netidee

STIPENDIEN

Optimizing Hybrid Workflows for Cloud-Based Quantum

Zwischenbericht | Call 19 | Stipendium ID 7413

Inhalt

1 Einleitung

Seit der Entstehung der Grundidee des Quantencomputings in den 1980er Jahren hat sich dieser Bereich von einer Idee zu ersten Prototypen und schließlich zu einem aktiven und interdisziplinären Forschungsfeld entwickelt. Darüber hinaus sind zahlreiche Wirtschaftsbranchen von den Auswirkungen der potenziellen Fähigkeiten dieser Technologie betroffen. Mögliche Anwendungsbereiche sind die Optimierung von Lieferketten und Herstellungsprozessen, die Simulation molekularer Strukturen und physikalischer Modelle sowie die Verbesserung von KI-Modellen.

Auch wenn die heute verfügbaren Quantencomputer (im Folgenden QC) im Vergleich zu den ersten Prototypen erhebliche Fortschritte gemacht haben und dies durch die Demonstration von Quantum Computational Advantage[1] für bestimmte Probleme bereits unter Beweis stellen konnten, sind heutige Maschinen aufgrund fehlender Fehlerkorrektur, der fragilen Zustände, mit denen sie hantieren, und der aktuell begrenzten Anzahl nutzbarer Qubits - der kleinsten Informationseinheit in einem QC - leider noch ein großes Stück von einer praktischen Realisierung eines solchen Vorteils entfernt.

Trotz bestehender Herausforderungen schreitet die Entwicklung kontinuierlich voran, sodass QC zunehmend auch für Forschungs- und Geschäftsfelder außerhalb der Physik relevant werden. In den vergangenen Jahren wurden die ersten Produktionssysteme in HPC-Einrichtungen bereitgestellt. Zudem ermöglichen es diverse Cloudanbieter ihren Nutzern, cloudbasiert auf QC-Ressourcen zuzugreifen. In Anlehnung an die Servicemodelle des klassischen Cloud-Computings wurde dafür die Bezeichnung „Quantum Computing as a Service“ (QCaaS) eingeführt.

Die breite Verfügbarkeit von Quantencomputing-Systemen über die Cloud ermöglicht es Forschern verschiedener Fachrichtungen, in diesem Bereich zu arbeiten. Dadurch ist echte Quantenhardware nicht mehr nur Experimentalphysikern vorbehalten, sondern steht auch Informatikern, theoretischen Physikern sowie Vertretern verschiedener relevanter Geschäftsfelder zur Verfügung. Das QCaaS-Modell erlaubt somit Experimente mit Quantencomputern, das Testen von Quantenalgorithmen für bestimmte Geschäftsanwendungen und die Ausbildung erforderlicher Fachkräfte, ohne dass die Nutzer selbst in die notwendige Infrastruktur investieren müssen.

[1] Die Fähigkeit eines Quantencomputers, ein bestimmtes Problem zu lösen, das kein klassischer Computer in praktisch relevanter Zeit bewältigen kann, wird als Quantenvorteil - Quantum Computational Advantage - bezeichnet.

Hinsichtlich der Ausführung von Quantenprogrammen hat sich ein hybrides Ausführungsmodell etabliert. Dabei werden QC- und klassische Rechenressourcen kombiniert, wobei der QC oftmals nur einen (kleinen) Teil des gesamten Anwendungsablaufs ausführt. Klassische Systeme übernehmen hingegen Aufgaben wie Kompilierung, Zugriffsmanagement, klassische Optimierung und Nachbearbeitung der berechneten Rohergebnisse. So kann etwa die Berechnung der Grundzustandsenergie eines Moleküls diese in eine parametrisierte Kostenfunktion kodieren, die dann auf einem QC ausgewertet wird. Die Kompilierung dieser Kostenfunktion, das Deployment und die notwendige Optimierungsschleife erfolgen jedoch durch klassische Rechner.

Die Integration dieser Abläufe ist nach wie vor mit Herausforderungen verbunden, darunter Skalierbarkeitsprobleme, die sich über den gesamten QC-Stack erstrecken. In dieser Arbeit konzentrieren wir uns auf die Kompilierungsebene von Quantenprogrammen. Im Gegensatz zu klassischen Programmen, die einmal kompiliert und auf Servern gehostet werden, müssen Quantenprogramme für jede Ausführung neu kompiliert und bereitgestellt werden. Dies ist ein zeitaufwändiger Vorgang, dessen Aufwand mit zunehmender Komplexität des Programms zunimmt und die praktische Ausführungszeit von Quantenanwendungen erheblich verlängert. In dieser Arbeit möchten wir uns dieser Herausforderung annehmen und die Nutzung von Quanten-Ressourcen sowie die Bereitstellung von Quantenanwendungen in der Cloud zeit- und ressourceneffizienter gestalten.

Bei bestehenden Lösungen für das Problem der erhöhten Laufzeit werden die resultierenden kompilierten Schaltkreise oder Pulsbefehle[2] zur späteren Wiederverwendung gespeichert. Diese Ansätze können jedoch eine übermäßige Speicherung von Daten erfordern. Zudem können nach bestimmten Hardwarekalibrierungen Neukompilierungen auf Schaltkreis- und Pulsebene erforderlich sein. Anstatt die Quantenkompilierung auf Schaltkreis- oder Pulsebene anzugehen, entwickeln wir neuartige Methoden, um den Kompilierungsaufwand auf Ebene der einzelnen *Compiler-Passes* - also der individuellen Schritte, die den Kompilierungsprozess durchführen - zu reduzieren.

[2] Pulsbefehle sind Hardware-Steuerbefehle auf niedrigster Ebene, die mittels zeitabhängiger Steuerimpulse die Qubits in den entsprechenden Zustand für die Berechnung manipulieren.

2 Status

2.1 **Meilenstein 1** – Implementierung eines experimentellen Setups

Wir möchten den Einfluss einzelner Compiler-Passes auf die Gesamtlaufzeit eines Quantenprogramms untersuchen. Zu diesem Zweck haben wir eine Pipeline konzeptioniert und mithilfe von Profiling-Tools implementiert. Die resultierende Pipeline ermöglicht es uns, die Runtime-Statistiken des Kompilierungsprozesses auszuwerten.

Der Einfluss einzelner Compiler-Passes auf die Gesamtlaufzeit von Quantenprogrammen wurde bisher nicht erforscht. Wir haben eine Pipeline konzipiert, um empirische Daten zur Laufzeit einzelner Compiler-Passes zu erheben. Abbildung 1 zeigt eine schematische Darstellung der Pipeline. Dazu bedienen wir uns Profiling-Tools wie cProfile aus Python. Das zu untersuchende Quantenprogramm kann als Skriptdatei in den Profiler importiert werden. Der Profiler erfasst die Ausführung des Programms und liefert ein Ausführungsprofil, das Informationen zur individuellen Laufzeit der einzelnen Compiler-Passes, zur Anzahl der Funktionsaufrufe und zur Zeit, die in Unterfunktionen aufgewendet wird, enthält. Anhand dieser Daten können wir die Funktionsaufrufe für diejenigen Compiler-Passes filtern, die die längste Laufzeit benötigen. Die zugehörige Pipeline haben wir vollständig in Python 3.9.2 implementiert. Basierend auf dieser Pipeline können wir nun verschiedenste Quantenprogramme profilieren bzw. auswerten.

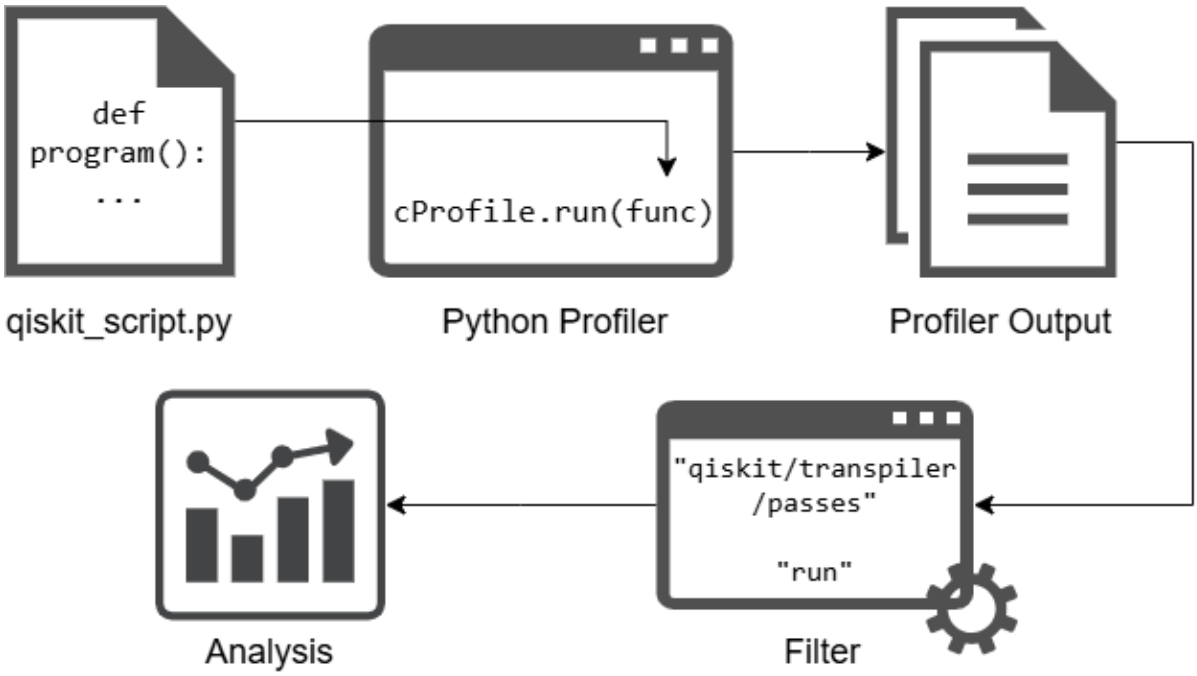

Abbildung 1: Schematische Darstellung der Profiling-Pipeline.

2.2 **Meilenstein 2 –** Auswertung erster Messreihen

Um erste Runtime-Profile studieren zu können, erheben wir mithilfe unserer Profiling-Pipeline zunächst Daten für verschiedene Quantenprogramme. In einer ersten Untersuchung fokussierten wir uns dabei auf die Gesamtlaufzeit der einzelnen Compiler-Passes.

Zur Evaluierung unserer Pipeline und zur Generierung erster Messreihen testeten wir diese mit zwei Programmen: der Quantum-Fourier-Transformation (QFT) und der Greenberger-Horne-Zeilinger (GHZ) State-Preparation. Wir untersuchten die kumulative Laufzeit einzelner Compiler-Passes und identifizierten diejenigen, die den größten Einfluss auf die Gesamtlaufzeit haben. Diese vorläufige Analyse bildet den Ausgangspunkt für unsere weitere Forschung. Unter Verwendung von Qiskit v. 1.3.2 stellten wir fest, dass die Kompilierung einer QFT etwa 82 % und die eines GHZ-Zustands etwa 95 % der Gesamtlaufzeit des Programms in Anspruch nehmen. Abbildung 2 zeigt eine Zusammenfassung der Beiträge der Kompilierungs- und QPU-Ausführungszeit zur Gesamtlaufzeit für verschiedene Compiler-Optimierungs-Level. Diese Beobachtungen basieren auf der Ausführung von Quantenprogrammen mit 100 Qubits und sind somit repräsentativ für aktuelle Workloads.

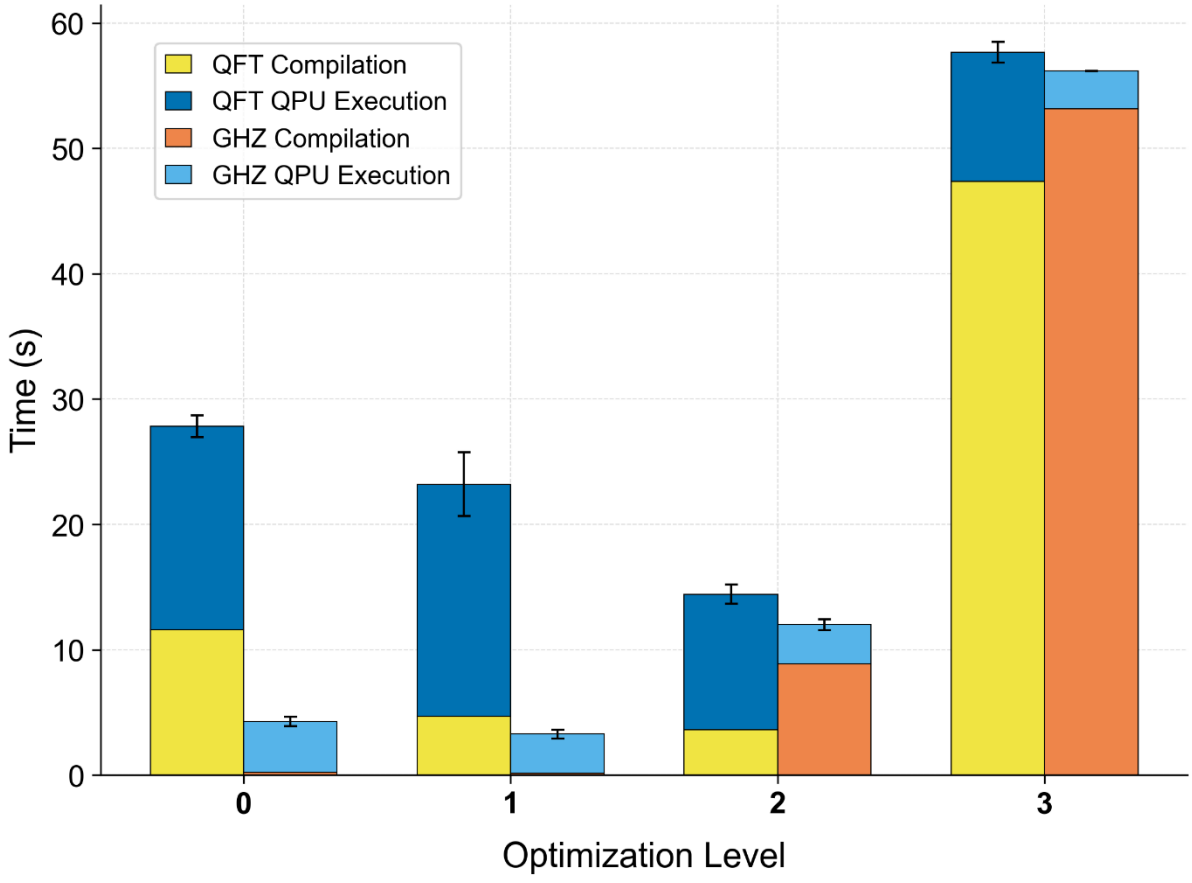

Abbildung 2: Kompilierungs- und QPU-Ausführungszeiten der QFT- und GHZ-Programme für verschiedene Compiler-Optimierungs-Level

Eine eingehendere Untersuchung der Beiträge der einzelnen Compiler-Passes ergab, dass bestimmte Passes den überwiegenden Teil der Kompilierungszeit für beide Programme ausmachten. Wir stellten außerdem fest, dass der Zeitaufwand für die Kompilierung sowohl als Prozentsatz der Gesamtlaufzeit als auch in tatsächlichen Sekunden bei GHZ höher war, obwohl GHZ weniger Gates (Operationen auf Qubits) als QFT umfasst. Diese Ergebnisse deuten darauf hin, dass die Kompilierungszeit nicht allein von der Anzahl der Gates oder Qubits abhängt. Sie wird auch von den vielschichtigen Beziehungen zwischen den Programmeigenschaften und dem resultierenden Verhalten der einzelnen Compiler-Passes beeinflusst.

2.3 **Meilenstein 3 –** Publikation der Zwischenergebnisse

Wir haben die Auswertung unserer Messreihen in Form eines Papers aufbereitet und zusammengefasst. In diesem beschreiben wir auch die nächsten Schritte, die auf den gewonnenen Erkenntnissen basieren. Das Paper wurde als Full Paper zur Präsentation beim „2025 IEEE International Workshop on Quantum Computing: Circuits, Systems, Automation and Applications (QC-CSAA)" angenommen.

Zur Validierung unserer Ideen haben wir ein Paper über unsere vorläufigen Analysen und unsere bisherige Arbeit verfasst. Zusammen mit den bisherigen Zwischenergebnissen haben wir das Paper als Beitrag zum „2025 IEEE International Workshop on Quantum Computing: Circuits, Systems, Automation and Applications (QC-CSAA)"[3] eingereicht, welcher im Rahmen der „ISVLSI 2025"-Konferenz stattfindet. Unser Beitrag wurde als Full Paper zur Präsentation angenommen. Zusätzlich zu dem eigentlichen Inhalt des Manuskripts haben wir im Rahmen des Einreichungsprozesses auch einen sogenannten „Double-blind Peer-Review"-Prozess durchlaufen, der uns wertvolle Rückmeldung für unsere Arbeit geliefert hat.

Der Preprint ist auf ArXiv: https://arxiv.org/abs/2504.15141

[3] https://www.ieee-isvlsi.org/ISVLSI_2025_Website/quantum-computing-workshop.html

3 Zusammenfassung Planaktualisierung

Zusammenfassend ist laut aktuellem Stand eine Anpassung der Stipendienplanung ab Punkt 7 um einen Monat nach hinten erforderlich. Dies ist die Folge der Verlängerung der Frist für die Einreichung von Beiträgen, die von den Organisatoren der ISVLSI 2025 vom 31. März 2025 auf den 30. April 2025 verschoben wurde. Die angepasste Version des Planungsdokuments wird zusammen mit diesem Zwischenbericht bei Netidee eingereicht. Der Rest der Arbeit verläuft bisher nach Plan. Wir freuen uns, dass unsere Zwischenergebnisse bereits als Beitrag auf der Konferenz akzeptiert wurden. Dies bestätigt uns in unserem Vorhaben, einen Beitrag zur verbesserten Ausführung von Quantenprogrammen, insbesondere im „as-a-Service“-Kontext, zu leisten. Im Juni beginnen wir mit der Refinement-Phase 1, in der wir unsere Experimente erweitern und die wertvollen Rückmeldungen aus dem Review-Prozess der Einreichung umsetzen werden.