# Foundations of Quantum Programming (extended abstract)

Mingsheng Ying*

Center for Quantum Computation and Intelligent Systems,
University of Technology, Sydney, Australia
and
State Key Laboratory of Intelligent Technology and Systems,
Tsinghua University, Beijing, China

**Keywords:** Quantum computation; loop programs; predicate transformer semantics; Floyd-Hoare logic

Progress in the techniques of quantum devices has made people widely believe that large-scale and functional quantum computers will be eventually built. By then, super-powered quantum computer will solve many problems affecting economic and social life that cannot be addressed by classical computing. However, our experiences with classical computing suggest that once quantum computers become available in the future, quantum software will play a key role in exploiting their power, and quantum software market will even be much larger than quantum hardware market. Unfortunately, today's software development techniques are not suited to quantum computers due to the essential differences between the nature of the classical world and that of the quantum world. To lay a solid foundation for tomorrow's quantum software industry, it is critically essential to pursue systematic research into quantum programming methodology and techniques.

Intensive research on quantum programming has been conducted in the last 15 years, and many exciting results have been reported. The existing research can be roughly classified into the following categories.

- Design of quantum programming languages [1], [11], [13], [14], [15], [23].
- Semantics of quantum programming languages [5].
- Verification of quantum programs [2], [3], [4], [6].
- Quantum software architecture [17].
- Quantum compilers [25], [12]
- Concurrent quantum programming and quantum process algebras [7], [9], [10], [21], [24]

There are already two excellent survey papers of quantum programming [8], [16]. This talk mainly summarizes the author and his collaborators' work in the foundations of quantum programming.

* This work was partly supported by the National Foundation of Natural Sciences of China (Grant No: 60736011)

## 1 Quantum Loop Programs [22]

Loops are a powerful program construct in classical computation. Some high-level control features such as loop and recursion are provided in Selinger's functional quantum programming language QFC [15]. The power of quantum loop programs is yet to be further exploited. The exploitation of such power requires a deep understanding of the mechanism of quantum loops. The author and Yuan Feng examined thoroughly the behaviors of quantum loops in a language-independent way and found some convenient criteria for deciding termination of a general quantum loop on a given input in the case of finite-dimensional state spaces. More precisely, in [22], a general scheme of quantum loop programs was introduced, the computational process of a quantum loop was described, and the essential difference between quantum loops and classical loops was analyzed. In addition, we introduced the notions of termination and almost termination of a quantum loop. The function computed by a quantum loop was also defined. Quantum walks were considered to show the expressive power of quantum loops. Then we found a necessary and sufficient condition under which a quantum loop program terminates on a given mixed input state. A similar condition is given for almost termination. Furthermore, we proved that a quantum loop is almost terminating if and only if it is uniformly almost terminating, and a small disturbance either on the unitary transformation in the loop body or on the measurement in the loop guard can make any quantum loop (almost) terminating, provided that some dimension restriction is satisfied. A representation of the function computed by a quantum loop was presented in terms of finite summations of complex matrices.

## 2 Predicate Transformer Semantics of Quantum Programs [20], [19]

Since it provides a goal-directed program development strategy and nondeterminacy can be accommodated well in it, predicate transformer semantics has a very wide influence in classical programming methodology. There have been already two approaches to predicate transformer semantics of quantum programs in the literature. The first approach was proposed by Sanders and Zuliani [14] in designing qGCL, a quantum extension of the guarded-command language. In this approach, quantum computation is reduced to probabilistic computation by the observation (measurement) procedure. Thus, predicate transformer semantics developed for probabilistic programs can be conveniently used for quantum programs. The second approach was proposed by D'Hondt and Panangaden [5], where the notion of a predicate is directly taken from quantum mechanics; that is, a quantum predicate is defined to be an observable (a Hermitian operator) with eigenvalues within the unit interval. In this approach, forward operational semantics of quantum programs is described by super-operators according to Selinger [15], and a beautiful Stone-type duality between state-transformer (forwards) and predicate-transformer (backwards) semantics of quantum programs can be established by employing the Kraus representation theorem for super-operators.

To further develop the second approach, we have to tackle some problems that would not arise in the realm of classical and probabilistic programming. One of such problems is the commutativity of quantum weakest preconditions. Various logical operations of quantum weakest preconditions such as conjunction and disjunction will be needed in reasoning about complicated quantum programs, but defining these operations requires commutativity between the involved quantum predicates. However, the author and his collaborators [19] noticed that the weakest preconditions of two commutative quantum predicates do not necessarily commute. This is an obvious obstacle in the further development of predicate transformer semantics for quantum programs, and it seems to be very difficult to overcome in the general setting. The author and his collaborators [20] decided to focus their attention on a special class of quantum predicates, namely projection operators. One reason for this decision is conceptual, and it comes from the following observation: the quantum predicates dealt with in [5] are Hermitian operators whose eigenvalues are within the unit interval, and in a sense, they can be envisaged as quantization of probabilistic predicates. On the other hand, projection operators are Hermitian operators with 0 or 1 as their eigenvalues, and they should be thought of as quantization of classical (Boolean) predicates. Physically, the simplest type of measuring instrument is one performing so-called yes-no measurement. Only a single change may be triggered on such an instrument, and it is often called an effect by physicists. Another reason is technical: there is a bijective correspondence between the projection operators in a Hilbert space and the closed subspaces of this space. The set of closed subspaces of a Hilbert space was recognized by Birkhoff and von Neumann as (the algebraic counterpart) of the logic of quantum mechanics, and its structure has been thoroughly investigated in the development of quantum logic for over 70 years. Thus, we are able to exploit the power of quantum logic in our research on predicate transformer semantics of quantum logic.

The author and his collaborators [20] developed a quite complete predicate transformer semantics of quantum programs by employing some powerful mathematical tools developed in Birkhoff-von Neumann quantum logic. In particular, they proved universal conjunctivity, termination law and Hoare's induction rule for quantum programs. The proof of termination law requires an essential application of Takeuti's technique of strong commutator introduced in his studies of quantum set theory.

## 3 Floyd-Hoare Logic for Quantum Programs [18]

The fact that human intuition is much better adapted to the classical world than the quantum world is one of the major reasons that it is difficult to find efficient quantum algorithms. It also implies that programmers will commit much more faults in designing programs for quantum computers than programming classical computers. Thus, it is even more critical than in classical computing to provide formal methods for reasoning about correctness of quantum programs. Indeed, several proof systems for verification of quantum programs and quantum communication protocols have been proposed in the recent literature. For example, Baltag and Smets [2] presented a dynamic logic formalism of information flows in quantum systems, which is capable of describing various quantum operations such as unitary evolutions and quantum measurements, and particularly entanglements in multi-partite quantum systems. Brunet and Jorrand [3] introduced a way of applying Birkhoff and von Neumann's quantum logic to the study of quantum programs by expanding the usual propositional languages with new primitives representing unitary transformations and quantum measurements. In [4], Chadha, Mateus and Sernadas proposed a Floyd-Hoare-style proof system for reasoning about imperative quantum programs using a quantitative state logic, but only bounded iterations are allowed in their programming language. Feng et al. [6] found some useful proof rules for reasoning about quantum loops, generalizing several effective proof rules for probabilistic loops.

Recently, the author [18] established of a full-fledged Floyd-Hoare logic for deterministic quantum programs based on Selinger's idea [15] of modeling quantum programs as super-operators and D'Hondt and Panangaden's notion of quantum predicate as an Hermitian operator [5]. This logic includes a proof system for partial correctness and a proof system for total correctness of deterministic quantum programs. In particular, we are able to prove its (relative) completeness by exploiting the power of weakest preconditions and weakest liberal preconditions for quantum programs. It is worth mentioning that the proof of the (relative) completeness requires techniques quite different from those for classical programs and tools from analytic (continuous) mathematics.

## References

1. T. Altenkirch and J. Grattage, A functional quantum programming language, in: *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS)*, 2005, pp. 249-258.
2. A. Baltag and S. Smets, LQP: the dynamic logic of quantum information, *Mathematical Structures in Computer Science*, 16(2006)491-525.
3. O. Brunet and P. Jorrand, Dynamic quantum logic for quantum programs, *International Journal of Quantum Information*, 2(2004)45-54.
4. R. Chadha, P. Mateus and A. Sernadas, Reasoning about imperative quantum programs, *Electronic Notes in Theoretical Computer Science*, 158(2006)19-39.
5. E. D'Hondt and P. Panangaden, Quantum weakest preconditions, *Mathematical Structures in Computer Science*, 16(2006)429-451.
6. Y. Feng, R. Y. Duan, Z. F. Ji and M. S. Ying, Proof rules for the correctness of quantum programs, *Theoretical Computer Science*, 386(2007)151-166.
7. Y. Feng, R. Y. Duan, Z. F. Ji and M. S. Ying, Probabilistic bisimulations for quantum processes, *Information and Computation*, 205(2007)1608-1639.
8. S. J. Gay. Quantum programming languages: survey and bibliography, *Mathematical Structures in Computer Science*, 16(2006)581-600.
9. S. J. Gay and R. Nagarajan, Communicating quantum processes, in: *Proceedings of the 32nd ACM Symposium on Principles of Programming Languages, Long Beach, California, USA*, ACM Press, 2005, pp. 145 - 157.

10. P. Jorrand and M. Lalire, Toward a quantum process algebra, in: *Proceedings of the 1st ACM Conference on Computing Frontiers, Ischia, Italy*, ACM Press, 2005, pp. 111-119.
11. E. H. Knill, *Conventions for quantum pseudocode*, Technical Report LAUR-96-2724, Los Alamos National Laboratory, 1996.
12. R. Nagarajan, N. Papanikolaou and D. Williams, Simulating and compiling code for the sequential quantum random access machine, *Electronic Notes in Theoretical Computer Science*, 170 (2007)101124.
13. B. Ömer, *Structural quantum programming*, Ph.D. Thesis, Technical University of Vienna, 2003.
14. J. W. Sanders and P. Zuliani, Quantum programming, in: *Proceedings, Mathematics of Program Construction*, LNCS 1837, Springer-Verlag, 2000, pp. 88-99.
15. P. Selinger, Towards a quantum programming language, *Mathematical Structures in Computer Science*, 14(2004)527-586.
16. P. Selinger, A brief survey of quantum programming languages, in: *Proceedings of the 7th International Symposium on Functional and Logic Programming*, LNCS 2998, Springer, 2004.
17. K. M. Svore, A. V. Aho, A. W. Cross, I. L. Chuang, I. L. Markov, A layered software architecture for quantum computing design tools, *IEEE Computer*, 39(2006)74-83.
18. M. S. Ying, Hoare logic for quantum programs, *ACM Transactions on Programming Languages and Systems* Volume 33 Issue 6, December 2011 Article No. 19; also see: http://xxx.lanl.gov/abs/0906.4586.
19. M. S. Ying, J. X. Chen, Y. Feng and R. Y. Duan, Commutativity of quantum weakest preconditions, *Information Processing Letters*, 104(2007)152-158.
20. M. S. Ying, R. Y. Duan, Y. Feng and Z. F. Ji, Predicate transformer semantics of quantum programs, in: I. Mackie and S. Gay (eds.), *Semantic Techniques in Quantum Computation*, Cambridge University Press, 2010, pp. 311-360.
21. M. S. Ying and Y. Feng, An algebraic language for distributed quantum computing, *IEEE Transactions on Computers*, 58(2009)728-743.
22. M. S. Ying and Y. Feng, Quantum loop programs, *Acta Informatica*, 47(2010)221-250.
23. M. S. Ying and Y. Feng, A flowchart language for quantum programming, submitted.
24. M. S. Ying, Y. Feng, R. Y. Duan and Z. F. Ji, An algebra of quantum processes, *ACM Transactions on Computational Logic*, 10(2009)19
25. P. Zuliani, Compiling quantum programs, *Acta Informatica*, 41(2005)435-473.
26. P. Zuliani, Reasoning about faulty quantum programs, *Acta Informatica*, 46(2009)403-432