International Journal of Education, Social Science & Humanities.
Finland Academic Research Science Publishers
ISSN: 2945-4492 (online) | (SJIF) = 8.09 Impact factor
Volume-12| Issue-7| 2024 Published: |22-07-2024|

UCHTA KVANT DASTURLASH TILLARI

https://doi.org/10.5281/zenodo.12701353

Abdurahmonov A.A.
abba@mail.ru
Muhamad Al-Xorazmiy nomidagi Toshkent axborot texnologiyalari universiteti.

Annotasiya

Agar siz kvant kompyuterlari bilan tanish bo'lsangiz, qubit zarracha ekanligini bilasiz. Bitlar holatida dasturlash tillarining ishlash printsipi aniq: barcha operatsiyalar mohiyatan nollar va birlar ketma-ketligi, ya'ni signalning mavjudligi yoki uning yo'qligi. Ammo dasturlash tillari zarrachalarni qanday boshqarishi mumkin? Ushbu maqolada biz sizga dasturlash tillari analog kubitlar bilan qanday ishlashini, qanday kvant tillari mavjudligini va ularning har birining afzalliklari nimada ekanligini aytib beramiz.

Kalit so'zlar

kvant kompyuterlari, kvant algoritmlari, kubit, kvant superpozitsiyasi, kvant chigallashuvi, kvant dekogerensiyasi, kvant dasturlash tili.

Kvant dasturlash tillari qanday ishlaydi

Qubit ikki darajali kvant tizimi bo'lib, bu ikki daraja odatda bitta zarracha (masalan, foton, elektron yoki atom) holatidir, lekin zarracha emas, balki har qanday kvant tizimining holati ham bo'lishi mumkin.

Masalan, agar biz elektron zarrachani kvant tizimi sifatida olsak, u holda qubit quyidagicha bo'lishi mumkin:

- elektron spini;
- uning mavjudligi/yo'qligi.

Agar biz Jozefson (Josephson junction) o‘tishini kvant tizimi sifatida olsak, u holda kubit:

- tok yo‘nalishi (shartli rvaishda chap/o‘ng);
- energetik daraja (noinchi/birinchi).

Ya’ni buning barchasi ixtiyoriy bir tizimning ikki darajali holatlaridir.

Zarralarning bir vaqtning o‘zida barcha holatlarda bo‘lishi superpozitsiya deb ataladi. U kubitlarga asoslangan kvant hisob-kitoblarida faol foydalaniladi.

Zarrachalar yana bir ajoyib xususiyatga ega: ular kuzatilgunga qadar superpozitsiya holatida bo'ladi, lekin kimdir ularni kuzatishni boshlashi bilanoq ular mumkin bo'lgan qiymatlar to'plamida qutb qiymatini oladi - 0 yoki 1.

International Journal of Education, Social Science & Humanities.
Finland Academic Research Science Publishers
ISSN: 2945-4492 (online) | (SJIF) = 8.09 Impact factor
Volume-12| Issue-7| 2024 Published: |22-07-2024|

Kvant protsessorlari qubitning ikkita asosiy xususiyatidan foydalanadi: superpozitsiya va qutb qiymatlaridan birini olish qobiliyati - 0 yoki 1. Bu kvant operatsiyalarining asosidir.

Kvant chigallashuvi (quantum entanglement) kvant mexanikasidagi fundamental hodisa bo‘lib, ikki yoki undan ortiq kubit (yoki boshqa kvant tizimlari) shu qadar zich bog‘lanib qoladiki, bir kubitning holati ikkinchisining holatiga darhol ta’sir qiladi.

Kvant protsessorlarida bir nechta kubitlar mavjud. Masalan, 2022 yilda IBM 433 kubitli kompyuterni taqdim etdi. Ular bir-biri bilan o'zaro ta'sirlashganda, qo'shma superpozitsiya effekti paydo bo'ladi.

Kvant protsessoridagi har bir kubit superpozitsiyada bo'ladi, ammo endi uning kuzatuv paytidagi qiymati u o'zaro ta'sir qiladigan boshqa kubitlrga ham bog'liq.

Qubitlar ustida amallar

Qubitlar ustida bajariladigan barcha amallar ularning fizik xususiyatlarini, qubitlarning holatini va boshqa zarralar bilan bog'lanish darajasiniini o'lchash yoki o'zgartirishni o'z ichiga oladi.

Bu erda kvant dasturlash tillarida eng ko‘p ishlatiladigan amallar va ularning kubitlarga qanday ta'sir qilishlari keltirilgan:

1. Yagona o‘zgartirish amallari (Unitary operations). Masalan, Hadamar operatori (H) holatlarning super pozitsiyasini yaratadi | 0 ⟩ va | 1 ⟩, Shi (S) operatori esa holatga faza siljishini qo‘llaydi.
2. O‘lchash amali(Measurement). Kubitning o‘lchanishi holatning bazis holatlaridan birida tugashiga olib keladi. Ularni o‘lchash kubitning holati haqida ma’lumotni olish va ulardan keyingi hisoblashlarda foydalanish imkonini beradi.
3. Chigallashtirish amali (Entanglement). U ikki yoki undan ortiq kubit bir kubit holatining o‘zgarishi zudlik bilan ikkinchisiga aks etadigan tarzda korrellanadigan holatlarni yaratishga imkon beradi. Axborotni uzatish va kvant shifrlash va kvant teleportatsiyasi muammolarini hal qilish uchun foydalaniladi.
4. Boshqariladigan aylantirish amali (Controlled rotations). Muayyan shartlarni bajarishda kubitlarga unitar operatsiyalarni qo‘llash imkonini beradi. Masalan, CNOT operatori (boshqariladigan INKOR) NOT amalini maqsadli kubitga faqat boshqaruvchi kubit holati | 1 ⟩ bo‘lganda qo‘llaydi.
5. Dekogerentsiya amali va xatolarni tuzatish (Decoherence and error correction). Dekogerensiya — kvant tizimlarining ajralmas qismidir. Kvant

International Journal of Education, Social Science & Humanities.
Finland Academic Research Science Publishers
ISSN: 2945-4492 (online) | (SJIF) = 8.09 Impact factor
Volume-12| Issue-7| 2024 Published: |22-07-2024|

dasturlashtirish tillari hisoblarning aniqligini oshirish uchun xatolarni topish va tuzatish imkonini beradi.

Kvant dasturlash tillari

1.Q# (Q sharp)

Bu Microsoft tomonidan ishlab chiqilgan yuqori darajadagi kvant dasturlash tili. Q# boshqa kvant tillariga nisbatan bir qancha muhim afzalliklarga ega: siz unda C# kvant bo'lmagan dasturlash tilidan foydalanib dasturlar yozishingiz mumkin va Q# Microsoft Quantum Development Kit tomonidan ham qo'llab-quvvatlanadi.

Q# dasturlash tilida dastur kodiga misol:

```
namespace Quantum.Examples {
open Microsoft.Quantum.Primitive;
operation SuperpositionExample() : Result {
using (qubit = Qubit()) {
H(qubit);
return M(qubit); // Qubitni o'lchash va natijani qaytarish
}
}
}
```

Kodni o‘qiymiz:

1. namespace Quantum.Examples – nomlar maydonini aniqlash. Kvant hisoblar uchun barcha amallar va ma’lumot turlari Quantum.Examples nomlar maydonida bo‘ladi.
2. open Microsoft.Quantum.Primitive- open operatori Microsoft.Quantum.Primitive. nomlar maydonini import qiladi. U kvant hisoblash bilan ishlash uchun zarur bo'lgan asosiy amallar va funktsiyalarni o'z ichiga oladi.
3. operation SuperpositionExample (): Result – SuperpositionExample nomi bilan amalni aniqlash. Amal hech qanday argument qabul qilmaydi va Result qiymatini qaytaradi. Result turi kubitni o‘lchash natijasini taqdim etadi va 0 yoki 1 qiymatlarini qabul qilishi mumkin.
4. using (qubit = Qubit) — using konstruksiyasi kubit yaratish va foydalangandan keyin resurslarni avtomatik ravishda ozod qilish uchun ishlatiladi. Kubit qubit o'zgaruvchisiga joylashtiriladi.
5. H (qubit) — Hadamar amalini qubit kubitiga qo‘llaydigan H operatsiyasining chaqiruvidir. Hadamard amali $|0\rangle$ va $|1\rangle$ holatlarining superpozitsiyasini yaratadi, ya'ni qubit bir vaqtning o'zida ikkala holatda ham bo'lishi mumkin.

International Journal of Education, Social Science & Humanities.
Finland Academic Research Science Publishers
ISSN: 2945-4492 (online) | (SJIF) = 8.09 Impact factor
Volume-12| Issue-7| 2024 Published: |22-07-2024|

6. return M (qubit) — qubit kubitini o‘lchaydi va o‘lchash natijasini qaytaradi. M amali kubit holatini o‘lchaydi va kubit qaysi holatda o‘lchanganiga qarab 0 yoki 1 ni qaytaradi.

2. pyQuil

pyQuil — Rigetti Computing tomonidan ishlab chiqilgan Python kutubxonasi. pyQuil sizga supero'tkazgich kubitlarida kvant dasturlarini yaratish, simulyatsiya qilish va ishga tushirish imkonini beradi. U boshqa Python ilmiy kutubxonalari bilan birlashtirilgan.

pyQuil dastur kodiga misol:

```
from pyquil import Program
from pyquil.gates import H, CNOT
from pyquil.api import WavefunctionSimulator

# Kvant dasturini yaratamiz
p = Program()

# Hadamar amalini birinchi kubitda qo‘llaymiz
p += H(0)

# CNOT amalini ikkala kubitda ham qo'llaymiz
p += CNOT(0, 1)

# Simulyator yaratamiz
simulator = WavefunctionSimulator()

# Biz dasturni bajargandan so'ng to'lqin funksiyasini olamiz
wavefunction = simulator.simulate(p)

# Natijalarni chiqamiz
print(wavefunction)
```

Kodni tushuntiramiz:

1. from ... import ... – zarur modullarni pyQuil dan import qilamiz.

2. p = Program () — kvant dasturini ifodalovchi Program ob’yektini yaratamiz.

International Journal of Education, Social Science & Humanities.
Finland Academic Research Science Publishers
ISSN: 2945-4492 (online) | (SJIF) = 8.09 Impact factor
Volume-12| Issue-7| 2024 Published: |22-07-2024|

3. p + = H (0) — Birinchi kubitda Hadamar (H) amalini qo‘llaymiz. adamard amali qubitning superpozitsiyasini yaratadi, uni boshlang'ich holatidan |0> holatiga (|0> + |1>)/sqrt(2) aylantiradi.

4. p + = CNOT (0, 1) – ikki kubitda ham CNOT amalini qo‘llaymiz. CNOT amali ikkinchi kubitni faqat birinchi kubit |1> holatida bo'lganda invertatsiya qiladi.

5. simulator = WavefunctionSimulator () – dasturni amalga oshirish imkonini beradigan WavefunctionSimulator ob’ektini yaratmoqdamiz.

6. wavefunction = simulator.simulate - biz dasturning bajarilishini simulyatsiya qilamiz va bajarilgandan so'ng to'lqin funksiyasini olamiz.

7. print (wavefuncion) - tolqin funktsiyasini chop etish.

3. Qiskit

Qiskit - bu kvant algoritmlarini ishlab chiqish va bajarish uchun IBM tomonidan ishlab chiqilgan ochiq kodli dasturiy ta'minot kutubxonasi. Ochiq kodli kod va faol ishlab chiquvchilar hamjamiyatiga ega Qiskit kvant dasturlashni o'rganish va qo'llash uchun keng ko'lamli vositalar va resurslarni taqdim etadi. Qiskit, shuningdek, Python kabi boshqa dasturlashtirish tillari bilan integratsiyalashish imkoniyatini beradi, bu esa uni u bilan allaqachon tanish bo‘lgan ishlab chiquvchilar uchun qulayroq qiladi.

Qiskit dastur kodiga misol:

```
from qiskit import QuantumCircuit, execute, Aer
# 2 kubit va 2 klassik bitli kvant sxemasini yaratamiz
qc = QuantumCircuit(2, 2)

qc.h(0)   # Hadamar amalini birinchi kubitda qo‘llaymiz
qc.cx(0, 1)  # CNOT amalini ikkala kubitda ham qo'llaymiz
# Qubitlarning holatini klassik bitlarga o'lchash
qc.measure([0,1], [0,1])
# Simulyatorda sxemani bajarish
simulator = Aer.get_backend('qasm_simulator')
job = execute(qc, simulator, shots=1000)
result = job.result()
# Natijalarni chop etamiz
counts = result.get_counts(qc)
print(counts)
```

Dastur nima qiladi:

International Journal of Education, Social Science & Humanities.
Finland Academic Research Science Publishers
ISSN: 2945-4492 (online) | (SJIF) = 8.09 Impact factor
Volume-12| Issue-7| 2024 Published: |22-07-2024|

1. from ... import ... – zarur modullarni Qimkitdan import qilamiz.

2. qc = (2, 2) — ikkita kubit va ikkita klassik bitga ega bo‘lgan ob’yektni yaratamiz.

3. qc.h (0) — birinchi kubitda Hadamar (H) amalini qo‘llaymiz. Hadamard amali qubitning dastlabki |0> holatining superpozitsiyasini yaratadi va uni (|0> + |1>)/sqrt(2) ga aylantiradi.

4. qc.cx (0, 1) – ikki kubitda ham CNOT (cx) amalini qo‘llanamiz. CNOT amali faqat birinchi kubit |1> holatida bo'lsa, ikkinchi kubitni o'zgartiradigan transformatsiyani qo'llaydi.

5. qc.measure ([0,1], [0,1]) — kubitlarning holatini o‘lchaymiz va natijalarni klassik bitlarga saqlaymiz.

6. simulator = Aer.get_backend (’qasm _ simulator’) - kvant simulyatorini ishga tushiramiz.

7. job = execute (qc, simulator, shots = 1000) — simulyatordagi sxemani bajarib, takrorlash sonini 1000 ga teng deb ko‘rsatamiz.

8. result = job.result () - simulyatsiya natijalarini saqlaymiz.

9. counts = result.get_counts (qc)//print (counts) — simulyatsiya natijalarini olib, ularni chiqaramiz.

Xulosa

Agar siz kvant dasturlashda o'z kuchingizni sinab ko'rishga qaror qilsangiz, yoqorida keltirilgan tillardan bittasini tanlashingiz va amalda qo’llab ko‘rishingiz mumkin.

FOYDALANILGAN ADABIYOTLAR:

1. Химено-Сеговиа, Хэрриган, Джонстон: Программирование квантовых компьютеров. Базовые алгоритмы и примеры код Издательство: Питер, 2021 г., ID 758841 ISBN: 978-5-4461-1531-0, 336 стр.

2. Гузик В.Ф., Гушанский С.М., Потапов В.С. Оценка сложности квантовых алгоритмов // Технологии разработки информационных систем. Сборник статей Международной научно-практической конференции, 01.09.2014, г. Геленджик, стр. 81-85;

3. В. М. Соловьев: Информатика. 2015. Т. 15, вып. 4, Научный отдел информатика,УДК 519.688 квантовые компьютеры и квантовые алгоритмы. Часть 1. Квантовые компьютеры