf{N}'$ are compatible, and each $x \in \mathbf{N}'$ is compatible with $\{w/z^*\}$ for any $w \in \mathbf{N}'$.

Now using Claim 2 and Lemma 13 we are able to show that $(y)P_1|P_2 \sim (y)(P_1|P_2)$ in π'_N, and this together with Claim 1 yields that $(y)P_1|P_2 \dot{\sim} (y)(P_1|P_2)$ in π_N provided $y \notin \mathrm{fn}^*(P_2)$.

(4) It may be carried out by combining the ideas of (3) and the proof of Theorem 8(d) in [44, II]. The details are very complicated and dull, and we omit them. □

In the $\pi-$calculus, it is merely required that $y \notin \mathrm{fn}(P_2)$ in order to guarantee $(y)P_1|P_2 \dot{\sim} (y)(P_1|P_2)$. But in Proposition 8(3) this condition is strengthened to $y \notin \mathrm{fn}^*(P_2)$. The following example illustrates that $y \notin \mathrm{fn}(P_2)$ is not sufficient for $(y)P_1|P_2 \sim (y)(P_1|P_2)$ in the π_N-calculus.

Example 9. Let $P_1 = \mathbf{0}$ and $P_2 = \overline{x}u.\mathbf{0}|x(v).\overline{v}z.\mathbf{0}$. Then $y \notin \mathrm{fn}(P_2)$. Suppose that $p_x(u|u) = 0.6$, $p_x(y|u) = 0.4$ and $p_y(z|z) = 1$. We have

$$(y)P_1|P_2 \stackrel{\overline{x}y[0.4]}{\longrightarrow} (y)\mathbf{0}|(\mathbf{0}|\overline{y}z.\mathbf{0}) \stackrel{\overline{y}z}{\longrightarrow} (y)\mathbf{0}|(\mathbf{0}|\mathbf{0})$$

and

$$(y)(P_1|P_2) \stackrel{\overline{x}y[0.4]}{\longrightarrow} (y)(\mathbf{0}|(\mathbf{0}|\overline{y}z.\mathbf{0}))$$

The second transition of $(y)P_1|P_2$ cannot be simulated by $(y)(P_1|P_2)$, and it does not hold that $(y)P_1|P_2 \dot{\sim}_\lambda (y)(P_1|P_2)$ for any $\lambda > 0$.

The expansion law in CCS or the $\pi-$calculus is a convenient mathematical tool in reasoning about behavior of parallel composition because it is able to unfold a composition of agents to a summation with all of its capabilities for action explicit. We now intend to establish an expansion law in the π_N-calculus. As pointed out before, π_N has the same syntax as π. However, the expansion law cannot be expressed in the original language of π. Instead, we have to introduce a probabilistic extension of π. This is an interesting feature of π_N distinct from π.

What is needed to add into the syntax of π is simply a probabilistic summation. Thus, the syntax of the extended π is given by

$$P ::= \mathbf{0} \mid \overline{y}x.P \mid y(x).P \mid \tau.P \mid P_1{+}P_2 \mid \sum_i [p_i]P_i \mid P_1|P_2 \mid (x)P \mid [x = y]P \mid A(y_1, ..., y_n)$$

where i ranges over a finite or countably infinite set of indexes, and it is required that $\sum_i [p_i] = 1$. It should be noted that in this new language both nondeterministic choice and probabilistic choice are allowed. The transitional semantics of the additional construct, probabilistic summation, is given by the following inference rule:

SUM:

$$\frac{P_j \xrightarrow{\alpha[q]} P'}{\sum_i [p_i] P_i \xrightarrow{\alpha[p_i q]} P'}$$

To present the expansion law in a compact way, we also need to make a convention of notation: the action symbols α_i and β_j in the following proposition are allowed to be not only ordinary prefixes but also derived prefixes of the form $(u)\overline{x}y$ with $x \neq y$. In other words, α_i and β_j may range over the set $\{\tau\} \cup \{\overline{x}y, x(y) : x, y \in \mathbf{N}\} \cup \{(u)\overline{x}y : u, x, y \in \mathbf{N} \text{ and } x \neq y\}$. Note that if $\alpha_i = (u)\overline{x}y$, then $\alpha_i.P_i$ stands for agent $(u)\overline{x}y.P_i$, and if $\beta_j = (u)\overline{x}y$, then $\beta_j.Q_j$ is agent $(u)\overline{x}y.Q_j$ (see [44, II], Definition 16).

With the above preliminaries, the expansion laws with respect to bisimilarity in the π_N-calculus can be stated as follows:

Proposition 9. *(The expansion law for $\dot{\sim}$) Suppose that $P = \sum_i \alpha_i.P_i$ and $Q = \sum_j \beta_j.Q_j$, and suppose that for all i, j, if $\alpha_i = x(y)$ then $y \notin \mathrm{fn}(Q)$, if $\alpha_i = (u)\overline{x}y$ then $u \notin \mathrm{fn}^*(Q)$, if $\beta_j = x(y)$ then $y \notin \mathrm{fn}(P)$, and if $\beta_j = (u)\overline{x}y$ then $u \notin \mathrm{fn}^*(P)$. Then we have*

$$P|Q \dot{\sim} \sum_i \alpha_i.(P_i|Q) + \sum_j \beta_j.(P|Q_j) + \sum_{\alpha_i \text{ comp } \beta_j} R_{ij}$$

where the relation "α_i comp β_j" holds in the following four cases:

(1) α_i is $\overline{x}u$ and β_j is $x(v)$; then

$$R_{ij} = \sum_{y:p_x(y|u)>0} [p_x(y|u)]\tau.(P_i|Q_j\{y/v\})$$

(2) α_i is $(u)\overline{x}y$ and β_j is $x(v)$; then

$$R_{ij} = \sum_{z:p_x(z|y)>0} [p_x(z|y)]\tau.((u)P_i|Q_j\{z/v\})$$

if $p_x(u|y) = 0$, and

$$R_{ij} = \sum_{z:p_x(z|y)>0} [p_x(z|y)]\tau.((u)P_i|Q_j\{z/v\}) + [p_x(u|y)]\tau.(w)(P_i\{w/u\}|Q_j\{w/v\})$$

if $p_x(u|y) > 0$, *where* $w \notin \mathrm{fn}((u)P_i, (v)Q_j)$;

(3) α_i *is* $x(v)$ *and* β_j *is* $\overline{x}u$; *then*

$$R_{ij} = \sum_{y:p_x(y|u)>0} [p_x(y|u)]\tau.(P\{y/v\}|Q_j)$$

(4) α_i *is* $x(v)$ *and* β_j *is* $(u)\overline{x}y$; *then*

$$R_{ij} = \sum_{z:p_x(z|y)>0} [p_x(z|y)]\tau.(P_i\{z/v\}|(u)Q_j)$$

if $p_x(u|y) = 0$, *and*

$$R_{ij} = \sum_{y:p_x(y|u)>0} [p_x(y|u)]\tau.(P\{y/v\}|(u)Q_j) + [p_x(u|y)]\tau.(w)(P_i\{w/v\}|Q_j\{w/u\}$$

if $p_x(u|y) > 0$, *where* $w \notin \mathrm{fn}((v)P_i, (u)Q_j)$.

Proof: For any agents P and Q, we write $R(P, Q)$ for the agent in the right-hand side of the bisimilarity in the conclusion part of the proposition. Let

$$S = \{(P|Q, R(P, Q)) : P, Q \text{ are agents}\}$$

Then it is easy to show that $S \cup Id$ is a bisimulation, where Id is the identity relation on agents. □

Originally, information about noise is hidden behind syntax and it cannot be seen directly. Now through the expansion law it is brought out to surface, and it appears explicitly in syntax. This is exactly why we need to extend the syntax of π (and π_N) in order to present the expansion law in π_N.

6. Strong (D-)Equivalence

It was observed in the last section that $\lambda-$bisimilarity is not preserved by substitutions of free names. To overcome this objection, in this section we introduce a refined version of $\lambda-$bisimilarity, namely, $\lambda-$equivalence. It is defined to be a $\lambda-$bisimilarity under all substitutions of free names. The notion of $\lambda-$equivalence is clearly a probabilistic generalization of equivalence introduced in Definition 10 of [44, II].

As was done in Definition 12 of [44, II], $\lambda-$equivalence can be further refined by allowing only some special substitutions of free names to be used. The constraint imposed on these substitutions is represented by the notion of distinction. Recall that a distinction is a symmetric and irreflexive relation D between names (see [44,

II], Definition 11). If $A \subseteq \mathbf{N}$, then we sometimes use A to abbreviate the distinction $A \times A - \mathrm{Id}_{\mathbf{N}}$, in which all different names in A are treated as distinct. For any distinction D, we write $D \upharpoonright A$ for $D \cap (A \times A)$, the restriction of D on A. Let $x \in \mathbf{N}$. Then $D \smallsetminus x$ means the distinction $D - ((\{x\} \times \mathbf{N}) \cup (\mathbf{N} \times \{x\}))$, where the name x is excluded from the distinction D.

A substitution is a function σ from $\mathbf{N}$ to itself such that $\sigma x = x$ for all but a finite number of names x. If $\sigma x \neq \sigma y$ for all $(x, y) \in D$, then we say that σ respects D. In other words, substitution σ respects a distinction D if and only if all names distinguished by D cannot be identified by σ

Now we are able to introduce the main notion of this section.

Definition 9. *Let D be a distinction and $\lambda \in (0, 1]$, and let P and Q be two agents.*

(1) If $P\sigma \dot{\sim}_\lambda Q\sigma$ for all substitutions σ respecting D, then P and Q are said to be (strongly) $(D, \lambda)-$equivalent, and we write $P \sim_{D,\lambda} Q$.

(2) The (strong) $D-$equivalence degree between P and Q is defined by

$$\mathrm{Eq}_D(P, Q) = \sup\{\lambda \in (0, 1] : P \sim_{D,\lambda} Q\}$$

In particular, if $D = \emptyset$, then $\sim_{D,\lambda}$ is simply called (strong) $\lambda-$equivalence and abbreviated to $\sim_\lambda$. It is obvious that P and Q are $\lambda-$equivalent if and only if $P\sigma \dot{\sim} Q\sigma$ for all substitutions σ. Similarly, $\emptyset-$equivalence degree $\mathrm{Eq}_\emptyset$ will be called (strong) equivalence degree, and the subscript $\emptyset$ will be dropped for simplicity. Also, we shall drop λ from $\sim_{D,\lambda}$ whenever $\lambda = 1$.

Intuitively, $\lambda-$bisimilarity can be thought of as an equivalence relation where different names will not be identified, whereas $\lambda-$equivalence is an equivalence relation under the assumption that all free names may be identified. In the middle, there is the notion of $D-$equivalence which only allows us to identify free names that are not related by the distinction D.

Propositions 3 and 6 may be then generalized to the case of $(D, \lambda)-$bisimilarity. It is indicated that $(D, \lambda)-$equivalence is transitive, and it is preserved by the combinators in π under certain conditions on channels.

Proposition 10. *For all distinctions D and $\lambda \in (0, 1]$, we have:*

(1) $\sim_{D,\lambda}$ is reflexive and symmetric, and $P \sim_{D,\lambda_1} Q$ and $Q \sim_{D,\lambda_2} R$ imply $P \sim_{D,\lambda_1\lambda_2} R$. In particular, $\sim_D$ is an equivalence relation.

(2) If $P \sim_{D,\lambda} Q$, then

$$\tau.P \sim_{D,\lambda} \tau.Q$$

$$\overline{x}y.P \sim_{D,\min[\lambda,\inf_{\sigma \text{ respects } D} \rho_{x\sigma}(y\sigma,z\sigma)]} \overline{x}z.Q$$

In particular,

$$\overline{x}y.P \sim_{D,\min[\lambda,\rho_x(y,z)]} \overline{x}z.Q$$

whenever for all σ respecting D, channel x is compatible with σ;

$$[x = y]P \sim_{D,\lambda} [x = y]Q; \text{ and}$$

$$P + R \sim_{D,\lambda} Q + R$$

(3) Suppose that for any

$$z \in \bigcup_{\sigma \text{ respects } D} \bigcup_{R \in D^*(P\sigma) \cup D^*(Q\sigma)} \mathrm{fn}(R),$$

$$w \notin \bigcap_{\sigma \text{ respects } D} \bigcap_{R \in D^*(P\sigma) \cup D^*(Q\sigma)} \mathrm{fn}(R)$$

and

$$x \in \bigcup_{\sigma \text{ respects } D} \bigcup_{R \in D^*(P\sigma) \cup D^*(Q\sigma)} \mathrm{so}(R),$$

z is a normal name, z and w are compatible, and x is compatible with $\{w/z\}$. If $P \sim_{D,\lambda} Q$, then $P|R \sim_{D,\lambda} Q|R$.

(4) Suppose that z is normal, and suppose that z and w are compatible, and x is compatible with $\{w/z\}$ for any

$$w \notin \bigcup_{\sigma \text{ respects } D} \bigcap_{R \in D^*(P\sigma) \cup D^*(Q\sigma)} \mathrm{n}(R)$$

and for any

$$x \in \bigcup_{\sigma \text{ respects } D} \bigcup_{R \in D^*(P\sigma) \cup D^*(Q\sigma)} \mathrm{so}(R)$$

If $P \sim_{D,\lambda} Q$, then $(z)P \sim_{D,\lambda} (z)Q$.

(5) Suppose that y is a normal name, and suppose that for any

$$z \notin \bigcap_{\sigma \text{ respects } D} \mathrm{fn}(P\sigma, Q\sigma, y)$$

and for any

$$y \in \bigcup_{\sigma \text{ respects } D} (\mathrm{so}(P\sigma) \cup \mathrm{so}(Q\sigma))$$

y and z are compatible and x is compatible with $\{z/y\}$. If $P \sim_{D,\lambda} Q$ and $P\{v/y\} \sim_{D,\lambda} Q\{v/y\}$ for all $v \in \mathrm{fn}(P, Q)$ such that $(v, y) \in D$, then

$$x(y).P \sim_{D,\lambda} x(y).Q$$

Proof: Immediate from Propositions 3 and 6. □

The above proposition may be equivalently presented in terms of $D-$equivalence degree as the following:

Corollary 4. *(1) It holds that*

$$\mathrm{Eq}_D(P,P)=1,\ \mathrm{Eq}_D(P,Q)=\mathrm{Eq}_D(Q,P)$$

and

$$\mathrm{Eq}_D(P,Q)\cdot\mathrm{Eq}_D(Q,R)\leq\mathrm{Eq}_D(P,R)$$

(2) We have:

$$\mathrm{Eq}_D(P,Q)\leq\mathrm{Eq}_D(\tau.P,\tau.Q)$$

$$\min[\mathrm{Eq}_D(P,Q),\inf_{\sigma\ \text{respects}\ D}\rho_x(y,z)]\leq\mathrm{Eq}_D(\overline{x}y.P,\overline{x}z.Q)$$

In particular,

$$\min[\mathrm{Eq}_D(P,Q),\rho_x(y,z)]\leq\mathrm{Eq}_D(\overline{x}y.P,\overline{x}z.Q)$$

if channel x is compatible with all substitutions σ which respect D;

$$\mathrm{Eq}_D(P,Q)\leq\mathrm{Eq}_D([x=y]P,[x=y]Q)$$

$$\mathrm{Eq}_D(P,Q)\leq\mathrm{Eq}_D(P+R,Q+R)$$

(3) With the same assumption as in Proposition 10(3), we have:

$$\mathrm{Eq}_D(P,Q)\leq\mathrm{Eq}_D(P|R,Q|R)$$

(4) With the same assumption as in Proposition 10(4), we have:

$$\mathrm{Eq}_D(P,Q)\leq\mathrm{Eq}_D((w)P,(w)Q)$$

(5) With the same assumption as in Proposition 10(5), we have:

$$\min[\mathrm{Eq}_D(P,Q),\inf_{v\in\mathrm{fn}(P,Q)\ \text{and}\ (v,y)\in D}\mathrm{Eq}_D(P\{v/y\},Q\{v/y\}]\leq\mathrm{Eq}_D(x(y).P,x(y).Q)$$

Proof. Immediate from Proposition 10 and Definition 9(2). □

Proposition 7 can also be generalized to the case of $D-$equivalence in a quite direct way, and this gives some basic equational laws for $D-$equivalence.

Let substitution $\sigma = \{u_1/y_1, ..., u_n/y_n\}$ and let $K(\subseteq \mathbf{N})$ be a set of names. If for any $i, j \leq n$ and $x \in K$, the three sets $\{z \in \mathbf{N} : p_{u_i}(z|x) > 0\}$, $\{z \in \mathbf{N} : p_{u_i}(z|u_j) > 0\}$ and $\{z \in \mathbf{N} : p_x(z|u_i) > 0\}$ are all finite, then we say that σ is support-finite with respect to K.

With this new concept of support-finiteness, the $(D-)$equivalence generalization of Proposition 7 may be stated as follows.

Proposition 11. *(1)* $P + \mathbf{0} \sim P$;

(2) $P + P \sim P$;

(3) $P_1 + P_2 \sim P_2 + P_1$;

(4) $P_1 + (P_2 + P_3) \sim (P_1 + P_2) + P_3$;

(5) If $A(\widetilde{x}) \stackrel{def}{=} P$, *then* $A(\widetilde{y}) \sim P\{\widetilde{y}/\widetilde{x}\}$;

(6) $[x = y]P \sim_{\{x,y\}} \mathbf{0}$;

(7) $[x = x]P \sim P$;

(8) Suppose that $\mathrm{fn}^*(P)$ *is a finite set, and any substitution* σ *respecting* D *is support-finite with respect to* $\mathrm{fn}(P)$. *Then* $(y)P \sim_D P$ *whenever* $y \notin \mathrm{fn}^*(P)$;

(9) $(y)(z)P \sim (z)(y)P$;

(10) $(y)(P + Q) \sim (y)P + (y)Q$;

(11) $(y)\alpha.P \sim_D \alpha.(y)P$ *if* α *is not a (free) output action and* $y \notin \bigcup_{\sigma \text{ respects } D} n(\alpha\sigma)$;

(12) $(y)\overline{x}z.P \sim_D \overline{x}z.(y)P$ *if* $y \neq x\sigma$ *and* $p_{x\sigma}(y|z\sigma) = 0$ *for all substitutions* σ *respecting* D;

(13) $(y)\alpha.P \sim \mathbf{0}$ *if* y *is the subject of* α;

(14) $P|\mathbf{0} \sim P$;

(15) $P_1|P_2 \sim P_2|P_1$;

(16) $(y)P_1|P_2 \sim_D (y)(P_1|P_2)$ *if*

$$y \notin \bigcup_{\sigma \text{ respects } D} \mathrm{fn}^*(P_2\sigma),$$

and $(y)(P_1|P_2) \sim_D (y)P_1|(y)P_2$ *if*

$$y \notin \bigcup_{\sigma \text{ respects } D} \mathrm{fn}^*(P_1\sigma) \cap \mathrm{fn}^*(P_2\sigma); \text{ and}$$

(17) $(P_1|P_2)|P_3 \sim P_1|(P_2|P_3)$.

Proof: We only prove (8), and the others are immediate from Proposition 7.

Suppose that $\sigma = \{u_1/y_1, ..., u_n/y_n\}$ is a substitution and it respects D. We

need to prove that $((y)P)\sigma \dot{\sim} P\sigma$. From the assumption we see that

$$M = \bigcup_{i\leq n, x\in \mathrm{fn}(P)} (\{z \in \mathbf{N} : p_{u_i}(z|x) > 0\} \cup \{z \in \mathbf{N} : p_x(z|u_i) > 0\})$$

$$\cup \bigcup_{i,j\leq n} \{z \in \mathbf{N} : p_{u_i}(z|u_j) > 0\}$$

is finite because $\mathrm{fn}(P)$ is a finite set. We choose $v \notin M \cup \mathrm{fn}^*(P)$. Since $y \notin \mathrm{fn}^*(P) \supseteq \mathrm{fn}(P)$, we have $(y)P \equiv_\alpha (v)P\{v/y\} = (v)P$ and $((y)P)\sigma \equiv_\alpha ((v)P)\sigma$. Then with Corollary 1(1) we obtain $((y)P)\sigma \dot{\sim} ((v)P)\sigma = (v)(P\sigma)$. Note that $v \notin M \cup \mathrm{fn}^*(P)$. This implies $v \notin \mathrm{fn}^*(P\sigma)$. Hence, from Proposition 7(8) it follows that $(v)(P\sigma) \dot{\sim} P\sigma$. □

Note that some items of Proposition 11 are given without subscripts D, λ in the equivalence symbol. This means that they are true when D is the empty set and $\lambda = 1$. Of course, they are also valid for the case of $D \neq \emptyset$ and $\lambda < 1$ because $\sim = \sim_{\emptyset,1}$ is the strongest one in the family of equivalence relations $\sim_{D,\lambda}$, namely, $\sim \subseteq \sim_{D,\lambda}$ for any $D \neq \emptyset$ and $\lambda < 1$.

We also have an expansion law for $(1-)$equivalence. The only difference between Proposition 9 and the following proposition is that certain matching constructs must be added in the case of strong equivalence in order to distinguish different names.

Proposition 12. *(The expansion law for $\sim$) Suppose that $P = \sum_i \alpha_i.P_i$ and $Q = \sum_j \beta_j.Q_j$, where α_i and β_j are as in Proposition 9, and suppose that for all i, j, if $\alpha_i = x(y)$ then $y \notin \mathrm{fn}(Q)$, if $\alpha_i = (u)\overline{x}y$ then $u \notin \mathrm{fn}^*(Q)$, if $\beta_j = x(y)$ then $y \notin \mathrm{fn}(P)$, and if $\beta_j = (u)\overline{x}y$ then $u \notin \mathrm{fn}^*(P)$. Then*

$$P|Q \sim \sum_i \alpha_i.(P_i|Q) + \sum_j \beta_j.(P|Q_j) + \sum_{\alpha_i \ \mathrm{opp} \ \beta_j} [x_i = y_j]R_{ij}$$

where the relation "α_i opp β_j" holds in the following four cases:

(1) α_i is $\overline{x_i}u$ and β_j is $y_j(v)$; then

$$R_{ij} = \sum_{z:p_{x_i}(z|u)>0} [p_{x_i}(z|u)]\tau.(P_i|Q_j\{z/v\})$$

(2) α_i is $(u)\overline{x_i}t$ and β_j is $y_j(v)$; then

$$R_{ij} = \sum_{z:p_{x_i}(z|t)>0} [p_{x_i}(z|t)]\tau.((u)P_i|Q_j\{z/v\})$$

if $p_{x_i}(u|t) = 0$, and

$$R_{ij} = \sum_{z:p_{x_i}(z|t)>0} [p_{x_i}(z|t)]\tau.((u)P_i|Q_j\{z/v\}) + [p_{x_i}(u|t)]\tau.(w)(P_i\{w/u\}|Q_j\{w/v\})$$

if $p_{x_i}(u|t) > 0$, *where* $w \notin \mathrm{fn}((u)P_i, (v)Q_j)$;

(3) α_i *is* $x_i(v)$ *and* β_j *is* $\overline{y_j}u$; *then*

$$R_{ij} = \sum_{z:p_{y_j}(z|u)>0} [p_{y_j}(z|u)]\tau.(P\{z/v\}|Q_j)$$

(4) α_i *is* $x_i(v)$ *and* β_j *is* $(u)\overline{y_j}t$; *then*

$$R_{ij} = \sum_{z:p_{y_j}(z|t)>0} [p_{y_j}(z|t)]\tau.(P_i\{z/v\}|(u)Q_j)$$

if $p_{y_j}(u|t) = 0$, *and*

$$R_{ij} = \sum_{z:p_{y_j}(z|u)>0} [p_{y_j}(z|u)]\tau.(P\{z/v\}|(u)Q_j) + [p_{y_j}(u|t)]\tau.(w)(P_i\{w/v\}|Q_j\{w/u\}$$

if $p_{y_j}(u|t) > 0$, *where* $w \notin \mathrm{fn}((v)P_i, (u)Q_j)$.

Proof: Proposition 9 enables us to show that applying a substitution σ to both sides of the expansion law yields two bisimilar agents. The key technique is using alpha-conversion to avoid name capture in $\alpha_i\sigma$ and $Q\sigma$ as well as in $\beta_j\sigma$ and $P\sigma$. □

The following proposition is a simple generalization of Theorem 16 in [44, II], and it exposes an interesting connection of bound names and distinctions.

Proposition 13. *(1) If* $P \sim_{D,\lambda} Q$, *then* $(x)P \sim_{D\setminus x,\lambda} (x)Q$.
(2) If $P \sim_{D\setminus x,\lambda} Q$, *then* $y(x).P \sim_{D,\lambda} y(x).Q$.
(3) If $P \sim_{D,\lambda} Q$, *then* $P \sim_{D\restriction \mathrm{fn}(P,Q),\lambda} Q$.
Or equivalently in terms of $D-$*equivalence degree, we have:*
(1)' $\mathrm{Eq}_D(P,Q) \leq \mathrm{Eq}_{D\setminus x}((x)P, (x)Q)$.
(2)' $\mathrm{Eq}_{D\setminus x}(P,Q) \leq \mathrm{Eq}_D(y(x).P, y(x).Q)$.
(3)' $\mathrm{Eq}_D(P,Q) \leq \mathrm{Eq}_{D\restriction \mathrm{fn}(P,Q)}(P,Q)$.

Proof: Similar to Theorem 16 in [44, II]. □

An axiomatization of strong bisimilarity in π was found in [44], and it was proved to be complete over finite agents. The main aim of introducing $D-$equivalence in [44] is to give a simple axiomatization of equivalence in π. To conclude this section, we would like to point out that in the π_N-calculus it is still an open problem to find a corresponding axiomatization for $\lambda-$bisimilarity or $D, \lambda-$equivalence.

7. Stratified (Strong) Bisimilarity

It was shown in Proposition 5 that $\lambda-$bisimilarity is preserved by some substitutions of a single name. This is a basis of many other important algebraic properties of $\lambda-$bisimilarity. However, in Proposition 5 the condition on the channel names involved in these substitutions is very strict. The main aim of this section is to remove part of this condition. This will be done by introducing the notion of $(\lambda,\mu)-$stratified bisimilarity, which is much weaker than $\lambda-$bisimilarity.

Definition 10. *Let $\{S_n\}_{n=0}^{\infty}$ be an increasing sequence of binary relations on agents, i.e. $S_n \subseteq S_{n+1}$ for every $n \geq 0$, and let $\lambda, \mu \in (0,1]$. Then $\{S_n\}_{n=0}^{\infty}$ is called a $(\lambda,\mu)-$stratified (strong) simulation if for any $n \geq 0$ and for any $(P,Q) \in S_n$,*

(i) whenever $P \xrightarrow{\alpha[p]} P'$, and α is a free action, then there are agent Q', and $q \in (0,1]$ such that $Q \xrightarrow{\alpha[q]} Q'$, $P' S_{n+1} Q'$ and $p * q \geq \lambda \cdot \mu^n$ (see page 27 for the definition of operation $*$);

(ii) if $P \xrightarrow{x(y)} P'$, and $y \notin \mathrm{n}(P,Q)$, then there is Q' such that $Q \xrightarrow{x(y)} Q'$, and $P'\{w/y\} S_{n+1} Q'\{w/y\}$ for all names w; and

(iii) if $P \xrightarrow{\overline{x}(y)[p]} P'$, and $y \notin \mathrm{n}(P,Q)$, then there exist Q', and $q \in (0,1]$ such that $Q \xrightarrow{\overline{x}(y)[q]} Q'$, $P' S_{n+1} Q'$ and $p * q \geq \lambda \cdot \mu^n$.

An increasing sequence $\{S_n\}_{n=0}^{\infty}$ of binary relations on agents is called a $(\lambda,\mu)-$stratified (strong) bisimulation if both $\{S_n\}_{n=0}^{\infty}$ and $\{S_n^{-1}\}_{n=0}^{\infty}$ are $(\lambda,\mu)-$stratified simulations.

Definition 11. *Let P and Q be two agents. If there exists a $(\lambda,\mu)-$stratified bisimulation $\{S_n\}_{n=0}^{\infty}$ such that $(P,Q) \in S_0$, then P and Q are said to be $(\lambda,\mu)-$stratification-bisimilar, and we write $P \dot{\sim}_{\lambda,\mu} Q$.*

Obviously, $(\lambda,\mu)-$stratified bisimulation is a generalization of $\lambda-$bisimulation. If S is a $\lambda-$bisimulation, then the constant sequence $\{S_n = S\}_{n=0}^{\infty}$ is a $(\lambda,1)-$stratified bisimulation. Conversely, if $\{S_n\}_{n=0}^{\infty}$ is a $(\lambda,1)-$stratified bisimulation, then $\cup_{n=0}^{\infty} S_n$ is a $\lambda-$bisimulation. Thus, $P \dot{\sim}_{\lambda} Q$ if and only if $P \dot{\sim}_{\lambda,1} Q$.

The intuition behind the above definitions is that two processes are equated by a stratified bisimilarity if , roughly speaking, in few steps the difference between their transition probabilities is small enough. Note that the recursive construction in a bisimulation is unfolded as a sequence $\{S_n\}_{n=0}^{\infty}$ of relations in a $(\lambda,\mu)-$stratified bisimulation. When considering a pair of agents related by relation S_n, the nearness of their transition probabilities, represented by $p * q$, is only required not to be less than the value $\lambda \cdot \mu^n$, which will become smaller and smaller as n increases.

The following example explains that $(\lambda,\mu)-$stratified bisimilarity is really weaker that $\lambda-$bisimilarity.

Example 10. Let $0 < p < q < 1$ and $p + q \leq 1$, and let

$$A_n \stackrel{def}{=} \overline{x_n}y.A_{n+1} \text{ and } B_n \stackrel{def}{=} \overline{x_n}z.B_{n+1}$$

for all $n \geq 0$. Moreover, suppose that

$$p_{x_n}(y|y) = p^{n+1},\ p_{x_n}(z|y) = 1 - p^{n+1},$$

$$p_{x_n}(y|z) = q^{n+1} \text{ and } p_{x_n}(z|z) = 1 - q^{n+1}$$

for each $n \geq 0$. Then it is easy to see that $A_0 \dot{\sim}_\lambda B_0$ does not hold for any $\lambda > 0$. On the other hand, let

$$S_n = \{(A_i, B_i) : 0 \leq i \leq n\}$$

for every $n \geq 0$. Then $\{S_n\}_{n=0}^{\infty}$ is a $(\frac{p}{q}, \frac{p}{q})-$stratified bisimulation, and we have $A_0 \dot{\sim}_{\frac{p}{q},\frac{p}{q}} B_0$. In fact,

$$A_n \stackrel{\overline{x_n}y[p^{n+1}]}{\longrightarrow} A_{n+1}$$

is simulated by

$$B_n \stackrel{\overline{x_n}y[q^{n+1}]}{\longrightarrow} B_{n+1}$$

and vice versa, and

$$A_n \stackrel{\overline{x_n}z[1-p^{n+1}]}{\longrightarrow} A_{n+1}$$

is simulated by

$$B_n \stackrel{\overline{x_n}z[1-q^{n+1}]}{\longrightarrow} B_{n+1}$$

and vice versa. The pair (A_n, B_n) of agents in S_n has the pair (A_{n+1}, B_{n+1}) of derivatives in S_{n+1}. In addition, we have

$$p^{n+1} * q^{n+1} = (1 - q^{n+1}) * (1 - p^{n+1}) = 1,$$

$$q^{n+1} * p^{n+1} = \frac{p^{n+1}}{q^{n+1}}$$

and

$$(1 - p^{n+1}) * (1 - q^{n+1}) = \frac{1 - q^{n+1}}{1 - p^{n+1}} \geq \frac{p^{n+1}}{q^{n+1}} = \frac{p}{q}(\frac{p}{q})^n$$

since $p^{n+1} + q^{n+1} < p + q \leq 1$.

We now are ready to present an improvement of Proposition 5. We can see that $(1-)$compatibility of z and w and $(1-)$compatibility of x with respect to $\{w/z\}$ in Proposition 5 are weakened to $\mu-$compatibility and $\theta-$compatibility, respectively, in the following proposition.

Proposition 14. *Suppose that $z \neq w$, and z is normal. If z and w are $\mu-$compatible, each $x \in \mathrm{so}(P) \cup \mathrm{so}(Q)$ is $\theta-$compatible with $\{w/z\}$, $P \dot{\sim}_\lambda Q$ and $w \notin \mathrm{fn}(P,Q)$, then*

$$P\{w/z\} \dot{\sim}_{\lambda,\mu\theta} Q\{w/z\}$$

Equivalently, if $P \dot{\sim}_\lambda Q$ and $w \notin \mathrm{fn}(P,Q)$, then

$$P\{w/z\} \dot{\sim}_{\lambda, l(z,w)\cdot \inf_{x\in \mathrm{so}(P)\cup \mathrm{so}(Q)} c(x,\{w/z\})} Q\{w/z\}$$

Proof. Similar to Proposition 5. □

Except improving Proposition 5, $(\lambda,\mu)-$stratified bisimilarity can be used in many other ways. It may be observed that in Lemmas 11 and 12 we only consider $\lambda-$bisimilarity up to $\dot{\sim}_\mu$ (and restriction) for the case of $\mu = 1$. Now by introducing the notion of $(\lambda,\mu)-$stratified bisimilarity we are allowed to deal with $\lambda-$bisimulation up to $\dot{\sim}_\mu$ (and restriction) for $\mu < 1$.

Lemma 13. *Let S be a $\lambda-$bisimulation up to $\dot{\sim}_\mu$. We write $\mathbf{N}_0$ for the set of all normal names. Suppose that $\mathbf{N}_0 \cap \mathrm{n}(P,Q,U,V) \neq \emptyset$ for any agents P,Q,U and V with $P \dot{\sim}_\mu USV \dot{\sim}_\mu Q$. Then we have*

$$S \subseteq \dot{\sim}_{\lambda,\mu^2\cdot\theta\cdot\delta}$$

where

$$\theta = \inf_{z\in\mathbf{N}_0, w\in\mathbf{N}} l(z,w)$$

and

$$\delta = \inf_{z\in\mathbf{N}_0, x, w\in\mathbf{N}} c(x,\{w/z\})$$

Proof. We set $S_0 = S$ and

$$S_{n+1} = \{(P\{w/z\}, Q\{w/z\}) : P \dot{\sim}_\mu S_n \dot{\sim}_\mu Q, z \text{ is a normal name, and } w \notin \mathrm{fn}(P,Q)\}$$

and want to show that $\{S_n\}_{n=0}^\infty$ is a $(\lambda, \mu^2\cdot\theta\cdot\delta)-$stratified bisimulation. We assume that PS_nQ and proceed by induction on n to demonstrate that the three conditions in Definition 10 are satisfied. The case of $n = 0$ is immediate from Definition 7. In general, if $PS_{n+1}Q$, $P \xrightarrow{\overline{x}(y)[p]} P'$ and $y \notin \mathrm{n}(P,Q)$, then there are agents U_1, U_2, V_1 and V_2 such that $P = U_1\{w/z\}$, $Q = V_1\{w/z\}$, $U_1 \dot{\sim}_\mu U_2 S_n V_2 \dot{\sim}_\mu V_1$, z is a normal name, and $w \notin \mathrm{fn}(U_1, V_1)$. By Lemma 7', there are name u, agent U_1' and $p' \in (0,1]$ such that $U_1 \xrightarrow{\overline{u}(y)[p']} U_1'$, $u\{w/z\} = x$, $P' \equiv_\alpha U_1'\{w/z\}$ and

$p * p' \geq \theta$. With the assumption, we can find a normal name $z' \notin \mathrm{n}(U_1, U_2, V_1, V_2)$. Then it holds that $U_1 \stackrel{\overline{u}(z')[p']}{\longrightarrow} U_1'\{z'/y\}$. Since $U_1 \dot{\sim}_\mu U_2$, we have $U_2 \stackrel{\overline{u}(z')[p"]}{\longrightarrow} U_2'$, $p' * p" \geq \mu$ and $U_1'\{z'/y\} \dot{\sim}_\mu U_2'$ for some U_2' and $p" \in (0,1]$. Now using the induction hypothesis, there should be V_2' and $q" \in (0,1]$ such that $V_2 \stackrel{\overline{u}(z')[q"]}{\longrightarrow} V_2'$, $U_2' S_{n+1} V_2'$ and $p" * q" \geq \lambda \cdot (\mu^2 \cdot \theta \cdot \delta)^n$. Furthermore, from $V_2 \dot{\sim}_\mu V_1$ we obtain $V_1 \stackrel{\overline{u}(z')[q']}{\longrightarrow} V_1'$, $q" * q' \geq \mu$ and $V_2' \dot{\sim}_\mu V_1'$ for some agent V_1' and $q' \in (0,1]$. And from Lemma 6' it follows that

$$Q = V_1\{w/z\} \stackrel{\overline{x}(z')[q]}{\longrightarrow} Q" \equiv_\alpha V_1'\{w/z\}$$

for some $Q"$ and $q \in (0,1]$ with $q' * q \geq \delta$. Therefore,

$$\begin{aligned} p * q &\geq (p * p') \cdot (p' * p") \cdot (p" * q") \cdot (q" * q') \cdot (q' * q) \\ &\geq \theta \cdot \mu \cdot \lambda \cdot (\mu^2 \cdot \theta \cdot \delta)^n \cdot \mu \cdot \delta = \lambda \cdot (\mu^2 \cdot \theta \cdot \delta)^{n+1} \end{aligned}$$

On the other hand, we have $Q \stackrel{\overline{x}(y)[q]}{\longrightarrow} V_1'\{w/z\}\{y/z'\}$,

$$P' \equiv_\alpha U_1'\{w/z\} \equiv_\alpha U_1'\{z'/y\}\{y/z'\}\{w/z\}$$

and $U_1'\{z'/y\} \dot{\sim}_\mu S_{n+1} \dot{\sim}_\mu V_1'$. This implies that $(P', V_1'\{w/z\}\{y/z') \in S_{n+2}$. The cases for other actions are similar, and we complete the proof. □

Lemma 14. *With the same assumption and notation in Lemma 13, if S is a $\lambda-$bisimulation up to $\dot{\sim}_\mu$ and restriction, then $S \subseteq \dot{\sim}_{\lambda,\mu^2\cdot\theta\cdot\delta}$.*

Proof. Similar to Lemma 13. □

Before concluding this section, it should be pointed out that $(\lambda,\mu)-$stratified bisimulation still needs an elaboration. It seems that $(\lambda,\mu)-$stratified bisimulation is a very useful mathematical tool in reasoning about the behavior of probabilistic systems, in particular, in coping with recursion constructs in probabilistic process algebras, and the above two lemmas will provide a powerful proof technique for this purpose.

8. Reliability of Agents in $\pi-$Calculus

In Sections 5, 6 and 7, what concerns us is a comparison of the behaviors of two different agents in the π_N-calculus. In contrast, the aim of this section is to compare the behavior of an agent in the environment of noiseless channels and that of the same agent in the environment of noisy channels. Such a comparison is carried out by introducing the notions of $\lambda-$reliability bisimulation and reliability degree.

Definition 12. *Let S be a binary relation on agents, and let $\lambda \in (0,1]$. Then S is called a $\lambda-$reliability bisimulation if for any agents P, Q, PSQ implies*

(i) whenever $P \xrightarrow{\alpha} P'$ in π and α is a free action then for some Q' and $p \geq \lambda$, $Q \xrightarrow{\alpha[p]} Q'$ in π_N and $P'SQ'$;

(ii) whenever $P \xrightarrow{x(y)} P'$ (in π) and $y \notin \mathrm{n}(P,Q)$ then for some Q', $Q \xrightarrow{x(y)} Q'$ (in π_N) and for all noiseless channels w, $P'\{w/y\}SQ'\{w/y\}$;

(iii) whenever $P \xrightarrow{\overline{x}(y)} P'$ in π and $y \notin \mathrm{n}(P,Q)$ then for some Q' and $p \geq \lambda$, $Q \xrightarrow{\overline{x}(y)[p]} Q'$ in π_N and $P'SQ'$;

(iv) whenever $Q \xrightarrow{\alpha[p]} Q'$ in π_N, α is a free action and $p > 1-\lambda$ then for some P', $P \xrightarrow{\alpha} P'$ in π and $P'SQ'$;

(v) whenever $Q \xrightarrow{x(y)} Q'$ (in π_N) and $y \notin \mathrm{n}(P,Q)$ then for some P', $P \xrightarrow{x(y)} P'$ (in π) and for all noiseless channels w, $P'\{w/y\}SQ'\{w/y\}$;

(vi) whenever $Q \xrightarrow{\overline{x}(y)[p]} Q'$ in π_N, $y \notin \mathrm{n}(P,Q)$ and $p > 1-\lambda$ then for some P', $P \xrightarrow{\overline{x}(y)} P'$ in π and $P'SQ'$.

Definition 13. *(1) An agent P is said to be $\lambda-$reliable if there is a $\lambda-$reliability bisimulation S such that PSP.*

(2) The reliability degree of an agent P is defined to be

$$\mathrm{rel}(P) = \sup\{\lambda \in (0,1] : P \text{ is a } \lambda - \text{reliable}\}$$

When agent P is $1-$reliable, we simply say that it is reliable. Note that in clauses (ii) and (v) of Definition 12, channels w are assumed to be noiseless. Also see clauses (2.iii) and (2.vii) in Definition 14 below. This requirement is mainly for the technical reason.

To illustrate the above two definitions, let us see a very simple example.

Example 11. Suppose that x is a binary symmetric channel, and its channel matrix is given by $p_x(y|y) = 0.85$, $p_x(z|y) = 0.15$, $p_x(y|z) = 0.15$ and $p_x(z|z) = 0.85$. Then a routine calculation shows that $P = \overline{x}y.\mathbf{0} + \overline{x}z.\mathbf{0}$ is $0.85-$reliable.

The following lemma indicates that the class of $\lambda-$reliability bisimulations is closed under the operation of any union. It will be frequently used in the proofs of some sequent propositions.

Lemma 15. *If S_i is a $\lambda-$reliability bisimulation for all $i \in I$, then $\bigcup_{i\in I} S_i$ is also a $\lambda-$reliability bisimulation.*

Proof. Immediate from Definition 12. □

The next proposition demonstrates certain left-continuity of $\lambda-$reliability bisimulation in variable λ. It is in a sense similar to Proposition 4.

Proposition 15. *Suppose that each P' in $D^*(P)$ is support-finite and the defining agent of each agent identifier in P' is weakly guarded. Then P is* $\mathrm{rel}(P)-$*reliable.*

Proof. From Definition 13 we know that there is an increasing sequence $\{\lambda_n\}_{n=1}^{\infty}$ of real numbers in the unit interval with $\lim_{n\to\infty}\lambda_n = \mathrm{rel}(P)$, $\lambda_n < \mathrm{rel}(P)$ and P is λ_n-reliable for all $n \geq 1$. Thus from Definition 13, there should be a λ_n-reliability bisimulation S_n such that PS_nP for each $n \geq 1$. Without any loss of generality, we can assume that $\{S_n\}_{n=1}^{\infty}$ is an decreasing sequence. (Indeed, if not so, we let $S_n^* = \bigcup_{k=n}^{\infty} S_k$. Note that S_k is a λ_n-reliability bisimulation when $k \geq n$. Then it follows from Lemma 15 that S_n^* is also a λ_n-reliability bisimulation, the sequence $\{S_n^*\}_{n=0}^{\infty}$ is decreasing, and we may use $\{S_n^*\}_{n=0}^{\infty}$ to replace $\{S_n\}_{n=1}^{\infty}$.)

We write $D_0^*(P)$ for the set of all derivatives of P in the $\pi-$calculus, and set

$$S = (\bigcap_{n=1}^{\infty} S_n) \cap (D_0^*(P) \times D^*(P))$$

An argument similar to the demonstration that $(\bigcap_{n=0}^{\infty} \dot{\sim}_{\lambda_n}) \cap (D^* \times D^*)$ is a $\mathrm{Sim}(P,Q)-$bisimulation in the proof of Proposition 4 enables us to assert that S is a $\mathrm{rel}(P)-$reliability bisimulation. Note that here we have to use Corollary 1.4.6 in [50] to warrant the image-finiteness of transition relation $\xrightarrow{\alpha}$ in the $\pi-$calculus for a fixed action α. Thus, it is obvious that PSP and we complete the proof. □

The following proposition establishes a link between $\lambda-$reliability and bisimilarity. Thus, it allows us to consider correctness properties and reliability of a system in a single framework.

Proposition 16. *If P is $\lambda-$reliable, and both in π and in π_N we have $P \dot{\sim} Q$, then Q is also $\lambda-$reliable. Consequently, if it holds that $P \dot{\sim} Q$ both in π and π_N, then we have* $\mathrm{rel}(P) = \mathrm{rel}(Q)$.

Proof. We assume that S is the greatest $\lambda-$reliability bisimulation. The existence of S is guaranteed by Lemma 15. Then similar to Proposition 3 we can prove that $\dot{\sim}\, S\, \dot{\sim}$ is a $\lambda-$reliability bisimulation, where the first $\dot{\sim}$ is the bisimilarity relation in π, and the second one is the $(1-)$bisimilarity relation in π_N. □

The above proposition enables us to infer that $\mathrm{rel}(P) = \mathrm{rel}(Q)$ when P and Q are bisimilar both in π and in π_N. Then, what can we infer if we only know that $P \dot{\sim}_\lambda Q$ with $\lambda < 1$ in π_N? Indeed, we hope to find a function $R(\lambda)$ of the bisimilarity parameter λ such that $|\mathrm{rel}(P) - \mathrm{rel}(Q)| = R(\lambda)$ when $P \dot{\sim} Q$ in π and $P \dot{\sim}_\lambda Q$ in π_N. This is still an open problem.

Corollary 4. *If* $P \equiv_\alpha Q$ *then* $\mathrm{rel}(P) = \mathrm{rel}(Q)$.

Proof. Obvious from Theorem 1 in [44, II] and Proposition 16. □

A simple application of Proposition 16 gives us a set of agent pairs that have the same reliability degree. This set of reliability laws are very useful in analyzing reliability of complex mobile systems, and they allow us to use equational reasoning techniques in reliability analysis.

Corollary 5. *(1)* $\mathrm{rel}(P + \mathbf{0}) = \mathrm{rel}(P)$*;*
(2) $\mathrm{rel}(P + P) = \mathrm{rel}(P)$*;*
(3) $\mathrm{rel}(P_1 + P_2) = \mathrm{rel}(P_2 + P_1)$*;*
(4) $\mathrm{rel}(P_1 + (P_2 + P_3)) = \mathrm{rel}((P_1 + P_2) + P_3)$;
(5) If $A(\widetilde{x}) \stackrel{def}{=} P$*, then* $\mathrm{rel}(A(\widetilde{y})) = \mathrm{rel}(P\{\widetilde{y}/\widetilde{x}\})$*;*
(6) $\mathrm{rel}((y)P) = \mathrm{rel}(P)$ *if* $y \notin \mathrm{fn}^*(P)$*;*
(7) $\mathrm{rel}(y)(z)P = \mathrm{rel}((z)(y)P)$*;*
(8) $\mathrm{rel}((y)(P + Q)) = \mathrm{rel}((y)P + (y)Q)$*;*
(9) $\mathrm{rel}((y)\alpha.P) = \mathrm{rel}(\alpha.(y)P)$ *if* α *is not a (free) output action and* $y \notin \mathrm{n}(\alpha)$*;*
(10) $\mathrm{rel}((y)\overline{x}z.P) = \mathrm{rel}(\overline{x}z.(y)P)$ *if* $y \neq x, z$ *and* $p_x(y|z) = 0$*;*
(11) $\mathrm{rel}((y)\alpha.P) = 1$ *if* y *is the subject of* α*;*
(12) $\mathrm{rel}(P|\mathbf{0}) = \mathrm{rel}(P)$*;*
(13) $\mathrm{rel}(P_1|P_2) = \mathrm{rel}(P_2|P_1)$*;*
(14) $\mathrm{rel}((y)P_1|P_2) = \mathrm{rel}((y)(P_1|P_2))$ *if* $y \notin \mathrm{fn}^*(P_2)$*, and* $\mathrm{rel}((y)(P_1|P_2)) = \mathrm{rel}((y)P_1|(y)P_2)$ *if* $y \notin \mathrm{fn}^*(P_1) \cap \mathrm{fn}^*(P_2)$*; and*
(15) $\mathrm{rel}((P_1|P_2)|P_3) = \mathrm{rel}(P_1|(P_2|P_3))$.

Proof. Immediate from Theorems 3-8 in [44, II] and Propositions 7, 8 and 16. □

The remainder of this section is devoted to derive some useful algebraic properties of reliability. To this end, we have to introduce a proof technique similar to $\lambda-$bisimulation up to restriction. We first give a counterpart of Proposition 5 in the setting of reliability.

Lemma 16. *Let S be the greatest $\lambda-$reliability bisimulation, i.e. the union of all $\lambda-$reliability bisimulations. Suppose that z is a normal name, z and w are compatible, and each $x \in \mathrm{so}(P) \cup \mathrm{so}(Q)$ is compatible with $\{w/z\}$ (see Definitions 1 and 2), and $w \notin \mathrm{fn}(P,Q)$. Then PSQ implies $P\{w/z\}SQ\{w/z\}$.*

Proof. It is given by an argument similar to the proof of Proposition 5. Note that here we need to use Lammas 3 and 4 in [44, II] as well. The details are omitted. □

It seems that we can improve Lemma 16 in the way of Proposition 14 by introducing a stratified generalization of the notion of $\lambda-$reliability bisimulation (cf. Definitions 10 and 11).

Definition 14. *Let $\lambda \in (0,1]$. Then a binary relation S on agents is called a $\lambda-$reliability bisimulation up to restriction if for all agents P, Q with PSQ, we have:*

(1) if $w \notin \mathrm{fn}(P,Q)$, then $P\{w/z\}SQ\{w/z\}$; and

(2.i) whenever $P \xrightarrow{\overline{x}y} P'$ in π, then for some Q' and $p \geq \lambda$, $Q \xrightarrow{\overline{x}y[p]} Q'$ in π_N and $P'SQ'$;

(2.ii) whenever $P \xrightarrow{\tau} P'$ in π, then for some Q' and $p \geq \lambda$, $Q \xrightarrow{\tau[p]} Q'$ in π_N and $P'SQ'$, or for some $P", Q", w$ and $p \geq \lambda$, $Q \xrightarrow{\tau[p]} Q'$ in π_N and $P' \equiv_\alpha (w)P"$, $Q \equiv_\alpha (w)Q'$ and $P"SQ"$;

(2.iii) whenever $P \xrightarrow{x(y)} P'$ (in π) and $y \notin \mathrm{n}(P,Q)$, then for some Q', $Q \xrightarrow{x(y)} Q'$ (in π_N) and $P'\{w/y\}SQ'\{w/y\}$ for all noiseless channels w;

(2.iv) whenever $P \xrightarrow{\overline{x}(y)} P'$ in π and $y \notin \mathrm{n}(P,Q)$, then for some Q' and $p \geq \lambda$, $Q \xrightarrow{\overline{x}(y)[p]} Q'$ in π_N and $P'SQ'$;

(2.v) whenever $Q \xrightarrow{\overline{x}y[p]} Q'$ in π_N and $p > 1-\lambda$, then for some P', $P \xrightarrow{\overline{x}y} P'$ in π and $P'SQ'$;

(2.vi) whenever $Q \xrightarrow{\tau[p]} Q'$ in π_N and $p > 1-\lambda$, then for some P', $P \xrightarrow{\tau} P'$ in π and $P'SQ'$, or for some $P", Q"$ and w, $P \xrightarrow{\tau} P'$ in π and $P' \equiv_\alpha (w)P"$, $Q' \equiv_\alpha (w)Q"$ and $P"SQ"$;

(2.vii) whenever $Q \xrightarrow{x(y)} Q'$ (in π_N) and $y \notin \mathrm{n}(P,Q)$, then for some P', $P \xrightarrow{x(y)} P'$ (in π) and $P'\{w/y\}SQ'\{w/y\}$ for all noiseless channels w; and

(2.viii) whenever $Q \xrightarrow{\overline{x}(y)[p]} Q'$ in π_N, $y \notin \mathrm{n}(P,Q)$ and $p > 1-\lambda$, then for some P', $P \xrightarrow{\overline{x}(y)} P'$ in π and $P'SQ'$.

The validity of the technique of $\lambda-$reliability bisimulation up to restriction is then guaranteed by the following lemma.

Lemma 17. *If S is a $\lambda-$reliability bisimulation up to restriction, then it is included in the greatest $\lambda-$reliability bisimulation.*

Proof. Similar to Lemma 10. □

We also have an equivalent characterization of the greatest $\lambda-$reliability bisimulation in a way similar to Proposition 2.

Lemma 18. *Let S be the greatest $\lambda-$reliability bisimulation. Then PSQ if and only if*

(i) whenever $P \xrightarrow{\alpha} P'$ in π and α is a free action then for some Q' and $p \geq \lambda$, $Q \xrightarrow{\alpha[p]} Q'$ in π_N and $P'SQ'$;

(ii) whenever $P \xrightarrow{x(y)} P'$ (in π) and $y \notin \mathrm{n}(P,Q)$ then for some Q', $Q \xrightarrow{x(y)} Q'$ (in π_N) and for all noiseless channels w, $P'\{w/y\}SQ'\{w/y\}$;

(iii) whenever $P \xrightarrow{\overline{x}(y)} P'$ in π and $y \notin \mathrm{n}(P,Q)$ then for some Q' and $p \geq \lambda$, $Q \xrightarrow{\overline{x}(y)[p]} Q'$ in π_N and $P'SQ'$;

(iv) whenever $Q \xrightarrow{\alpha[p]} Q'$ in π_N, α is a free action and $p > 1-\lambda$ then for some P', $P \xrightarrow{\alpha} P'$ in π and $P'SQ'$;

(v) whenever $Q \xrightarrow{x(y)} Q'$ (in π_N) and $y \notin \mathrm{n}(P,Q)$ then for some P', $P \xrightarrow{x(y)} P'$ (in π) and for all noiseless channels w, $P'\{w/y\}SQ'\{w/y\}$; and

(vi) whenever $Q \xrightarrow{\overline{x}(y)[p]} Q'$ in π_N, $y \notin \mathrm{n}(P,Q)$ and $p > 1-\lambda$ then for some P', $P \xrightarrow{\overline{x}(y)} P'$ in π and $P'SQ'$.

Proof. Similar to Proposition 4.4 in [42]. □

We are now able to prove the following proposition. It asserts that reliability is preserved by the basic combinators of π, and thus provides us with some useful mathematical tools to analyze reliability of a complex system in a modularized (or compositional) way.

Proposition 17. *(1) $\tau.P$ is $\lambda-$reliable if and only if P is $\lambda-$reliable. Thus,* $\mathrm{rel}(\tau.P) = \mathrm{rel}(P)$.

(2) If P is $\lambda-$reliable, then $\overline{x}y.P$ is $\min(\lambda, p_x(y|y))-$reliable; and

$$\mathrm{rel}(\overline{x}y.P) \geq \min(p_x(y|y), \mathrm{rel}(P))$$

(3) $[x = y]P$ *is reliable, that is,* $\mathrm{rel}([x = y]P) = 1$ *when* $x \neq y$*, and* $[x = x]P$ *is* $\lambda-$*reliable if and only if* P *is* $\lambda-$*reliable and* $\mathrm{rel}([x = x]P) = \mathrm{rel}(P)$.

(4) If both P *and* Q *are* $\lambda-$*reliable, then* $P + Q$ *is also* $\lambda-$*reliable; and*

$$\mathrm{rel}(P + Q) \geq \min(\mathrm{rel}(P), \mathrm{rel}(Q))$$

(5) Suppose that for any

$$z \in \bigcup_{R \in D^*(P) \cup D^*(Q)} \mathrm{fn}(R),$$

$$w \notin \bigcap_{R \in D^*(P) \cup D^*(Q)} \mathrm{fn}(R)$$

and

$$x \in \bigcup_{R \in D^*(P) \cup D^*(Q)} \mathrm{so}(R),$$

z *is a normal name,* z *and* w *are compatible, and* x *is compatible with* $\{w/z\}$ *(see Definitions 1 and 2). If* P *and* Q *are all* $\lambda-$*reliable, then* $P|Q$ *is* $\lambda-$*reliable too; and*

$$\mathrm{rel}(P|Q) \geq \min(\mathrm{rel}(P), \mathrm{rel}(Q))$$

(6) Suppose that z *is a normal name, and suppose that* z *and* w *are compatible, and* x *is compatible with* $\{w/z\}$ *for all*

$$w \notin \bigcap_{Q \in D^*(P)} \mathrm{n}(Q) \text{ and } x \in \bigcup_{Q \in D^*(P)} \mathrm{so}(Q)$$

If P *is* $\lambda-$*reliable then so is* $(z)P$*; and* $\mathrm{rel}((z)P) \geq \mathrm{rel}(P)$.

(7) Suppose that y *is a normal name, and suppose that for any* $z \notin \mathrm{fn}(P, y)$ *and for any* $x \in \mathrm{so}(P)$*,* y *and* z *are compatible, and* x *is compatible with* $\{z/y\}$*. If* $P\{w/y\}$ *is* $\lambda-$*reliable for all* $w \notin \mathrm{fn}(P, y)$*, then* $x(y).P$ *is also* $\lambda-$*reliable; and*

$$\mathrm{rel}(x(y).P) \geq \inf_{w \notin \mathrm{fn}(P,y)} \mathrm{rel}(P\{w/y\})$$

Proof. (1) If $\tau.P$ is $\lambda-$reliable, then there is a $\lambda-$reliability bisimulation S such that $(\tau.P, \tau.P) \in S$. With Definition 12 it is easy to see that $(P, P) \in S$. So P is $\lambda-$reliable too. Conversely, if PSP and S is a $\lambda-$reliability bisimulation, then $S \cup \{(\tau.P, \tau.P)\}$ is also a $\lambda-$reliability bisimulation and we know that $\tau.P$ is $\lambda-$reliable.

(2) Suppose that S is a $\lambda-$reliability bisimulation and PSP. Then we put

$$S' = S \cup \{(\overline{x}y.P, \overline{x}y.P)\}$$

and want to show that S' is a λ'-reliability bisimulation, where $\lambda' = \min(\lambda, p_x(y|y))$. Indeed, $\overline{x}y.P \xrightarrow{\overline{x}y} P$ in π is simulated by

$$\overline{x}y.P \stackrel{\overline{x}y[p_x(y|y)]}{\longrightarrow} P$$

in π_N and $p_x(y|y) \geq \lambda'$. Conversely, if

$$\overline{x}z.P \stackrel{\overline{x}z[p_x(z|y)]}{\longrightarrow} P$$

in π_N and $p_x(z|y) > 1 - \lambda$, then $z = y$. Otherwise, we have $1 = \lambda' + (1 - \lambda') < p_x(y|y) + p_x(z|y) \leq 1$, a contradiction. Thus, the probabilistic transition in π_N may be simulated by $\overline{x}y.P \xrightarrow{\overline{x}y} P$ in π.

(3) Immediate from Theorem 5 in [44, II] and Propositions 7(6), (7) and 16.

(4) Let S_1 and S_2 be two $\lambda-$reliability bisimulations, and let PS_1P and QS_2Q. Then it is easy to show that $\{(P+Q, P+Q)\} \cup S_1 \cup S_2$ is a $\lambda-$reliability bisimulation.

(5) With Lemma 15, we may assume that S is the greatest $\lambda-$reliability bisimulation. Then PSP and QSQ. Let

$$S = \{(P_1|Q_1, P_2|Q_2) : P_1SP_2,\ Q_1SQ_2\ P_1, P_2 \in D^*(P) \text{ and } Q_1, Q_2 \in D^*(Q)\}$$

With Lemma 16, it suffices to show that S is a $\lambda-$reliability bisimulation up to restriction. The condition (1) of Definition 14 is verified by using Lemma 16, and the condition (2) of Definition 14 is proven by a thorough case analysis.

(6) Similar to the proof of Proposition 6(3). In the definitions S_0 and S there, the relation of $\lambda-$bisimilarity, $\dot{\sim}_\lambda$, must be replaced by the greatest $\lambda-$reliability bisimulation.

(7) We assume that for each $w \notin \text{fn}(P, y)$, $P\{w/z\}$ is $\lambda-$reliable and want to show that $x(y).P$ is $\lambda-$reliable. Let S be the greatest $\lambda-$reliability bisimulation. Then we only need to show that $(x(y).P, x(y).P) \in S$. This may be done by using Lemma 18 and an argument similar to the proof of Proposition 6(4). $\square$

9. Conclusion

The $\pi-$calculus is one of the most important models of mobile processes, in which agents may not only communicate with each other but also dynamically reconfigure their communication topology with passing of channel names. An implicit but fundamental assumption in the $\pi-$calculus is that communication channels are all noiseless. Such an abstraction from noise in channels considerably simplifies the presentation of π and gives us an elegant theory of mobile processes. However, it is not realistic in many applications where channel noise cannot be neglected.

This paper proposes an extension of π that accommodates noise in channels well, namely, π_N. The two calculi of π and π_N have the same syntax but different semantics. The semantical difference between them is due to their different assumptions on the nature of channels. In the π_N-calculus, noise in a channel is assumed, and it is represented by a family of probability distributions, which evaluates the probability that a channel name is sent via this channel but another name is received at its output. This statistical representation of noise follows a basic idea from Shannon's information theory. The operational semantics of π_N is given in terms of probabilistic transition systems.

The notion of $\lambda-$bisimulation is introduced in π_N, and it equates two processes whenever the difference between their probabilities of performing the same actions does not exceed the given threshold λ. A set of algebraic (equational) laws for $\lambda-$bisimiliarity $\dot{\sim}_\lambda$ is derived, and it is shown that $\dot{\sim}_\lambda$ is preserved by the basic combinators in π. This generalizes the corresponding results in the $\pi-$calculus. At the same time, some interesting distinctions related to bisimilarity between π and π_N are also observed. From Example 8 we see that there exist agents P and Q which are bisimilar in π, but not $\lambda-$bisimilar in π_N for any $\lambda > 0$. On the other hand, Example 7 shows that for some agents P and Q, $P \dot{\sim} Q$ does not hold in π, but $P \dot{\sim}_\lambda Q$ is valid in π_N with a certain $\lambda > 0$.

The relations of $\lambda-$bisimulation compare the behaviors of two different agents in the same framework of π_N. We also propose the notion of $\lambda-$reliability bisimulation which may be used to compare the different behaviors of the same agent in π and π_N. An agent is defined to be $\lambda-$reliable when its behavior in π_N is not far from that in π, with λ binding the deviation probability. It is proved that two agents bisimilar both in π and π_N have the same reliability degree. This establishes an interesting link between the two notions of bisimulation and reliability. Also, we show that reliability is preserved by compositions of agents using the basic constructs provided in π.

Of course, this paper is only a preliminary approach to the π_N-calculus. A more thorough study of π_N is anticipated, and we believe that the full power of π_N is still to be exploited. To conclude this paper, we propose some problems for further research.

We only treat the strong version of $\lambda-$bisimulation in the present paper. A weak version of $\lambda-$bisimulation in π_N, which ignores invisible internal actions, and an algebraic theory for it will be introduced in a continuation of this paper.

Recursive definitions are very useful in modelling infinite behavior of processes, and uniqueness of recursive equations in the sense of bisimilarity provides us with a powerful tool for reasoning about correctness of implementations with respect to specifications in various process algebras, including CCS [42], π [44] and higher-order process calculi [49, 54, 55, 62]. However, the construct of recursion is not carefully examined in this paper. The reason is that some technical difficulties are not easy to overcome when dealing with iterations in the probabilistic setting. Since the operation that we use to evaluate the probability of consecutive events is product,

powers p^n of a probability value $p < 1$ will often be involved when we consider bisimulation between two recursively defined agents. This causes sometimes a very serious difficulty because as the exponential n increases rapidly, p^n will vanish and probability information carried by p will be lost. A similar situation also arises in predicate transformer semantics of probabilistic (sequential) recursive programs; see [60] for a discussion. A notion of stratified bisimulation is already introduced in Section 7. The basic idea behind stratified bisimulation is that two processes can be (approximately) equated if the difference between their probabilities of performing the same actions is small enough in few steps. We believe that it will become a suitable proof technique in reasoning about recursive agents not only in π_N but also in other probabilistic extensions of process algebras, such as those introduced in [3, 19, 25, 31-33, 40, 48, 52, 59]. However, it seems that this technique still needs a further development in order to cope with recursion construct in π_N.

An interesting question is concerned with the notion of reliability: are there any agents P and Q such that $P \dot{\sim} Q$ in π (but not in π_N) and $\mathrm{rel}(P) > \mathrm{rel}(Q)$? Furthermore, if the answer to the previous question is positive, then given an agent P, how to find an agent Q with

$$\mathrm{rel}(Q) = \max\{\mathrm{rel}(P') : P \dot{\sim} P' \text{ holds in } \pi\}?$$

A solution to this question is very useful in real applications. For example, suppose a formal specification of a safety-critical system is described as agent P in the $\pi-$calculus. If we are able to find an agent Q which is bisimilar to P in π and which has the greatest reliability degree, then it is reasonable to choose Q as the optimal implementation of P. Corollary 5 tells us that it is impossible to find a solution only by simple equational reasoning.

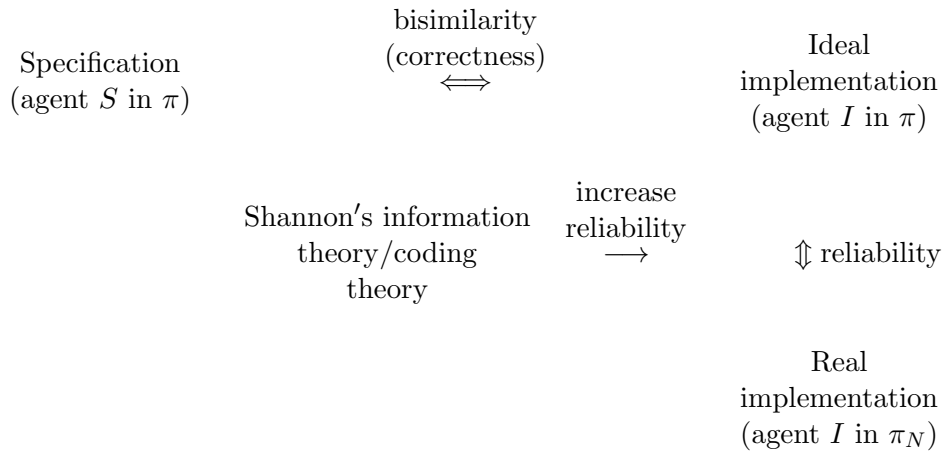

Fig.2

Indeed, the question concerning reliability has much deeper implications. Recall that in the introduction the relation among specification, ideal implementation and real implementation was visualized by Figure 1, and it can be described very well by the two notions of bisimulation and reliability developed in this paper. One of the most interesting and important problems for further studies is how to use techniques from coding theory [2, 53] to increase reliability of channels involved in the real implementation, and eventually to improve reliability of the whole system of the real implementation. Note that the potential improvement of reliability of a channel is bound by its transmission rate. It is obvious that this problem has a close link to performance analysis of mobile systems [20]. An extended version of the $\pi-$calculus, called stochastic $\pi-$calculus, in which it is possible to effectively compute performance measures, has already been introduced by Priami [48]. Thus, a solution to this problem might require us to establish a stochastic $\pi-$calculus with noisy channels, and then to find a tradeoff point between reliability of a mobile system modelled in such a calculus and its performance measures by employing Shannon's channel coding theorem and source coding theorem. This idea can be visualized by the above Figure 2. Furthermore, as is well-known, one of the current trends in information technology is that computation and communication are merging together due to the development of computer networks. Thus, a unified formal framework which is able to accommodate both computation and communication is highly expected. This suggests us to find a unification of the $\pi-$calculus and Shannon's information theory. The π_N-calculus developed in the current paper may be seen as the first step toward such an unification.

Finally, we consider the influence of noise in communication channels on information security. The Spi-calculus was proposed by Abadi and Gordon [1] as an extension of π for the description and analysis of cryptographic protocols. In Spi, some cryptographic primitives are introduced, and the security properties of protocols may be represented in terms of coarse-grained notions of protocol equivalences. Security protocols rely not only on cryptography but also on authenticity and privacy of communication channels. Thus, noise in communication channels will bring essential difficulty to formal reasoning about security properties of protocols because noise may cause that a private channel name is wrongly sent to an enemy. A reexamination of the Spi-calculus in the new environment of noisy channels would be another interesting problem for future research.

Acknowledgement: The work reported in this paper was carried out when the author was visiting Institute of Computer Science, University of Mannheim, Germany. The author is very grateful to Professor Mila Majster-Cederbaum for her stimulating discussions and for providing the excellent working environment. The author would like to thank Professor Martin Wirsing for his inspiring discussions. The author also would like to thank the anonymous referees for their invaluable criticisms, comments and suggestions which helped to improve considerably the presentation of this paper. In particular, they pointed out some mistakes in the

original versions of Lemmas 3 and 8 and the definition of $\mathrm{fn}^*(\cdot)$ and provided Example 5, and one of them kindly outline an asynchronous and reduction-based variant [17, 36, 37] of π_N, which is a very interesting topic for the further studies.

References

[1] Abadi, M., and Gordon, A. D, A calculus for cryptographic protocols: The spi calculus, *Information and Computation*, 148(1999)1–70. An extended version appeared as Digital Equipment Corporation Systems Research Center Report No. 149, 1998

[2] P. Abdulla, A. Annichini and A. Bouajjani, Symbolic verification of lossy channel systems: Application to the bounded retransmission protocol, in: *Tools and Algorithms for the Construction and Analysis of Systems, Proceedings*, LNCS 1579, Springer, 1999, pp. 208-222

[3] P. A. Abdulla, L. Boasson and A. Bouajjani, Effective lossy queue languages, in: *Automata, Languages and Programming, Proceedings*, LNCS 2076, Springer, 2001, pp. 639-651

[4] P. A. Abdulla, A. Bouajjani and B. Jonsson, On-the-fly analysis of systems with unbounded, lossy FIFO channels, in: *Computer-Aided Verification, Proceedings*, LNCS 1427, Springer, 1998, pp. 305-318

[5] P. A. Abdulla, A. Collomb-Annichini, A. Bouajjani and B Jonsson, Using forward reachability analysis for verification of lossy channel systems, *Formal Methods in System Design*, 25(2004)39-65

[6] P. A. Abdulla and B. Jonsson, Verifying programs with unreliable channel, *Information and Computation*, 127(1996)91-101

[7] P. A. Abdulla and Jonsson B, Undecidable verification problems for programs with unreliable channels, *Information and Computation*, 130 (1996)71-90

[8] P. A. Abdulla and A. Rabinovich, Verification of probabilistic systems with faulty communication, in: *Foundations of Software Science and Computation Structures, Proceedings*, LNCS 2620, Springer, 2003, pp. 39-53

[9] R. B. Ash, *Information Theory*, Dover, New York, 1990

[10] J. Baeten, J. Bergstra and S. Smolka, Axiomatizing probabilistic processes: ACP with generative probability, *Information and Computation* 122(1995)234-255 (Preliminary version appeared in: *CONCUR'92, Third International Conference on Concurrency Theory, Stony Brook, NY, USA, August 24-27, 1992, Proceedings,* LNCS 630, Springer, 1992, pp. 472-485)

[11] C. Baier and B. Engelen, Establishing qualitative properties for probabilistic lossy channel systems, in: J.-P. Katoen (Ed.), *Formal Methods for Real-Time and Probabilistic Systems: 5th International AMAST Workshop, ARTS'99, Bamberg, Germany, May 1999. Proceedings*, LNCS 1601, Springer, 1999, pp. 34-53

[12] M. Berger, Basic theory of reduction congruence for two timed asynchronous $\pi-$calculi, in: P. Gardner, N. Yoshida (eds.), *CONCUR 2004 - Concurrency Theory: 15th International Conference, London, UK, August 31 - September 3, 2004. Proceedings*, LNCS 3170, Springer, 2004, pp. 115-130

[13] M. Berger, *Towards Abstractions for Distributed Systems*, Ph.D. Thesis, Imperial College, London, 2002

[14] M. Berger and K. Honda, The two-phase commit protocol in an extended $\pi-$calculus, in: *Proc. EXPRESS'00, 7th International Workshop on Expressiveness in Concurrency, Penn State University, USA, August 21, 2000*

[15] J. A. Bergstra and J. -W. Klop, Algebra of communicating processes with abstraction, *Theoretical Computer Science*, 33(1985)77-121

[16] M. Boreale and D. Sangiorgi, Some congruence properties for $\pi-$calculus bisimilarities, *Theoretical Computer Science*, 198(1998)159-176

[17] G. Boudol, *Asynchrony and the $\pi-$calculus,* Rapport de Recherche 1702, INRIA, Sophia-Antipolis, 1992

[18] G. Boudol and I. Castellani, A non-interleaving semantics for CCS based on proved transitions, *Fundamenta Informaticae*, 11(1988)433-452

[19] F. van Breugel and J. Worrell, Towards quantitative verification of probabilistic transition systems, in: *Proceedings of the 23th International Colloquium on Automata, Languages and Programming,* LNCS 2076, Springer, Berlin, 2001, pp. 421-432

[20] E. Brinksma, H. Hermanns and J. -P. Katoen (eds.), *Lectures on Formal Methods and Performance Analysis*, LNCS 2090, Springer, Berlin, 2001

[21] S. D. Brookes, C. A. R. Hoare and A. W. Roscoe, A theory of communicating sequential processes, *Journal of the ACM*, 31(1984)560-599

[22] C. C. Chang and H. J. Keisler, *Model Theory* (3rd edition), Studies in Logic and the Foundations of Mathematics 73, North-Holland, Amsterdam, 1990

[23] R. Cleaveland, Z. Dayar, S. A. Smolka and S. Yuen, Testing preorders for probabilistic processes, *Information and Computation*, 154(1999)93-148

[24] R. Cleaveland, S. A. Smolka and A. E. Zwarico, Testing preorders for probabilistic processes, in: W. Kuich (ed.), *Automata, Languages and Programming (ICALP'92), Viena, Proceedings,* LNCS 623, Springer, 1992, pp. 708-719

[25] D. Desharnais, V. Gupta, R. Jagadeesan and P. Panangaden, Metrics for labelled Markov systems, in: *Proceedings of the 10th International Conference on Concurrency Theory,* LNCS 1664, Springer, 1999, pp. 258-273

[26] U. Engberg and M. Nielsen, A calculus of communicating systems with label passing, Report DAIMI PB-208, Computer Science Department, University of Aarhus, 1986

[27] Y. Feng and S. Y. Zhang, Approximate bisimilarity in probabilistic process algebras, unpublished note, 2002

[28] N. Francez, *Program Verification*, Addison-Wesley, Wokingham, 1992

[29] R. J. van Glabbeek, S. A. Smolka and B. Steffen, Reactive, generative, and stratified models of probabilistic processes, *Information and Computation*, 121(1995)59-80

[30] R. J. van Glabbeek, S. A. Smolka, B. Steffen and C. M. N. Tofts, Reactive, generative, and stratified models of probabilistic processes, in: *Proc. 5th Annual IEEE Symposium on Logic in Computer Science, Philadelphia*, 1990, pp. 130-141

[31] H. A. Hansson and B. Jonsson, A calculus for communicating systems with time and probability, in: *Proceedings of the 11th IEEE Symposium on Real-Time Systems, Lake Buena Vista, Florida, USA,* IEEE Computer Society Press, 1990, pp. 278-287.

[32] O. M. Herescu and C. Palamidessi, Probabilistic asynchronous pi-calculus, in: J. Tiuryn (ed.), *Proceedings of the Third International Conference on Foundations of Software Science and Computation Structures, Held as Part of the Joint European Conferences on Theory and Practice of Software*, LNCS 1784, Springer, 2000, pp. 146–160.

[33] Herescu, *The Probabilistic Asynchronous PI-Calculus*, Ph.D. Thesis, The Pennsylvania State University, 2002.

[34] C. A. R. Hoare, Communicating sequential processes, *Communications of the ACM,* 21(1978)666-677

[35] C. A. R. Hoare, *Communicating Sequential Processes*, Prentice Hall, New York, 1985

[36] K. Honda and M. Tokoro, An object calculus for asynchronous communication, in: P. America (ed.), *Proceedings of the European Conference on Object-Oriented Programming,* LNCS 512, pp. 133-147, Springer-Verlag (Heidelberg, 1991)

[37] K. Honda and N. Yoshida, On reduction-based process semantics, *Theoretical Computer Science*, 151(1995)437-486

[38] P. Iyer and M. Narasimha, Probabilistic lossy channel systems, in: M. Bidoit and M. Dauchet (eds.), *TAPSOFT'97, Theory and Practice of Software Development, Proceedings*, LNCS 1214, Springer, 1997, pp. 667-681

[39] K. G. Larsen and A. Skou, Bisimulation through probabilistic testing, *Information and Computation*, 94(1991)371-384

[40] R. Q. Lu and Z. C. Wei, *Truly Probabilistic Pi-Calculus and Risk Semantics*, Technical Report, Mathematical Institute, Academia Sinica, 2004.

[41] R. Milner, *A Calculus of Communicating Systems*, Lecture Notes in Computer Science 92, Springer-Verlag, Berlin, 1980

[42] R. Milner, *Communication and Concurrency*, Prentice-Hall, Englewood Cliffs, NJ, 1989

[43] R. Milner, *The polyadic $\pi-$calculus: a tutorial,* Technical Report ECS-LFCS-91-180, University of Edinburgh, 1991

[44] R. Milner, J. Parrow and D. Walker, A calculus of mobile processes, Parts I and II, Technical Report ECS-LFCS-89-85 and 86, University of Edinburgh, 1989; also see *Information and Computation* 100(1992)1-77

[45] R. Milner and D. Sangiorgi, Barbed bisimulation, in: *ICALP'92, Automata, Languages and Programming*, LNCS 623, Springer, 1992, 685-695

[46] M. Núñez, Algebraic theory of probabilistic processes, *Journal of Logic and Algebraic Programming*, 56(2003)117-177

[47] C. A. Petri, *Kommunikation mit Automaten,* Schriften des Rheinisch-Westfaelischen, Institutes fuer instrumentelle Mathematik an der Universitaet, Nr. 2, Bonn, 1962

[48] C. Priami, Stochastic $\pi-$calculus, *The Computer Journal*, 38(1995)578-589

[49] D. Sangiorgi, Bisimulation for higher-order process calculi, *Information and Computation*, 131(1996)141-178

[50] D. Sangiorgi and D. Walker, *The $\pi-$Calculus: A Theory of Mobile Processes*, Cambridge University Press, Cambridge, 2001

[51] P. Schnoebelen, The Verification of Probabilistic Lossy Channel Systems, in: Christel Baier et al. (eds.), *Validation of Stochastic Systems: A Guide to Current Research Editors*, LNCS 2925, Springer, 2004, pp. 445-465

[52] K. Seidel, Probabilistic communicating processes, *Theoretical Computer Science*, 152(1995)219-249

[53] C. E. Shannon, A mathematical theory of communication, I, II, *Bell System Technical Journal*, 27(1948)379-423; 623-656

[54] B. Thomsen, *Calculi for Higher Order Communicating Systems*, ph. D. thesis, Dept. of Computing, Imperial College, 1990

[55] B. Thomsen, Plain CHOCS, a second generation calculus for higher-order process, *Acta Informatica*, 30(1993)1-59

[56] P. Walley, *Statistical Reasoning with Imprecise Probabilities*, Chapman and Hall, London, 1991

[57] M. S. Ying, *Topology in Process Calculus: Approximate Correctness and Infinite Evolution of Concurrent Programs*, Springer-Verlag, New York, 2001

[58] M. S. Ying, Bisimulation indexes and their applications, *Theoretical Computer Science*, 275(2002)1-68

[59] M. S. Ying, Additive models of probabilistic processes, *Theoretical Computer Science*, 275(2002)481-519

[60] M. S. Ying, Reasoning about probabilistic sequential programs in a probabilistic logic, *Acta Informatica*, 39(2003)315-389

[61] M. S. Ying and M. Wirsing, Approximate bisimilarity, in: T. Rus (ed.), *Algebraic Methodology and Software Technology, 8th International Conference, AMAST 2000, Iowa City, USA, May 20-27, 2000, Proceedings*, Lecture Notes in Computer Science 1816, Springer-Verlag, Berlin, pp. 309-321

[62] M. S. Ying and M. Wirsing, Recursive equations in higher-order process calculi. *Theoretical Computer Science*, 266(2001)839-852