An Analysis of Quito Coverage for Quantum Testing

Ganesh Nanduru (bae9wk) and Alex Walsh (ajw2qy)

***Abstract*—With the potential to make significant changes in the landscape of computing, it is essential to test quantum programs in order to understand qubit behavior and continue to develop more complex quantum computers and functions. This position paper surveys the Quito testing approach presented in "Assessing the Effectiveness of Input and Output Coverage Criteria for Testing Quantum Programs", and we advocate for the use of input coverage and output coverage to test quantum programs. We make this claim based on the evidence included in the visionary paper and our analysis of the other testing approaches that have been presented. Furthermore, we renounce the use of input-output coverage and the Wrong Output Oracle to assess quantum test suites. While we acknowledge that quantum testing must be conducted as thoroughly as possible, we argue that the use of input-output coverage and the Wrong Output Oracle are unnecessarily expensive components of the Quito approach. In this position paper, we aim to support and critique our chosen visionary paper as well as offer additional aspects of quantum computing that we believe should be accounted for in testing methodologies such as quantum noise.**

I. Introduction

Quantum computing is an approach to computing that harnesses laws of quantum mechanics to perform calculations more efficiently than classic computing. While classic computers use bits as their basic unit of information, quantum computers use qubits to operate quantum gates, making them extremely powerful and complex. Most notably, qubits can exist in multiple states at any given moment in time, due to a quantum mechanics principle known as superposition. This also means that the value of a qubit can fluctuate between one and zero, so a quantum program outcome cannot be predicted in advance, only observed when it is done executing. This has major significance in the world of quantum testing since researchers must devise new ways to test the behavior and accuracy of quantum programs.

Considering that the exact value of a qubit is held in superposition, many different branches of quantum programs must be accounted for in a test suite, which creates the need for expansive test requirements. To help simplify test design, there is another property of qubits that enables programmers to predict the outcome of a quantum program. Stochasticity is a characteristic of qubits that asserts they are well-described using a probability distribution function. By prioritizing more probable branches, the amount of tests needed to adequately assess quantum programs can be optimized despite superposition [1].

Due to the deterministic nature of classical number generation, random numbers created by classical programs are not truly random, and can be predicted and reproduced. However, because qubits can exist in superposition, quantum computers are able to produce actual physically random numbers, making them a robust tool for applications requiring randomness, such as encryption algorithms and optimization problems [2].

As new quantum technology is being developed to expand the capabilities of computing, testing criteria must also be improved to cover the wider breadth of software bugs that may propagate in these unique conditions. Based on the data provided in our chosen visionary paper, it is our position that input coverage and output coverage are effective coverage types for quantum programs. We make this claim based on the evidence included in the visionary paper and comparisons to the QuanFuzz and QSharpCheck test suites. However, we do not agree with the use of input-output coverage and the implementation of the Wrong Output Oracle test oracle. We view these components of the Quito approach as too expensive for what they contribute to the testing methodology. In addition to these claims, we acknowledge that there are many more factors to consider when approaching quantum testing. In particular, we propose that the researchers consider quantum noise as they continue to develop effective quantum testing. We believe that considering this, as well as other unique characteristics of qubits, will allow quantum testing to be more thorough, efficient, and applicable as quantum computers become more readily available and integral in modern technology.

II. Survey of The Literature

The visionary paper we assessed was "Assessing the Effectiveness of Input and Output Coverage Criteria for Testing Quantum Programs". In their work, the researchers presented a testing methodology for quantum programs called Quito (QUantum InpuT Output coverage). The Quito approach consists of three coverage criteria: input coverage (IC), output coverage (OC), and input-output coverage (IOC). Further, the researchers defined two types of test oracles to assess test suites: Wrong Output Oracles (WOO) and Output Probability Oracle (OPO). For clarity before defining the different types of coverage, a valid output is one in which the probability of a quantum program returning that output is not zero. IC requires that there exists

a test for every possible input value in the test suite. OC requires that there exists a test which results in each valid output being observed at least once in the test suite. And finally, IOC requires that for each pair of valid inputs and outputs, there exists a test in the test suite such that running the program with the input results in the output.

The researchers generated test suites using each of the levels of coverage for five basic quantum programs: Ent, Swap, RCR, Inc, and Dec. Every test suite was then subject to assessment under one or both of the testing oracles. First, the test suite is run against WOO, which checks that the tests in a given suite produce valid outputs. If a test does not produce a valid output, the test is labeled as a definite fail, the assessment terminates, and the test suite fails the WOO oracle. If the test suite satisfies the WOO, it is evaluated against the OPO, which returns an expected output with its corresponding expected probability and assesses failure by evaluating the statistical significance of the output and probability. The OPO runs through all tests in a given test suite and evaluates whether they are a likely fail or inconclusive. After assessing their test suites with the test oracles, the researchers used mutation analysis to determine the effectiveness of each coverage criteria using four mutation operators: Add Gate, Delete Gate, Replace Gate, and Replacement Mathematical Operator.

Two uncertainties the researchers attempted to address in their experiments were how cost-effective each level of coverage was overall and how cost-effective each level of coverage was under the assessment of the test oracles. To describe test coverage effectiveness, researchers reported the mutation score, a metric calculated by dividing the number of mutations killed by the number of unique mutants for each program. [3]

III. Analysis and Discussion

The results of our chosen visionary paper are displayed in Table II of our visionary paper. The table contains the overall mutation scores and average number of test cases for each test suite of the five basic quantum programs. Data for both test oracles and their combined results are displayed for every IC, OC, and IOC.

We noticed a lack of mutation score improvement from IC and OC to IOC. In fact, the IOC coverage was less effective than IC and OC for the Inc and Dec programs, respectively, with 7.7 percent decrease in coverage for both. Additionally, IOC required over 80,000 test cases for 4 our of 5 of their programs under test, whereas IC and OC never used more than 32,000 tests. This absence of improvement combined with the significant increase in the average number of test cases for all five IOC test suites under the OPO was a stark indication that we needed to reevaluate the cost-benefit component that IOC has in the Quito testing approach. Furthermore, we noticed that the OPO was used even if a test suite had passed the WOO, and this prompted us to consider the role of the WOO and how realistic it is as a test oracle, should the Quito approach be used in other real-world applications.

While the visionary paper provides a convincing case for how IC and OC analyze mutations well with a relatively lower number of test cases, we will examine two contemporary test suites in the quantum testing research field to determine how Quito compares with respect to efficiency, coverage, and mutation testing.

QuanFuzz is an implementation of fuzz testing on quantum programs. Fuzz testing, or fuzzing, tests programs by supplying them random malformed data and analyzing how they react to these inputs. QuanFuzz implements fuzzing by generating matrices to represent the probabilities of random qubit states, then uses a search algorithm to find the most likely test paths. To generate mutants, QuanFuzz applies six quantum gates to random qubits. The researchers found QuanFuzz obtained 20 to 60 percent more branch coverage than fully random fuzz testing, which shows promise but cannot be extrapolated as the paper did not feature any comparisons to other quantum test suites. Additionally, performance of the testing suite slowed down dramatically as the number of qubits being tested rose, reaching an iteration time upwards of 10,000 seconds for a seven-qubit program, compared to 1.39 seconds for a two-qubit program. While QuanFuzz scaled poorly in more complex quantum programs, we have not observed results suggesting the same for Quito's coverage criteria, as the programs Quito assessed used at most two qubits. [4]

QSharpCheck is a property-based test suite. It borrows the use of probability density matrices from QuanFuzz and implements a statistical analysis similar to Quito's OPO to determine whether a program has failed a test suite. The difference with QSharpCheck lies in how mutations and test conditions are generated. Instead of applying gate transformations to qubits to mutate tests, QSharpCheck will simply mutate a line of code to switch or misdirect an operation, similar to classical computing mutations. One example is changing a boolean expression from an equals comparator to a not equals comparator. Using QSharpCheck, researchers reported an 80 percent mutation score with an average runtime of 51 seconds for a three-qubit program and a 60 percent mutation score with an average runtime of 66 milliseconds for a two-qubit program. Once again, we observe a harsh tradeoff in time efficiency as the number of qubits tested is increased. [5]

While both QuanFuzz and QSharpCheck present innovative and effective ways to test quantum programs, we believe that Quito has advantages which make it more effective than the others. After analyzing the QuanFuzz methodology, we view the matrix generation of test paths as an efficient way to quickly kill

mutants, but the explicit coverage of known inputs and outputs implemented by Quito is more effective in covering edge cases of quantum programs, hence the higher mutation scores. We recognize the approach of iteratively generating mutants could potentially be viable in quantum computing due to the principles of quantum interference and noise which we will explain later in the paper. However, we observe an immediate disadvantage of QuanFuzz's method of probabilistically generating mutants is a lack of mutant coverage which could lead to test paths missing vital faults in the code.

An advantage of Quito over QSharpCheck is the use of quantum-specific mutators, as the authors of QSharpCheck note a potential threat to the validity of their design is whether or not their mutation operators represent actual faults. Further, Quito offers a greater coverage of faults through mutations, since it uses actual quantum gates to mutate programs instead of code line swaps. Additionally, Quito's criteria of IC and OC provide simple, concrete amounts of test requirements necessary for a quantum test suite, whereas QSharpCheck relies on changing arbitrary program properties and running a statistical analysis to guess whether or not the test suite adequately covers program faults. Quito may see improvements in covering program faults by adopting QSharpCheck's more classical approach to mutating code, but its use of quantum gates for mutation highlights its effectiveness in adequately assessing program faults.

While we agree that Quito's use of IC and OC is effective at testing quantum programs, we do not agree in the implementation of IOC with respect to testing quantum programs. We make this claim based on our analysis of Table II from the visionary paper. This data shows that the average total number of test cases in IOC test suites is significantly higher for each of the five programs. Furthermore, the total IOC mutation scores for four of the five programs are either equal to or less than the mutation scores for input coverage and output coverage test suites. The only exception is with the RCR program; here, IOC provided a mutation score of 95 percent whereas input and output coverage provided equal mutation scores of 80 percent.

Focusing on the Ent and Swap programs, the results of Quito's research suggest that there are no differences between the IC, OC, and IOC criteria for these two basic qubit programs. This is seen by the fact that the mutation coverage for each of the three criteria were the same for both Ent and Swap. This observation highlights the drastic tradeoff of efficiency upon implementing IOC, as the article provides that IOC criteria required 145,556 more test cases than the output coverage criterion and 152,768 more test cases than the IC criterion. This is too steep of an efficiency tradeoff for better code coverage, and we think Quito could benefit from omitting coverage for input-output pairs. In our opinion, the cost of hundreds of thousands of additional tests for virtually the same mutation score does not seem like productive investment. We believe that there are other factors to consider which we discuss later in this paper that could serve as a better investment into Quito as opposed to the use of IOC.

Our final critique of the Quito approach is the use of the WOO to assess the generated test suites. After observing the data presented in Figure I, we questioned whether or not the WOO was contributing data that the OPO could account for on its own. Since the OPO identifies likely fails, definite fails identified by the WOO would fall under this category and be accounted for by the OPO had it run first. In the cases where a program passed the WOO, the OPO was less effective in killing mutants. However, in cases where a program did fail the WOO, the OPO was effective and had consistently higher mutation scores than when a program passed the WOO; this is evident in the data for the Swap and RCR programs. Despite having a higher average number of tests, the consistently higher mutation scores of the OPO show us that it offers greater effectiveness at testing quantum programs.

	Probability							
State	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
Ideal	0.495	-	-	-	-	-	-	0.505
Noisy	0.476	0.013	0.007	0.016	0.008	0.019	0.020	0.443

Fig. 1. The ideal and noisy outputs of a three-qubit entanglement program [6]

In addition to being unnecessary, we have found that the WOO is also an unrealistic test oracle for real-world quantum applications. In software, noise refers to unwanted discrepancies that impact signals and their processing. Noise can be the result of several factors including hardware details, magnetic fields, and radiation. While noise affects classical computer operations, quantum computers are especially susceptible due to their immature hardware and the unique characteristics of qubits. More specifically, qubit interactions can be disturbed by environmental factors and by other qubits in the system; these types of disturbances are referred to as decoherence and crosstalk noise, respectively. When affected by noise, qubit states are at risk of being altered or lost, which can result in a quantum program returning an incorrect output. In their report, the authors of "Noise-Aware Quantum Software Testing" show that noise has very real and significant effects in the output of even basic quantum programs. Figure I displays the probabilities of states of a three-qubit entanglement program run on both an ideal and noisy simulator. We can see in the table that the noisy simulator facilitates probabilities of qubit states whose probabilities are non-existent in their ideal, noise-free, simulator. In fact, the noisy simulator caused an 8.3 percent chance of the three-qubit entanglement process

to be in a state that had a zero percent chance of occurring in an ideal simulator.

Considering this, we do not agree with the decisions of the authors of our chosen visionary paper to overlook noise and use the WOO as the first assessment of the generated test suites. We believe that the risk of noise causing an incorrect output is too significant to justify the use of the WOO if the OPO will always be used after a test suite fails the WOO assessment. In a real-world application of the Quito quantum testing approach, noise will likely cause most test suites to fail the WOO, especially if the test suites are large in size. Thus, we think that the WOO will be unnecessary and unhelpful in future implementations of Quito. Furthermore, we urge the researchers of Quito to consider the effects that noise has on quantum programs and address this in the future. There is evidence that machine learning can be used to alleviate the effect that noise has on the test results of quantum programs. Specifically, the approach, QOIN, reduced the noise effect on quantum test outputs by more than 80 percent on a noisy backend. We believe that if Quito can implement a strategy similar to QOIN, their mutation scores will be greater, indicating a better testing methodology. [6]

IV. Conclusions

After reviewing the results of Quito testing and comparing it to contemporary work, we maintain our position that input coverage and output coverage along with the Output Probability Oracle specified by Quito compose an effective test suite for assessing quantum programs. We were able to compare results as well as the analyses presented on Quito, QuanFuzz, and QSharpTest to observe advantages in the usage of quantum gates for mutation and the comprehensive, efficient coverage offered by input and output criteria. We also analyzed the role of input-output coverage and the Wrong Output Oracle and provided evidence that these components of Quito are not cost effective. Specifically, we believe that the impact that noise has on quantum programs is a severe deterrent to the use of the Wrong Output Oracle.

We want to acknowledge that not only is the applicability of Quito to higher-qubit programs unknown, but several additional phenomena and discoveries of quantum computing have yet to be explored by testing approaches. In particular, we believe that quantum entanglement and interference of qubits can be harnessed to improve quantum testing in future development.

1) Entanglement: The principle of entanglement is a phenomenon in quantum mechanics where the state of one entity determines another. In quantum computing, qubits can be entangled with each other. This can simplify testing, as the amount of tests required to cover unique configurations of two entangled qubits is reduced if the value of one qubit determines the value of another. In future research, we would expect advances in efficiency from identifying and skipping tests that are made redundant by entanglement. [7]

2) Interference: A behavior of waves shared in the quantum world is constructive/destructive interference. Qubits, similarly to waves, can interfere to amplify some outcomes while destroying others, and this can be exploited to find a solution to a problem. Grover's algorithm is a search technique that implements this idea by representing locations as quantum states, then amplifying states based on their likelihood of being the target of the search. QuanFuzz accounts for interference by running multiple passes through a mutation matrix before generating a test path. We think Quito can improve in coverage effectiveness by relying on interference to identify important test paths, by using QuanFuzz's matrix mutant generator or representing mutants as states like in Grover's algorithm. [8]

In summary, while we have noted properties of quantum mechanics that could be used to improve performance of the Quito test suite, we recognize that quantum computing is a rapidly advancing field and the testing approach that Quito implements is still more efficient or more effective at finding faults than its competitors. Finally, we recognize the advantages of using statistical analysis when determining the mutant coverage of a test suite, especially when considering the presence of quantum noise in real applications of quantum computing. These factors strongly support our claim that Quito's input coverage and output coverage are effective criteria when implemented exclusively with the Output Probability Oracle to generate a quantum test suite.

References

[1] P. Rouchon, "A tutorial introduction to quantum stochastic master equations based on the qubit/photon system," 8 2022.

[2] M. Moeini, M. Akbari, M. Mirsadeghi, H. R. Naeij, N. Haghkish, A. Hayeri, and M. Malekian, "Quantum random number generator based on led," 2023.

[3] S. Ali, P. Arcaini, X. Wang, and T. Yue, "Assessing the effectiveness of input and output coverage criteria for testing quantum programs," in *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, 2021, pp. 13–23.

[4] J. Wang, M. Gao, Y. Jiang, J. Lou, Y. Gao, D. Zhang, and J. Sun, "Quanfuzz: Fuzz testing of quantum program," *CoRR*, vol. abs/1810.10310, 2018. [Online]. Available: http://arxiv.org/abs/1810.10310

[5] S. Honarvar, M. R. Mousavi, and R. Nagarajan, "Property-based testing of quantum programs in q#," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, ser. ICSEW'20. New York, NY, USA: Association for Computing Machinery, 2020, p. 430–435. [Online]. Available: https://doi.org/10.1145/3387940.3391459

[6] A. Muqeet, T. Yue, S. Ali, and P. Arcaini, "Noise-aware quantum software testing," 2023. [Online]. Available: https://arxiv.org/abs/2306.16992

[7] G. Burkard, T. D. Ladd, A. Pan, J. M. Nichol, and J. R. Petta, "Semiconductor spin qubits," *Reviews of Modern Physics*, vol. 95, no. 2, Jun. 2023. [Online]. Available: http://dx.doi.org/10.1103/RevModPhys.95.025003

[8] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, ser. STOC '96. New York, NY, USA: Association for Computing Machinery, 1996, p. 212–219. [Online]. Available: https://doi.org/10.1145/237814.237866