Making Quantum Computing Open: Lessons from Open-Source Projects

Ruslan Shaydulin, Caleb Thomas, and Paige Rodeghero
School of Computing
Clemson University
Clemson, SC, USA
Email: {rshaydu, caleb8, prodegh}@g.clemson.edu

Abstract—Quantum computing (QC) is an emerging computing paradigm with potential to revolutionize the field of computing. QC is a field that is quickly developing globally and has high barriers of entry. In this paper we explore both successful contributors to the field as well as wider QC community with the goal of understanding the backgrounds and training that helped them succeed. We gather data on 148 contributors to open-source quantum computing projects hosted on GitHub and survey 46 members of QC community. Our findings show that QC practitioners and enthusiasts have diverse backgrounds, with most of them having a PhD and trained in physics or computer science. We observe a lack of educational resources on quantum computing. Our goal for these findings is to start a conversation about how best to prepare the next generation of QC researchers and practitioners.

Index Terms—Quantum computing, Software engineering, Software design, Computer science education, Physics education

I. INTRODUCTION

The recent years saw a sequence of rapid scientific and engineering advancements in the field of quantum computing (QC). In less than two decades QC evolved from being mostly theoretical to being able to solve practically interesting problems [1]–[4]. A number of private companies, universities and government labs worldwide successfully demonstrated a hardware implementation of QC [5]–[9], and a number of quantum computing projects have emerged, aiming to provide a platform for integration of quantum computation in scientific and business workflows.

Quantum computing has the potential to provide exponential speedups to many important problems [10], making possible to compute things that are unimaginable today. The most famous example of the potential capabilities of quantum computing is Shor's [11] algorithm. It describes a way to factor integers in polynomial time and has implications for RSA-based cybersecurity [10]. Many more algorithms have been developed, with applications to quantum chemistry [12], combinatorial optimization [3], [13] and machine learning [1].

The key problem in the field of QC software engineering is that to fully realize the promise of quantum computing the community needs scientists and developers with skills relevant to the evolving field. In these early days, researchers working in quantum computing need an understanding of both the unique logic of quantum computation, as well as

the computer science experience needed to integrate it into existing classical workflows. One example of an area where a diverse set of skills spanning multiple fields is required is development of quantum algorithms. Quantum algorithms like Shor's [11] are significantly different from their classical counterparts. Just understanding them requires knowledge of quantum mechanics and computational complexity theory, two disciplines that usually do not go together in a university curriculum. Development of new algorithms requires mastery of both, as well as a deep computer science background.

Our data shows that currently most QC practitioners deal with the problem of lack of educational resources by combining their formal training in fields like physics or computer science with reading papers and textbooks on their own and following tutorials online. This makes the field of QC hard to enter and hard to navigate for the newcomers. We address this problem by performing an analysis of the background and skills that practitioners find important to their work. To our knowledge this is the first attempt to explore quantum computing from the software engineering perspective.

In this paper, we present the data about the background and training of QC researchers and practitioners. The methodology that we follow consists of two parts, with possible overlap between the two. First, we scrape the data about 148 contributors to open-source QC projects from GitHub. Second, we survey 46 QC professionals to augment the data. We found that most QC specialists are trained in non-computing fields like physics, chemistry or mathematics. Our goals for this paper is to receive community feedback on this line of research which could include further investigation of factors contributing to success in QC field through surveying of practitioners, panels at conferences and gathering data on open-source contributions. We believe our findings can help develop educational resources targeted at preparing a new generation of QC researchers and practitioners.

To the best of our knowledge, this is the first exploration of the quantum computing field from the software engineering perspective, and the first attempt to rigorously explore the contributors to open-source quantum computing projects. This makes our approach novel and interesting to both software engineering and quantum computing communities.

II. PROBLEM

Today there are very few training programs in quantum computing and there is no consensus on the curriculum structure. This is supported by our data: out of 46 survey respondents, only one received formal training in quantum computation. This indicates a lack of understanding of what training is required for success in QC field. Without looking further into what is required to become a part of the QC field, it is difficult for aspiring QC researchers to know what they should do to become a part of this field, which will inevitably stunt the growth of the QC industry. There is a need for a deeper exploration of the challenges facing QC researchers and practitioners. This paper addresses this by exploring the contributors to open-source quantum computing projects, their backgrounds, and the challenges they face.

III. BACKGROUND AND RELATED WORK

Quantum computing is still a very young field, with software projects constantly pushing the boundary of human knowledge. As such, quantum computing software projects have to deal with all the problems that plague scientific software projects: unforeseen changes in the requirements, lack of software development expertise and limited budgets [14]. The cross-disciplinary nature of quantum computing adds to the complexity of the domain.

A. Quantum Computing

Unlike in classical computation, where the computation happens by manipulating bits, the fundamental computational unit in QC is qubit. A bit can have one of two states: 0 or 1. Similarly, qubit state is a unit vector in a two-dimensional complex vector space [10]. A qubit state can be encoded in a state of a quantum mechanical object, for example as polarization of a single photon [10].

The field of quantum computing is in a state of constant change, and is generally expected to continue changing in the foreseeable future. In the past few years multiple Near-term Intermediate-Scale Quantum (NISQ) hardware implementations have been developed [15] and demonstrated to provide a potential for quantum speedups [16]. Naturally, different implementations come with certain trade-offs. For example, trapped ion qubits are generally less noisy and offer better connectivity, whereas superconducting qubits offer faster gate clock speeds and more clear path to scalability [17]. This diversity of hardware introduces an additional degree of complexity for the development of QC algorithms and software, forcing algorithm developers to stay aware of the trade-offs presented by hardware.

A plethora of algorithms leveraging the power of quantum computation have been developed over the years. Shor's [11] and Grover's [18] algorithms are two most well-known examples of quantum algorithms for practical problems with theoretically proven speed-ups over classical state-of-the-art. However, the limitations of NISQ-era hardware make most of them impossible to run in the near-term. Near-term quantum computers are widely believed to be able to provide no more than a few hundreds of non error-corrected qubits. To address this challenge, a number of NISQ approaches have been proposed, most prominent of them Variational Quantum Eigensolver (VQE) [19] and Quantum Approximate Optimization Algorithm (QAOA) [13]. The limitations of near-term hardware make development of practical algorithms especially challenging.

B. Open-source Scientific Software Projects

There has been a push towards open source scientific software projects. Government labs, such as Sandia National Laboratories, create and use open source projects [20], [21]. This allows for easier collaboration with scientists and programmers both within industry and academia.

Many software engineering studies have been conducting by mining GitHub [22]–[26]. These projects look at contributors, their efforts, the number of bugs created or fixed, etc. One problem with open source projects is that it can be unclear who is contributing more than others [27]. Today, conferences such as "The Mining Software Repositories" (MSR) exist to better understand software repositories and the contributors.

IV. METHODOLOGY

In order to understand the structure of quantum computing community and the challenges facing the contributors, we focus on three open-source and open-development quantum computing projects from three companies working in quantum computing field: Qiskit (IBM), PyQuil/Grove (Rigetti) and Cirq (Google, not an official product).

Qiskit is open-source project developed by IBM. At the time of writing, it consists of two main parts: Qiskit Terra [28] provides the basic building blocks and Qiskit Aqua [29] provides a library of algorithms upon which applications can be built. Qiskit is in active development and new modules with new functionality has been integrated into it while this paper was in preparation.

PyQuil/Grove is an open-source QC framework developed by Rigetti. Rigetti has recently announced Quantum Cloud Services (QCS), a new closed-source project aimed at providing cloud access to quantum computers to researchers and developers. Rigetti QCS is currently in closed beta-testing. In this work we focus on PyQuil/Grove, previous open-source iteration of Rigetti quantum effort. Rigetti PyQuil [30] provides the low-level building blocks and Rigetti Grove [31] provide a library of algorithms implemented with PyQuil.

Cirq [32] is developed by researchers at Google. Cirq is not an official Google product. Cirq provides low-level tools for building programs to be run on noisy intermediate-scale quantum (NISQ) computers.

Our approach consists of two parts. First, we scraped the contribution data from the projects' GitHub repositories using PyDriller framework [33]. These data provide us with a quantitative understanding of who contributes to the quantum computing projects. QC field is still relatively small when compared with software development as a whole. Analyzing the data on between a hundred and two hundred QC

project contributors provides a valuable insight about the QC practitioners. Second, we augment the data from by sending out a questionnaire to quantum computing developers and enthusiasts through IBM Qiskit and Rigetti Slack channels, as well as to emails of contributors we scraped from GitHub.

The questionnaire extends the data harvested from GitHub by exploring the background, motivation and the challenges facing the practitioners. The questionnaire consists of three parts. First part includes questions on the participants background, education and professional experience (e.g. "What was your major in college?"). Second part explores participants' experience with different QC frameworks, what educational resources they find the most helpful and how confident they are in their CS and physics knowledge (e.g. "How adequate do you find your computer science (CS) background for your work in QC (e.g. coding skills, understanding of CS concepts etc)?"). Third part consists of open-ended questions, e.g. "What kind of CS training would have better prepared you for your work in QC? What CS class you wish you took but didnt?".

V. RESULTS

We collected data about 148 quantum computing researchers by scraping data on contributions from GitHub repositories of popular QC projects and received 46 responses to our survey. We can say that the first part of the data (GitHub data) give us an insight about the people who *already contribute* to quantum computing projects, whereas the second part of the data (survey data) gives us a broader picture of the community of people *interested* in quantum computing. We observe a significant difference between the two, indicating that QC companies need to engage the broader community in software development process to make these projects truly collaborative. The GitHub data also demonstrates how different companies approach the challenging task of building a quantum computing framework in different ways.

Through analysis of the GitHub data we get an idea of people who are already successful in the field. Concretely, we can measure the success by number of commits to the state-of-the-art open source projects. If success is defined like that, the data shows that most successful contributors have a PhD. 61% of contributors with more than 10 commits in repository's master branch have a PhD, with 81% of them having a PhD in physics (55%) or computer science (26%). This is unsurprising, since the projects in questions are fundamentally research projects, requiring precisely the skills a PhD program develops. GitHub data demonstrates how the company culture varies between the three big players: while in IBM and Rigetti projects 62% and 64% top contributors have a PhD, for Google's Cirq the number is only 50%, indicating that Google treats its quantum computing initiative differently. What unites all the projects we looked at is that they were all developed almost exclusively by people employed at that company, with the ratio of employees among top contributors being 100% for all three companies. This indicates a need for a broader collaboration, both between companies, as well as between companies and other research institutions.

In contrast to the GitHub data, which contains almost exclusively employees of the QC companies, out of 64 survey respondents only 30% indicated that they work for a quantum-computing-centered company. This shows that there is a broader interest in quantum computing, both in industry (35% of respondents) as well as in national laboratories (9%) and among PhD students (9%). Still, we observe a very high number of people with PhD among our respondents (28%), confirming our observations from GitHub data. Similarly to GitHub data, most respondents (70%) studied either physics (29%) or computer science (41%).

The predominance of people with PhDs (i.e. people with research backgrounds) is indicative of a young field, where the barrier to entry is still very high. There is a clear need to lower those barriers by developing educational materials that could help bring people into the field. The lack of educational materials is confirmed by multiple observations. First, our data shows almost no people with degree in Quantum Computation, Quantum Information or related fields (only one respondent indicated that they have a masters degree in Quantum Computing). Second, in ranking educational materials by their impact and helpfulness the respondents rate "reading papers on their own" (mean score of 2.98, less is better) and "reading textbooks on their own" (mean score of 3.04) as most helpful, followed by "following tutorials in software repositories" (mean score of 3.25). This shows that the effort software companies have put into developing introductory tutorials to accompany their frameworks is paying off and that the community finds them helpful. Surprisingly, Massive Open Online Courses (MOOCs) scored fairly low (second to last with mean score of 3.91), which in our opinion says more about their accessibility rather than availability (that is how easy their are to follow, rather then how easy their are to find). The only online MOOCs known to us are parts of MIT Quantum Computing Curriculum, require a strong mathematical background and teach mostly theoretical (and somewhat challenging) material.

The data we collected shows that training for quantum computing should combine physics, computer science (CS) and mathematics. We find that respondents trained in predominantly non-CS fields find their computer science skills lacking (skills like coding, software development etc), whereas respondents with predominantly CS training find their physics knowledge inadequate. Interestingly, the ratio of respondents who found their physics skills moderately or extremely inadequate (20% of respondents) is much higher that the ratio of respondents who said the same about their CS skills (9%). This indicates that the community finds the physics side harder, suggesting that the training should focus more on developing relevant physics knowledge. In an open-ended question, where the respondents were encouraged to discuss what training they wish they received, 17 respondents indicated that they didn't receive sufficient training in quantum computing and quantum information. Another commonly received suggestion was linear algebra and mathematical training in general.

VI. Discussion and Conclusion

In this paper we present the data on 148 contributors to quantum computing projects and 46 members of wider quantum computing community (with possible overlap between the two). Two parts of our dataset complement each other, providing a valuable insight into that factors contributing to success in the field of quantum computing.

Our finding that most respondents rate studying on their own as the most useful highlights the need for universities and other educational institutions to step in and offer training, be it in form of a MOOC available on the internet or an in-person degree program. These educational resources should adequately prepare QC practitioners, with more focus given to the most challenging parts of QC curriculum. Our findings have implications for designs of such offerings, indicating that the respondents find computer science and software engineering aspect of their work in QC the least challenging, while struggling more with underlying fundamental mathematical and physical concepts.

The majority of the respondents have less than two years of experience in QC (80%), with only 7% having worked in the field for more than 5 years. This is indicative of a young and booming field, with a lot of new entrants. The relative immaturity of the field as well as its interdisciplinarity make it very challenging to join. As a community, we need to develop more resources to bringing quantum computing to undergraduate students through classes, online courses and tutorials. But as the field is booming and hype is high, it's important to keep a cool head. As one of our respondents wisely noted, one should "be careful not to train people for something that will bust and not get them jobs."

We hope this project will have a positive impact by bringing software engineering community and the software engineering methods to quantum computing. Moreover, we believe that the data we collected will contribute to the discussion on how best to prepare the next generation of quantum computing specialists and researchers.

Note Added. – After the completion of this work, we became aware of a related work appearing in Ref. [34]. They provide an exhaustive review of open-source quantum computing projects. Unlike our work, they don't directly survey the contributors to these projects. Additionally, we review the educational and professional background of the contributors.

References

[1] R. Shaydulin, H. Ushijima-Mwesigwa, I. Safro, S. Mniszewski, and Y. Alexeev, "Network community detection on small quantum computers," *arXiv preprint arXiv:1810.12484*, 2018.

[2] ——, "Community detection across emerging quantum architectures," *arXiv preprint arXiv:1810.07765*, 2018.

[3] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm applied to a bounded occurrence constraint problem," *arXiv preprint arXiv:1412.6062*, 2014.

[4] H. Ushijima-Mwesigwa, C. F. Negre, and S. M. Mniszewski, "Graph partitioning using quantum annealing on the d-wave system," in *Proceedings of the Second International Workshop on Post Moores Era Supercomputing*. ACM, 2017, pp. 22–29.

[5] C. Ballance, T. Harty, N. Linke, M. Sepiol, and D. Lucas, "High-fidelity quantum logic gates using trapped-ion hyperfine qubits," *Physical review letters*, vol. 117, no. 6, p. 060504, 2016.

[6] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, B. Campbell *et al.*, "Superconducting quantum circuits at the surface code threshold for fault tolerance," *Nature*, vol. 508, no. 7497, p. 500, 2014.

[7] J. R. Petta, A. C. Johnson, J. M. Taylor, E. A. Laird, A. Yacoby, M. D. Lukin, C. M. Marcus, M. P. Hanson, and A. C. Gossard, "Coherent manipulation of coupled electron spins in semiconductor quantum dots," *Science*, vol. 309, no. 5744, pp. 2180–2184, 2005.

[8] A. Acín, I. Bloch, H. Buhrman, T. Calarco, C. Eichler, J. Eisert, D. Esteve, N. Gisin, S. J. Glaser, F. Jelezko *et al.*, "The quantum technologies roadmap: a european community view," *New Journal of Physics*, vol. 20, no. 8, p. 080201, 2018.

[9] M. Saffman, T. G. Walker, and K. Mølmer, "Quantum information with rydberg atoms," *Reviews of Modern Physics*, vol. 82, no. 3, p. 2313, 2010.

[10] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2002.

[11] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*. Ieee, 1994, pp. 124–134.

[12] J. Romero, R. Babbush, J. McClean, C. Hempel, P. Love, and A. Aspuru-Guzik, "Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz," *Quantum Science and Technology*, 2018.

[13] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," *arXiv preprint arXiv:1411.4028*, 2014.

[14] C. Letondal and U. Zdun, "Anticipating scientific software evolution as a combined technological and design approach," in *Second International Workshop on Unanticipated Software Evolution*, 2003.

[15] J. Preskill, "Quantum computing in the nisq era and beyond," *arXiv preprint arXiv:1801.00862*, 2018.

[16] V. Dunjko, Y. Ge, and J. I. Cirac, "Computational speedups using small quantum devices," *arXiv preprint arXiv:1807.08970*, 2018.

[17] N. M. Linke, D. Maslov, M. Roetteler, S. Debnath, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, "Experimental comparison of two quantum computing architectures," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3305–3310, 2017.

[18] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. ACM, 1996, pp. 212–219.

[19] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. Obrien, "A variational eigenvalue solver on a photonic quantum processor," *Nature communications*, vol. 5, p. 4213, 2014.

[20] "Software Policy," https://tinyurl.com/yaoyd8kl, accessed: 2019-02-03.

[21] "Federal Source Code Policy: Achieving Efficiency, Transparency, and Innovation through Reusable and Open Source Software," https://sourcecode.cio.gov/, accessed: 2019-02-03.

[22] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian, "An in-depth study of the promises and perils of mining github," *Empirical Software Engineering*, vol. 21, no. 5, pp. 2035–2071, 2016.

[23] V. Cosentino, J. Luis, and J. Cabot, "Findings from github: methods, datasets and limitations," in *Proceedings of the 13th International Conference on Mining Software Repositories*. ACM, 2016, pp. 137–141.

[24] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian, "The promises and perils of mining github," in *Proceedings of the 11th working conference on mining software repositories*. ACM, 2014, pp. 92–101.

[25] D. Pletea, B. Vasilescu, and A. Serebrenik, "Security and emotion: sentiment analysis of security discussions on github," in *Proceedings of the 11th working conference on mining software repositories*. ACM, 2014, pp. 348–351.

[26] B. Vasilescu, D. Posnett, B. Ray, M. G. van den Brand, A. Serebrenik, P. Devanbu, and V. Filkov, "Gender and tenure diversity in github teams," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2015, pp. 3789–3798.

[27] M. Goldman, G. Little, and R. C. Miller, "Real-time collaborative coding in a web ide," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '11.

New York, NY, USA: ACM, 2011, pp. 155–164. [Online]. Available: http://doi.acm.org/10.1145/2047196.2047215

[28] "Qiskit/qiskit-terra repository." [Online]. Available: https://github.com/Qiskit/qiskit-terra

[29] "Qiskit/aqua repository." [Online]. Available: https://github.com/Qiskit/aqua

[30] "rigetticomputing/pyquil repository." [Online]. Available: https://github.com/rigetticomputing/pyquil

[31] "rigetticomputing/grove repository." [Online]. Available: https://github.com/rigetticomputing/grove

[32] "quantumlib/Cirq repository." [Online]. Available: https://github.com/quantumlib/Cirq

[33] D. Spadini, M. Aniche, and A. Bacchelli, *PyDriller: Python Framework for Mining Software Repositories*, 2018.

[34] M. Fingerhuth, T. Babej, and P. Wittek, "Open source software in quantum computing," *PLOS ONE*, vol. 13, no. 12, pp. 1–28, 12 2018.