Quantum search:
Algoritmul lui Grover

Milan MUJDAR (333)
Iulia TAMAȘ (333)
Andrei-Ioan LEGIAN (333)

Cuprins

Descriere

Algoritmul lui Grover constituie un important punct de referință în domeniul calculului cuantic, fiind semnificativ mai avantajos în comparație cu metodele clasice de căutare într-o bază de date nesortată. Propus în 1996 de Lov Grover, algoritmul permite identificarea unui element țintă într-un spațiu de căutare de dimensiune N în timp $O(\sqrt{N})$, comparativ cu algoritmii clasici, ce necesită $O(N)$. Acest progres este esențial în vederea rezolvării eficiente a problemelor de căutare unde nu există o structură ce ar putea fi exploatată algoritmic, iar soluția trebuie identificată prin testarea tuturor opțiunilor posibile.

La baza algoritmului stau câteva principii fundamentale ale mecanicii cuantice, în special cea de superpoziție și de interferență. Algoritmul lui Grover utilizează o funcție oracol care indică dacă o anumită superpoziție este soluția căutată, aplicând apoi procesul iterativ de amplificare a amplitudinii. Această procedură are rolul de a crește progresiv probabilitatea de a măsura starea dorită, profitând de comportamentul cuantic pentru a favoriza soluțiile corecte în detrimentul celorlalte variante.

Aplicabilitate și limitări

Datorită caracterului său general, algoritmul lui Grover are aplicații extinse în domenii variate, reușind să acopere majoritatea problemelor de tip NP-complete. Printre acestea se numără factorizarea numerelor întregi, problemele de satisfiabilitate booleană (SAT), problema comis-voiajorului (TSP), detecția coliziunilor în funcții hash și inversarea funcțiilor criptografice.

Totuși, progresele nu sunt atât de mari încât să ridice semne de întrebare serioase referitor la securitatea cibernetică: de exemplu, funcția SHA256, de la un număr de 2^{256} pași va fi redusă la $\frac{\pi}{4}2^{128}$ pași, iar AES-128 va avea rezistența efectivă a unei chei de 64 de biți, implicit 128 pentru AES-256.

Totodată, necesitatea unui număr mare de qubiți face aproape imposibilă utilizarea într-un mediu real al acestui tip de algoritm, calculatoarele cuantice actuale nefiind suficient de avansate. Deși oferă un avantaj semnificativ pentru problemele cu un spațiu mare de căutare, eficientizarea algoritmului în cazul seturilor de date mai restrânse nu rentează. În același timp, la fel ca orice alt algoritm cuantic, algoritmul lui Grover poate fi susceptibil la erori provocate de noise și decoherence.

Pasul 1 - Pregătirea stării inițiale: algoritmul Hadamard

Primul pas al algoritmului lui Grover reprezintă pregătirea stării inițiale, ce implică aducerea sistemului cuantic într-o superpoziție uniformă a tuturor stărilor posibile. Pentru spațiul de căutare $N = 2^n$ elemente, unde n este numărul de qubiți, sistemul este inițializat în starea $|0>^{\otimes n}$. Această stare reprezintă configurația în care toți qubiții se află în starea de bază.

Ulterior, asupra fiecărui qubit se aplică independent poarta Hadamard, notată în general cu H. Pentru un sistem compus din n qubiți aplicarea transformării Hadamard rezultă o transformare tensorială notată $H^{\otimes n}$. Astfel, starea inițială $|0>^{\otimes n}$, care corespunde vectorului canonic $|0 \ldots 0>$, se transformă într-o superpoziție uniformă a tuturor celor 2^n stări posibile ale spațiului de căutare:

$$H^{\otimes n}|0>^{\otimes n} = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x> \ .$$

Superpoziția rezultată constituie fundamentul pe care algoritmul lui Grover construiește avantajul cantitativ. Prin distribuirea amplitudinilor în mod egal asupra întregului spațiu de căutare se creează condițiile necesare pentru amplificarea ulterioară a stării soluției. Acest pas poate fi interpretat ca o "inițializare" a întregului domeniu, permițând algoritmului să exploreze simultan toate posibilitățile în paralel.

Pasul 2 - Aplicarea Oracolului U_f: marcarea soluției prin inversarea fazei

În această etapă a algoritmului lui Grover, obiectivul este să identificăm soluția (sau soluțiile) dorită(e) dintr-un spațiu de căutare de dimensiune 2^n, marcându-le într-un mod care va permite amplificarea probabilității lor în pasul următor. Acest lucru se realizează cu ajutorul unei componente fundamentale: **Oracolul**, notat Uf.

Scopul oracolului:

Oracolul implementează o funcție booleană:

$$f: \{0,1\}^n \rightarrow \{0,1\}$$

- $f(x) = 1$, dacă x este soluția caută ($x = x_{target}$)
- $f(x) = 0$, altfel

În cadrul algoritmului, oracolul aplică transformarea:

$$U_f|x> = (-1)^{f(x)}|x>$$

Aceasta înseamnă că **doar starea soluție** $|x_{target}>$ își schimbă semnul (faza), adică:

$$U_f|x_{target}>=-|x_{target}>$$

Iar toate celelalte stări rămân neschimbate.

Ce înseamnă "inversarea fazei" ?

Inversarea fazei nu modifică probabilitatea de măsurare, dar joacă un rol esențial în **interferența cu amplitudinile celorlaltor stări** în pasul următor (difuzia). Prin schimbarea semnului **doar pentru soluția corectă**, o eventuala reflexie față de media amplitudinilor va avea ca efect **amplificarea amplitudinii** acesteia.

Pasul 3 - Aplicarea Oracolului de Difuzie U_s: amplificarea probabilității soluției corecte

După ce oracolul U_f a marcat soluția corectă printr-o **inversare de fază**, următorul pas al algoritmului lui Grover este să amplificăm probabilitatea acesteia de a fi măsurată. Această amplificare se realizează prin **operatorul de difuzie**, notat U_s, care execută o **reflexie față de starea medie**.

Ce face operatorul de difuzie?

Operatorul de difuzie aplică transformarea:

$$U_s = 2|\psi><\psi| - I$$

unde $|\psi>=H^{\otimes n}|0>^{\otimes n}$ este **starea uniformă**, adică superpoziția egală a tuturor stărilor de bază. Geometric, U_s reflectă toate stările în jurul vectorului medie (adică în jurul stării uniforme). Deoarece soluția a fost "trasă în jos" de oracol (prin faza negativă), reflexia va "împinge" acea amplitudine mai sus decat media. Cu alte cuvinte: **inversarea fazei** + **reflexia fata de medie** = **creșterea probabilității soluției.**

De ce se folosește $|0>$ în construcția acestui operator ?

Pentru că starea uniformă $|\psi>$ este de fapt:

$$|\psi>=H^{\otimes n}|0>^{\otimes n}$$

Astfel, pentru a reflecta fata de $|\psi>$, este echivalent să:

- trecem în baza Hadamard;
- reflectăm față de $|0>$;
- ne întoarcem înapoi.

Efectul pașilor 2 și 3:

După fiecare aplicare a combinației $U_s\ U_f$, amplitudinea soluției crește, iar a celorlaltor scade. După aproximativ $\frac{\pi}{4} * \sqrt{N}$ iterații (unde $N = 2^n$), probabilitatea soluției ajunge foarte aproape de 1.

Pasul 4 - Repetarea pașilor 2 și 3: Iterațiile algoritmului Grover

Această secțiune construiește circuitul principal al algoritmului Grover.

Inițial se creează o suprapunere uniformă a tuturor stărilor posibile, aplicând porti Hadamard pe toți cei n qubiți. Astfel se formează starea:

$$|s>=\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x>,$$

unde $N = 2^n$ este dimensiunea totală a spațiului de căutare. Pentru 8 qubiți rezultă 256 stări posibile.

După actualizare se construiește:

- **Oracolul**, care identifică starea ținta și aplică o inversare de fază doar asupra acesteia.
- **Operatorul de difuzie**, care amplifică amplitudinea stării ținta prin reflectarea amplitudinilor față de medie.

Numărul optim de iterații Grover este calculat cu formula:

$$\left[\frac{\pi}{4}\sqrt{\frac{N}{s}}\right]$$

unde “s” reprezinta numarul de solutii a problemei (în cazul nostru = 1)

Această valoare asigură probabilitatea maximă de succes la măsurare.

În cadrul fiecărei iterații, se aplică mai întâi oracolul, urmat de operatorul de difuzie. Cu fiecare repetare probabilitatea de a obține starea țintă crește semnificativ.

Pasul 5 - Măsurarea și simularea circuitului

După finalizarea tuturor iterațiilor Grover, fiecare qubit este măsurat în registrul clasic corespunzător. În Qiskit, această măsurare transferă starea cuantică într-o stare clasică binară (de exemplu: 10101010), care poate fi analizată.

Pentru execuția circuitului se utilizează AerSimulator, un simulator ideal (fără zgomot), parte a pachetului qiskit-aer. Circuitul este tradus și compilat (“transpile”) înainte de rulare pentru a fi optimizat în funcție de backend-ul ales (simulatorul), prin operații precum reordonarea qubiților sau simplificarea porților.

Circuitul este executat de 1024 de ori ($shots - 1024$) pentru a obține o distribuție statistică a rezultatelor. Astfel, se poate observa frecvența fiecărei stări măsurate. Dacă

algoritmul a fost implementat corect, starea ținta ('10101010') va apărea cu o frecvență foarte mare, indicând succesul căutării cuantice.