Research Statement

David Lo
School of Information Systems
Singapore Management University

Date of research statement revision: 1 Jan 2009

1. Research Problem and Directions

Difficulties in managing legacy systems and presence of bugs have cost billions of dollars annually. It is estimated that a substantial proportion of software cost is due to the difficulties in understanding existing/legacy systems especially during maintenance tasks, i.e. when new feature updates, bug fix, etc. are performed. US National Institute of Standards and Technology (NIST) estimated that software bugs have caused US economy to lose $59.5 billion dollars annually. One of the root causes associated with high cost of software maintenance and presence of bugs is the fact that documented software specification is often missing, incomplete and outdated in the industry. Also, many documented specifications are often too vague and can be interpreted in various ways.

As a step forward to reduce software maintenance cost and detect bugs, machine learning and data mining techniques have been employed to infer or reverse engineer high-level specifications from existing programs either statically (i.e. from code) or dynamically (i.e. from execution traces). This is termed as specification mining and has been one of the new, hot topics in software engineering. Dedicated sessions for this research direction have appeared in flagship conferences of software engineering in recent years. The specification mined can be used for understanding legacy systems, reducing software maintenance cost, re-engineering legacy system, improving regression tests, aiding verification of programs, and detecting bugs.

My research in general addresses the goal of reducing maintenance cost and improving dependability of systems. In particular, I focus on extending the frontiers of research in specification mining.

2. Research Outputs

I worked on extraction or mining of program specifications in various forms from program execution traces. Together with co-authors and colleagues, we have shown that the mined specifications are useful for program understanding, aiding regression tests and detecting bugs.

Mining Finite State Machines (FSMs) from Execution Traces. We have performed a comprehensive study on the accuracy of existing specification mining techniques in inferring specifications of dynamic API protocols in the form of Finite State Machines

(FSMs) from program execution traces. The proposed performance metrics have also been used by consequent studies (by us and others) in assessing the quality of new specification mining algorithms. Next, we have also developed a new architecture to improve accuracy, robustness and scalability of existing techniques mining specifications in the form of FSMs. These two pieces of work have appeared in [1] and [2] respectively.

Mining Repetitive Program Behavioral Patterns from Execution Traces. We have proposed a new algorithm to mine repetitive program behavioral patterns from execution traces. Program behavioral patterns are commonly described in software documentations, e.g., locking protocol: <lock, unlock>, telecommunication protocol: <off_hook, dial_tone_on, dial_tone_off, seizure_interrupt, ring_tone, answer, connection_on>, Java Transaction Architecture (JTA) protocol: <TxManager.begin, TxManager.commit>, <TxManager. begin, TxManager,rollback>, etc. Frequent repetitive software behavioral patterns shed light to high-level program specifications. It has been shown that the method is able to reveal the specifications of industrial-scale software. Effective search space pruning strategy has been employed to render the technique efficient to work on traces of an industrial program. The description of the work is available in [3].

Mining Temporal Rules from Execution Traces. To detect bugs and ensure correctness of software systems formal analysis techniques like model checking have been proposed. One of the major challenges hampering the wide application of the technique in the industry is the difficulties and programmers' reluctance in writing formal specifications. Without any specification to verify, the effectiveness of model checking technique in ensuring the correctness of systems and finding bugs is limited. To push the applications of verification techniques further, we mine temporal rules in the form of Linear Temporal Logic, one of the most commonly used formalisms for model checking. Effective search space pruning strategies are employed to render the technique efficient to work on traces of industrial program. We have shown the effectiveness of the technique in revealing software design rules and in mining bug-identifying properties. The work has been awarded second position at SIGPLAN Symposium on Programming Language Design and Implementation Student Research Competition 2007. The algorithmic part of the work is described in [4]. The application part of the work is described in [5].

Mining Live Sequence Charts from Execution Traces. Sequence diagram is one of the most commonly used formats in representing documented software specifications. In the software modeling community, a formal version of sequence diagram, termed as Live Sequence Chart (LSC) has been proposed. LSC has good tool supports from the wealth of existing research in the area, and hence a good target for mining. We have developed an algorithm to efficiently mine Live Sequence Charts from program execution traces. We have shown the applicability of the technique in mining sequence diagram from traces of an open-source medium-scale program. The description of the work is available in [6]. As an extension of the above, we have proposed a technique to mine for scenario-based triggers and effects in the semantics of LSC, which is described in [7].

Frequent Pattern Mining from Sequences of Events. I have also worked on frequent pattern mining from general sequences of events. One work is on mining generators of

sequential patterns [8]. Another work is on mining frequent closed repetitive subsequences [9]. The techniques are aimed to be general purpose and work on any data in the format of sequences of events.

For the above studies, I benefited from collaborations with co-authors from National University of Singapore, University of Illinois Urbana-Champaign, US, the Weizmann Institute of Science, Israel, and Nanyang Technological University.

3. Future Research Directions

In the future, as extensions to the above studies, the following are planned research directions:

- Application of existing mining techniques to more case studies on:
  - Security and intrusion detection
  - Program comprehension
  - Verification
  - Debugging
  - Testing
  - Re-engineering
- Further improvement to the efficiency of existing mining techniques
- Investigation to the theoretical foundations of existing specification mining techniques
- Utilization of the synergy of static and dynamic analysis in specification mining
- Construction of more research “bridges” joining the areas of data mining and software engineering

References

[1] David Lo and Siau-Cheng Khoo. QUARK: Empirical Assessment of Automaton-based Specification Miners. In proceedings of the 13th Working Conference on Reverse Engineering (WCRE'06) . Benevento, Italy. Oct 23-27, 2006.

[2] David Lo and Siau-Cheng Khoo. SMArTIC: Towards Building an Accurate, Robust and Scalable Specification Miner. In proceedings of the 14th SIGSOFT Symposium on Foundation of Software Engineering (FSE'06). Portland, Oregon. Nov 5-11, 2006.

[3] David Lo, Siau-Cheng Khoo and Chao Liu. Efficient Mining of Iterative Patterns for Software Specification Discovery. In proceedings of the 13th SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'07). San Jose, California. Aug 12-15, 2007.

[4] David Lo, Siau-Cheng Khoo and Chao Liu. Efficient Mining of Recurrent Rules from a Sequence Database. In proceedings of the 13rd International Conference on Database Systems for Advance Applications (DASFAA'08). New Delhi, India. March 19-21, 2008.

[5] David Lo, Siau-Cheng Khoo, Chao Liu. Mining temporal rules for software maintenance, Journal of Software Maintenance and Evolution: Research and Practice, vol. 20, no. 4, pp. 227–247, John Wiley & Sons, Inc., New York, NY, USA, 2008

[6] David Lo, Shahar Maoz and Siau-Cheng Khoo. Mining Modal Scenario-based Specifications from Execution Traces of Reactive Systems. In proceedings of the 22nd

IEEE/SIGSOFT International Conference on Automated Software Engineering (ASE'07). Atlanta, Georgia. Nov 5-9, 2007.

[7] David Lo and Shahar Maoz. Mining Scenario-Based Triggers and Effects. In proceedings of the 23rd IEEE/SIGSOFT International Conference on Automated Software Engineering (ASE'08). L'Aquila, Italy. September 15-19, 2008.

[8] David Lo, Siau-Cheng Khoo and Jinyan Li. Mining and Ranking Generators of Sequential Patterns. In proceedings of the 8th SIAM International Conference on Data Mining (SDM'08). Atlanta, USA. April 24-26, 2008.

[9] Bolin Ding, David Lo, Jiawei Han and Siau-Cheng Khoo. Efficient Mining of Closed Repetitive Gapped Subsequences from a Sequence Database, to appear in Proceedings of the 25th International Conference on Data Engineering (ICDE'09), Shanghai, China. March 29-April 4, 2009