Raport: Komputery kwantowe

Jerzy Szyjut
193064
Hubert Malinowski
193088

May 20, 2025

1 Wstęp

W ramach *µProjektu* z przedmiotu Zaawansowane architektury komputerów przygotowaliśmy implementację kilku algorytmów kwantowych przy pomocy biblioteki Qiskit w języku Python. W projekcie wzięliśmy pod uwagę algorytmy:

- Grovera
- Shora
- Teleportaacji

Dzięki funkcjonalności biblioteki Qiskit algortmy te można uruchomić na prawdziwych komputerach kwantowych, które są dostępne w chmurze IBM.

2 Wstęp teoretyczny

Komputery kwantowe to urządzenia obliczeniowe, które wykorzystują zjawiska mechaniki kwantowej do przetwarzania informacji. Podstawową jednostką informacji w komputerze kwantowym jest **kubity** (ang. *qubit*), który w przeciwieństwie do klasycznego bitu może znajdować się nie tylko w stanie 0 lub 1, ale także w ich superpozycji, czyli w stanie będącym kombinacją obu tych wartości:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

gdzie α i β są liczbami spełniającymi $|\alpha|^2 + |\beta|^2 = 1$.

Kubity charakteryzują się dwoma kluczowymi właściwościami:

- **Superpozycja** - możliwość jednoczesnego przebywania w wielu stanach.
- **Splątanie** - zjawisko, w którym stan jednego kubitu jest nierozerwalnie związany ze stanem innego, nawet jeśli są one od siebie oddalone.

Obliczenia w komputerach kwantowych realizowane są za pomocą **bramek kwantowych**, które odpowiadają operacjom na kubitach. Bramki te, takie jak bramka Hadamarda, bramka Pauli-X czy bramka CNOT, pozwalają na manipulowanie stanami kubitów i tworzenie złożonych algorytmów kwantowych.

Komputery kwantowe mają potencjał do rozwiązywania pewnych problemów znacznie szybciej niż komputery klasyczne, wykorzystując równoległość obliczeń wynikającą z superpozycji oraz korelacje wynikające ze splątania.

3 Uruchomienie projektu

Aby uruchomić projekt wystarczy podążać za instrukcjami zawartymi w pliku `README.md`. Kopię tych instrukcji zamieszczam poniżej:

3.1 Wymagania wstępne

Aby rozpocząć pracę, upewnij się, że masz zainstalowane następujące elementy:

- Python 3.11 lub nowszy
- Qiskit
- Jupyter Notebook

3.2 Instalacja

1. Utwórz środowisko wirtualne (opcjonalnie, ale zalecane):

   ```
   python -m venv venv
   source venv/bin/activate  # W systemie Windows użyj ‘venv\Scripts\activate‘
   ```

2. Zainstaluj wymagane zależności:

   ```
   pip install -r requirements.txt
   ```

3. Uruchom Jupyter Notebook:

   ```
   jupyter notebook
   ```

4. Otwórz notatniki Jupyter znajdujące się w katalogu `notebooks/`.

5. Jeśli chcesz uruchomić kod na sprzęcie IBM Quantum, skonfiguruj swoje konto IBM Quantum i wprowadź token API w odpowiednim miejscu w kodzie.

3.3 Użytkowanie

1. Uruchom notatniki Jupyter w katalogu `notebooks/`.
2. Modyfikuj i eksperymentuj z kodem, aby pogłębić swoją wiedzę.

4 Dostęp do komputerów kwantowych IBM

Aby uzyskać dostęp do komputerów kwantowych IBM, należy zarejestrować się na stronie IBM Quantum i uzyskać token API. Następnie można skonfigurować token w kodzie, aby uzyskać dostęp do zasobów kwantowych. W momencie w którym piszę ten raport, dostępne są komputery kwantowe o różnej liczbie kubitów, o architekturach różnych. W planie darmowym można przeprowadzać obliczenia na komputerach kwantowych przez 10 minut miesięcznie.

5 Budowa biblioteki Qiskit

Qiskit to modułowa, open-source'owa biblioteka kwantowa rozwijana przez IBM i społeczność, zaprojektowana do tworzenia, analizowania i wykonywania obliczeń kwantowych. Architektura Qiskit została zbudowana z myślą o skalowalności i wsparciu dla różnych backendów kwantowych i klasycznych.

5.1 Quantum Circuit

Komponent `QuantumCircuit` stanowi podstawowy interfejs programowania algorytmów kwantowych. Umożliwia definiowanie rejestrów (klasycznych i kwantowych), aplikowanie bramek, pomiarów, a także kontrolę przepływu programu. Obwody są obiektami wysokiego poziomu, które można analizować, optymalizować oraz kompilować przy użyciu narzędzi takich jak `transpile`.

5.2 Pass Manager

`PassManager` odpowiada za kompilację obwodów kwantowych, przeprowadzając je przez szereg transformacji zwanych *passami*. Każdy pass może optymalizować, przekształcać lub analizować obwód. Passy są organizowane w etapy (*stages*) takie jak unifikacja bramek, mapowanie na topologię sprzętową oraz redukcja głębokości obwodu. Użytkownik może tworzyć własne sekwencje passów lub korzystać z domyślnych przepływów.

5.3 Primitives

`Primitives` to uproszczony i nowoczesny interfejs programistyczny umożliwiający użytkownikom wykonywanie zadań kwantowych takich jak próbkowanie (`Sampler`), szacowanie oczekiwanej wartości (`Estimator`) czy pomiary stanu. Odseparowują one szczegóły backendu i umożliwiają programowanie na wyższym poziomie abstrakcji, z zachowaniem precyzyjnej kontroli nad parametrami eksperymentu.

5.4 Inne komponenty

- **Qiskit Runtime**: Architektura wykonawcza zapewniająca elastyczną i wydajną obsługę zadań kwantowych, działającą lokalnie lub w chmurze.
- **Qiskit Terra**: Podstawowy pakiet odpowiedzialny za kompilację i reprezentację obwodów.

- **Qiskit Aer**: Symulator kwantowy umożliwiający emulację obliczeń kwantowych na klasycznym sprzęcie.
- **Qiskit Experiments**: Framework do prowadzenia eksperymentów kalibracyjnych i charakterystyki urządzeń kwantowych.