Institute of Architecture of Application Systems

About a Criterion of Successfully Executing a Circuit in the NISQ era: What $wd \ll 1/\epsilon_{\text{eff}}$ Really Means

Marie Salm, Johanna Barzen,
Frank Leymann, Benjamin Weder

Institute of Architecture of Application Systems,
University of Stuttgart, Germany
{salm, barzen, leymann, weder}@iaas.uni-stuttgart.de

Marie Salm, Johanna Barzen, Frank Leymann, and Benjamin Weder. 2020. About a Criterion of Successfully Executing a Circuit in the NISQ Era: What wd $\ll 1/\epsilon_{\text{eff}}$ Really Means. In Proceedings of the 1st ACM SIGSOFT International Workshop on Architectures and Paradigms for Engineering Quantum Software (APEQS 2020). Association for Computing Machinery, New York, NY, USA, 10–13.
DOI: https://doi.org/10.1145/3412451.3428498

BibTeX

```
@inproceedings{Salm2020_CriterionExecutingCircuit,
  author    = {Salm, Marie and Barzen, Johanna and Leymann, Frank
               and Weder, Benjamin},
  title     = {{About a Criterion of Successfully Executing a
                Circuit in the NISQ Era: What $wd \ll
               1/\epsilon_\text{eff}$ Really Means}},
  booktitle = {Proceedings of the 1st ACM SIGSOFT International
                Workshop on Architectures and Paradigms for
                Engineering Quantum Software (APEQS)},
  publisher = {ACM},
  year      = 2020,
  month     = nov,
  pages     = {10--13},
  doi       = {10.1145/3412451.3428498}
}
```


Universität Stuttgart
Germany

About a Criterion of Successfully Executing a Circuit in the NISQ Era: What $wd \ll 1/\epsilon_{\text{eff}}$ Really Means

Marie Salm
salm@iaas.uni-stuttgart.de
Institute of Architecture of Application Systems, University of Stuttgart, Germany

Johanna Barzen
barzen@iaas.uni-stuttgart.de
Institute of Architecture of Application Systems, University of Stuttgart, Germany

Frank Leymann
leymann@iaas.uni-stuttgart.de
Institute of Architecture of Application Systems, University of Stuttgart, Germany

Benjamin Weder
weder@iaas.uni-stuttgart.de
Institute of Architecture of Application Systems, University of Stuttgart, Germany

ABSTRACT

To evaluate classical software, a huge variety of software metrics exists. Similar metrics of quantum algorithms especially in context of near-term quantum computers are only rudimentary investigated. Hereby, metrics are particularly important to be able to estimate what is already possible with current quantum computers. Thus, in this paper, we discuss existing quantum performance metrics and focus on a metric that determines whether a quantum circuit is successfully executable on a given gate-based quantum computer. Thereby, we give an overview of various factors that can affect an execution and present our plan to concretize and apply the metric.

CCS CONCEPTS

• **Computer systems organization → Quantum computing**; • **General and reference → Metrics**; **Surveys and overviews**.

KEYWORDS

Quantum Computing, Quantum Circuits, Metrics, NISQ, Error Rates

ACM Reference Format:

Marie Salm, Johanna Barzen, Frank Leymann, and Benjamin Weder. 2020. About a Criterion of Successfully Executing a Circuit in the NISQ Era: What $wd \ll 1/\epsilon_{\text{eff}}$ Really Means. In *Proceedings of the 1st ACM SIGSOFT International Workshop on Architectures and Paradigms for Engineering Quantum Software (APEQS '20), November 13, 2020, Virtual, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3412451.3428498

1 INTRODUCTION

To quantify and finally compare the capabilities of software to be executed on classical computers, software metrics have been established and are tremendously important [15]. Today's quantum computers are limited by high error rates and only small numbers of qubits [21]. Such *Noisy Intermediate-Scale Quantum (NISQ)* computers consequently offer only short stable execution times which considerably limit the number of sequential executable gates, i.e. the *depth*, of concrete quantum circuits (Note that we are dealing with the gate-based model of quantum computing in this paper).

With these limitations, corresponding metrics are also extremely important to be able to estimate what is already possible on a given quantum computer. However, it is not possible to simply transfer the metrics of classical software to quantum algorithms or circuits, as quantum computers are fundamentally different in their design [22]. In addition, noise and resulting errors on today's quantum computers are hard to characterize in terms of their impact on quantum circuits. However, different quantum metrics to determine errors and the fidelity of individual gates already exist [12, 19, 31]. There are also quantum metrics that determine the overall performance of a given quantum computer, like *Total Quantum Factor (TQF)* [25] and *Quantum Volume* (V_Q) [2].

Our work focuses on another well-known quantum metric that estimates if a given arbitrary quantum circuit is successfully executable on a given quantum computer: $wd \ll \frac{1}{\epsilon_{\text{eff}}}$ [2, 7, 13, 18, 21]. Whereby the size wd of a circuit is limited depending on the effective error rate ϵ_{eff} of the quantum computer. Thereby, two questions arise: *How is the effective error rate composed?* Furthermore, the circuit size should be "much less" than the multiplicative inverse of the effective error rate according to the formula, which is not a precise threshold: *How to sharpen the equation from $\ll$ to $<$?* Therefore, we give an overview of the various factors contributing to the effective error rate, propose a refinement of the quantum metric, and present our vision on how to measure the concrete values.

2 CLASSICAL SOFTWARE METRICS

A whole plethora of metrics has been developed to measure the "suitability" of classical software [15]. "Suitability" covers aspects such as maintainability, modularization etc. A straightforward software metric that measures the complexity of a software application on the basis of its size is Lines Of Code (LOC) [15]. Another set of software metrics are the Halstead Complexity Metrics (HCM) [9] that describe the logical size of a software application [32]. First, the number of operators and operands is determined in the code and then parameters such as difficulty, programming effort, programming time, and software vocabulary are calculated. Another

APEQS '20, November 13, 2020, Virtual, USA

ACM ISBN 978-1-4503-8100-0/20/11…$15.00
https://doi.org/10.1145/3412451.3428498

well-known complexity measure is the McCabe Cyclomatic Complexity Metric (CCM) [16]. It examines the control flow graph of a software application and determines the number of linear independent paths through this graph [32]. Also metrics exist that especially focus on object-oriented software, such as Coupling Between Object classes (CBO) [5], Number Of Methods (NOM) [10], and Lack of Cohesion of Methods (LCOM) [5, 15]. As testing of software is an important concept to detect errors, software metrics are developed to measure the test coverage of the implemented code [4, 11]. Thereby, e.g., the coverage of code blocks, control flow, and the data flow can be considered for measurement [11]. The Maintainability Index (MI) is a software metric that estimates, amongst others, the maintainability of a software application [30]. Hereby, several software metrics are combined in MI, such as the already mentioned HCM, CCM, LOC, and the number of comments in the code. Comments are provided for a better understanding of the code, but must also be maintained for correctness and actuality. However, multiple variations of MI exist [30].

3 QUANTUM METRICS

In contrast to classical metrics, the number of metrics that measure the "suitability" of quantum circuits is small (Sec. 4). The current focus is on metrics that measure "the power" of quantum computers themselves. In the following, we discuss two existing quantum metrics that both measure the performance of a given quantum computer depending on the size of successfully executable quantum circuits. Thereby, we address their considered factors and discuss their shortcomings.

3.1 Total Quantum Factor

A quantum metric to measure and compare the performance of quantum computers is the *Total Quantum Factor (TQF)* presented by Sete et al. [25]:

$$TQF := \frac{T_1}{t_g} \cdot n_q \qquad (1)$$

T_1 is defined as the average coherence time of the implemented qubits, t_g is the maximum gate time of the implemented gate set, and n_q is the available number of qubits [25], whereas n_q restricts the circuit width, T_1/t_g estimates the maximum depth. Thus, TQF indicates the maximum circuit size the quantum computer can execute before the outcome could not be clearly assigned to the correct result. However, a single gate in the set with a long execution time worsens the total TQF of the quantum computer, even if it is rarely applied and the others only require short gate times. In addition, beneath decoherence, no other factors, such as gate and measurement errors, are considered that further influence the result.

3.2 Quantum Volume

Another quantum metric that is currently used by many quantum computer vendors as a measure of performance is *Quantum Volume (V_Q)* [2, 18]:

$$V_Q = \max_{n' \leq n} \min[n', \frac{1}{n' \epsilon_{\text{eff}}(n')}]^2 \qquad (2)$$

n' is defined as the circuit width, whereas n is the number of qubits of the quantum computer and forms the upper limit [18]. $\epsilon_{\text{eff}}(n')$ is the *effective error rate*. It is the average error rate per two-qubit gate determined by executing many circuits of the same depth of 1 on a given quantum computer. Thereby, it also considers several hardware characteristics, e.g. qubit connectivity, parallelism of operations, and implemented gate set. The maximum depth d of a faultless circuit can be estimated by $d = 1/n'\epsilon_{\text{eff}}(n')$. Eq. 2 squares the minimum of the two values n' and d with choosing n' such that V_Q is maximized [18]. In comparison to TQF, V_Q also determines the maximum circuit size a quantum computer is capable of, but it includes further factors, summarized as effective error rate [2].

Applied to real quantum computers, a simplified formula for V_Q is used: $V_Q = 2^{\min(d,n)}$ [7, 20], with depth d and width n. Cross et al. generate squared random circuit models, such that $d = n$, and check via a success metric called *heavy output probability* if more than $2/3$ of the executions are successful [7]. Then, d and n of the maximum permitted circuit model are inserted as exponents and V_Q is determined. Thereby, a different V_Q value results compared to Eq. 2. For example, if $d = n = 6$ is the size of a permitted circuit model $V_Q = 6^2 = 36$ results, where the ideal case without errors is assumed. With the simplified formula the value is $V_Q = 2^6 = 64$. Hence, either only one of the formulas should be used in practice, or quantum computer vendors should explicitly declare which formula was used to measure V_Q. However, Blume-Kohout and Young state that most of the current circuits of quantum algorithms are not of squared shapes [3]. Therefore, it is important to also consider non-squared circuits to determine more precisely the capabilities of a quantum computer, as one may have a small number of qubits but supports good coherence times and error rates whereas another is the exact opposite.

4 SIMPLE METRIC - COMPLEX FACTORS

The quantum metrics presented in Sec. 3 quantify the performance of quantum computers by a single number. However, to get an estimation if an arbitrary given quantum circuit is successfully executable on a given quantum computer or not, we are focusing on an additional quantum metric in this section.

The metric is described by a small formula which can be derived from the work on Quantum Volume [2, 7, 18] and is discussed in [13, 21] as a "rule of thumb":

$$wd \ll \frac{1}{\epsilon_{\text{eff}}} \qquad (3)$$

Thereby, w describes the *width* and d the *depth* of the circuit. ϵ_{eff} is the effective error rate of the quantum computer, as explained in Sec. 3 [13, 18]. Eq. 3 limits the size of a circuit depending on the effective error rate of the quantum computer. If $wd \approx 1/\epsilon_{\text{eff}}$, Moll et al. argue that the execution is highly probable to fail [18]. Therefore, the closer wd gets to the value of $1/\epsilon_{\text{eff}}$, the less accurate the result will be due to noise and resulting errors [13].

The simplicity of Eq. 3 has its pitfalls, e.g.: *How is the effective error rate composed?* Bishop et al. and Moll et al. argue that the effective error rate depends on various factors, e.g. connectivity, number of qubits, available gate set, parallelism, gate error rates, mapping algorithm, and quantum system complexity [2, 18]. In the following, we give an overview of the individual factors and their effects to show that they are highly interdependent and, currently, difficult to characterize individually. This hardens predicting and modelling error behaviour [22, 29].

Gate Errors. Implementations of quantum gates are erroneous and, therefore, can lead to incorrect qubit states when applied [29]. With each additional operation the deviation from the correct result increases. In particular, two-qubit gates, such as CNOT, have a much higher probability to be erroneous during their execution than single-qubit gates [29]. To evaluate the error rate of a gate, different metrics, such as average gate fidelity [19] and diamond distance [12] exist. But results of experiments show that these metrics do not represent the behavior when gates are applied several times, like in a circuit of a quantum algorithm [31].

Measurement Errors. Errors caused by measurements are not directly mentioned by [2, 18] but have a certain impact on the overall error and on the precision of the circuit result [28]. In comparison to quantum gates, a measurement operation introduces significant delays in which the qubit further decoheres [13]. Therefore, measurements have the highest operational error rate [28].

Connectivity. Qubits of a quantum computer are often not completely connected as the realization of a graph with a two-digit number of qubits is already hard to realize [29]. Additionally, high connectivity may lead to unintended interactions between nearby qubits, also known as *crosstalk* [17, 22]. The lack of connectivity implies that additional SWAP gates are needed to execute a two-qubit operation on two not directly connected qubits. This may also cause an increase of the required number of qubits for the circuit as additional qubits may be required for the state exchange, as shown by [13]. Additional gates mean additional errors and an increase of the depth of the circuit that must be taken into account [13].

Gate Set. If the gate set used in the circuit is not physically supported by the quantum computer the missing gates have to be replaced by a subroutine of supported gates [14]. This can result in a significant increase in the number of gates and depth of the circuit which in turn leads to additional gate error rates [13].

Qubits. As the state of a qubit decoheres after a certain amount of time due to unintended interaction with the environment, operations have to be applied before the state is too erroneous [13, 22]. Therefore, to get a meaningful outcome the depth or respectively the execution time of the circuit should be within the coherence time. With the combination of decoherence and crosstalk, the qubit state can further be disrupted [22]. Beneath, e.g., depolarization, amplitude, and phase damping, there is the probability that qubits accidentally leave the defined computational state space, also known as *leakage*, as they can contain further states depending on their realization [1, 8]. A qubit *loss* is possible when a qubit disappears [8].

Complexity of the System. Moll et al. argue that ϵ_{eff} is influenced by the "system complexity" including, e.g., the number of hardware components, which increases with the number of qubits [18]. However, the term is kept very abstract and should probably describe the different constellations of noise and resulting error rates that can occur on quantum computers. For example, with more qubits on the hardware, general crosstalk may increase. Thereby, crosstalk can occur when qubits are idle or operations are applied [24]. It can even occur during state preparation, caused by the imperfect controls and measurement processes. However, a concrete characterization of the composition of crosstalk does not yet exist [3, 24].

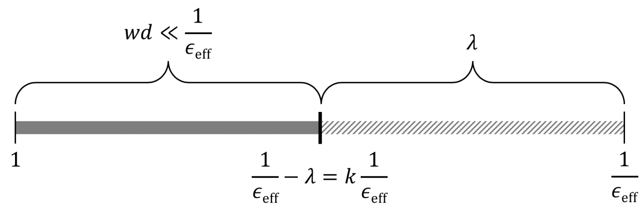

Figure 1: Introduction of a concrete threshold for $wd \ll \frac{1}{\epsilon_{\text{eff}}}$.

Parallelism. By parallelizing the execution of mutually independent operations on different qubits as far as possible, the depth of the circuit and, thus, the execution time can be considerably reduced [13]. As a consequence, errors caused by decoherence will be reduced.

Mapping Algorithm. How the circuit is mapped to the physical realization of the quantum computer depends on the mapping algorithm of the used compiler and has a strong influence on the effective error rate ϵ_{eff} [2]. Thereby, the compiler determines how the *logical* qubits of the circuit are initially mapped to the *physical* qubits on the quantum computer [13]. It is of advantage to place logical qubits that often share a two-qubit gate nearby on the connectivity graph such that the number of SWAP gates is minimal [29]. This process is known to be NP-hard [6, 26]. However, the connections between qubits and the qubits itself have a changing error rate which must be considered for finding the shortest path in the weighted connectivity graph [13, 29].

After the subsequent gate mapping procedure, the circuit size has reached its peak [13]. Therefore, many of the already existing compilers use optimization algorithms to reduce the resulting depth [27]. Thereby, the gates are parallelized as far as possible, redundancy in gate sequences are removed, and gates with high error rates are avoided. Future compilers may additionally consider general errors like crosstalk to further improve the precision of the result of a quantum circuit [17].

5 SHARPENING THE SIMPLE METRIC

A very simple metric for the "suitability" of a quantum circuit itself is given by Eq. 3: The size of the circuit should be much less than the multiplicative inverse of the effective error rate. But "much less" is not precise, i.e. not helpful at all. It is hard to exactly determine when the limit for a successful execution on a given quantum computer is reached. Therefore, in this section we investigate the question: *How to sharpen Eq. 3 from $\ll$ to $<$?*

Eq. 3 implies, that there exists an interval $]1/\epsilon_{\text{eff}} - \lambda, 1/\epsilon_{\text{eff}}[$ with some, yet unknown, λ; if the circuit size wd lies in this interval the noise and resulting errors are too high such that the result would be too imprecise, as shown with the shaded part in Fig. 1. Therefore, multiplying $1/\epsilon_{\text{eff}}$ with a factor k results in a threshold that determines the onset of the above interval (Fig. 1): $k \cdot 1/\epsilon_{\text{eff}} = 1/\epsilon_{\text{eff}} - \lambda$. With this in mind, the following refined and sharpened equation should hold:

$$wd < k \frac{1}{\epsilon_{\text{eff}}} \tag{4}$$

To determine concrete values for k, λ, and ϵ_{eff}, available quantum computers have to be benchmarked. Thereby, a wide spectrum of circuit classes, e.g. random circuits, periodic circuits, and current quantum algorithm circuits, as presented in [3], should be considered. Required quantum computing properties, like connectivity graph, gate set, and coherence times of qubits, should be extracted from collected provenance data [13]. Additionally, the depth and width of already compiled circuits in Eq. 4 should be inserted, because during compilation the resulting depth and width can vary greatly. Cross et al. consider sizes of non-compiled circuits, as they want a general measure to compare quantum computers, while the sharpened metric focuses on a more accurate feasibility estimate for an arbitrary circuit [7].

The resulting values and metadata can then in turn be saved as provenance data and further be reused [14]. The feasibility metric itself can be used to refine the automated selection of quantum circuits and suitable quantum computers [23].

6 CONCLUSION & FUTURE WORK

In this paper, we gave an overview of metrics for classical software and presented performance metrics TQF and Quantum Volume for quantum computers. Furthermore, we especially focused our work on the metric $wd \ll \frac{1}{\epsilon_{\text{eff}}}$ and discussed in more detail how the quantum system, noise, and errors affect ϵ_{eff} and how this equation can be refined for precise estimation, i.e. a sharp upper limit. Finally, we presented our vision for the determination of the metric variables via benchmarking and provenance. As a next step, we want to realize our vision by benchmarking given quantum computers and evaluating the validity of the refined quantum metric in Eq. 4 as discussed in Sec. 5.

ACKNOWLEDGMENTS

This work was partially funded by the BMWi project *PlanQK* (01MK20005N) and the DFG's Excellence Initiative project *SimTech* (EXC 2075 - 390740016).

REFERENCES

[1] Dorit Aharonov and Michael Ben-Or. 2008. Fault-Tolerant Quantum Computation with Constant Error Rate. *SIAM J. Comput.* 38, 4 (2008), 1207–1282. https://doi.org/10.1137/S0097539799359385
[2] L. Bishop, S. Bravyi, A. Cross, J. Gambetta, J. Smolin, and March. 2017. Quantum Volume.
[3] Robin Blume-Kohout and Kevin C. Young. 2019. A volumetric framework for quantum computer benchmarks. arXiv:quant-ph/1904.05546
[4] Xia Cai and Michael R. Lyu. 2005. The Effect of Code Coverage on Fault Detection under Different Testing Profiles. In *Proceedings of the 1st International Workshop on Advances in Model-Based Testing (A-MOST '05)*. Association for Computing Machinery, 1–7. https://doi.org/10.1145/1083274.1083288
[5] Shyam R. Chidamber and Chris F. Kemerer. 1994. A metrics suite for object oriented design. *IEEE Transactions on Software Engineering* 20, 6 (1994), 476–493. https://doi.org/10.1109/32.295895
[6] Alexander Cowtan, Silas Dilkes, Ross Duncan, Alexandre Krajenbrink, Will Simmons, et al. 2019. On the qubit routing problem. arXiv:quant-ph/1902.08091
[7] Andrew W. Cross, Lev S. Bishop, Sarah Sheldon, Paul D. Nation, and Jay M. Gambetta. 2019. Validating quantum computers using randomized model circuits. *Phys. Rev. A* 100 (2019), 032328. Issue 3. https://doi.org/10.1103/PhysRevA.100.032328
[8] Austin G. Fowler. 2013. Coping with qubit leakage in topological codes. *Phys. Rev. A* 88 (2013), 042308. Issue 4. https://doi.org/10.1103/PhysRevA.88.042308
[9] Maurice H. Halstead. 1977. Elements of software science.
[10] Brian Henderson-Sellers. 1995. *Object-Oriented Metrics: Measures of Complexity.* Prentice-Hall, Inc.
[11] Joseph R. Horgan, Saul London, and Michael R. Lyu. 1994. Achieving software quality with testing coverage measures. *Computer* 27, 9 (1994), 60–69. https://doi.org/10.1109/2.312032
[12] A Yu Kitaev. 1997. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys* 52, 6 (1997), 1191–1249. https://doi.org/10.1070/rm1997v052n06abeh002155
[13] Frank Leymann and Johanna Barzen. 2020. The bitter truth about gate-based quantum algorithms in the NISQ era. *Quantum Science and Technology* 5, 4 (2020), 044007. https://doi.org/10.1088/2058-9565/abae7d
[14] Frank Leymann, Johanna Barzen, Michael Falkenthal, Daniel Vietz, Benjamin Weder, et al. 2020. Quantum in the Cloud: Application Potentials and Research Opportunities. In *Proceedings of the 10th International Conference on Cloud Computing and Service Science (CLOSER 2020)*. SciTePress, 9–24. https://doi.org/10.5220/0009819800090024
[15] Rüdiger Lincke, Jonas Lundberg, and Welf Löwe. 2008. Comparing Software Metrics Tools. In *Proceedings of the 2008 International Symposium on Software Testing and Analysis (ISSTA '08)*. Association for Computing Machinery, 131–142. https://doi.org/10.1145/1390630.1390648
[16] Thomas J. McCabe. 1976. A Complexity Measure. *IEEE Transactions on Software Engineering* SE-2, 4 (1976), 308–320. https://doi.org/10.1109/TSE.1976.233837
[17] Daniel Mills, Seyon Sivarajah, Travis L. Scholten, and Ross Duncan. 2020. Application-Motivated, Holistic Benchmarking of a Full Quantum Computing Stack. arXiv:quant-ph/2006.01273
[18] Nikolaj Moll, Panagiotis Barkoutsos, Lev S Bishop, Jerry M Chow, Andrew Cross, et al. 2018. Quantum optimization using variational algorithms on near-term quantum devices. *Quantum Science and Technology* 3, 3 (2018), 030503. https://doi.org/10.1088/2058-9565/aab822
[19] Michael A Nielsen. 2002. A simple formula for the average gate fidelity of a quantum dynamical operation. *Physics Letters A* 303, 4 (2002), 249 – 252. https://doi.org/10.1016/S0375-9601(02)01272-0
[20] Juan M. Pino, Jennifer M. Dreiling, Caroline Figgatt, John P. Gaebler, Steven A. Moses, et al. 2020. Demonstration of the QCCD trapped-ion quantum computer architecture. arXiv:quant-ph/2003.01293
[21] John Preskill. 2018. Quantum Computing in the NISQ era and beyond. *Quantum* 2 (2018), 79. https://doi.org/10.22331/q-2018-08-06-79
[22] Salonik Resch and Ulya R. Karpuzcu. 2019. Benchmarking Quantum Computers and the Impact of Quantum Noise. arXiv:quant-ph/1912.00546
[23] Marie Salm, Johanna Barzen, Uwe Breitenbücher, Frank Leymann, Benjamin Weder, et al. 2020. The NISQ Analyzer: Automating the Selection of Quantum Computers for Quantum Algorithms. *Communications in Computer and Information Science (CCIS)* (2020). to appear.
[24] Mohan Sarovar, Timothy Proctor, Kenneth Rudinger, Kevin Young, Erik Nielsen, et al. 2019. Detecting crosstalk errors in quantum information processors. https://doi.org/10.22331/q-2020-09-11-321 arXiv:quant-ph/1908.09855
[25] Eyob A. Sete, William J. Zeng, and Chad T. Rigetti. 2016. A functional architecture for scalable quantum computing. In *2016 IEEE International Conference on Rebooting Computing (ICRC)*. 1–6. https://doi.org/10.1109/ICRC.2016.7738703
[26] Marcos Yukio Siraichi, Vinicius Fernandes dos Santos, Sylvain Collange, and Fernando Magno Quintão Pereira. 2018. Qubit Allocation. In *CGO 2018 - International Symposium on Code Generation and Optimization*. 1–12. https://doi.org/10.1145/3168822
[27] Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, et al. 2020. t|ket⟩: A retargetable compiler for NISQ devices. *Quantum Science and Technology* (2020). https://doi.org/10.1088/2058-9565/ab8e92
[28] Swamit S. Tannu and Moinuddin K. Qureshi. 2019. Mitigating Measurement Errors in Quantum Computers by Exploiting State-Dependent Bias. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '52)*. Association for Computing Machinery, 279–290. https://doi.org/10.1145/3352460.3358265
[29] Swamit S. Tannu and Moinuddin K. Qureshi. 2019. Not All Qubits Are Created Equal: A Case for Variability-Aware Policies for NISQ-Era Quantum Computers. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '19)*. Association for Computing Machinery, 987–999. https://doi.org/10.1145/3297858.3304007
[30] Kurt Dean Welker. 2001. Software Maintainability Index Revisited. *CrossTalk - The Journal of Defense Software Engineering* (2001).
[31] Dennis Willsch, Madita Nocon, Fengping Jin, Hans De Raedt, and Kristel Michielsen. 2017. Gate-error analysis in simulations of quantum computers with transmon qubits. *Phys. Rev. A* 96 (2017), 062302. Issue 6. https://doi.org/10.1103/PhysRevA.96.062302
[32] S. Yu and S. Zhou. 2010. A survey on metric of software complexity. In *2010 2nd IEEE International Conference on Information Management and Engineering*. 352–356.