Topology, Randomness and Noise in Process Calculus*

Mingsheng Ying†
State Key Laboratory of Intelligent Technology and Systems,
Department of Computer Science and Technology,
Tsinghua University, Beijing 100084, China

Abstract

Formal models of communicating and concurrent systems are one of the most important topics in formal methods, and process calculus is one of the most successful formal models of communicating and concurrent systems. In the previous works, the author systematically studied topology in process calculus, probabilistic process calculus and pi-calculus with noisy channels in order to describe approximate behaviors of communicating and concurrent systems, and randomness and noise in them. This article is a brief survey of these works.

Key Words: Communicating and concurrent systems, Process calculus, Topology, Randomness, Noise

1 Introduction

Communication and concurency are one of the essential features of complex dynamic systems. Since 1960s, computer scientists have proposed various formal models to describe communication and concurrent behaviors of systems, e.g., Petri nets, Hoare's Communicating Sequential Processes (CSP), Milner's Calculus of Communicating Systems (CCS), pi-calculus and Bigraphs, Pneuli's Temporal logic, and Cardelli and Gordon's Ambient calculus. These works have been successfully applied in various fields such as

*This work was partly supported by the National Foundation of Natural Sciences of China (Grant No: 60621062)

†Email: yingmsh@tsinghua.edu.cn

concurrent programming, protocol verification for communication networks, and they have become one of the main directions in the mainstream of theoretical computer science. Among these models, Milner's process calculus, including CCS and pi [3,4], is one of the most important and mathematically beautiful theories of communicating and concurrent systems.

In 1980s , some researchers in China started their works on process calculus, and since then they have made a series of important contributions to this area. For example, Professor Huimin Lin and his colleagues proposed the notion of symbolic bisimulation, which have been widely used by foreign computer scientists in their researches; Professor Ruqian Lu introduced a truly concurrent model of CCS by employing Petri nets, and he presented a risk semantics for probabilistic pi-calculus; Professor Yuxi Fu proposed Chi calculus, and Professor Xinxin Liu established a Milner-Hennessy logic for value-passing CCS.

The author of the present article have also worked in the area of process calculus for years. This article is a short survey of his works in this area, and at the same time it will give a brief review on the related works by other researchers.

2 Topology in Process Calculus

Halmos, a famous mathematician, claimed that problems are the heart of mathematics. More generally, the author believes that prolems are the heart of all purely theoretic subjects. The author's work on process calculus was just motivated by an important problem raised by N. Francez in 1992. In the introduction of his research monograph "Program Verification" [1], Francez emphatically 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 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."

According to the author's understanding, Francez's statement mentioned above can be divided into two aspects: (1) In many complex realistic problems, software implementations cannot satisfy exactly their formal specifications, and they can only approximate the latter. Thus, we hope to find some formal methods which are suited to describe and reason about approximation between systems. (2) Up to now, correctness and reliability of softwares are considered separately, and they belong to two different areas in computer science. Is it possible to establish a unified framework in which both correctness and reliability of softwares may be treated well?

The author proposed topology in process calculus [19] as an attempt to solve the first aspect of the Francez's problem. The driving idea is to introduce suitable topological structures into various models of computation as our mathematical tools of describing approximation between systems, and we hope that they will provide some new formal methods in supporting development of softwares.

In the history of about 70 years of computer science, many models of computation have been invented. Roughly speaking, they can be divided into two categories: models of sequential computation and models of concurrent computation. Due to the rapid development of networking, one has to consider the environment of network in any application of computer technology. A computer network is a typical communicating and concurrent system. As stated above, Milner's process calculus is one of the most important formal models of communicating and concurrent systems. Thus, as the first step, the author focused his attention on the studies of topology in process calculus. It is worth pointing out that the main ideas can also be used in solving corresponding problems in other models of computation.

One of the central notions in process calculus is behavior equivalence between processes, including strong and weak bisimulations, observational equivalence, trace equivalence, testing equivalence and failure equivalence. In real applications, specification and implementation are usually represented by two processes, and correctness of programs is then defined to be behavior equivalence of these two processes. However, various behavior equivalences like bisimilarity require that the external behaviors of the two processes under consideration must be exactly matched. Consequently, they are not suited to describe approximation between specifications and implementations. There are mainly two schemes of approximation between specifications and implementations: static one and dynamic one. We call the former approximate correctness of programs, and the latter is in fact an (finite or infinite) evolution of programs. In the author's work, two different formal methods have been introduced for the description of these two

schemes of approximation.

In order to depict the evolution mechanism of communicating and concurrent systems, we proposed the notion of bisimulation limit based on the Moore-Smith theory of convergence. It was proved that bisimulation limits form a convergence class, and thus they naturally determine a topology on the set of processes. Furthermore, we examined consistency of bisimulation limits and structural operational semantics of CCS, and it was shown that the basic combinators in CCS is continuous with respect to bisimulation limits. This illustrates compatibility of the modularization technique in software methodology with the evolution behaviors of systems. These results are useful in our analysis, understanding and modeling of the evolution and approximation of communicating and concurrent systems. In addition, recently process calculus was used by Sumpter et al at the Center of Biomathematics, Oxford University, in modeling the behaviors of insects. Hopefully, our notion of bisimulation limit will become an effective mathematical tool for describing evolution of biological systems whose subsystems communicate with each other. On the other hand, bisimulation limits also have potential applications in the further theoretic development of process calculus itself. As is well known, in formal semantics of sequential computation, the solutions of recursive equations are often expressed as a certain kind of limits in some semantic domains. This happens in the failure semantics of CSP too. It is also an important problem to solve recursive equations in process calculus, and the uniqueness of solutions of recursive equations in process calculus was studied by Milner in depth. However, due to lack of a notion of limit consistent with bisimulations, the solutions of recursive equations in CCS could not be represented in the form of limits. Now the notion of bisimulation limit enables us to give a limit representation of solutions of recursive equations in CCS.

As we saw above, bisimulation limits are important in both the theoretic development of process calculus and its applications. One may wonder why such a useful notion was not proposed before. The main reason might be the difficulty of mathematical techniques. We know that semantic domains possess a natural ordering in either denotational semantics of sequential programming languages or in failure semantics of CSP, and a notion of limit can be derived directly from the order topology equipped automatically. As one more example, let us consider Nivat and de Bakker's metric semantics of nondeterministic programs. Its semantic domain is equipped with a metric which can be easily used to define a notion of limit. However, bisimulations are defined in a recursive way, and it is not easy to find an order or metric suited to define a notion of limit consistent with them. Therefore, the author

gave up the idea of finding such an order or metric and turn to directly define the notion of bisimulation limit from the external behaviors of systems by employing techniques from the theory of net convergence.

To formulate the notion of approximate correctness of programs in process calculus, a metric or a closure structure is assume on the set of basic actions, and then the author introduced the concepts of near bisimulation and bisimulation index. In a sense, they are two orthogonal approximate versions of the notion of bisimulation. The concept of bisimulation index provides with us a continuous spectrum of precision degrees in which we can depict approximate behavior equivalence of communicating and concurrent systems, and it allows us to adjust precision degree according the need from real applications. The concrete results obtained in this work includes: (1) in the case of ultra-metric, a logic of Hennessy-Milner type is established for bisimulation indexes; (2) a uniqueness theorem of solutions of recursive equations is proven in the sense of approximate bisimulation; (3) the rule of approximate communication is introduced to attack the problem of congruence of bisimulation indexes with respect to parallel composition; (4) the notion of approximate bisimulation is applied in describing approximate correctness of real time systems modeled in the languages of real time CCS and real time ACP.

Actually, two different way of defining topological structures are considered in the author's studies on topology in process calculus. One may be called the extensional way in which topologies are completely determined by the observable external behaviors of systems, and bisimulation limit topology is given in this way. The other is the intensional way, and topological structures appear in this way when considering near bisimulations and bisimulation indexes. In defining topologies in the intensional way, a topology on the set of basic actions is presumed, and it describe the nearness relation between basic actions. Then this presumed topology is reasonably extended to the set of all processes. It should be pointed out that the presumed topology on the set of basic actions comes from our real problems that we want to solve.

In conclusion, four methods are introduced in this section to deal with the problem of approximation between communicating and concurrent systems, and indeed they reflects four profiles of this problem. Because of limit of space, only basic ideas are outlined here, for details we referee to [19, 20, 28]. For related works, the notion of topological bisimulation was introduced by Cuijpers and Reniers [5] at Eindhoven University of Technology. A behavioral pseudo-metric was presented by van Breugel [1] at York University for metric labeled transition systems. Some ideas simi-

lar to approximate bisimulation were applied by Julius, Girard and Pappas [10, 11, 12, 8, 13, 4, 7, 18] from University of Pennsylvania in the their studies on stochastic hybrid systems.

3 Probabilistic Process Calculus

Classical formal methods such as VDM, Z, Modula 2, CCS, CSP, occam and the hardware description language HDL are mainly used in specifying and reasoning about the functions of systems. However, randomness resides in many complex software and hardware systems. For example, the ISO standards of communication protocols provide convenient description at the abstract level for the design of communication systems, but the implementation of these systems heavily depends on the reliability of communication channels, which is usually treated in a statistic way. This forces us to find formal models of probabilistic systems and to develop formal methods for verifying such systems.

To describe probabilistic behaviors in concurrency and communication, Morgan et al suggested a stratified probabilistic models of Hoare's CSP. At the same time, randomization of other models of concurrent computation was also carried out. Larsen, Skou, Jou, Smolka, Steffen and Tofts proposed reactive model, generative model and stratified model of probabilistic CCS, respectively. Lynch, Wu, Smolka and Stark studied carefully probabilistic I/O automata. Baeten and Bergstra built probabilistic ACP. In a word, there have been already a big amount of work on formal methods for stochastic (and/or probabilistic) concurrent computation, but some fundamental problems are still open.

One of the key problems in probabilistic CCS is the randomization of parallel composition. So far, all of the (relatively) successfully established models of probabilistic CCS adopt synchronous version of parallel composition. In such a composition, parallel components interact in each step, and its actions are formed by composing the actions of the components. Hence, nondeterministic choices are not involved in such a composition, and its randomization is much easier. However, asynchronous parallel composition, often called standard parallel composition, is much more important in CCS, and its randomization is quite difficult. Although D'Argenio, Hermanns and Katoen have made some essential contributions to the solution of this problem, it is still a long distance to reach a complete solution. The author [21] tried to propose a so-called additive model of probabilistic processes. He introduced a new kind of parallel composition with more than

two variables and more than one probability parameters. This enables us to overcome the difficulty that the "probabilization"of asynchronous parallel composition does not enjoy associativity. Among these probability parameters, some are used to express probabilities of the actions by a single process in the composition, and the others are used to indicate the probabilities of interactions between two processes in the composition. These probability parameters are required to satisfy a normalization condition. To be consistent with these probability parameters in parallel composition, we need to put probability parameters in prefix operators, and an explicit condition for the transitional semantics of the choice operator has to be changed into an implicit condition.

It may be conceived that various behavioral equivalences are still at the center of probabilistic process calculus. In the previous works, bisimulation semantics of probabilistic process calculus sets a strict constraint on probability parameters, and it requires that the behavior probabilities of two processes must be the same. The idea of approximate bisimulation discussed in the last section can be naturally applied in probabilistic process calculus, and in this way we are able to give a subtler semantic description of probabilistic concurrent programs. In [21], the author defined a class of behavioral equivalences with looser constraints on probability parameters for probabilistic concurrent systems. Some fundamental properties of these equivalences are established; for example, it was shown that they possess congruence with respect to various combinators. It should be mentioned that the techniques used in the treatment of these behavioral equivalence with probability parameters are much more complicated than those for classical behavioral equivalence.

For the details of the work discussed in this section, we referee to [21]. In addition, the author would like to point out that van Breugel [2, 3] at York University, Panangaden at McGill University and Mislove and Worrell from Tulane University studied approximate bisimulations of probabilistic processes in some different ways.

4 Semantics of Probabilistic Sequential Programs Based on Probabilistic Logic

To the author's best knowledge, the logical tool employed in all of previous works on reasoning about probabilistic program is classical two-valued logic. It is not difficult to see that approximate bisimulation for probabilistic processes considered in the last section is in fact a refinement of the classical

notion of bisimulation in a probabilistic logic. The idea of using a probabilistic logic as the meta-logical tool in describing probabilistic programs should also be very useful in the studies of probabilistic sequential programs.

In the 1990s, a research group consisting of Morgan et al at Oxford University introduced probabilistic information into the formal models of sequential computing systems. They generalized the concept of Dijkstra's weakest precondition into probabilistic setting and proposed the notion of probabilistic predicate transformer, and finally a refinement calculus for probabilistic programs was established [14, 17]. By the way, we would like to mention that Butler at University of Southampton found an application of probabilistic refinement calculus in quantum computation, and he analyzed Grover's quantum search algorithm by using the theoretical tools developed by Morgan et al.

Morgan et al have already established a correspondence between the relational model of probabilistic programs and probabilistic predicate transformers. Therefore, a key problem in the further development of probabilistic refinement calculus is the studies of various healthiness conditions of probabilistic predicate transformers, such as monotonicity, conjunctivity, disjunctivity, and continuity. As pointed out above, the logic tool used by Morgan et al in their work is classical two-valued logic. Since two-valued logic is not fine enough with respect to probabilistic programs, some essential difficulties will arise in solving the problems at a deeper level. In particular, the normal form theorem for various healthiness conditions cannot be generalized to the case of probabilistic programs.

Recently, the author [22] suggested to use probabilistic logic as a meta-logic tool in describing probabilistic programs, and proposed so-called semantic analysis approach of probabilistic logic to do so. The normal forms of probabilistic predicate transformers fulfilling various healthiness conditions were found, and a normal form theorem is proven so that each probabilistic sequential program can be decomposed to a composition of an angelic choice and a demonic choice. Furthermore, a game semantics is established for probabilistic sequential programming languages, and a winning strategy theorem is proven for probabilistic programs.

5 Pi-Calculus with Noisy Channels

In (non-probabilistic or probabilistic) value-passing process calculus, there is an implicit assumption that communication channels are always noiseless. It is not the case in many applications, and in particular, this assumption

is not true when describing the behaviors of networks like Internet. In a recent paper [26], the author combined some ideas from Shannon information theory and process algebras and he proposed a value-passing calculus with noisy channels, namely, π_N, the pi-calculus with noisy channels.

In considering correctness of programs, we often use a process to represent the specification of software, its implementation is described as another process, and then we check whether these two processes are behaviorally equivalent. In this case, we are comparing two processes in the same calculus. Interestingly, pi-calculus with noisy channels enables us to compare the behaviors of a process in the noiseless environment and its different behaviors in the noisy environment. If the implementation of a software specification is represented by a process, then the behavior of this process in pi-calculus and that in pi-calculus with noisy channels can be thought of as an ideal implementation and a realistic implementation, respectively, and their behavioral equivalence depicts reliability of the system under consideration, instead of its correctness. Thus, pi-calculus provides with us a unified framework in which both correctness and reliability of programs can be examined, and to a certain extent it give a solution to the second aspect of the Francez's problem.

The idea of pi-calculus was used by Abadi and Gordon in analyzing security protocols, and they proposed Spi-calculus. Noise must bring some serious new problems to the analysis of security protocols. Recently, the author started a research on Spi-calculus with noisy channels [27].

6 Conclusion

According to the Halmos's point of view, the Francez problem may be seen as the heart of the research area considered in this article. As progressing in solving this problem, the function of this heart becomes weaker and weaker, and we have to get a new heart as the driving force of our further studies. To conclude this article, we try to create such a new heart and hope it will be strong enough:

Open problem: *It should be realized that the combination of Shannon information theory and process algebras in the pi-calculus with nosy channels is not so deep. A much more interesting and difficult problem would be to estimate reliability measure of processes in pi-calculus with noisy channels by using the fundamental theorem of information theory, channel coding/sournce coding theorem. It is highly anticipated that a theory that unifies*

information theory and process algebras will be finally established.

References

[1] F. van Breugel, A behavioural pseudometric for metric labelled transition systems, in: M. Abadi and L. de Alfaro (eds.), *Proceedings of the 16th International Conference on Concurrency Theory (CONCUR), San Francisco, August 2005*, LNCS 3653, pages 141-155, Springer-Verlag

[2] F. van Breugel, The metric monad for probabilistic nondeterminism, preparing

[3] F. van Breugel, B. Sharma and J. Worrell, Approximating a behavioural pseudometric without discount for probabilistic systems, preparing

[4] A.A. Julius, A. Girard, G.J. Pappas, Approximate bisimulation for a class of stochastic hybrid systems, in the *Proc. American Control Conference 2006*, Minneapolis, USA

[5] P. J. L. Cuijpers and M. A. Reniers, Topological (Bi-)Simulation, *Electronic Notes in Theoretical Computer Science*, 100(2004)49-64

[6] N. Francez, *Program Verification,* Addison-Wesley, Wokingham, 1992

[7] A. Girard, A.A. Julius, G.J. Pappas, Approximate simulation relations for hybrid systems, in *Proc. IFAC Conf. Analysis and Design of Hybrid Systems 2006*, Alghero, Sardinia, Italy

[8] A. Girard, A.A. Julius, G.J. Pappas, Approximate simulation relations for hybrid systems, submitted to the *Int. J. Discrete Event Dynamic Systems*, 2006

[9] 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

[10] A.A. Julius, Approximate abstraction of stochastic hybrid automata, in *Hybrid Systems: Computation and Control*, LNCS Volume 3927, pp. 318-332, Springer Verlag, 2006

[11] A.A. Julius, G.J. Pappas, Approximate abstraction of stochastic hybrid systems, provisionally accepted to the *IEEE Trans. Automatic Control*, 2006

[12] A.A. Julius, G.J. Pappas, Approximate equivalence and synchronization of metric transition systems, submitted to the *Systems and Control Letters*, 2006

[13] A.A. Julius, G.J. Pappas, Approximate equivalence and approximate synchronization of metric transition systems, to appear in the *Proc. 45th IEEE Conf. Decision and Control 2006*, San Diego, USA

[14] A. McIver and C. Morgan, *Abstraction, Refinement and Proof for Probabilistic Systems,* Springer-Verlag, New York, 2005

[15] R. Milner, *Communication and Concurrency,* Prentice Hall, New York, 1989

[16] R. Milner, J. Parrow and D. Walker, A calculus of mobile processes, I, II, *Information and Computation,* 100(1992)1-77

[17] C. C. Morgan, A. McIver and K. Seidel, Probabilistic predicate transformers, *ACM Transactions on Programming Languages and Systems*, 18(1996)325-353

[18] P. Tabuada, A. Ames, A.A. Julius, G.J. Pappas, Approximate Reduction of Dynamical Systems, to appear in the *Proc. 45th IEEE Conf. Decision and Control 2006*, San Diego, USA.

[19] M. S. Ying, *Topology in Process Calculus: Approximate Correctness and Infinite Evolution of Concurrent Programs,* Springer-Verlag, New York, 2001

[20] M. S. Ying, Bisimulation indexes and their applications, *Theoretical Computer Science*, 275:1(2002), 1-68

[21] M. S. Ying, Additive models of probabilistic processes, *Theoretical Computer Science*, 275:2(2002), 481-519

[22] M. S. Ying, Reasoning about probabilistic sequential programs in a probabilistic logic, *Acta Informatica*, 39:5(2003)315-389

[23] M. S. Ying and M. Wirsing, Recursive equations in higher-order process calculi, *Theoretical Computer Science*, 266:1(2001), 839-852

[24] M. S. Ying, Weak confluence and tau-inertness, *Theoretical Computer Science*, 238:2(2000), 465-475

[25] M. S. Ying, A shorter proof to uniqueness of solutions of equations, *Theoretical Computer Science*, 216:2(1999), 395-397

[26] M. S. Ying, pi-calculus with noisy channels, *Acta Informatica*, 41:9(2005)525-593

[27] M. S. Ying, The spi calculus and noisy communication channels, preparing

[28] M. S. Ying and M. Wirsing, Approximate bisimilarity, in: T. Rus (Ed.), *Algebraic Methodology and Software Technology, 8th International Conference, AMAST 2000*, Proceedings, LNCS 1816, Springer, pp. 309-322