[1] G. J. Myers, C. Sandler, T. Badgett. The art of software testing. John Wiley and Sons, 2011.

[2] S Anand, E. K. Burke, T. Y. Chen, J. Clark, M. B. Cohen, W. Grieskamp, M. Harman, M. J. Harrold, P. McMinn. An orchestrated survey of methodologies for automated software test case generation. *Journal of Systems and Software*, Elsevier, 2013, 86(8): 1978-2001.

[3] R. Baldoni, E. Coppa, D. C. Delia, C. Demetrescu, I. Finocchi. A survey of symbolic execution techniques. *ACM Computing Surveys (CSUR)*, ACM Press, 2018, 51(3): 50:1-50:39.

[4] J. C. King. A new approach to program testing. *Proceedings of the International Conference on Reliable Software*. ACM Press, 1975, pp. 228-233.

[5] D. M. Leonardo, N. Bjørner. Z3: An efficient SMT solver. *Proceedings of the 14th International conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2008)*. Springer, 2008: pp. 337-340.

[6] V. Ganesh, D. L. Dill. A decision procedure for bit-vectors and arrays. *Proceedings of the 19th International Conference on Computer Aided Verification (CAV 2007)*. Springer, 2007: pp. 519-531.

[7] V. Chipounov, V. Georgescu, C. Zamfir. Selective symbolic execution. Proceedings of the 5th Workshop on Hot Topics in System Dependability (HotDep). 2009.

[8] Odefroid P, Klarlund N, Sen K. DART: directed automated random testing[C]. In Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI’15). ACM, 2005, 213-223.

[9] Cadar C, Dunbar D, Engler D R. KLEE: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs[C]// In Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation (OSDI’08). 2008, 209-224.

[10] Godefroid P, Levin M Y, Molnar D A. Automated whitebox fuzz testing[C]// In Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS’08). 2008, 151-166.

[11] Dias Neto A C, Subramanyan R, Vieira M, et al. A survey on model-based testing approaches: a systematic review[C]. In Proceedings of the 1st ACM International Workshop on Empirical Assessment of Software Engineering Languages and Technologies, in conjunction with the 22nd IEEE/ACM International Conference on Automated Software Engineering (ASE’07). ACM, 2007: 31-36.

[12] Lee D, Yannakakis M. Principles and methods of testing finite state machines-a survey[C]. In Proceedings of the IEEE. IEEE Computer Society, 1996, 84(8): 1090-1123.

[13] Kansomkeat S, Rivepiboon W. Automated-generating test case using UML statechart diagrams[C]// In Proceedings of the 2003 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on Enablement Through Technology (SAICSIT’03). South African Institute for Computer Scientists and Information Technologists, 2003: 296-300.

[14] Han, Xue, Tingting Yu, and David Lo. "PerfLearner: learning from bug reports to understand and generate performance test frames." Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. ACM, 2018.

[15] Cohen D M, Dalal S R, Fredman M L, et al. The AETG system: An approach to testing based on combinatorial design[J]. IEEE Transactions on Software Engineering, 1997, 23(7): 437-444.

[16] Lei Y, Kacker R, Kuhn D R, et al. IPOG/IPOG-D: efficient test generation for multi‐way combinatorial testing[J]. Software Testing, Verification and Reliability, 2008, 18(3): 125-148.

[17] Ghazi S A, Ahmed M A. Pair-wise test coverage using genetic algorithms[C]. In Proceedings of the Congress on Evolutionary Computation (CEC’03). IEEE Computer Society, 2003, 1420-1424.

[18] Bryce R C, Colbourn C J. One-test-at-a-time heuristic search for interaction test suites[C]. In Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO’07). ACM, 2007: 1082-1089.

[19] Kobayashi N, Tsuchiya T, Kikuno T. A new method for constructing pair-wise covering designs for software testing[J]. Information Processing Letters, 2002, 81(2): 85-91.

[20] Medeiros F, Kästner C, Ribeiro M, et al. A comparison of 10 sampling algorithms for configurable systems[C]. In Proceedings of the 38th International Conference on Software Engineering (ICSE’16). ACM, 2016: 643-654.

[21] Chen T Y, Leung H, Mak I K. Adaptive random testing[C]. In Proceedings of the 9th Annual Asian Computing Science Conference (ASIAN’04). Springer, 2004: 320-329.

[22] Chen T Y, Kuo F C, Merkel R G, et al. Mirror adaptive random testing[J]. Information and Software Technology, 2004, 46(15): 1001-1010.

[23] Chen T Y, Merkel R, Wong P K, et al. Adaptive random testing through dynamic partitioning[C]. In Proceedings of the 4th International Conference on Quality Software (QSIC’04). IEEE, 2004: 79-86.

[24] Harman M, Mansouri S A, Zhang Y. Search based software engineering: A comprehensive analysis and review of trends techniques and applications[R]. Technical Report TR-09-03, Department of Computer Science, King’s College London, 2009.

[25] Ahmed M A, Hermadi I. GA-based multiple paths test data generator[J]. Computers & Operations Research, 2008, 35(10): 3107-3124.

[26] Xiao M, El-Attar M, Reformat M, et al. Empirical evaluation of optimization algorithms when used in goal-oriented automated test data generation techniques[J]. Empirical Software Engineering, 2007, 12(2): 183-239.

[27] Ghiduk A S, Harrold M J, Girgis M R. Using genetic algorithms to aid test-data generation for data-flow coverage[C]//In Proceedings of the 14th Asia-Pacific Software Engineering Conference (APSEC’07). IEEE, 2007: 41-48.

[28] Li H, Lam C P. An ant colony optimization approach to test sequence generation for state-based software testing[C]// In Proceedings of the 5th International Conference on Quality Software (QSIC’05). IEEE Computer Society, 2005, 255-264.

[29] Lefticaru R, Ipate F. Automatic state-based test generation using genetic algorithms[C]//In Proceedings of the 9th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC’07). IEEE, 2007: 188-195.

[30] Baudry B, Fleurey F, Jézéquel J M, et al. From genetic to bacteriological algorithms for mutation-based testing[J]. Software Testing, Verification and Reliability, 2005, 15(2): 73-96.

[31] K.-Y. Cai. Optimal software testing and adaptive software testing in the context of software cybernetics [J]. Information and Software Technology, 2002, 44(14): 841–855.

[32] G. B. Finelli. NASA Software Failure Characterization Experiments [J]. Reliability Engineering and System Safety, 1991, 32(1): 155–169.

[33] K.-Y. Cai, H. Hu, and F. Ye. Random Testing with Dynamically Updated Test Profile [C]. Proceedings of the 20th International Symposium on Software Reliability Engineering (ISSRE’09), IEEE Computer Society, 2009, pp. 198.

[34] Y. Li, B. B. Yin, J. Lv, and K.-Y. Cai. Approach for Test Profile Optimization in Dynamic Random Testing [C]. Proceedings of the 39th IEEE Annual International Computer Software and Applications Conference (COMPSAC’15), IEEE Computer Society, 2015, pp. 466–471.

[35] Z. Yang, B. Yin, J. Lv, K.-Y. Cai, S. S. Yau, and J. Yu. Dynamic Random Testing with Parameter Adjustment [C]. Proceedings of the 6th IEEE International Workshop on Software Test Automationthe, Co-located with the 38th IEEE Annual International Computer Software and Applications Conference (COMPSAC’14), IEEE Computer Society, 2014, pp. 37–42.

[36] J. Lv, H. Hu, and K.-Y. Cai. A Sufficient Condition for Parameters Estimation in Dynamic Random Testing [C]. Proceedings of the 3rd IEEE International Workshop on Software Test Automationthe, Co-located with the 35th IEEE Annual International Computer Software and Applications Conference (COMPSAC’11), IEEE Computer Society, 2011, pp. 19–24.

[37] K. Patel and R. M. Hierons, “A mapping study on testing nontestable systems,” Software Quality Journal, vol. 26, no. 4, pp. 1373–1413, 2018.

[38] S. S. Brilliant, J. C. Knight, P. E. Ammann. On the Performance of Software Testing Using Multiple Versions. Proceedings of the 20th international symposium on fault-tolerant computing (FTCS’90), IEEE Computer Society, 1990, pp. 408–415.

[39] K. Y. Sim, C. S. Low, F. C. Kuo. Eliminating Human Visual Judgment from Testing of Financial Charting Software. Journal of Software, 2014, 9(2): 298-312.

[40] W. K. Chan, S. C. Cheung. PAT: A Pattern Classification Approach to Automatic Reference Oracles for the Testing of Mesh Simplification Programs. Journal of Systems and Software, 2009, 82(3), 422–434.

[41] R. Guderlei, J. Mayer. Statistical Metamorphic Testing Testing Programs with Random Output by Means of Statistical Hypothesis Tests and Metamorphic Testing. Proceedings of the 7th international conference on quality software (QSIC’07), IEEE Computer Society, 2007, pp. 404–409.

[42] T. Y. Chen, S. C. Cheung, S. M. Yiu. Metamorphic Testing: A New Approach for Generating Next Test Cases. Technical Report HKUST-CS98-01, Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, 1998.

[43] S. Segura, G. Fraser, A. B. Sanchez, A. R. Cortes. A survey on metamorphic testing. IEEE Transactions on Software Engineering, 2016, 42(9): 805-824.

[44] T. Y. Chen, P. L. Poon, X. Xie. METRIC: Metamorphic Relation Identification Based on the Category-Choice Framework. Journal of Systems and Software, 2016, 116: 177-190.

[45] J. Zhang, J. Chen, D. Hao. Search-Based Inference of Polynomial Metamorphic Relations. Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering (ASE’14), ACM Press, 2014, pp. 701-712.

[46] F. H. Su, J. Bell, C. Murphy. Dynamic Inference of Likely Metamorphic Properties to Support Differential Testing. Proceedings of the 10th International Workshop on Automation of Software Test (AST’15), Co-located with the 37th IEEE International Conference on Software Engineering (ICSE’15), IEEE Computer Society, 2015, pp. 55-59.

[47] C. Sun, Y. Liu, Z. Wang, W.K. Chan. μMT: A Data Mutation Directed Metamorphic Relation Acquisition Methodology. Proceeding of the First International Workshop on Metamorphic Testing (MET 2016), collocated with ICSE 2016, IEEE Computer Society, 2016, pp.12-18.

[48] A. Gotlieb, B. Botella. Automated Metamorphic Testing. Proceedings of the 27th Annual International Conference on Computer Software and Applications (COMPSAC’03), IEEE Computer Society, 2003, pp. 34-40.

[49] H. Liu, X. Liu, T. Y. Chen. A New Method for Constructing Metamorphic Relations. Proceedings of the 12th International Conference on Quality Software (QSIC’12), IEEE Computer Society, 2013, pp. 59-68.

[50] G. Batra, J. Sengupta. An Efficient Metamorphic Testing Technique Using Genetic Algorithm. Proceedings of 5th International Conference on Information Intelligence, Systems, Technology and Management (ICISTM’11), Springer, 2011, pp. 180-188.

[51] G. Dong, T. Guo, P. Zhang. Security Assurance with Program Path Analysis And

Metamorphic Testing. Proceedings of the 4th International Conference on Software Engineering and Service Science (ICSESS’13), IEEE Computer Society, 2013, pp. 193-197.

[52] T. Y. Chen, F. C. Kuo, Y. Liu. Metamorphic Testing and Testing with Special Values. Proceedings of the 5th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD’04), 2004, pp. 128-134.

[53] X. Xie, W. E. Wong, T. Y. Chen, B. Xu. Metamorphic Slice: An Application in SpectrumBased Fault Localization. Information and Software Technology, 2013, 55(5): 866-879.

[54] G. Dong, S. Wu, G. Wang, T. Guo, Y. Huang. Security Assurance with Metamorphic Testing and Genetic Algorithm. Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT’10), IEEE Computer Society, 2010, pp. 397-401.

[55] C. Sun, G. Wang, B. Mu, H. Liu, Z. Wang, T. Y. Chen. A Metamorphic Relation-Based Approach to Testing Web Services without Oracles. International Journal of Web Services Research, 2012, 9(1): 51-73.

[56] W. K. Chan, S. C. Cheung, K. R. Leung. A Metamorphic Testing Approach for Online Testing of Service-Oriented Software Applications. International Journal of Web Services Research, 2007, 4(2): 61-72.

[57] J. Mayer, R. Guderlei. On Random Testing of Image Processing Applications. Proceedings of the 6th International Conference on Quality Software (QSIC’06), IEEE Computer Society, 2006, pp. 85-92.

[58] F. C. Kuo, S. Liu, T. Y. Chen. Testing a Binary Space Partitioning Algorithm with Metamorphic Testing. Proceedings of the ACM Symposium on Applied Computing (SAC’11), ACM Press, 2011, pp. 1482-1489.

[59] T. H. Tse, S. S. Yau. Testing Context-Sensitive Middleware-Based Software Applications. Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC’04). IEEE Computer Society, 2004, pp. 458-466.

[60] W. K. Chan, T. Y. Chen, S. C. Cheung, T. H. Tse, Z. Zhang. Towards the Testing of PowerAware Software Applications for Wireless Sensor Networks. Proceedings of the 12th AdaEurope International Conference on Reliable Software Technologies (ICRST’07), Springer, 2007, pp. 84-99.

[61] C. Sun, Z. Wang, G. Wang. A Property-based Testing Framework for Encryption Programs. Frontiers of Computer Science, Springer, 2014, 8(3): 478-489.

[62] I. H. Witten, E. Frank, M. A. Hall, and J. P. Christopher. Data Mining:Practical Machine Learning Tools and Techniques. Journal of Management Science, Ubon Ratchathani University, 2005, 3(6): 92-96.

[63] http://www.cleverhans.io/security/privacy/ml/2017/06/14/verifcation.html, The Challenge of Verifcation and Testing of Machine Learning.

[64] K. Pei, Y. Cao, J. Yang, S. Jana. Deepxplore: Automated Whitebox Testing of Deep Learning Systems. Proceedings of the 26th Symposium on Operating Systems Principles (SOSP’17), ACM Press, 2017, pp. 1-18.

[65] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, A. Swami. The Limitations of Deep Learning in Adversarial Settings. Proceedings of the 1st IEEE European Symposium on Security and Privacy (EuroS&P’16), IEEE Computer Society, 2016, pp. 372-387.

[66] O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, A. Criminisi. Measuring Neural Net Robustness with Constraints. Proceedings of the 10th Annual Conference on Neural Information Processing Systems (NIPS’16), 2016, pp. 2613-2621.

[67] N. Carlini, D. Wagner. Towards Evaluating the Robustness of Neural Networks. Proceedings of the IEEE Symposium on Security and Privacy (SP’17), IEEE Computer Society, 2017, pp. 39-57.

[68] O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, A. Criminisi. Measuring Neural Net Robustness with Constraints. Proceedings of the International Conference on Machine Learning (ML’17), IEEE Computer Society, 2017, pp. 2613-2621.

[69] S. Gu, L. Rigazio. Towards Deep Neural Network Architectures Robust to Adversarial Examples. Proceedings of the International Conference on Learning Representations (ICLR’15), IEEE Computer Society, 2015, pp. 777-780.

[70] X. Huang, M. Kwiatkowska, S. Wang, M. Wu. Safety Verification of Deep Neural Networks. Proceedings of the International Conference on Computer Aided Verification (CAV’17), Springer, 2017, pp. 3-29.

[71] Li L, Huang W L, Liu Y, et al. Intelligence testing for autonomous vehicles: a new approach[J]. IEEE Transactions on Intelligent Vehicles, 2016, 1(2): 158-166.

[72] Huang W L, Wen D, Geng J, et al. Task-specific performance evaluation of UGVs: case studies at the IVFC[J]. IEEE transactions on Intelligent Transportation Systems, 2014, 15(5): 1969-1979.

[73] Li L, Wen D, Zheng N N, et al. Cognitive cars: A new frontier for ADAS research[J]. IEEE Transactions on Intelligent Transportation Systems, 2012, 13(1): 395-407.

[74] Hernández-Orallo J. Evaluation in artificial intelligence: from task-oriented to ability-oriented measurement[J]. Artificial Intelligence Review, 2017, 48(3): 397-447.

[75] Argall B D, Chernova S, Veloso M, et al. A survey of robot learning from demonstration[J]. Robotics and autonomous systems, 2009, 57(5): 469-483.

[76] Kuefler A, Morton J, Wheeler T, et al. Imitating driver behavior with generative adversarial networks[C]// In Proceedings of IEEE Intelligent Vehicles Symposium (IV). IEEE, 2017: 204-211.

[77] Li L, Lin Y L, Zheng N N, et al. Artificial intelligence test: a case study of intelligent vehicles[J]. Artificial Intelligence Review, 2018: 1-25.