

Implementacija kvantnih algoritama

Kovačević, Matej

Master's thesis / Diplomski rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Science / Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:217:460998>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2025-03-27**



Repository / Repozitorij:

[Repository of the Faculty of Science - University of Zagreb](#)



SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
FIZIČKI ODSJEK

Matej Kovačević

IMPLEMENTACIJA KVANTNIH ALGORITAMA

Diplomski rad

Zagreb, 2024.

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
FIZIČKI ODSJEK

INTEGRIRANI PREDDIPLOMSKI I DIPLOMSKI SVEUČILIŠNI STUDIJ
FIZIKA; SMJER ISTRAŽIVAČKI

Matej Kovačević

Diplomski rad

Implementacija kvantnih algoritama

Voditelj diplomskog rada: prof. dr. sc. Hrvoje Buljan

Ocjena diplomskog rada: _____

Povjerenstvo: 1. _____

2. _____

3. _____

Datum polaganja: _____

Zagreb, 2024.

Zahvaljujem mentoru prof. dr. sc. Hrvoju Buljanu i izv. prof. dr. sc. Dariju Jukiću na strpljenju, usmjeravanju i savjetima.

Hvala roditeljima, sestrama i cijeloj obitelji na podršci koju su mi pružali cijeli život.

Hvala kolegama-prijateljima Tonki, Glogaru, Gašpi s Nikom, AMB, Evi i Anji na svim zajedničkim druženjima, učenjima i podsjetnicima na rokove predaja, termine ispita i administrativne obaveze. Hvala Begsu i Rebsu na svim nefakultetskim druženjima.

Hvala Vani na ljubavi, inspiraciji i motivaciji, ostalo mi ovdje ne stane.

Hvala mama!

Sažetak

Kvantno računanje prvo se pojavilo s idejom lakšeg simuliranja prirodnih kvantnih procesa za što treba ogromna količina klasičnih resursa. Od početaka do danas to područje je formalizirano u vidu kvantne teorije informacije i pokazalo se da postoje neki problemi koje bi takav pristup mogao riješiti efikasnije od bilo kojeg poznatog klasičnog algoritma. Najpoznatiji takav primjer je Shorov algoritam za faktoriziranje cijelih brojeva koji je privukao veliku pažnju javnosti zbog potencijalne primjene na razbijanje kriptografije.

U ovom radu dan je teorijski uvod u kvantnu teoriju informacije i opisani su principi kojima kvantno računanje stječe eventualnu prednost nad klasičnim. Ukratko je dan osvrt na postojeća hardverska rješenja i na tehnike kojima se može poboljšati njihova pouzdanost.

Objašnjen je Deutsch problem i pokazano je kako implementirati Deutsch algoritam formalizmom kvantnih krugova koristeći Qiskit paket u Pythonu.

Detaljno je objašnjen Harrow - Hassidim - Lloyd (HHL) algoritam za rješavanje sustava linearnih jednadžbi i sve njegove subrutine koji je zatim primijenjen na konkretan linearni sustav dimenzija 2^{20} i pokazana je implementacija svih kvantnih logičkih vrata potrebnih za to. Pripremljeni kvantni krug izvršen je uzorkovanjem, simuliranjem na klasičnom računalu i na stvarnom IBM-ovom kvantnom računalu. Usporedbom rezultata klasične simulacije i stvarnog mjerenja na kvantnom računalu dobivamo uvid u izazove koji stoje pred kvantnim računanjem prije nego ono postane pouzdana metoda računanja na velikim skalama. Klasičnom simulacijom pokazana je poanta HHL algoritma, a izvedbom na stvarnom kvantnom računalu vidljiva je eksperimentalna nesavršenost realnih qubita i njihova podložnost smetnjama i dekoherenciji.

Ključne riječi: kvantno računanje, qubit, superpozicija, isprepletenost, kvantna logička vrata, kvantni paralelizam, kvantni algoritam, kvantni krug

Implementation of quantum algorithms

Abstract

Quantum computing first emerged with the idea of facilitating the simulation of natural quantum processes, which requires an immense amount of classical resources. From its beginnings until today, this field has been formalized as quantum information theory, and it has been shown that there are some problems that such an approach could solve more efficiently than any known classical algorithm. The most famous example of this is Shor's algorithm for factoring integers, which attracted significant public attention due to its potential application in breaking cryptography.

In this paper, a theoretical introduction to quantum information theory is provided, and the principles by which quantum computing could gain an advantage over classical computing are described. A brief overview is given of existing hardware solutions and techniques that can improve their reliability.

Deutsch's problem is explained, along with a demonstration of how to implement Deutsch's algorithm using the formalism of quantum circuits with the Qiskit package in Python. The Harrow–Hassidim–Lloyd (HHL) algorithm for solving systems of linear equations is explained in detail, including all its subroutines, and is then applied to a specific linear system, with a step-by-step implementation of all the quantum logic gates needed. The prepared quantum circuit was executed through sampling, simulation on a classical computer, and on a real IBM quantum computer.

By comparing the results from classical simulation and actual measurement on a quantum computer, we gain insight into the challenges that quantum computing faces before it can become a reliable method for large-scale computation. The classical simulation demonstrates the essence of the HHL algorithm, while the execution on a real quantum computer reveals the experimental imperfections of real qubits and their susceptibility to interference and decoherence.

Key words: quantum computing, qubit, superposition, entanglement, quantum logic gates, quantum parallelism, quantum algorithm, quantum circuit

Sadržaj

1	Uvod	1
2	Teorijska osnova kvantnog računanja	3
2.1	Qubit	3
2.2	Više qubita	5
2.2.1	Komputacijska baza - notacija	6
2.3	Kvantna logička vrata	7
2.3.1	Jedno-qubitna kvantna logička vrata	8
2.3.2	Više-qubitna kvantna logička vrata	9
2.3.3	Reverzibilnost	9
2.4	Kvantni paralelizam	11
2.5	Kvantni algoritmi	12
2.5.1	Deutsch problem	13
2.5.2	Shorov algoritam	15
2.5.3	Groverov algoritam	16
2.6	Izvedba qubita i korekcija greške	17
3	Harrow - Hassidim - Lloyd (HHL) algoritam	19
3.1	Priprema stanja	21
3.2	Kvantna procjena faze	21
3.2.1	Superpozicija Walsh-Hadamardovim vratima	22
3.2.2	Kontrolirana rotacija	23
3.2.3	Inverzni kvantni Fourierov transformat	24
3.3	Rotacija pomoćnog qubita	26
3.4	Inverzna kvantna procjena faze	26
3.5	Mjerenje pomoćnog qubita	27
3.6	Dodatni komentari i napomene	28
3.6.1	Što ako matrica nije hermitska	28
3.6.2	Normiranost vektora $\mathbf{b}$ i $\mathbf{x}$	29
4	Implementacija HHL algoritma	31
4.1	Zadavanje sustava i priprema stanja	31
4.2	Implementacija kvantnih vrata za procjenu faze	31

4.3 Implementacija rotacije pomoćnog qubita	34
4.4 Inverzna KPF i mjerenje	36
5 Rezultati i diskusija	38
5.1 Egzaktno rješenje	38
5.2 Distribucija vjerojatnosti	38
5.3 Simulator	40
5.4 Kvantno računalo	41
6 Zaključak	43
Dodatci	44
A Implementacija Deutschovog algoritma	44
B Implementacija HHL algoritma	45

1 Uvod

Kvantna teorija informacije, ili kraće, kvantno računanje može se opisati kao primjena kvantnih pravila na teoriju informacije. Svoje početke nalazi u 1980-im godinama kad je Richard Feynman iznio ideju kvantnog računala kao mogući odgovor na problem koji klasična računala imaju sa simuliranjem kvantnih sustava. David Deutsch je zatim formalizirao tu ideju uvođenjem teorijskog modela univerzalnog kvantnog računala demonstrirajući usput i fundamentalnu razliku u obradi informacija u kvantnom i klasičnom računalu što ćemo, između ostaloga, eksplicitno i pokazati u ovom radu.

Vjerojatno najzaslužniji za popularnost koju je kvantno računanje steklo čak i među općom populacijom je Peter Shor koji je 1994. godine otkrio kvantni algoritam za efikasno faktoriziranje cijelih brojeva. To je problem koji se smatrao dovoljno teškim da se na nemogućnost njegova efikasnog rješenja oslanjala sigurnost kriptografije pa je njegovo otkriće uznemirilo javnost i stvorilo sliku da je kvantno računanje gotovo svemoguće ili da se svaki problem može eksponencijalno brže riješiti kvantnim nego klasičnim računalom. To je mit koji za sobom povlači velika očekivanja s kojima se kvantno računanje mora nositi, iako je takva percepcija omogućila tom području istraživanja da prikupi veliku količinu sredstava za razvoj algoritama i eksperimentalnih tehnika pomoću kojih bi se ti algoritmi mogli implementirati na samom hardveru.

Predviđanja o postizanju tzv. kvantne premoći (engl. *quantum supremacy*) se stalno mijenjaju i najčešće pomiču dalje prema budućnosti zbog velikih izazova koji se javljaju pri pokušaju konstrukcije kvantnih računala s velikim brojem qubita. Konkretna izvedba samih qubita može biti svakojaka; od fotoničkih rješenja, preko zatočenih iona ili atoma, supravodljivih qubita do topoloških qubita. Svaka metoda sa sobom nosi svoje prednosti i nedostatke i teško je prognozirati koja će se od postojećih, ili nekih novih, ideja pokazati najboljom.

Najveći dio ovog rada posvetit ćemo Harrow - Hassidim - Lloyd (HHL) algoritmu za rješavanje sustava linearnih jednadžbi. To je problem koji se javlja gotovo u svakom području istraživanja koje se oslanja na matematički opis modela. Dat ćemo detaljan opis načela na kojima počivaju taj algoritam i subrutine koje koristi te ćemo ga primijeniti na jedan konkretan numerički primjer. Također ćemo ga implementi-

rati koristeći Qiskit paket unutar programskog jezika Python. Kod ćemo pokrenuti lokalno pomoću Aer simulatora, ali i na stvarnom IBM-ovom kvantnom računalu. Usput ćemo se kratko osvrnuti na izazove koji stoje pred kvantnim računanjem i prokomentirati koja će područja najviše profitirati eventualnom izgradnjom korisnih, skalabilnih kvantnih računala.

2 Teorijska osnova kvantnog računanja

2.1 Qubit

U klasičnoj teoriji informacije *bit* predstavlja najmanju jedinicu informacije. Može poprimiti dva stanja, 0 ili 1. Kvantna mehanika nudi mogućnost da uvedemo superpoziciju te tako dobijemo sustav u stanju:

(2.1)

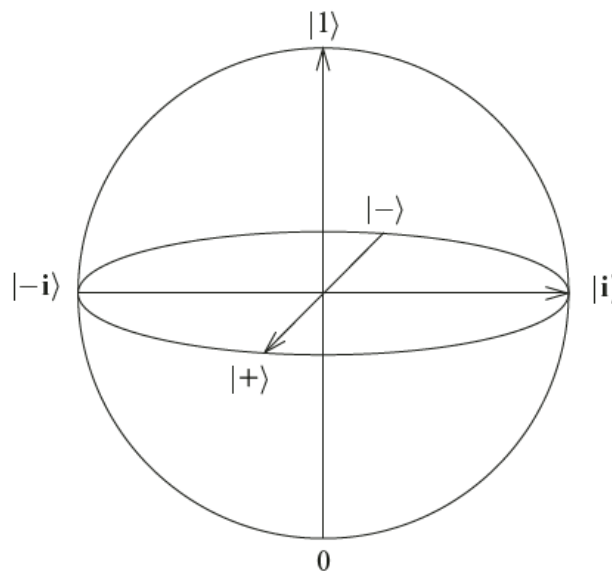
To stanje onda predstavlja tzv. *kvantni bit* ili kraće *qubit*. Za stanje superpozicije se često, pomalo nespretno, navodi da je to stanje u kojem je sustav u isto vrijeme "i nula i jedan" ili da je stanje negdje "između", no činjenica je da je stanje superpozicije isključivo kvantno svojstvo za koje nemamo dobar klasični analogon. Na ovaj način qubit poprima kontinuirani skup vrijednosti za razliku od diskretnog skupa mogućnosti bita. Upravo superpoziciju, zajedno s dodatnim, isključivo kvantnim svojstvom, isprepletenošću, nastoji iskoristiti kvantna teorija informacije kako bi računske operacije i algoritme učinila efikasnijima i uvela mogućnosti koje nisu ostvarive na klasičnim računalima. Formalno je qubit, dakle, jedinični vektor iz kompleksnog dvodimenzionalnog vektorskog prostora s ortonormiranom bazom, po analogiji s klasičnim bitovima, označenom $\{|0\rangle, |1\rangle\}$. Važno je napomenuti da u ovoj reprezentaciji qubita, zbog jedinog ograničenja da taj vektor bude jediničan, postoji višeznačnost, odnosno dva vektora koja se razlikuju samo do na globalnu fazu $e^{i\phi}$ i $e^{i\phi}|\psi\rangle$ predstavljaju isto kvantno stanje jer je globalna faza kompleksni broj modula koji ne mijenja vjerojatnost mjerenja pojedinih ishoda pa time nije fizikalno opaziva.

Kao što je uobičajeno u gotovo svakoj literaturi o kvantnoj fizici koristi se tzv. braket notacija, a u matričnoj reprezentaciji vektori baze mogu se zapisati kao sljedeći vektori stupci:

(2.2)

Bilo koji drugi izbor dva ortonormirana dvodimenzionalna vektora bi bio jednako dobar, no po konvenciji ćemo koristiti ovu standardnu bazu, osim ako je drugačije eksplicitno napomenuto. Koristan način prikaza mogućih stanja jednog qubita je tzv. Blochova sfera prikazana na Slici 2.1. Točke na Blochovoj sferi, za razliku od prikaza jediničnim kompleksnim vektorom, odgovaraju jedinstveno kvantnom stanju qubita, odnosno to preslikavanje je jedan-za-jedan. "Šetanje" po Blochovoj sferi ekvivalentno je rotacijama vektora stanja qubita koje se postižu kvantnim logičkim vratima o kojima će više biti riječi u poglavlju 2.3.1.

Iako se qubit zbog superpozicije može naći u bilo kojem od beskonačno stanja, iz njega nije moguće izvući više od jednog klasičnog bita informacije. To je tako jer mjerenje qubita u odnosu na neku bazu kolabira stanje iz superpozicije u jedno od svojstvenih stanja s vjerojatnošću koja je proporcionalna kvadratu amplitude tog stanja u superpoziciji. Rezultat mjerenja je i dalje jedna od dvije mogućnosti, s vjerojatnošću ili s vjerojatnošću. Dakle, jedan qubit nije nadmoćan nad klasičnim bitom, ali moć kvantnog računanja dolazi do izražaja pametnim redoslijedom primjene kvantnih vrata ili kombinacijom više qubita.



Slika 2.1: Blochova sfera. Preuzeto iz [1].

2.2 Više qubita

Sustav (registar) od klasičnih bitova se može naći u jednom od diskretnih stanja, odnosno pomoću klasičnih bitova informacija možemo razlikovati različitih mogućnosti. Qubiti se kombiniraju, kao i ostali kvantni sustavi, pomoću tenzorskog produkta. To znači da, ukoliko imamo qubita, svaki sa svojom bazom njihovo ukupno stanje će biti zapisano preko baze:

$$\vdots \quad (2.3)$$

Oznake rednog broja qubita i tenzorskog produkta se često izostavljaju u svrhu lakšeg čitanja pa se koristi i oznaka , a najčešće samo za prvi element baze i analogno za ostale. Pogledajmo primjer qubita. Njihovo zajedničko stanje bit će zapisano preko baze . Elementi baze predstavljali bi sve mogućnosti koje bi mogli poprimiti klasični bitovi, no qubiti mogu biti u općenitoj normiranoj superpoziciji tih stanja, , što opet čini prostor mogućnosti kontinuirano beskonačnim. Čak i ako fiksiramo koeficijente u superpoziciji i samo izabiremo od kojih članova ćemo uopće graditi superpoziciju to nam ostavlja mogućih kombinacija. Tako smo se i eksplicitno uvjerali koliko više mogućnosti dopušta prostor kombiniranih qubita, a sad ćemo ukratko objasniti sljedeće važno svojstvo sustava više qubita, isprepletenost.

Od svih spomenutih mogućih superpozicija, odnosno stanja qubita, postoje neka stanja koja se ne mogu dekomponirati i zapisati kao tenzorski produkt stanja pojedinačnih qubita. Takva stanja zovemo isprepletena (engl. *entangled*) stanja. Primjerice, stanje:

$$(2.4)$$

nije isprepleteno jer se, kao što vidimo, može zapisati kao tenzorski produkt stanja pojedinih qubita. S druge strane, stanje:

(2.5)

jest isprepleteno i ne može se dekomponirati na tenzorski produkt stanja dva qubita jer bi raspis desne strane vodio na zaključak da $|0\rangle$ i $|1\rangle$ moraju biti $|0\rangle$, a bez da $|0\rangle$ ili $|1\rangle$ budu $|0\rangle$ što je očito nemoguće. Usput, ovo stanje se zove EPR (Einstein - Podolsky - Rosen) par te je jedno od Bellovih stanja i ima važnu ulogu u kvantnom računanju.

Povećavanjem broja qubita, većinu prostora stanja sustava iscrpljuju upravo isprepletena stanja stoga su ona izrazito važna za kvantne algoritme i često će se sustav nalaziti upravo u takvim stanjima kroz neke dijelove algoritama.

2.2.1 Komputacijska baza - notacija

Ranije smo bazu za dva qubita pisali skraćenom notacijom tenzorskog produkta elemenata baze pojedinih qubita, $|00\rangle, |01\rangle, |10\rangle, |11\rangle$. Uočimo li da elementi baze nalikuju brojevima $00, 01, 10, 11$ u binarnom sustavu, možemo uvesti sljedeću notaciju

Brojevi unutar ket-ova su prikaz u dekadskoj bazi onog broja koji odgovara tom kvantnom stanju, a broj znamenki pomoću kojeg taj broj zapisati u binarnoj bazi nam govori indeks koji ujedno i odgovara broju qubita, u ovom slučaju 2. Pogledajmo još jedan primjer:

Broj 10 u binarnom sustavu glasi 2 , a indeks 1 nam govori da ga moramo zapisati u binarnom sustavu s 2 znamenki. Fizikalno ovaj primjer znači da imamo 2 qubita od kojih su svi u stanju $|0\rangle$, osim trećeg koji je u stanju $|1\rangle$. Pomoću ove notacije ćemo onda bazu sustava od 2 qubita pisati:

(2.6)

2.3 Kvantna logička vrata

Transformacija nekog općenitog kvantnog stanja rezultira ponovno kvantnim stanjem. Zahtjev normiranosti novog stanja nameće uvjet da ta transformacija bude unitarna, odnosno da je prikaz tog operatora u nekoj bazi unitarna matrica što znači da joj je inverz jednak svom hermitski adjungiranom operatoru, $U^\dagger = U^{-1}$. Očita posljedica unitarnosti je reverzibilnost svih transformacija kvantnog sustava, a jedna manje očita posljedica je činjenica da se općenito kvantno stanje ne može klonirati što je poznato kao Teorem o zabrani kloniranja (engl. *No-cloning theorem*). Pretpostavimo li da postoji klonirajući operator $\mathcal{K}$ koji djeluje na općenito stanje $|\psi\rangle$ na sljedeći način:

(2.7)

i pretpostavimo li da je stanje $|\psi\rangle$ ortogonalno stanju $|\phi\rangle$, možemo pogledati djelovanje operatora $\mathcal{K}$ na stanje $|\psi\rangle$:

(2.8)

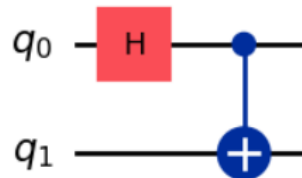
U prvom koraku smo iskoristili definiciju stanja $|\psi\rangle$, u drugom linearnost unitarnog operatora, a u trećem djelovanje operatora kloniranja. S druge strane, djelovanje operatora $\mathcal{K}$ na stanje $|\psi\rangle$ klonira to stanje:

(2.9)

Uspoređujući desne strane izraza 2.8 i 2.9 vidimo da smo došli do kontradikcije jer ni $|\psi\rangle$ ni $|\phi\rangle$ ne mogu biti jednaki pa zaključujemo da ne može postojati operator koji klonira nepoznato kvantno stanje.

Po uzoru na klasična logička vrata koja djeluju na bitove, operacije na qubitima nazivaju se kvantnim logičkim vratima. Niz kvantnih logičkih vrata naziva se kvantni

krug i može se apstraktno grafički prikazati za lakšu vizualizaciju kvantnih algoritama. Jednostavni primjer je dan na Slici 2.2, a konkretno taj prikazani krug služi za pripremu spomenutog EPR para. Može se pokazati da se bilo koja operacija na registru od n qubita može dekomponirati na niz dvo-qubitnih i jedno-qubitnih kvantnih logičkih vrata [2-4].



Slika 2.2: Primjer jednostavnog kvantnog kruga koji priprema EPR par. Generirano u Pythonu koristeći Qiskit paket.

2.3.1 Jedno-qubitna kvantna logička vrata

U ovom poglavlju pogledat ćemo neke transformacije stanja jednog qubita. Trivijalna kvantna logička vrata su ona koja ostavljaju qubit u istom stanju, odnosno identiteta I . Po uzoru na klasična NE-vrata postoje i kvantna logička NE-vrata koja označavamo s N . Korisna su i kvantna vrata koja mijenjaju relativnu fazu između stanja, a od iznimne važnosti su i Hadamardova vrata H koja stavljaju qubit u stanje uniformne superpozicije dva osnovna stanja. Djelovanje ovih vrata na elemente baze i njihove matrične reprezentacije su sljedeće:

(2.10)

$$\begin{aligned}
 I &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\
 N &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}
 \end{aligned}$$

2.3.2 Više-qubitna kvantna logička vrata

Najvažnija dvo-qubitna kvantna logička vrata su tzv. kontrolirana NE-vrata (engl. *Controlled NOT-gate*, $CNOT$) koja djeluju na dva qubita na način da negiraju drugi, **ciljni** (engl. *target*) qubit ukoliko je prvi, **kontrolni** (engl. *control*) qubit u stanju $|1\rangle$. Njihovo djelovanje na standardnu bazu dva qubita je sljedeće:

$$(2.11)$$

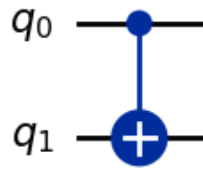
$CNOT$ vrata ne mogu se zapisati kao tenzorski produkt dvaju jedno-qubitnih vrata pa se u kombinaciji s Hadamardovim vratima mogu iskoristiti za pripremu isprepletenog EPR para:

$$(2.12)$$

Indeks na Hadamardovim vratima označava na koji qubit se njihovo djelovanje odnosi. Formalno primjenjujemo transformaciju U_{ij} , ali radi preglednosti zapisa indeksom naglašavamo koji qubit transformiramo, a na ostalima podrazumijevamo djelovanje identitete. U_{ii} i $U_{ii}^\dagger$ vrata su sama svoj inverz. Grafički prikaz $CNOT$ vrata dan je na Slici 2.3. Puni kružić označava kontrolni qubit, a krug sa znakom plus na njemu označava ciljani qubit koji se negira ukoliko je kontrolni qubit u stanju $|1\rangle$. Osim kontroliranih NE-vrata, mogu postojati i neka druga kontrolirana U_{ij} -vrata, gdje može biti bilo koja operacija koju ćemo primijeniti na ciljani qubit samo ako je kontrolni qubit u stanju $|1\rangle$. (Za konkretan primjer pogledati 3.2.2).

2.3.3 Reverzibilnost

Kao što smo već spomenuli, kvantne transformacije (kvantna vrata) moraju biti reverzibilne, odnosno reverzibilnost je posljedica unitarnosti jer se inverzna transformacija dobije samo hermitskim adjungiranjem originalne, $U^\dagger$. S druge strane, postoje klasična vrata koja nisu reverzibilna, odnosno ne možemo se iz rezultata vra-



Slika 2.3: Grafički prikaz kontroliranih NE-vrata. Generirano u Pythonu koristeći Qiskit paket.

titi unatrag jedinstveno do ulazne vrijednosti. Kao najjednostavniji primjer možemo pogledati klasična jedno-bitna NULL vrata čije je djelovanje definirano na sljedeći način:

S obzirom da za obje mogućnosti ulaznih vrijednosti, ova funkcija vraća istu vrijednost kao izlaz (nije injektivna), vidimo li kao rezultat, ne možemo znati koja je bila ulazna vrijednost. Klasična NE-vrata, I-vrata ni ILI-vrata također nisu reverzibilna. Bilo koju klasičnu operaciju nad registrom od bitova možemo formalno gledati kao funkciju:

$$\mathbb{Z} \rightarrow \mathbb{Z} \quad (2.13)$$

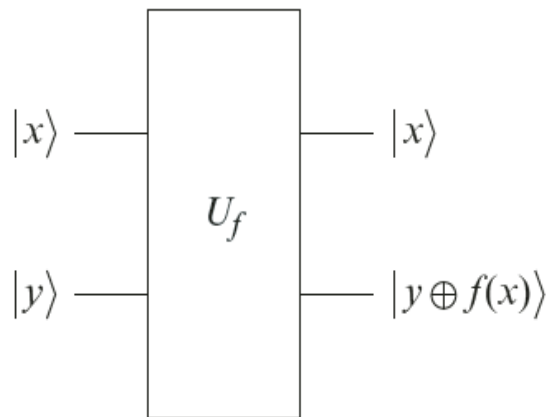
čiji je izlaz zapisan u registru od bitova, a predstavljaju broj mogućih ulaznih (izlaznih) bitnih nizova (engl. *bitstrings*). Takvu funkciju možemo kanonski proširiti na način da uvedemo pomoćni registar koji možemo iskoristiti za razlikovanje različitih ulaza koji daju iste izlaze. Pomoćni ulazni registar ima istu dimenzionalnost kao i izlaz funkcije, . Formalno tada funkcija glasi:

$$\mathbb{Z} \rightarrow \mathbb{Z} \quad (2.14)$$

U drugom članu izlaza, označava isključivo ILI (engl. *exclusive-OR*) koje primjenjujemo bit po bit između bitnog niza i bitnog niza . Uzmemo li u ulaznom registru da je , onda reproduciramo djelovanje originalne funkcije . Ovakva konstrukcija će uvijek dati reverzibilnu funkciju [1].

S obzirom da smo sada konstruirali reverzibilnu funkciju, možemo je zamijeniti uni-

tarnim operatorom koji predstavlja kvantna logička vrata. Grafički prikaz tog operatora dan je na slici 2.4.



Slika 2.4: Grafički prikaz unitarnog operatora čije djelovanje odgovara reverzibilnoj konstrukciji funkcije f . Preuzeto iz [1].

2.4 Kvantni paralelizam

Unitarni operator iz prošlog odjeljka je naravno linearan pa njegovo djelovanje na neku superpoziciju opisujemo kao:

(2.15)

Bit će nam korisna superpozicija svih mogućih stanja sustava, a nju postizemo tzv. Walsh-Hadamardovom transformacijom, odnosno primjenom Hadamardovih vrata na svaki pojedini qubit. Ako imamo qubita to izgleda ovako:

$$\begin{array}{c}
 \text{---} \quad \text{---} \quad \text{---} \quad \text{---} \\
 \text{puta} \quad \text{puta} \quad \text{puta} \quad \text{puta} \\
 \text{---} \quad \text{---} \quad \text{---} \quad \text{---} \\
 \text{---} \quad \text{---} \quad \text{---} \quad \text{---}
 \end{array}
 \quad (2.16)$$

gdje smo u zadnjem koraku iskoristili kompaktni zapis preko komputacijske baze opisane u poglavlju 2.2.1. Znači djelovanje Walsh-Hadamardove transformacije na početno stanje gdje su svi qubiti u stanju daje uniformnu superpoziciju svih

mogućih stanja. Ubacimo li to u unitarna vrata i izaberemo li pomoćni registar također u stanju dobijemo:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (2.17)$$

Primijetimo da smo zbog superpozicije jednim djelovanjem kvantnih logičkih vrata na neki način primijenili funkciju na svaki mogući ulaz jer se sada sustav nalazi u superpoziciji svih mogućih vrijednosti funkcije i to je ono što nazivamo kvantnim paralelizmom.

Iako naivno djeluje nevjerojatno moćno, da bismo saznali bilo koju vrijednost funkcije, sustav bismo trebali izmjeriti što bi nam dalo samo jednu i još k tome nasumično odabranu vrijednost u skladu s njenom vjerojatnošću da bude izmjerena, što znači da ne možemo magično očitati sve vrijednosti funkcije unatoč tome što smo ih "magično" sve izračunali odjednom. Ipak, ovaj korak nije beskoristan, već je često jedan od prvih koraka u brojnim kvantnim algoritmima u kojima onda daljnjim transformacijama tog sustava slijede korisne stvari, kao što se može vidjeti u poglavlju 3.2.1.

2.5 Kvantni algoritmi

Kvantni algoritam se od klasičnog algoritma razlikuje po tome što transformira stanje sustava unitarnim, kvantnim logičkim vratima i iskorištava tijekom svog izvođenja kvantna svojstva superpozicije i isprepletenosti. Ideja je vidjeti postoji li neki kvantni algoritam koji postiže nešto što klasični algoritam ne može. Odgovor na to je negativan; naime svaki kvantni algoritam mogao bi se u teoriji simulirati na klasičnom računalu, samo što bismo potencijalno potrošili ogromnu količinu resursa (memorije i vremena) za to učiniti. Ideja onda postaje vidjeti možemo li naći kvantni algoritam koji neki problem rješava brže nego ijedan klasični algoritam.

Najjednostavnija klasa algoritama su oni koji tretiraju funkciju kao crnu kutiju kojoj možemo slati upite (engl. *queries*). Poslati upit znači pogledati što crna kutija izbaci kao odgovor na zadani ulaz. Ne možemo pogledati unutar crne kutije što ona točno radi, otuda i taj naziv, nego su nam dostupni samo ulaz i izlaz i pokušavamo na temelju njih zaključiti nešto o unutarnjem mehanizmu, odnosno matematičkim rječnikom, o funkciji unutar te kutije. Efikasnost algoritma se u ovakvim proble-

mima mjeri brojem upita koje je potrebno postaviti crnoj kutiji da bi se dobio odgovor na pitanje koje nas zanima. Demonstrirat ćemo da postoji problem koji je moguće kvantnim algoritmom riješiti uz manje upita nego ijednim klasičnim algoritmom.

2.5.1 Deutschov problem

U ovom problemu zanima nas je li funkcija $f: \mathbb{Z}_2 \rightarrow \mathbb{Z}_2$ konstantna.

Budući da je ova funkcija koja djeluje na jedan qubit nije teško eksplicitno navesti sve četiri mogućnosti za takvu funkciju. Možemo imati:

Prve dvije funkcije su konstantne, odnosno daju isti izlaz za oba ulaza, dok zadnje dvije funkcije nisu konstantne, a u ovom slučaju su i balansirane, što znači da za polovicu svih ulaza daju vrijednost 0, a za drugu polovicu daju vrijednost 1. U ovom trivijalnom dvodimenzionalnom slučaju čim funkcija nije konstantna automatski je balansirana, ali u višim dimenzijama to ne mora biti slučaj.

Lako je vidjeti da klasičnim algoritmom moramo poslati minimalno dva upita funkciji, odnosno ispitati izlaze za obje mogućnosti ulaza i vidjeti poklapaju li se ti izlazi kako bismo mogli odrediti je li funkcija konstantna. U kvantnom algoritmu upite šaljem unitarnim vratima opisanim u poglavlju 2.3.3 čije je djelovanje . Pogledajmo što se događa ukoliko pomoćni qubit pripremimo u stanju $|0\rangle$.

$$|0\rangle$$

$$|0\rangle$$

(2.18)

Djeluje kao da nismo puno postigli s obzirom da smo samo dobili globalnu fazu, međutim pogledajmo dalje što ako nam je ulaz u superpoziciji $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$.

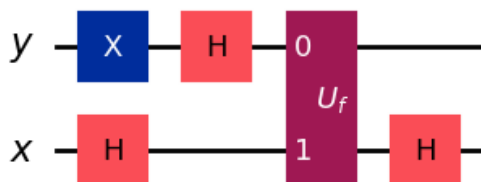
$$\begin{aligned} &= \\ &= \end{aligned}$$

(2.19)

const.

const.

Sad je jasno da smo, zanemarujući globalnu fazu, dobili dvije mogućnosti za slučajeve kad funkcija je ili nije konstantna i te dvije mogućnosti možemo sa sigurnošću razlikovati primjenjujući Hadamardova vrata na prvi qubit. Zbog toga što, a mjerenje prvog qubita nakon opisane procedure daje sa sigurnošću ako je funkcija konstantna, a ako nije. Opisani kvantni algoritam garantira da jednim upitom, odnosno jednim izvršavanjem algoritma, dobijemo odgovor na pitanje je li funkcija konstantna ili ne, a to nije moguće postići nijednim klasičnim algoritmom. Shematski prikaz Deutschovog algoritma dan je na Slici 2.5.



Slika 2.5: Kvantni krug Deutschovog algoritma. Generirano u Pythonu koristeći Qiskit paket.

Mjerenje gornjeg qubita daje nam odgovor na pitanje kao što je opisano ranije. Na toj shemi eksplicitno vidimo da su vrata crna kutija i ne znamo njihovo unutarnje djelovanje, ali svejedno ovim krugom saznajemo je li funkcija konstantna ili ne. Na ovoj slici može se primijetiti da je položaj qubita i obrnut u odnosu na Sliku 2.4 i to je konvencija iz Qiskit paketa koje ćemo se držati; qubit koji je skroz lijevo u ket-vektoru ide na dno kvantnog kruga i onda redom dalje. Također je konvencija da svi qubiti počinju u stanju $|0\rangle$. U Dodatku A dan je kod u Pythonu koji implementira Deutschov algoritam za izabranu funkciju f .

Deutschov problem je jednostavan, a čak i njegova generalizacija koju nazivamo

Deutsch-Josza problem [5] nije neki problem koji ćemo susresti u realnoj situaciji u životu, ali važan je jer dokazuje da postoje problemi koje kvantni algoritmi mogu riješiti brže od klasičnih. Bernstein-Vazirani i Simonov problem još su dva primjera problema crnih kutija (engl. *black box* ili *oracle*) koji se rješavaju algoritmima upita (engl. *query*) koji su nadmoćniji od klasičnih algoritama. Iako su navedeni problemi pomalo umjetni, korisno je eksplicitno se uvjeriti u činjenicu da za njih postoji nadmoćni kvantni algoritam. Otkriće tih problema je dalo nadu i inspiraciju za kompliciranije kvantne algoritme koji bi mogli imati i konkretnu primjenu, kao što je, vjerojatno najpoznatiji od svih, Shorov algoritam.

2.5.2 Shorov algoritam

Peter Shor je 1994. [6] predstavio svoj algoritam za faktoriziranje cijelih brojeva koji je polinomijalan u vremenu, što je izazvalo veliko zanimanje znanstvene zajednice, a i šire jer je većina kriptografije zasnovana na uvjerenju da je faktoriziranje cijelih brojeva dovoljno težak problem koji se ne može efikasno riješiti. Inspiracija mu je bio Simonov algoritam [7] koji smo kratko spomenuli, a kojem je cilj za funkciju sa svojstvom $f(x) = f(y)$ pronaći bitni niz x . Simon je otkrio algoritam koji to može učiniti sa $O(N^2)$ poziva funkciji i dodatnih $O(N)$ klasičnih koraka, za razliku od klasičnih algoritama kojima treba $O(N^3)$ poziva [1].

Shor je, koristeći kvantni paralelizam i kvantni Fourierov transformat, pronašao algoritam za efikasno pronalaženje perioda funkcije. Veza između pronalaska perioda određene funkcije i faktoriziranja cijelih brojeva bila je već poznata u to vrijeme tako da je klasičnom obradom perioda dobivenog kvantnim dijelom algoritma uspio efikasno faktorizirati željeni cijeli broj. Shorov algoritam se uvelike oslanja na rezultat da primjena kvantnog Fourierovog transformata na stanje čije su amplitude zadane diskretnom periodičnom funkcijom $f(x)$ s periodom P rezultira stanjem koje je superpozicija samo onih stanja koja su višekratnici broja P , gdje je N broj qubita, odnosno kad je $x = kP$ za neki $k \in \mathbb{N}$. Funkcija čiji period želimo pronaći je $f(x) = x^r \bmod N$, gdje je r broj kojeg želimo faktorizirati. Ukratko, koraci Shorovog algoritma su sljedeći:

- nasumično izaberemo x i provjerimo je li x dijeli N . Ako da, posrećilo nam se i našli smo jedan faktor od N . U puno vjerojatnijem slučaju da su x i N relativno prosti nastavljamo dalje.

- Pripremimo superpoziciju i kvantnim paralelizmom izračunamo funkciju $f(x)$ te primijenimo kvantni Fourierov transformat. Ovaj korak je potpuno kvantni korak.
- Mjerenje s velikom vjerojatnošću daje rezultat koji je blizu višekratnika — (nemamo garanciju jer period općenito ne mora biti potencija baze N).
- Klasičnim metodama izračunamo pravi period r iz rezultata dobivenog mjerenjem u prošlom koraku. Za parni r provjerimo imaju li N ili $N/2$ i zajedničkih djelitelja.
- Ako do ovog koraka nismo uspjeli pronaći neki djelitelj od N , vratimo se na prvi korak i ponavljamo algoritam.

Prvi i četvrti korak ovog algoritma mogu se efikasno provesti korištenjem Euklidovog algoritma za pronalaženje najvećeg zajedničkog djelitelja. Kompleksnost Shorovog algoritma je polinomijalna u N , što je znatno bolje od najboljeg poznatog klasičnog algoritma koji je sub-eskponencijalan.

2.5.3 Groverov algoritam

Groverov algoritam [8] omogućuje kvadratičnu prednost nad klasičnim algoritmima u zadatku pretraživanja nestrukturiranih podataka. Formalno ga možemo opisati funkcijom $f: \mathbb{Z}_N \rightarrow \mathbb{Z}_2$ gdje nam je cilj saznati onaj $x \in \mathbb{Z}_N$ za koji vrijedi $f(x) = 1$. S obzirom da imamo N mogućih ulaza, u najgorem slučaju klasični algoritam mora provjeriti svaki, jer ne postoji neka struktura u podacima koja bi nam olakšala, pa je kompleksnost klasičnog algoritma $\mathcal{O}(N)$.

Groverov algoritam koristi fazna vrata upita (engl. *phase query gates*), definirana djelovanjem: $U_f = I - 2|f^{-1}(1)\rangle\langle f^{-1}(1)|$. Usporedba s izrazom (2.18) pokazuje da fazna vrata upita možemo implementirati unitarnim vratima upita U_f smatramo li pomoćni qubit dijelom samih vrata. Koristimo U_f za dvije funkcije; zadanu funkciju f i OR funkciju koja vraća 1 ako joj je ulaz 1, a 0 inače.

Implementacija Groverovog algoritma prati sljedeće korake:

- Na registar pripremljen u $|0\rangle$ stanju djelujemo Walsh-Hadamardovom transformacijom ostavljajući sustav u uniformnoj superpoziciji svih mogućih stanja.

- Iterativno ponavljamo korak $\frac{1}{2}$ OR odabrani broj puta.
- Mjerenje svih qubita u standardnoj bazi daje bitni niz koji uzimamo kao kandidat za rješenje.

Broj iteracija u drugom koraku ovisi o pouzdanosti koju želimo postići s obzirom da želimo povećati amplitudu stanja koje sadržava rješenje kako bismo povećali vjerojatnost njegovog mjerenja u idućem koraku. Broj iteracija se najčešće uzima da je $\frac{1}{2}$ otkud slijedi da je kompleksnost Groverovog algoritma $\frac{1}{2}$.

S obzirom da Groverov algoritam ne nudi eksponencijalno nego kvadratično ubrzanje njegova korisnost bi bila vidljiva tek za veliki $\frac{1}{2}$ što znači i velik broj qubita. Zašto to predstavlja problem objašnjeno je u sljedećem poglavlju.

2.6 Izvedba qubita i korekcija greške

Nameće se pitanje kako konstruirati qubit u fizičkom svijetu. U principu svaki kvantni sustav s dva nivoa može biti qubit kojem jedan nivo predstavlja $|0\rangle$, a drugi $|1\rangle$. Najveći inženjerski problem očekivano predstavlja dekoherencija koja narušava nužna kvantna svojstva qubita. Potrebna je vrlo visoka razina izoliranosti qubita od okoline kako bi se sačuvala koherencija na vremenskim skalama duljim od onih potrebnih za obavljanje željene operacije. Nekoliko je predloženih i eksperimentalno dobivenih realizacija qubita, od kojih su neke: supravodljivi qubiti, fotonski qubiti, zatočeni atomi i ioni te topološki qubiti.

Fotonski qubiti su prilično otporni na dekoherenciju i mogu prelaziti velike udaljenosti što je korisno za kvantnu komunikaciju, npr. gusto kodiranje (engl. *dense coding*), ali manipulacija pojedinih fotona je eksperimentalno zahtjevna kao i primjena kvantnih logičkih vrata.

Zatočeni ioni se laserski ohlade u osnovno stanje i radiofrekventnim zamkama se drže lokalizirani u prostoru [9, 10]. 1995. su eksperimentalno realizirana vrata ovom tehnikom [11] za što je 2012. dobivena i Nobelova nagrada. Otpornost i očuvanje koherencije zatočenih iona je relativno veliko u odnosu na druge izvedbe, ali skalabilnost predstavlja problem.

Neutralni atomi pak pružaju mogućnost dobrog skaliranja na veće sustave, ali su im vremena koherencije nešto manja nego ionima [12]. QuEra omogućuje javnosti pristup kvantnom računalu zasnovanom na ovom tipu qubita [19].

Trenutno su najnapredniji supravodljivi qubiti, ponajviše jer su se za njih odlučili Google i IBM. Oni omogućavaju brzu primjenu kvantnih vrata i relativno dobro se skaliraju, a prednost im je što se mogu stavljati na čipove gdje se mogu ostvariti razne povezivosti između qubita. Iako krajem prošle godine IBM je službeno predstavio kvantno računalo s 5 qubita, najavili su da će svoje napore usmjeriti prema boljoj pouzdanosti čipova s manjim brojem qubita, a ne na daljnje povećanje broja qubita. Najveće javno dostupno IBM-ovo kvantno računalo je iz obitelji "Heron" i sadrži 5 qubita [20] što znači da smo još uvijek u tzv. NISQ (engl. *noisy intermediate-scale quantum*) eri.

Sve dosad opisane izvedbe qubita podložne su, u većoj ili manjoj mjeri, šumu i dekoherenciji što predstavlja veliki izazov u dobivanju pouzdanih rezultata jer se kvantna informacija brzo izgubi. Načela kvantne teorije informacije kao što su Teorem o zabrani kloniranja i formalizam kvantnog mjerenja dodatno kompliciraju stvari jer onemogućuju da se direktno uvede redundancija. Zbog toga je vrlo važno područje kvantne korekcije greške (engl. *quantum error correction*) koje nastoji osmisliti načine da se kvantna informacija očuva, a bez kloniranja ili mjerenja. Najveći problem pri tome predstavlja višak (engl. *overhead*) fizičkih qubita koji se generira kako bi se stvorio jedan logički qubit koji je onda otporan na greške. Kad se uzme u obzir taj višak, nijedno današnje kvantno računalo nije ni blizu tome da bi moglo provesti algoritam s uključenom korekcijom greške jer se tu radi o redu veličine milijuna fizičkih qubita. Zbog toga je primamljiva ideja topoloških qubita kod kojih je kvantna informacija pohranjena u globalno stanje para anyona i time nije osjetljiva na lokalne perturbacije. Eksperimentalno manipuliranje anyonima je ekstremno zahtjevno pa će onaj ko prvi uspije realizirati i skalirati topološke qubite biti u velikoj prednosti.

Iako je kvantno računanje koje tolerira greške (engl. *fault-tolerant*) zasad neostvarivo, postoje nedavna istraživanja koja ukazuju na to da bi mogli postojati problemi kod kojih je kvantno računanje korisno i sa prisutnim šumom, pogotovo u slučajevima kad nas zanima očekivana vrijednost operatora na kvantnom krugu [17].

3 Harrow - Hassidim - Lloyd (HHL) algoritam

Harrow, Hassidim i Lloyd su 2009. godine predložili kvantni algoritam za pronalaženje aproksimativnog rješenja sustava linearnih jednadžbi [13]. Matematička formulacija problema glasi:

(3.1)

gdje je hermitska matrica, a i predstavljaju vektore poznatih i nepoznatih vrijednosti respektivno. Cilj problema je naći vektor kao:

(3.2)

no HHL algoritam najviše obećava u situacijama gdje nas zanimaju veličine poput , odnosno očekivana vrijednost operatora reprezentiranog matricom . Zapis ovog problema u formalizmu kvantnog računanja jest:

(3.3)

pri čemu će komponente vektora biti pohranjene kao koeficijenti superpozicije qubita u komputacijskoj bazi (2.6).

(3.4)

Matricu možemo zapisati u bazi njenih svojstvenih vektora pomoću svojstvenih vrijednosti , a s obzirom da je u toj bazi matrica dijagonalna, lako se dobije i njen inverz.

(3.5)

—

Uz činjenicu da su svojstveni vektori ortonormirani, , lako se provjeri ispravnost ovog zapisa, odnosno da vrijedi . Vektor također možemo razviti i po ovoj bazi što će biti važno pri izvođenju algoritma,

(3.6)

Kombinirajući izraze (3.5) i (3.6), izraz (3.2) možemo zapisati kao:

— (3.7)

no mi ćemo htjeti imati zapisanog u komputacijskoj bazi jer u odnosu na nju mjerimo pojedine qubite u registru u kojem će biti pohranjeno rješenje.

(3.8)

Osim registra u kojem će na početku biti pohranjen , a na kraju , za kojeg smo već rekli da se sastoji od qubita, HHL algoritam koristi još dva registra. Prvi od njih će služiti za pohranjivanje faze svojstvenih vrijednosti u koraku kvantne procjene faze i sastojat će se od qubita, a zadnji registar se sastoji od samo jednog, tzv. pomoćnog qubita. Po običaju su svi qubiti na početku inicijalizirani u stanju što možemo, uz naglašavanje pojedinih registara, zapisati kao:

(3.9)

Oznaku koristit ćemo za zapisivanje ukupnog stanja cijelog sustava u -tom koraku.

Nakon tog nultog koraka, odnosno samog alociranja qubita, slijede ostali koraci HHL algoritma:

- priprema stanja,
- kvantna procjena faze,
- rotacija pomoćnog qubita,
- inverzna kvantna procjena faze,
- mjerenje.

3.1 Priprema stanja

Cilj ovog koraka je u prvi registar učitati vektor $|0\rangle$, što znači zarotirati taj registar od qubita iz osnovnog stanja u stanje superpozicije s koeficijentima koji odgovaraju komponentama vektora $|0\rangle$. To je takozvano kodiranje amplitudama. Taj proces u principu ovisi o konkretnom vektoru $|0\rangle$ i nakon što smo proveli tu transformaciju (kao npr. u [16]) sustav se nalazi u stanju:

$$(3.10)$$

Primijetimo ovdje da smo izostavili indeks i iz prvog registra. Naime, držimo se konvencije da pisanje indeksa nakon ket-a ukazuje na to da taj ket reprezentira kvantno stanje koje dobijemo kad broj unutar ket-a zapišemo binarno s brojem znamenki koji odgovara indeksu, kao što je opisano u 2.2.1. U ovom primjeru $|0\rangle$ bi značilo da broj i moramo zapisati u binarnom obliku s n znamenki. To očito nema smisla jer i uopće ne predstavlja neku varijablu ili broj, već je to ime vektora pa zbog konzistentnosti notacije izostavljamo indeks i iako je i dalje taj registar u kojem je pohranjen $|0\rangle$ sastavljen od n qubita. Za razliku od ovog primjera, u izrazu (3.23) ostaje indeks jer se tamo upravo radi o situaciji gdje broj i zapisujemo binarno s znamenki. Takvo kodiranje nekih veličina u kvantno računalo nazivamo kodiranjem bazom, za razliku od primjera kodiranja amplitudama, kao što je slučaj u situaciji s $|0\rangle$.

3.2 Kvantna procjena faze

Kvantna procjena faze (KPF) (engl. *Quantum phase estimation* (QPE)) će rezultirati time da u drugom registru, koji sadržava n qubita, budu bazom kodirane faze svojstvenih vrijednosti matrice H koje ćemo povezati sa svojstvenim vrijednostima matrice H . Ovakav način kodiranja matrice H naziva se Hamiltonijansko kodiranje i osigurava da operator U kojeg ćemo primjenjivati na prvi registar bude unitaran jer je hermitska matrica. Parametar t možemo smatrati vremenom do kojeg evoluiramo sustav operatorom U .

Unitarne matrice imaju svojstvene vrijednosti apsolutne vrijednosti jednake 1 pa ih možemo pisati u formi $e^{i\phi}$, pri čemu ϕ predstavlja fazu te svojstvene vrijednosti. Međutim, mi smo ranije spominjali svojstvene vektore i svojstvene vrijednosti matrice

(označavali smo ih s $|i\rangle$ i $|j\rangle$) pa nam treba sljedeći izraz koji povezuje svojstvene vrijednosti matrice H i J , budući da je

$$(3.11)$$

Ovaj izraz nam također govori i da su svojstveni vektori matrice H jednaki onima matrice J pa onda možemo pisati:

$$(3.12)$$

Uvrstimo li definiciju matrice H u izraz (3.11), odnosno J , dobivamo:

$$(3.13)$$

a onda iz izraza (3.12) i (3.13) uočavamo korespodenciju:

$$(3.14)$$

U potpoglavljima koja slijede detaljnije ćemo pogledati pojedine dijelove ovog koraka algoritma.

3.2.1 Superpozicija Walsh-Hadamardovim vratima

Početno stanje sustava u ovoj fazi algoritma je ono iz izraza (3.10) što znači da se drugi registar nalazi u stanju $|0\rangle$. Slijedi djelovanje Hadamardovih vrata na svaki od n qubita drugog registra kako bi se postigla njihova uniformna superpozicija. Primjenjivanje Hadamardovih vrata na svaki qubit, $\frac{1}{\sqrt{2}} \sum_{j=0}^{1} |j\rangle$ nazivamo Walsh-Hadamardovom transformacijom, kao što je spomenuto u 2.4.

$$(3.15)$$

Ovaj izraz bi se mogao i kompaktnije zapisati, ali ovako će biti eksplicitno jasnije što se događa u ostalim dijelovima KPF.

3.2.2 Kontrolirana rotacija

Kontrolirana rotacija ukazuje na činjenicu da će postojati kontrolni i ciljni qubit, ili u ovom slučaju čak cijeli registri. Drugi registar će biti kontrolni te će on kontrolirati prvi registar koji je ciljni. Operacija koja će se primjenjivati na prvi registar je U^{2^r} , gdje r predstavlja redni broj qubita u drugom registru počevši s desne strane, $r = 0, \dots, m - 1$, te se operacija primjenjuje samo ako je taj r -ti qubit u stanju $|1\rangle$. Znači svaki od m qubita drugog registra kontrolira po jedna U^{2^r} vrata koja djeluju na cijeli prvi registar ukoliko se njihov qubit nađe u stanju $|1\rangle$. Opisanu operaciju nazvat ćemo C_U .

U ovom trenutku ćemo pretpostaviti da je u prvom registru, umjesto $|b\rangle$, učitani neki svojstveni vektor $|u_j\rangle$. Izvod će dalje ići s tom pretpostavkom, a da to u principu ne smanjuje općenitost pokazuje izraz (3.6) gdje smo $|b\rangle$ razvili upravo po svojstvenim vektorima tako da ćemo na kraju izvoda jednostavno napisati općeniti izraz kad je u registru zaista $|b\rangle$. Ova pretpostavka je ekvivalentna pretpostavci da $|b\rangle$ u tom razvoju ima samo jedan član, tj. da vrijedi $|b\rangle = |u_j\rangle$.

Djelovanje operatora U^{2^r} na vektor $|u_j\rangle$ je poznato i glasi:

$$U^{2^r} |u_j\rangle = e^{2\pi i \phi_j 2^r} |u_j\rangle. \quad (3.16)$$

S obzirom da taj operator primjenjujemo samo kad je r -ti qubit u stanju $|1\rangle$, možemo faktor $e^{2\pi i \phi_j 2^r}$ dodati ispred $|1\rangle$ u svakom qubitu drugog registra u izrazu (3.15), s njegovim odgovarajućim r , jer skalari slobodno "šetaju" kroz tenzorski produkt.

$$\begin{aligned} |\Psi_{2b}\rangle &= C_U |\Psi_{1a}\rangle \\ &= |u_j\rangle \frac{1}{2^{\frac{m}{2}}} \underbrace{\left((|0\rangle + e^{2\pi i \phi_j 2^{m-1}} |1\rangle) \otimes \dots \otimes (|0\rangle + e^{2\pi i \phi_j 2^0} |1\rangle) \right)}_{m \text{ puta}} |0\rangle_1 \\ &= |u_j\rangle \frac{1}{2^{\frac{m}{2}}} \sum_{k=0}^{2^m-1} e^{2\pi i \phi_j k} |k\rangle_m |0\rangle_1. \end{aligned} \quad (3.17)$$

U trećoj jednakosti smo iskoristili zapis pomoću komputacijske baze s m qubita. Raspis tenzorskog produkta qubita drugog registra daje $e^{2\pi i \phi_j (2^{m-1} a_{m-1} + \dots + 2^0 a_0)}$ gdje prepoznavamo izraz koji pretvara broj k zapisan kao vektor $|a_{m-1} \dots a_0\rangle$ iz binarnog u

dekadski zapis, gdje je $\log_{10} N$, $\lceil \cdot \rceil$ -ta znamenka zdesna tog broja u binarnom zapisu.

3.2.3 Inverzni kvantni Fourierov transformat

Slijedi primjena inverznog kvantnog Fourierovog transformata (IKFT) (engl. *Inverse quantum Fourier transform* (IQFT)) koji je zapravo kvantna verzija inverznog diskretnog Fourierovog transformata. Djelovanje IKFT na neko općenito kvantno stanje qubita možemo zapisati kao:

$$IKFT : | \psi \rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{-i 2 \pi k x} | k \rangle \quad (3.18)$$

gdje $| \psi \rangle$ predstavlja inverzni diskretni Fourierov razvoj funkcije $f(x)$:

$$f(x) = \sum_{k=0}^{N-1} \hat{f}(k) e^{i 2 \pi k x} \quad (3.19)$$

a $\hat{f}(k)$. Napomenimo da u dijelu literature ovakav transformat nosi naziv kvantni Fourierov transformat, ali mi ćemo se držati konvencije u kojoj je kvantni Fourierov transformat onaj bez minusa u eksponentu.

Nas će zanimati djelovanje IKFT na pojedini vektor baze $| x \rangle$:

$$IKFT : | x \rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{-i 2 \pi k x} | k \rangle \quad (3.20)$$

i iskoristit ćemo u jednom koraku raspisa sljedeći rezultat:

$$\sum_{k=0}^{N-1} e^{-i 2 \pi k x} = N \delta_{x,0} \quad (3.21)$$

osim ako je x mod N [1].

IKFT se primjenjuje na drugi registar stanja (3.17) te koristeći linearnost, definiciju IKFT, promjenu redoslijeda sumacije i izraz (3.21) dobivamo:

$$\begin{aligned}
& \text{IKFT} \\
& \text{IKFT} \quad \text{---} \\
& \text{---} \quad \text{IKFT} \\
& \text{---} \quad \text{---} \\
& \text{---} \quad \text{---} \\
& \text{---} \quad \text{---}
\end{aligned} \tag{3.22}$$

Vidimo da je postupak kvantne procjene faze rezultirao time da se u drugom registru nalazi n -teroznamenasta binarna reprezentacija broja n , odnosno imamo informaciju o fazi svojstvene vrijednosti koja pripada svojstvenom vektoru $|n\rangle$ zapisanog u prvom registru. Prisjetimo se izraza (3.14) gdje smo pokazali poveznicu između faze svojstvene vrijednosti operatora U i svojstvene vrijednosti operatora V . Iz njega slijedi $e^{i\phi} = \langle n | U | n \rangle$, što u općenitom slučaju nije cijeli broj, pa onda ni izraz zapisan u drugom registru stanja (3.22) nije cijeli broj. Ovdje je korisno što nam je preostao parametar ϕ kojeg onda izabiremo tako da $e^{i\phi}$ bude cijeli broj i još možemo uvesti pokratu $\phi = 2\pi k$. S tim svim na umu, izraz (3.22) možemo zapisati kao:

(3.23)

Finalno nam je još ostalo prisjetiti se da smo cijelo vrijeme radili s pretpostavkom da je u prvi registar učitani svojstveni vektor $|0\rangle$, a ne vektor $|1\rangle$. To ne predstavlja problem jer smo u izrazu (3.6) zapisali $|1\rangle$ kao superpoziciju svojstvenih vektora pa zbog linearnosti svih provedenih transformacija možemo ukupno stanje sustava jednostavno zapisati na sljedeći način:

(3.24)

3.3 Rotacija pomoćnog qubita

Pomoćni qubit će u ovom koraku biti zarotiran u stanje:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad (3.25)$$

za svaki θ . $\frac{1}{\sqrt{2}}$ predstavlja normalizacijsku konstantu i u principu je poželjno da bude što veća jer će $\frac{1}{2}$ biti vjerojatnost da se pomoćni qubit izmjeri u stanju o čemu ćemo detaljnije kasnije. θ je ograničen odozgora najmanjom reskaliranom svojstvenom vrijednošću $\frac{1}{2}$, što znači da mi želimo maksimalni θ za koji i dalje vrijedi $\frac{1}{2} \leq \cos^2 \theta$.

Stanje sustava nakon rotacije pomoćnog qubita jest:

$$\frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (3.26)$$

3.4 Inverzna kvantna procjena faze

Cilj ovog koraka je rasplesti prvi i drugi registar koji su trenutno isprepleteni pa ne možemo prvi registar vratiti u bazu $|0\rangle$ u kojoj bi ga htjeli mjeriti, iz baze $|0\rangle$ u kojoj je trenutno zapisan. U suštini ovaj korak vrći unatrag korak 3.2; prvo ćemo djelovati, ovog puta kvantnim Fourierovim transformatom na drugi registar, zatim inverznom kontroliranom rotacijom, operatorom $R(\theta)$ na prvi registar ako je θ -ti qubit drugog registra jednak 1 i naposljetku Walsh-Hadamardovim operatorom na drugi registar.

Djelovanje kvantnog Fourierovog transformata (KFT) na drugi registar stanja (3.26) daje:

$$\text{KFT} \left(\frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \right) = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix} \quad (3.27)$$

Slično kao i ranije u izrazu (3.17) i uz korištenje izraza (3.14), djelovanje inverzne kontrolirane rotacije možemo pisati na sljedeći način:

$$\begin{aligned}
|\Psi_{4b}\rangle &= C_{U^{-1}} |\Psi_{4a}\rangle \\
&= \sum_{j=0}^{2^n-1} \tilde{b}_j |u_j\rangle \left(\frac{1}{2^{\frac{m}{2}}} \sum_{y=0}^{2^m-1} e^{-\lambda_j t y} e^{\frac{2\pi i}{M} \tilde{\lambda}_j y} |y\rangle_m \right) \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} |0\rangle_1 + \frac{C}{\tilde{\lambda}_j} |1\rangle_1 \right) \\
&= \sum_{j=0}^{2^n-1} \tilde{b}_j |u_j\rangle \left(\frac{1}{2^{\frac{m}{2}}} \sum_{y=0}^{2^m-1} |y\rangle_m \right) \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} |0\rangle_1 + \frac{C}{\tilde{\lambda}_j} |1\rangle_1 \right). \tag{3.28}
\end{aligned}$$

U zadnjem redu smo iskoristili definiciju pokrate $\tilde{\lambda}_j$ tako da se eksponenti zbroje u 0. Sada vidimo da prvi i drugi registar više nisu isprepleteni što smo i htjeli postići. Preostaje još samo djelovati Walsh-Hadamardovim vratima kako bi drugi registar vratili u osnovno stanje zahvaljujući činjenici da su Hadamardova vrata sama sebi inverz.

$$\begin{aligned}
|\Psi_4\rangle &= (I^{\otimes n} \otimes H^{\otimes m} \otimes I) |\Psi_{4b}\rangle \\
&= \sum_{j=0}^{2^n-1} \tilde{b}_j |u_j\rangle |0\rangle_m \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} |0\rangle_1 + \frac{C}{\tilde{\lambda}_j} |1\rangle_1 \right). \tag{3.29}
\end{aligned}$$

3.5 Mjerenje pomoćnog qubita

Završni korak HHL algoritma je mjerenje pomoćnog qubita. Taj korak se u principu može odraditi i odmah nakon rotacije pomoćnog qubita, prije inverzne procjene faze, jer je pomoćni qubit neovisan, odnosno nije isprepleten s prva dva registra. Bude li rezultat mjerenja pomoćnog qubita $|0\rangle$, rezultat zanemarujemo i ponavljamo algoritam ispočetka dok ne izmjerimo $|1\rangle$. Kao što smo već napomenuli, vjerojatnost željenog ishoda je $\left| \frac{C}{\tilde{\lambda}_j} \right|^2$ pa je poželjno izabrati što veći C . Nakon što je pomoćni qubit izmjeren u stanju $|1\rangle$, sustav kolabira u stanje:

$$|\Psi_5\rangle = \frac{1}{\sqrt{\sum_{j=0}^{2^n-1} \left| \frac{\tilde{b}_j C}{\tilde{\lambda}_j} \right|^2}} \sum_{j=0}^{2^n-1} \frac{\tilde{b}_j C}{\tilde{\lambda}_j} |u_j\rangle |0\rangle_m |1\rangle_1. \tag{3.30}$$

Predfaktor koji se pojavio je tu zbog normalizacije stanja. Pod pretpostavkom da je C realna konstanta i uvrštavanjem $\tilde{\lambda}_j$ možemo sve konstante izlučiti iz suma u nazivniku predfaktora i u valnoj funkciji stanja sustava te pokratiti, što onda vodi na izraz:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (3.31)$$

Sad koristeći razvoj vektora $\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ iz izraza (3.7) i normiranost tog istog stanja, odnosno izraza (3.37) možemo konačno zapisati završno stanje:

$$\frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (3.32)$$

Postigli smo ono što smo htjeli; rješenje početne jednadžbe (3.1) je sada spremljeno u prvom registru kvantnog računala na kojem smo proveli HHL algoritam. Komponente (reskaliranog) $\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ možemo saznati mjerenjem qubita prvog registra u odnosu na standardnu bazu $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$. Pojedina komponenta $\frac{1}{\sqrt{2}}$ predstavlja amplitudu vjerojatnosti mjerenja prvog registra u stanju $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$. Formalno to možemo zapisati kao:

$$\frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (3.33)$$

i vidimo da bismo algoritam trebali ponoviti minimalno $\frac{1}{\epsilon}$ puta ako želimo saznati sama rješenja početnog sustava jednadžbi.

Napomenimo još da je kompleksnost HHL algoritma $\frac{1}{\epsilon}$, gdje je κ kondicijski broj matrice A , a ϵ željena preciznost rješenja. U članu 3.2.2 se očituje hipotetska eksponencijalna prednost u odnosu na klasične algoritme, pogotovo u slučaju kad nas ne zanimaju konkretne komponente rješenja.

3.6 Dodatni komentari i napomene

3.6.1 Što ako matrica nije hermitska

U prethodnim poglavljima smo pretpostavljali da je matrica A hermitska, odnosno da vrijedi $A = A^\dagger$. To je bitno da bismo mogli provesti simulaciju operatora u koraku 3.2.2, ali u općenitom slučaju matrica A ne mora biti hermitska. Tada možemo proširiti sustav trivijalnim jednadžbama i raditi s matricom

$$\begin{pmatrix} A & 0 \\ 0 & 0 \end{pmatrix} \quad (3.34)$$

koja je očito hermitska. U tom slučaju poznati vektor $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ proširujemo s nul-vektorom $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

jednake dimenzije i rješavamo sustav

(3.35)

čije rješenje onda nalazimo u formi [13].

3.6.2 Normiranost vektora $\mathbf{b}$ i $\mathbf{x}$

Vektor $\mathbf{b}$ je učitao u prvi registar, te njegove komponente predstavljaju koeficijente u kvantnoj superpoziciji $\mathbf{x}$, a za kvantna stanja znamo da moraju biti normirana. Vektor $\mathbf{x}$ naravno općenito ne mora biti normiran, ali on nam je poznat pa lako možemo izračunati njegovu normu i reskalirati ga tako da bude normiran. Zato bez smanjenja općenitosti možemo pretpostaviti da nam je stanje $\mathbf{x}$ iz izraza (3.6) normirano, odnosno da vrijedi:

(3.36)

Da bi stanje $\mathbf{x}$ bilo normirano, matricu A isto možemo reskalirati pa onda vrijedi i:

— (3.37)

U slučaju da ne reskaliramo matricu onda možemo jednostavno pisati

— (3.38)

što znači da tada nakon izvođenja algoritma u prvom registru imamo pohranjen normalizirani vektor $\mathbf{x}$, odnosno:

— (3.39)

Bit će nam koristan i izraz koji povezuje normu rješenja $\|\mathbf{x}\|$ s vjerojatnošću mjerenja pomoćnog qubita u stanju $|0\rangle$. Izraz (3.38) daje formulu za normu egzaktnog rješenja, a gledajući izraze (3.29) i (3.30) vidimo da vjerojatnost mjerenja pomoćnog

qubita u stanju možemo zapisati kao:

$$\frac{1}{\sqrt{2}}(\lvert 0 \rangle + \lvert 1 \rangle) \tag{3.40}$$

Uvrštavajući pokratu — izvede se željeni izraz.

$$\begin{aligned} & \frac{1}{\sqrt{2}}(\lvert 0 \rangle + \lvert 1 \rangle) \\ &= \frac{1}{\sqrt{2}} \left(\frac{1}{\sqrt{2}}(\lvert 00 \rangle + \lvert 01 \rangle) + \frac{1}{\sqrt{2}}(\lvert 10 \rangle + \lvert 11 \rangle) \right) \\ &= \frac{1}{2}(\lvert 00 \rangle + \lvert 01 \rangle + \lvert 10 \rangle + \lvert 11 \rangle) \end{aligned} \tag{3.41}$$

4 Implementacija HHL algoritma

U ovom poglavlju ćemo riješiti konkretan sustav jednažbi na kojem ćemo implementirati HHL algoritam koristeći Qiskit paket u Pythonu. Sustav je jednostavan, dimenzija 2^3 , ali ćemo na njemu demonstrirati sve korake algoritma, usput prateći i eksplicitno evoluciju vektora stanja.

4.1 Zadavanje sustava i priprema stanja

Matrica sustava A i vektor b su:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad (4.1)$$

U ovom primjeru priprema stanja $|b\rangle$ je jednostavna s obzirom da vektor b u matricnom zapisu odgovara točno vektoru baze $|000\rangle$. Znači samo trebamo primijeniti kvantna NE-vrata, X na početno stanje registra koji će se u ovom slučaju sastojati od jednog qubita jer je vektor b dvodimenzionalan. Znači prvi korak algoritma će biti od početnog stanja:

$$(4.2)$$

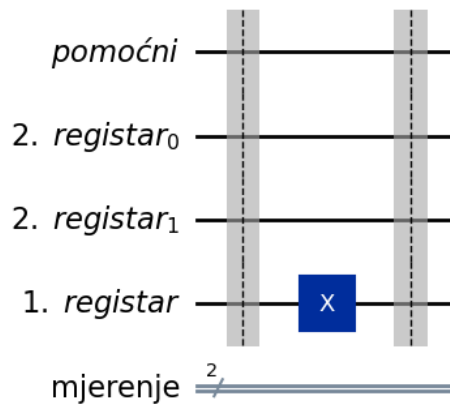
dobiti stanje s učitanim vektorom b u prvi registar.

$$(4.3)$$

Pripadni kvantni krug koji priprema početno stanje dan je na Slici 4.1. Na toj slici osim spomenuta 3 kvantna registra vidimo i klasični registar s dva bita koji će služiti da u njega pohranimo mjerenja pomoćnog qubita i prvog registra.

4.2 Implementacija kvantnih vrata za procjenu faze

Svojstvene vrijednosti zadane matrice su $\lambda_1 = 1$ i $\lambda_2 = -1$. S obzirom da omjer λ_1/λ_2 mora vrijediti i za λ_1/λ_2 , izbor parametra $\alpha = \pi/4$ daje $\lambda_1/\lambda_2 = 1$. Vidimo da su nam 2 qubita u drugom registru dovoljna za kodiranje bazom, $|00\rangle$ i $|11\rangle$.



Slika 4.1: Kvantni krug koji priprema stanje. Generirano u Pythonu koristeći Qiskit paket.

Za implementirati kontroliranu rotaciju definiranu u potpoglavlju 3.2.2 se općenito oslanjamo na hamiltonijansko simuliranje [14], no u ovom jednostavnom primjeru ju možemo eksplicitno izvesti. Primijetimo prvo da matricu U u bazi njenih svojstvenih vektora:

$$U = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}$$

možemo zapisati na sljedeći način:

$$U = e^{-i\theta H} \quad (4.4)$$

Matrica je u toj bazi naravno dijagonalna, što će nam olakšati njeno eksponenciranje da bismo dobili $e^{-i\theta H}$ i što su jedine matrice koje trebamo s obzirom da je $H^2 = I$. Uzimajući u obzir da je $H = \frac{1}{2}(I + \sigma_z)$ imamo:

$$e^{-i\theta H} = e^{-i\theta \frac{1}{2}(I + \sigma_z)} = e^{-i\theta/2} e^{-i\theta/2 \sigma_z} = e^{-i\theta/2} \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix} \quad (4.5)$$

Dobili smo vrata U izražena u bazi svojstvenih vektora matrice H , no da bismo ih implementirali želimo ih zapisati u standardnoj bazi. To ćemo postići transfor-

macijom sličnosti: $U = VDV^\dagger$. Matrica V je ista ona koja pretvara matricu D u dijagonalnu D , i njeni stupci su jednaki svojstvenim vektorima matrice D . Da bismo dobili reprezentaciju željenih vrata U u standardnoj bazi onda imamo:

$$U = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} e^{-i\theta} & 0 \\ 0 & e^{i\theta} \end{pmatrix} \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (4.6)$$

Slično imamo i za V :

$$V = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} e^{-i\theta} & 0 \\ 0 & e^{i\theta} \end{pmatrix} \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (4.7)$$

Unutar paketa Qiskit u Pythonu postoji klasa `QuantumCircuit` koja ima ugrađenu metodu za kontrolirana u vrata, `QuantumCircuit().cu(theta, phi, lambda, gamma, control, target)`, s 4 parametara (kuta) kojom se mogu implementirati općenita unitarna vrata

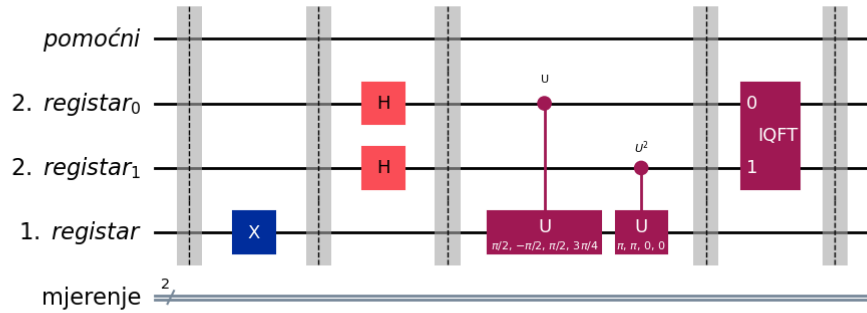
Parametri `control` i `target` odnose se na kontrolni i ciljni qubit (registar) što će kod nas biti pojedini qubiti drugog registra i prvi registar.

U našem slučaju trebat ćemo implementirati U , V prilikom procjene faze i $U^\dagger$ prilikom inverzne kvantne procjene faze, a odgovarajući parametri dani su u Tablici 4.1.

Vrata	Parametri
U	$\theta, \phi, \lambda, \gamma$
V	$\theta, \phi, \lambda, \gamma$
$U^\dagger$	$-\theta, -\phi, -\lambda, -\gamma$

Tablica 4.1: Tablica parametara za implementaciju pojedinih kontroliranih vrata.

Za implementirati (inverzni) kvantni Fourierov transformat također ćemo koristiti postojeću QFT klasu unutar Qiskit paketa. Prošireni krug koji implementira kvantnu procjenu faze nakon pripreme stanja dan je na Slici 4.2.



Slika 4.2: Kvantni krug proširen odgovarajućim vratima tako da implementira kvantnu procjenu faze. Generirano u Pythonu koristeći Qiskit paket.

Pregradama su odvojeni koraci pripreme superpozicije, kontrolirane rotacije i inverznog kvantnog Fourierovog transformata. Eksplicitno stanje sustava u ovom trenutku izgleda:

$$\begin{array}{cc} \text{---} & \text{---} \\ \text{---} & \text{---} \end{array} \quad (4.8)$$

u skladu s jednadžbom (3.24). Pri prelasku u drugi red smo komputacijsku bazu eksplicitno zapisali preko standardne baze i iskoristili najkraći način pisanja tenzorskog produkta između drugog registra i pomoćnog qubita.

4.3 Implementacija rotacije pomoćnog qubita

Cilj nam je zarotirati pomoćni qubit u stanje opisano izrazom (3.25). Minimalna vrijednost reskalirane svojstvene vrijednosti je pa biramo i . Pogledajmo za početak djelovanje općenitog operatora rotacije, konvencionalno nazvanog , na stanje .

$$\begin{array}{cccc} \text{---} & \text{---} & \text{---} & \text{---} \\ \text{---} & \text{---} & \text{---} & \text{---} \end{array} \quad (4.9)$$

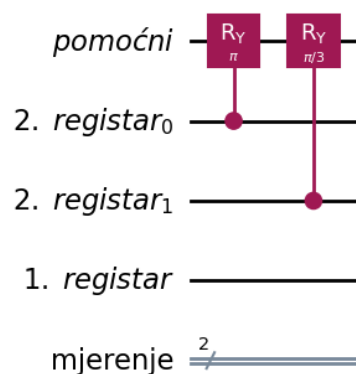
Uspoređujući dobiveni izraz s izrazom (3.25) i primjenjujući osnovni trigonometrijski identitet vidimo da uz odabir parametra — dobivamo željeno stanje. S obzirom da imamo dvije opcije za — moramo smisliti način kako implementirati vrata koja primjenjuju različiti — ovisno o stanju drugog registra gdje će nakon kvantne procjene faze biti pohranjene reskalirane svojstvene vrijednosti. Pogledajmo što želimo dobiti:

$$\begin{aligned} & - \\ & - \quad - \end{aligned} \quad (4.10)$$

Lako se uvjeriti da ukoliko binarnu reprezentaciju stanja registra označimo —, gornje dvije relacije možemo objediniti na sljedeći način:

$$- \quad (4.11)$$

Drugim riječima, svaki od dva qubita drugog registra može nam poslužiti kao kontrolni qubit za primijeniti kontroliranu rotaciju —. Kad je — primijenit ćemo —, a — kad je —. Za to ćemo ponovno iskoristiti metodu `QuantumCircuit().cry(alpha, control, target)` koja primjenjuje kontroliranu rotaciju za odabrani parametar —. Dio kvantnog kruga koji implementira opisane rotacije dan je na Slici 4.3.



Slika 4.3: Dio kvantnog kruga zadužen za rotaciju pomoćnog qubita. Generirano u Pythonu koristeći Qiskit paket.

U svrhu jasnoće opet ćemo navesti i eksplicitno stanje u kojem se trenutno nalazi kvantno računalo, odnosno konkretan izgled izraza (3.26).

$$\begin{array}{cc}
 - & - \\
 \hline
 - & - \\
 - & - \\
 \hline
 - & - \\
 - & - \\
 \hline
 \end{array}
 \quad (4.15)$$

Ovdje eksplicitno vidimo u prvom registru pohranjeno normalizirano rješenje početnog sustava jednažbi.

5 Rezultati i diskusija

U ovom poglavlju ćemo pokrenuti pripremljeni kvantni krug na tri načina i objasniti rezultate svake metode, a prije toga ćemo pokazati i analitički dobiveno egzaktno rješenje. Kod koji obavlja opisani zadatak, kao i pripremu samog kruga nalazi se u Dodatku B.

5.1 Egzaktno rješenje

Primjer koji razmatramo je jednostavan, malih dimenzija, tako da smo relativno jednostavno mogli eksplicitno pratiti evoluciju vektora stanja do kraja algoritma. U izrazu (4.15) vidimo normalizirani vektor rješenja sustava, no možemo li dobiti i egzaktno rješenje? Odgovor je potvrđan ako možemo saznati normu egzaktnog rješenja. Za to ćemo iskoristiti izraz (3.41) iz kojeg slijedi:

$$\| \psi \|^2 = \langle \psi | \psi \rangle \quad (5.1)$$

Uvrštavajući brojeve iz našeg primjera dobivamo da je norma egzaktnog rješenja sustava jednaka $\frac{1}{\sqrt{2}}$. Egzaktni vektor onda nađemo tako što prvi registar pomnožimo s tom normom, a to daje:

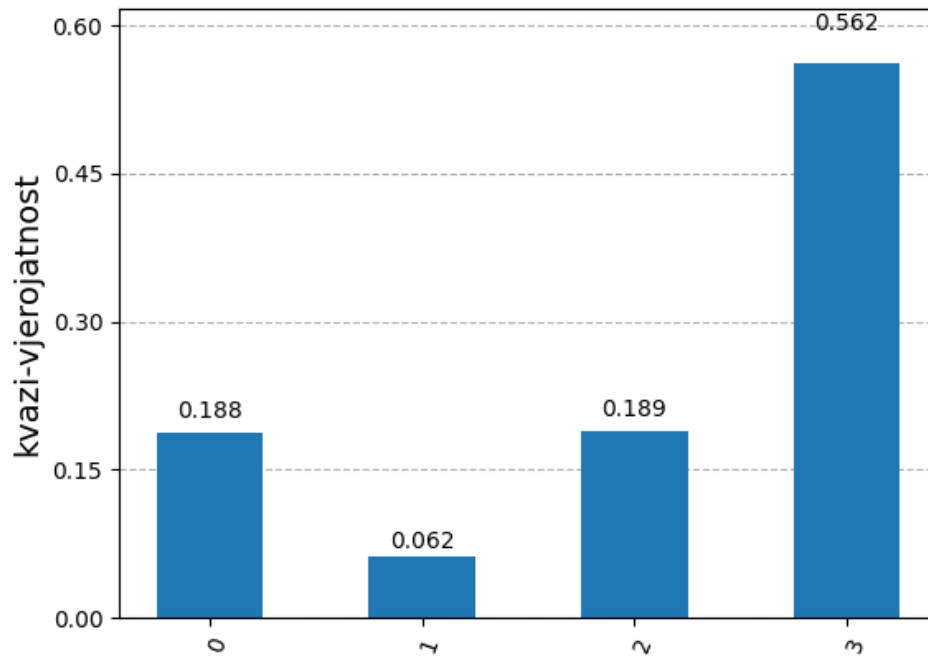
$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \quad (5.2)$$

Lako se uvjeriti da je to točno egzaktno rješenje početnog sustava. Ovaj postupak u općenitom slučaju HHL algoritma naravno ne bismo radili, ali korisno se uvjeriti kako stvarno radi algoritam.

5.2 Distribucija vjerojatnosti

Nakon što smo izgradili krug za implementaciju HHL algoritma prikazan na Slici 4.4, postoje mnogi načini na koje možemo iskoristiti taj krug za saznati nešto što nas zanima. Prvo ćemo primijeniti temeljnu funkciju (engl. *primitive*) `Sampler()` koja omogućuje uzorkovanje kvantnih mjerenja. `Sampler()` ne simulira svaki korak izvođenja algoritma već na temelju zadanog kruga računa distribuciju vjerojatnosti i vraća uzorak iz te distribucije. To je korisno za uvjeriti se radi li algoritam

kako očekujemo prije nego upogonimo simulator ili stvarno kvantno računalo. Naziv "kvazi-vjerojatnost" koji se susreće u ovakvim situacijama odražava činjenicu da se pri njihovom izračunu koriste neke metode korekcije greške i aproksimacije pa je moguće da se sve dobivene vrijednosti neće zbrojiti točno u 1.



Slika 5.1: Prikaz kvazi-vjerojatnosti mjerenja prvog registra i pomoćnog qubita dobiven korištenjem `Sampler()` temeljne funkcije.

Rezultat je prikazan na Slici 5.1. Oznake na x-osi predstavljaju dekadsku reprezentaciju rezultata mjerenja prvog registra i pomoćnog qubita. Npr. oznaka 3 odgovara mjerenju stanja 11. Nas zanima samo slučaj kad je pomoćni qubit izmjerjen u stanju 1 što odgovara drugom stupcu i četvrtom stupcu.

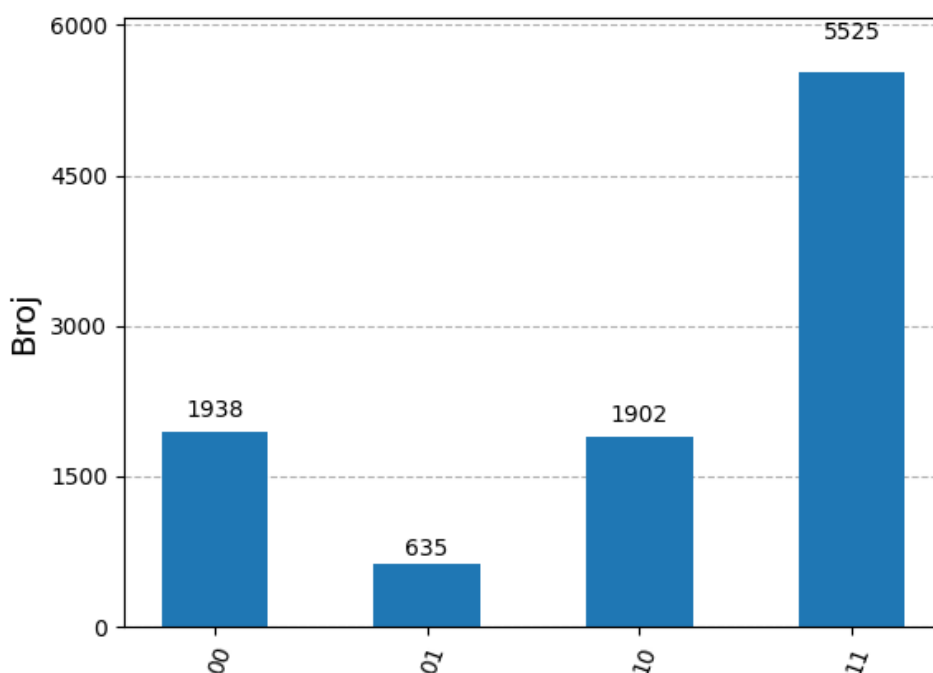
Omjer vjerojatnosti mjerenja prvog registra u stanju 1 i 3 je

$$(5.3)$$

što je u skladu s izrazom (4.15) iz kojeg je jasno da očekujemo da će mjerenje prvog registra u stanju 1 biti 3 puta vjerojatnije nego u stanju 3.

5.3 Simulator

Nakon uzorkovanja, pripremljeni kvantni krug smo pokrenuli koristeći `aer_simulator` simulator iz `Qiskit_aer` paketa. Simulator pokreće kvantni krug kao da se izvršava na stvarnom kvantnom računalu, simulirajući evoluciju vektora stanja i mjerenja. Prije pokretanja trebali smo iskoristiti `transpile()` funkciju koja svede krug na osnovna kvantna logička vrata u slučaju kad se koriste neka kompliciranija vrata. U našem slučaju to je bilo potrebno zbog QFT vrata koja nisu fundamentalna. Simulaciju smo ponovili puta i broj pojedinih ishoda prikazan je u histogramu na Slici 5.2. Iako ovaj histogram izgleda slično onom sa Slike 5.1 postoji fundamentalna razlika u tome kako smo došli do prikazanih podataka.



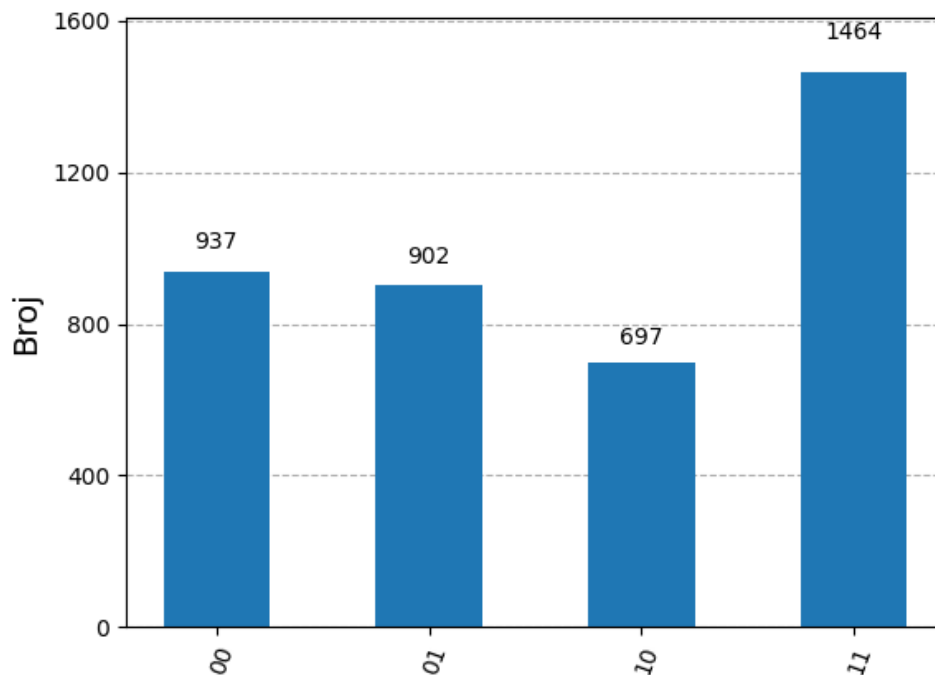
Slika 5.2: Prikaz broja pojedinih ishoda mjerenja prvog registra i pomoćnog qubita dobiven korištenjem `aer_simulator` simulatora.

I ovdje smo dobili očekivan rezultat; gledajući samo slučajeve kad je pomoćni qubit u stanju $|0\rangle$, otprilike puta je vjerojatnije da ćemo izmjeriti prvi registar u stanju $|0\rangle$ nego u stanju $|1\rangle$ jer je . Pogledajmo kako bismo iz ovih rezultata mogli saznati egzaktno rješenje sustava . Komponente normaliziranog rješenja koje zapravo mjerimo označit ćemo s s_0 i s_1 . Iz omjera vjerojatnosti mjerenja pojedinih stanja prvog registra zaključujemo ,

a iz činjenice da je to stanje normirano (3.39) slijedi $\frac{1}{\sqrt{2}}$. Oдавde lako vidimo da $\frac{1}{\sqrt{2}}$ i $\frac{1}{\sqrt{2}}$. Da bismo dobili komponente egzaktnog rješenja trebamo pomnožiti dobivene vrijednosti s normom $\frac{1}{\sqrt{2}}$, a i nju možemo izračunati iz dobivenih rezultata koristeći izraz (5.1). Vjerojatnost mjerenja promoćnog qubita u stanju $|0\rangle$ dobijemo tako da zbrojimo koliko puta smo dobili stanje $|00\rangle$ i stanje $|10\rangle$ i podijelimo s brojem ponavljanja simulacije. Dobivamo $\frac{937 + 697}{1464} \approx 0.5$ što od teorijske vrijednosti iz izraza (4.14) odstupa manje od 1%.

5.4 Kvantno računalo

Za kraj smo pokrenuli pripremljeni kvantni krug na stvarnom IBM-ovom kvantnom računalu. Odabrali smo uređaj imena "ibm_brisbane" koji sadrži 5 qubita i jedan je od uređaja iz "Eagle" obitelji procesora. I ovdje moramo iskoristiti `transpile()` funkciju u kojoj specificiramo odabrani uređaj i provedemo optimizaciju kruga za izvedbu na tom uređaju, ovisno o njegovoj arhitekturi, odnosno povezanosti (engl. *connectivity*.) Informacije o uređajima i praćenje poslanih i odrađenih zahtijeva dostupno je na IBM-ovim stranicama [18, 20].



Slika 5.3: Prikaz broja pojedinih ishoda mjerenja prvog registra i pomoćnog qubita dobiven korištenjem `ibm_brisbane` kvantnog računala.

Provedeno je 10^4 mjerenja (engl. *shots*), a broj pojavljivanja pojedinih ishoda prikazan je na Slici 5.3. Ovdje vidimo da je omjer mjerenja prvog registra u stanju $|0\rangle$ i stanju $|1\rangle$ jednak $1:1$, što znači da se rezultati ne slažu s teorijom niti sa simulacijom. To je tako zbog eksperimentalne nesavršenosti qubita koji su podložni šumu i dekoherenciji. Pokazali smo da čak i na malom broju qubita i relativno malom broju provedenih kvantnih vrata nad njima dolazi do velikih odstupanja pri mjerenju od teorijski očekivanih vrijednosti. U daljnjem radu rezultat bi se mogao probati poboljšati koristeći naprednije tehnike optimizacije kruga i neke tehnike ublažavanja greške (engl. *error mitigation techniques*). Usporedba dobivenih rezultata s postojećom literaturom [15] je zadovoljavajuća.

6 Zaključak

U ovom radu dali smo teorijski uvod u kvantno računanje i objasnili glavne posljedice promjene paradigme pri prelasku s klasične na kvantnu teoriju informacije. Osvrnuli smo se na početke kvantnog računanja i eksplicitno pokazali prvi algoritam koji je pokazao kvantnu prednost nad klasičnim pristupom pri rješavanju Deutschovog problema.

Detaljno smo opisali HHL algoritam za rješavanje sustava linearnih jednadžbi te smo ga primijenili na minimalni sustav. Također smo pokazali kako izgraditi kvantni krug koristeći Qiskit paket u Pythonu za Deutschov i HHL algoritam. Pripremljeni kvantni krug pokrenuli smo na tri načina: uzorkovanjem, simulacijom na klasičnom računalu i izvedbom na stvarnom kvantnom računalu. Prve dvije metode dale su rezultate koje su u slaganju od teorijske vrijednosti i odstupaju od nje manje od 10^{-3} , dok je rezultat dobiven na kvantnom računalu daleko od očekivane vrijednosti od koje odstupa čak više od 10^{-1} . Rezultat bi se daljnjim radom mogao poboljšati koristeći naprednije tehnike ispravljanja greške, ali to bi povećalo vrijeme izvođenja i količinu potrebnih qubita.

Sve u svemu ovaj rad je predstavio moguće prednosti kvantnog računanja nad klasičnim s teorijske strane, ali daje i realnu sliku o pouzdanosti trenutnih qubita i dočarava koliki je još put pred istraživačima i inženjerima prije nego se postigne realna korisnost i pouzdanost kvantnih računala. S druge strane, smatram da nikad ne treba podcijeniti upornost i inovativnost čovječanstva i posve odbaciti mogućnost da ćemo jednom i kvantno računanje na korisnoj skali moći uvrstiti među velika postignuća naše civilizacije.

Dodatci

Dodatak A Implementacija Deutschovog algoritma

```
1
2 from qiskit.circuit import QuantumCircuit, Gate
3 from qiskit import QuantumRegister
4 from qiskit_aer import AerSimulator
5
6 def U_f(i): #funkcija koja ce implementirati jednu of 4 funkcije
7     #pomocu kvantnih logickih vrata
8     Uf=QuantumCircuit(2)
9
10    if i==2:
11        Uf.x(0)
12    elif i==3:
13        Uf.cx(1,0)
14    elif i==4:
15        Uf.x(1)
16        Uf.cx(1,0)
17
18    return Uf
19
20 i=int(input("Unesi Deutschovu funkciju:")) #izabiremo jednu od 4
21     Deutschove funkcije f_i
22
23 Uf=U_f(i)
24
25 Uf.draw(output="mpl") #crtamo konkretnu implementaciju izabrane
26     funkcije f_i
27
28 qr1 = QuantumRegister(1, name="y")
29 qr2 = QuantumRegister(1, name="x")
30
31 alg=QuantumCircuit(qr1,qr2) #kvantni krug koji provodi Deutschov
32     algoritam
33
34 alg.x(0)
35 alg.h(range(2))
36 alg.barrier() #priprema upita završena
37
38 alg.compose(Uf, inplace=True)
39 alg.barrier() #implementacija crne kutije završena
40
41 alg.h(1) #obrada izlaza
42 alg.measure_all()
43
44 alg.draw(output="mpl")
45
46 rez = AerSimulator().run(alg, shots=1, memory=True).result()
47 mjerjenja = rez.get_memory()
48
49 if(mjerjenja[0][0]=="0"):
50     print("Funkcija je konstantna!")
51 else:
52     print("Funkcija nije konstantna!")
```

Kod A.1: Implementacija Deutschovog algoritma

Dodatak B Implementacija HHL algoritma

```
1
2 #uvozenje modula klasa i ostalog
3 from qiskit import QuantumCircuit, QuantumRegister, ClassicalRegister,
   transpile
4 from qiskit_aer import Aer
5 from qiskit.primitives import Sampler
6 from qiskit.circuit.library import QFT
7 from qiskit.visualization import plot_histogram
8 from qiskit_ibm_runtime import QiskitRuntimeService
9 import numpy as np
10
11 #alociranje potrebnih registara
12 reg1=QuantumRegister(1, name="1. registar")
13 reg2=QuantumRegister(2, name="2. registar")
14 pom=QuantumRegister(1, name="pomocni")
15 mjer=ClassicalRegister(2, name="mjerenje")
16
17 krug=QuantumCircuit(pom, reg2, reg1, mjer)
18
19 #superpozicija Walsh-Hadamardovim vratima
20 krug.h(reg2)
21 #krug.barrier() #po potrebi otkomentirati za dodatnu jasnocu
   vizualizacije
22
23 #kontrolirana rotacija
24 krug.cu(np.pi/2, -np.pi/2, np.pi/2, 3*np.pi/4, reg2[0], reg1, label='U
   ')
25 krug.cu(np.pi, np.pi, 0, 0, reg2[1], reg1, label=r'$U^{\{2\}}$')
26 #krug.barrier()
27
28 #inverzni kvantni Fourierov transformat
29 krug.append(QFT(2, inverse=True), reg2)
30 krug.barrier()
31
32 #rotacija pomocnog qubita
33 krug.cry(np.pi, reg2[0], pom)
34 krug.cry(np.pi/3, reg2[1], pom)
35 krug.barrier()
36
37 #inverzna kpf i mjerenje
38 krug.append(QFT(2), reg2)
39 #krug.barrier()
40 krug.cu(np.pi, np.pi, 0, 0, reg2[1], reg1, label=r'$U^{\{-2\}}$')
41 krug.cu(np.pi/2, np.pi/2, -np.pi/2, -3*np.pi/4, reg2[0], reg1, label=r
   '$U^{\{-1\}}$')
42 #krug.barrier()
43 krug.h(reg2)
44 krug.barrier()
45
46 krug.measure(pom, mjer[0])
47 krug.measure(reg1, mjer[1])
48 krug.draw(output="mpl")
49
50 #distribucija vjerojatnosti
51 rez1 = Sampler().run(krug, shots=100000).result()
52 hist1=plot_histogram(rez1.quasi_dists)
53 ax1=hist1.gca().set_ylabel("kvazi-vjerojatnost")
54 display(hist1)
55
56 #simulator
```

```

57 simulator = Aer.get_backend("aer_simulator")
58 transpiled_krug=transpile(krug, simulator)
59
60 rez2=simulator.run(transpiled_krug, shots=10000).result()
61 hist2=plot_histogram(rez2.get_counts())
62 ax2=hist2.gca().set_ylabel("Broj")
63 display(hist2)
64
65 #stvarni kvantni hardver
66 from qiskit_ibm_runtime import QiskitRuntimeService
67 service = QiskitRuntimeService(channel="ibm_quantum", token="unijeti
    svoj token")
68 uredaj = service.backend(name="unijeti ime uredaja")
69 transpiled_krug2=transpile(krug, backend=uredaj, optimization_level=1)
70 job2=uredaj.run([transpiled_krug2])
71 #obrada mjerenja
72 rez3=job2.result()
73 hist3=plot_histogram(rez3.get_counts())
74 ax3=hist3.gca().set_ylabel("Broj")
75 display(hist3)

```

Kod B.1: Implementacija HHL algoritma

Literatura

- [1] Eleanor Rieffel and Wolfgang Polak. 2011. Quantum Computing: A Gentle Introduction (1st. ed.). The MIT Press.
- [2] DiVincenzo, D. P., “Two-bit gates are universal for quantum computation,” *Physical Review A*, Vol. 51, No. 2, 1995, pp. 1015–1022. <https://doi.org/10.1103/physreva.51.1015>.
- [3] Barenco, A., Bennett, C. H., Cleve, R., DiVincenzo, D. P., Margolus, N., Shor, P., Sleator, T., Smolin, J. A., and Weinfurter, H., “Elementary gates for quantum computation,” *Phys. Rev. A*, Vol. 52, 1995, pp. 3457–3467. <https://doi.org/10.1103/PhysRevA.52.3457>.
- [4] Barenco, A., Deutsch, D., Ekert, A., and Jozsa, R., “Conditional Quantum Dynamics and Logic Gates,” *Phys. Rev. Lett.*, Vol. 74, 1995, pp. 4083–4086. <https://doi.org/10.1103/PhysRevLett.74.4083>.
- [5] Deutsch David and Jozsa Richard (1992). Rapid solution of problems by quantum computation, *Proc. R. Soc. Lond.* A439553–558, <https://doi.org/10.1098/rspa.1992.0167>
- [6] Shor, P. W. (1997). Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26(5), 1484–1509. <https://doi.org/10.1137/s0097539795293172>
- [7] Simon, D. R. (1997). On the Power of Quantum Computation. *SIAM Journal on Computing*, 26(5), 1474–1483. <https://doi.org/10.1137/S0097539796298637>
- [8] Grover, L. K. (1997). Quantum Mechanics Helps in Searching for a Needle in a Haystack. *Phys. Rev. Lett.*, 79(2), 325–328. <https://doi.org/10.1103/PhysRevLett.79.325>
- [9] Cirac, J. I., and Zoller, P., “Quantum Computations with Cold Trapped Ions,” *Phys. Rev. Lett.*, Vol. 74, 1995, pp. 4091–4094. <https://doi.org/10.1103/PhysRevLett.74.4091>.

- [10] Eschner, J., Morigi, G., Schmidt-Kaler, F., and Blatt, R., “Laser cooling of trapped ions,” *J. Opt. Soc. Am. B*, Vol. 20, No. 5, 2003, pp. 1003–1015. <https://doi.org/10.1364/JOSAB.20.001003>.
- [11] Monroe, C., Meekhof, D. M., King, B. E., Itano, W. M., and Wineland, D. J., “Demonstration of a Fundamental Quantum Logic Gate,” *Phys. Rev. Lett.*, Vol. 75, 1995, pp. 4714–4717. <https://doi.org/10.1103/PhysRevLett.75.4714>.
- [12] Wintersperger, K., Dommert, F., Ehmer, T. et al. Neutral atom quantum computing hardware: performance and end-user perspective. *EPJ Quantum Technol.* 10, 32 (2023). <https://doi.org/10.1140/epjqt/s40507-023-00190-1>
- [13] Harrow, Aram; Hassidim, Avinatan; Lloyd, Seth. (2009). Quantum Algorithm for Linear Systems of Equations. *Phys. rev. lett.* 103. 150502. DOI:10.1103/PhysRevLett.103.150502.
- [14] Berry, D. W., Ahokas, G., Cleve, R., & Sanders, B. C. (2006). Efficient Quantum Algorithms for Simulating Sparse Hamiltonians. *Communications in Mathematical Physics*, 270(2), 359–371. <https://doi.org/10.1007/s00220-006-0150-x>
- [15] Zaman, A., Morrell, H. J., & Wong, H. Y. (2023). A Step-by-Step HHL Algorithm Walkthrough, *IEEE Access*, 11, 77117–77131. <https://doi.org/10.1109/access.2023.3297658>
- [16] Grover, L.; Rudolph, T.; ”Creating superpositions that correspond to efficiently integrable probability distributions”, 2002., <https://doi.org/10.1103/PhysRevLett.103.150502>
- [17] Kim, Y., Eddins, A., Anand, S. et al. Evidence for the utility of quantum computing before fault tolerance. *Nature* 618, 500–505 (2023). <https://doi.org/10.1038/s41586-023-06096-3>
- [18] IBM Quantum Platform, <https://quantum.ibm.com/> (pogledano 05. 11. 2024.)
- [19] QuEra, <https://www.quera.com/> (pogledano 07.11.2024.)
- [20] IBM Processor types, <https://docs.quantum.ibm.com/guides/processor-types> (pogledano 09.11.2024.)