Universitatea “POLITEHNICA” Bucureşti
Facultatea de Automatică şi Calculatoare

Noi abordări în tehnica de calcul şi teoria informaţiei

Calcul Cuantic

Doctorand: **Lucian Dragne**
Îndrumător: **Prof. dr. ing. Florica Moldoveanu**,
Catedra de Calculatoare, Facultatea de Automatică şi Calculatoare, UPB

1. Concepte fundamentale

Informatica şi calculul cuantic reprezintă studiul proceselor informatice care pot fi realizate folosind sisteme ce se supun legilor mecanicii cuantice, aşa cum sunt ele formulate în prezent. Totuşi, extinderea tehnicilor de procesare cuantică la scară largă a informaţiei rămân în continuare o provocare atât pentru oamenii de ştiinţă, cât şi pentru ingineri. Metodele clasice de implementare a sistemelor de calcul încep să se lovească de barierele impuse de miniaturizarea din ce în ce mai mare a componentelor electronice, care se apropie de dimensiunile cuantice.

Una din soluţiile propuse pentru rezolvarea acestor dificultăţi este aceea de a schimba paradigma de calcul. O astfel de nouă paradigmă este oferită de teoria informaticii cuantice care se bazează pe folosirea principiilor ne-intuitive ale mecanicii cuantice pentru efectuarea calculelor, în locul folosirii sistemelor fizice clasice. S-a demonstrat că, în timp ce orice calculator clasic actual poate fi folosit pentru simularea unui calculator cuantic, această simulare nu se poate face eficient (adică prin mărirea costurilor de timp cu o valoare dependentă de intrare în mod cel mult polinomial) [1]. Astfel, calculatoarele cuantice oferă un avantaj semnificativ în viteza de prelucrare. Acest avantaj este aşa de mare încât unii specialişti sunt de părere că, indiferent de viitoarea dezvoltare a calculatoarelor clasice, ele nu vor putea niciodată să egaleze calculatoarele cuantice.

1.1. Teza Church – Turing

Bazele teoretice ale calculatoarelor clasice au fost puse de Alan Turing, care a demonstrat că se poate construi o *Maşină Turing Universală*, capabilă să simuleze orice altă Maşină Turing. Mai mult chiar, el a postulat că Maşina Turing Universală poate implementa orice proces definit prin acţiuni algoritmice:

Orice proces algoritmic poate fi simulat folosind o Maşină Turing.

Pornind de la observaţia că Maşina Turing nu numai că poate implementa orice proces algoritmic, ci mai mult, poate face acest lucru în mod eficient, s-a formulat varianta *tare* a tezei Church – Turing:

Orice model de calcul poate fi simulat folosind o Maşină Turing probabilistă prin adăugarea a cel mult un număr polinomial de operaţii.

1.2. Teza Church – Turing – Deutsch

Pornind de la aceste întrebări, David Deutsch a încercat să găsească o metodă de a deduce o variantă şi mai tare a tezei Church – Turing, pornind de legile fizice**Error! Reference source not found.**. Mai exact, Deutsch a încercat să găsească un model computaţional care să fie capabil să simuleze orice sistem fizic [15]. În acest fel, orice maşină de calcul implementata pe baza legilor acestui sistem fizic va putea fi simulat în acest model. Pornind de la presupunerea larg (dar nu pe deplin) acceptată în comunitatea ştiinţifică, că legile fizice sunt în ultimă instanţă cuantice, Deutsch s-a orientat spre maşinile de calcul bazate pe principiile mecanicii cuantice. Aceste maşini sunt analogele în domeniul cuantic al maşinilor din domeniul clasic, considerate cu jumătate de secol în urmă de către Turing. Ele constituie baza a ceea ce în termeni moderni se denumeşte *calculator cuantic*. Ceea ce modelul de calcul al lui Deutsch a permis, a fost să lanseze o provocare pentru varianta tare a tezei Church – Turing. Deutsch a demonstrat că este posibil ca un calculator cuantic să rezolve eficient probleme de calcul care nu au soluţie eficientă cunoscută până în prezent, pe un calculator clasic – modelat de o Maşină Turing probabilistă. Demonstraţia lui Deutsch este bazată pe un exemplu simplu care sugerează că într-adevăr calculatoarele cuantice pot să aibă putere de

calcul mai mare decât calculatoarele clasice. În conformitate cu aceste rezultate, teza Church – Turing – Deutsch s-ar formula astfel:

Orice proces fizic algoritmic poate fi simulat eficient folosind o Maşină Cuantică.

Aceste prime rezultate obţinute de Deutsch au fost îmbogăţite de altele în deceniile care au urmat. Poate printre cele mai importante se numără rezultatul obţinut în 1994 de către Peter Shor, care demonstrează că două probleme foarte importante [45], care, ca şi problema lui Deutsch, nu au soluţii eficiente cunoscute pe Maşina Turing, pot fi rezolvate eficient pe un calculator cuantic [46]. Aceste două probleme, care se bazează pe transformarea Fourier discretă sunt: problema calculului factorilor primi ai unui număr şi problema logaritmului discret. Un alt rezultat care demonstrează puterea calculatoarelor cuantice a fost obţinut în 1995 de Lov Grover [28], care a demonstrat că o altă problemă importantă – căutarea într-un spaţiu ne-structurat – poate fi de asemenea fi rezolvată mai rapid pe un calculator cuantic decât pe un calculator clasic. Deşi îmbunătăţirea vitezei nu este aşa de mare ca în cazul problemei numerelor prime, larga dependenţă a sistemelor actuale de tehnicile de căutare a provocat atenţia asupra acestui algoritm. Cam în acelaşi timp, alte grupuri de cercetători se concentrau asupra dezvoltării unei idei introduse de Richard Feynman în 1982. Feynman a arătat că există dificultăţi esenţiale în simularea sistemelor mecanice cuantice pe calculatoarele clasice şi a sugerat că dezvoltarea unor calculatoare cuantice ar permite eliminarea acestor dificultăţi. Această sugestie s-a dovedit între timp a fi adevărată. [5]

1.3. Teoria cuantică a informaţiei

Teoria comunicaţiei a căror baze au fost puse de Claude Shannon [35] reprezintă un alt aspect care ar trebui modificat în conformitate cu această nouă abordare cuantică a informaţiei. Realizarea cheie care este probabil cea mai importantă în teoria lui Shannon este aceea de a defini matematic conceptul de informaţie. Shannon a demonstrat două teoreme fundamentale care au constituit punctele de plecare pentru dezvoltările ulterioare din acest domeniu:

- teorema de transmisie a informaţiei printr-un canal de comunicaţie ideal (fără zgomot): cuantifică resursele necesare stocării informaţiei emise de o sursă.
- teorema de transmisie a informaţiei printr-un canal de comunicaţie real (cu zgomot): cuantifică informaţia care poate fi transmisă printr-un canal cu zgomot. Pentru a avea o transmisie sigură în prezenţa zgomotului, Shannon a demonstrat că pot fi folosite coduri de corectare a erorilor pentru protejarea informaţiei.

Teoria cuantică a informaţiei se dezvoltă urmând aceleaşi procedee. Ben Schumacher a oferit analogul cuantic al primei teoreme a lui Shannon – transmisia informaţiei printr-un canal fără zgomot, definind noţiunea de bit cuantic ca o resursă fizică. Dar, spre deosebire de teoria clasică a informaţiei, nu s-a descoperit încă varianta cuantică a celei de-a doua teoreme a lui Shannon – transmisia informaţiei printr-un canal cu zgomot. Cu toate acestea, există o teorie cuantică a corecţiei erorilor cu ajutorul căreia s-a permis dezvoltarea calculatoarelor cuantice operabile în prezenţa zgomotului. Folosind ideile clasice din teoria corecţiei erorilor se pot proteja stările cuantice de efectele zgomotului. De asemenea, această teorie permite transmisia sigură a informaţiei printr-un canal de comunicaţie cuantic cu zgomot.

În 1992, Charles Bennett şi Stephen Wiesner au demonstrat o altă aplicaţie a teoriei cuantice a informaţiei, şi anume transmisia informaţiei clasice printr-un canal de comunicaţie cuantic. Prin acest rezultat, denumit în literatura de specialitate „codificare supra-densă", se arată cum se pot transmite doi biţi de informaţie clasici prin trimiterea unui singur qubit de la sursă la destinaţie. Studiul teoriei informaţiei începe prin considerarea unui singur canal de comunicaţie. Acest fapt însă s-a generalizat, în prezent existând deja o teorie bine definită a reţelelor informatice. Tot ca o problemă deschisă, rămasă deocamdată fără răspuns, trebuie menţionată şi găsirea unor metode de interconectare a mai multor calculatoare în reţele cuantice.

1.4. Criptografia cuantică

Cel mai utilizate sisteme criptografice sunt în prezent cele bazate pe criptarea cu cheie publică. Ideea de bază pentru transmiterea secretă a mesajelor este aceea de a folosi funcţiile matematice greu inversabile. Funcţionarea acestor sisteme se bazează pe presupunerea că mesajul criptat de către transmiţător folosind o cheia publică nu poate fi decriptat în mod eficient decât de posesorul cheii secrete corespunzătoare. De asemenea, trebuie ca deducerea cheii secrete din cheia publică să nu poată fi făcută eficient. Multe din sistemele de criptare cu cheie publică (între care şi RSA) folosesc drept funcţii matematice greu inversabile, probleme legate de descompunerea în factori primi a numerelor naturale. S-a dovedit însă că aceste probleme nu sunt deloc greu rezolvabile folosind calculul cuantic. Aşadar, în eventualitatea construirii unui calculator cuantic aceste sisteme de criptare cu cheie publică devin nesigure.

O altă categorie de sisteme de criptare sunt cele care folosesc chei secrete. Sistemele clasice din această categorie s-au lovit însă de problema distribuţiei cheilor. Problema este cum se pot transmite aceste chei fără ca ele să fie compromise. O rezolvare a acestei probleme a fost oferită prin folosirea unui canal de comunicaţie cuantic. Această procedură de transmisie a cheilor secrete se numeşte distribuţie cuantică a cheilor, sau criptografie cuantică. Ideea care stă la baza acestor sisteme este folosirea principiului de observare din mecanica cuantică: observarea unui sistem cuantic conduce inevitabil la perturbarea sa. Conform acestui principiu aşadar, orice încercare de observare a unei chei transmisă printr-un canal cuantic poate fi descoperită de către receptorul cheii. Acelaşi principiu care conferă puterea acestor sisteme, ridică însă şi cea mai mare problemă de implementare. Aceste canale cuantice de comunicaţie nu pot fi prelungite prin folosirea aşa numitelor repetoare de semnal, aşa cum sunt ele înţelese în mod clasic. Cu toate acestea însă, prototipuri experimentale a acestor sisteme criptografice cuantice au intrat deja în sfera comercială.

1.5. Reprezentarea cuantică a informaţiei

Bitul reprezintă conceptul fundamental în teoria clasică a informaţiei şi al calculului. Teoria cuantică a calculului si teoria cuantică a informaţiei sunt construite pe baza unui concept fundamental asemănător: bitul cuantic, sau pe scurt qubit [36]. Făcând abstracţie de implementarea fizică a qubiţilor, ei se definesc ca entităţi matematice abstracte. Aşa cum un bit clasic este definit printr-o stare care poate avea valoarea 0 sau 1, un qubit este definit şi el printr-o stare cuantică. Folosind notaţia Dirac din mecanica cuantică, două stări posibile ale unui qubit, corespunzătoare celor două stări ale unui bit clasic, sunt $|0\rangle$ şi $|1\rangle$ – denumite stări computaţionale de bază. Un qubit se poate afla în orice stare normată, definită ca o superpoziţie liniară complexă de stări computaţionale de bază: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$,

unde α şi β sunt numere complexe, satisfăcând relaţia de normare: $|\alpha|^2 + |\beta|^2 = 1$

Cu alte cuvinte, starea unui qubit este reprezentată printr-un vector normat într-un spaţiu vectorial complex bidimensional, în care cele două stări computaţionale de bază formează o bază ortonormată. Prin măsurarea qubitului se poate obţine rezultatul 0 cu probabilitatea $|\alpha|^2$, sau rezultatul 1, cu probabilitatea $|\beta|^2$. În mod natural, se observă că pentru a respecta principiul din teoria probabilităţii conform căruia suma probabilităţilor evenimentelor posibile trebuie să fie 1, vectorul de stare trebuie să fie normat. În mod aparent paradoxal, α şi β, chiar cu restricţia de normare impusă, pot lua o infinitate de valori complexe. Deci, s-ar părea că un qubit reprezintă o cantitate de informaţie infinită. Această concluzie însă nu este corectă, deoarece măsura care interesează este cantitatea de informaţie ce poate fi observată. Prin măsurarea qubitului se poate obţine doar una din cele două valori; mai mult, după

măsurare starea qubitului se schimbă. Aşadar, efectuând o singură măsurare asupra unui qubit se obţine un singur bit de informaţie despre starea qubitului. Determinarea exactă a stării qubitului, adică determinarea exactă a valorilor α şi β, se poate face numai dacă s-ar putea efectua o infinitate de măsurători asupra unei infinităţi de qubiţi preparaţi identic. Generalizarea la reprezentarea sistemelor pe mai mulţi qubiţi se face folosind produsul tensorial între spaţii vectoriale. Starea unui sistem format din n qubiţi, fiecare qubit fiind aflat în starea $|\psi_i\rangle$, este reprezentată prin vectorul normat: $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes ..... \otimes |\psi_n\rangle$

1.6. Realizări practice ale sistemelor de calcul cuantice

Problema fundamentală care a fost ridicată este dacă există vreun principiu care să împiedice construcţia fizică a calculatoarelor cuantice. S-au identificat două posibile astfel de piedici:

- funcţionarea în mediul real poate fi diferită de cazul ideal din cauza zgomotului
- mecanica cuantică a căror principii stau la baza acestui domeniu poate fi o descriere incompletă a realităţilor fizice

Zgomotul reprezintă o piedică fundamentală în funcţionarea sistemelor de procesare a informaţiei, de care nu se poate face abstracţie. În acest sens, s-a demonstrat că presupunând că zgomotul poate fi redus sub o anume valoare prag, corectarea cuantică a codurilor poate fi folosită pentru a diminua şi mai mult zgomotul; practic acest procedeu putând fi repetabil la infinit fără a încărca semnificativ complexitatea calculului efectiv. La scară mică, de câţiva qubiţi, s-au realizat deja sisteme fizice de prelucrare cuantică a informaţiei.

1.7. Calcul cuantic – porţi cuantice

Calculul cuantic studiază transformările care se efectuează asupra stărilor cuantice. La fel cum un calculator clasic este construit din circuite electronice conţinând porţi logice conectate între ele, un calculator cuantic este alcătuit din circuite cuantice conţinând porţi cuantice inter-conectate. Formalismul matematic folosit în descrierea transformărilor aplicate de circuitele cuantice asupra qubiţilor este cel al operatorilor liniari. Aşadar starea unui qubit este reprezentată printr-un vector într-un spaţiu vectorial complex bidimensional, în timp ce un circuit cuantic acţionând asupra qubitului respectiv este reprezentat printr-un operator liniar unitar definit pe acel spaţiu vectorial. Orice circuit cuantic este reversibil: cunoscându-se starea qubiţilor de la ieşirea unui circuit, este întotdeauna posibil a se determina starea lor de la intrarea circuitului. Această proprietate nu este respectată de toate circuitele clasice: spre exemplu poarta NAND. Cu toate astea, circuitele clasice nu au capacitate de calcul sporită prin faptul că permit şi altfel de porţi decât cele reversibile. Există de exemplu o poartă reversibilă pe trei biţi, numită poartă Toffoli, cu ajutorul căreia se poate implementa orice circuit clasic. Poarta Toffoli inversează cel de-al treilea bit (numit şi bit rezultat) dacă şi numai dacă primii doi biţi (numiţi şi biţi de control) sunt setaţi. Poarta CNOT inversează qubitul rezultat dacă şi numai dacă qubitul de control este setat. Altfel spus, poarta CNOT implementează adunarea modulo 2 pe doi biţi. Un circuit cuantic care implementează adunarea modulo 4 pe 2 biţi: $|x_1 x_0 y_1 y_0\rangle \rightarrow |x_1 x_0\rangle |(x_1 x_0 + y_1 y_0) \bmod 4\rangle$, $x_1, x_0, y_1, y_0 \in \{0,1\}$

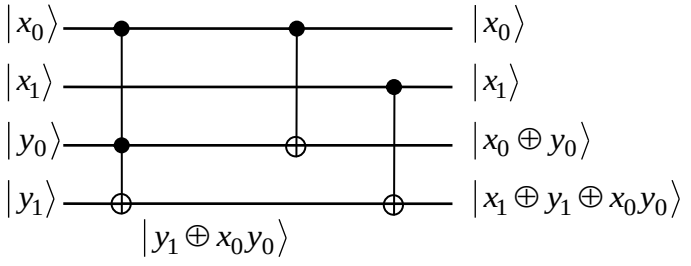


Figura 1. Circuitul cuantic care implementează adunarea modulo 4

Trebuie menţionat că măsurarea unuia, sau a mai multor qubiţi nu este o operaţie unitară. Deşi este necesar ca uneori să se efectueze măsurători asupra qubiţilor unui circuit cuantic, o măsurătoare nu este o poartă cuantică. După efectuarea unei măsurători starea qubitului, sau a qubiţilor respectivi nu mai are nici o relevanţă; ceea ce contează este informaţia obţinută din rezultatele măsurătorii – informaţie codificată sub formă de biţi clasici probabilistici. Liniile de legătură din reprezentarea circuitelor cuantice nu au semnificaţia unor legături fizice (fire conductoare de curent, sau altceva de acest gen), ci ele reprezintă durata de viaţă a unui qubit de-a lungul timpului. Circuitele cuantice sunt astfel întotdeauna aciclice.

1.8. Imposibilitatea de copiere a unui qubit

În calculul clasic, există un circuit foarte simplu care realizează copierea oricărui bit.

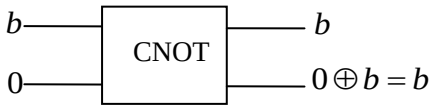


Figura 2. Circuitul clasic de copiere al unui bit

Una din deosebirile cele mai frapante între circuitele cuantice şi cele clasice este faptul că este imposibil de construit circuit cuantic care să copieze cu exactitate orice stare a unui qubit. Acesta este enunţul teoremei de non-clonare a unui qubit. Se pot construi numai circuite care să copieze cu exactitate două stări ortonormate ale unui qubit.

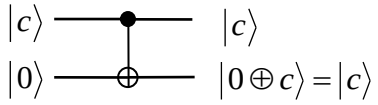


Figura 3. Circuitul cuantic de copiere a stărilor ortonormate $|0\rangle$ sau $|1\rangle$

Chiar dacă se acceptă implementarea circuitului de copiere folosind şi transformări ne-unitare se constată că se pot copia doar stări ortogonale. S-a demonstrat că un circuit de copiere cuantic pentru stări ne-ortogonale poate fi creat numai dacă se acceptă copii ne-exacte.

2. Eficienţa calculului cuantic

2.1. Paralelism cuantic

Paralelismul cuantic este o caracteristică fundamentală a multor algoritmi cuantici. Paralelismul generalizat la funcţii pe un număr arbitrar de biţi, foloseşte operaţia de transformare Walsh-Hadamard, o compunere de n porţi Hadamard care acţionează in paralel pe n qubiţi. Prin generalizare, după aplicarea transformării Walsh-Hadamard pe n qubiţi, toţi aflaţi iniţial in starea $|0\rangle$, se obţine: $|0\rangle^{\otimes n} \xrightarrow{H^{\otimes n}} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle$. Asta înseamnă că, transformarea Walsh-Hadamard produce o superpoziţie egală de toate stările computaţionale de bază. În plus, face asta foarte eficient, producând o superpoziţie de 2^n stări folosind numai n porţi. Evaluarea cuantică paralelă a unei funcţii $f:\{0,1\}^n \mapsto \{0,1\}$ poate fi efectuată astfel. Se prepară o stare formată din $n+1$ qubiţi: $|0\rangle^{\otimes n}|0\rangle$, apoi se aplică transformarea Hadamard pe primii n qubiţi, urmată de circuitul cuantic implementând U_f. Aceste transformări produc stările: $|0\rangle^{\otimes n}|0\rangle \xrightarrow{H^{\otimes n}} \left[\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle\right]|0\rangle \xrightarrow{U_f} \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle$

Într-un sens, paralelismul cuantic permite evaluarea simultană a tuturor valorilor posibile ale lui f, cu toate că, aparent, funcţia s-a evaluat numai o singură data. Totuşi, acest fel de paralelism nu este imediat şi folositor. Măsurarea stării $\sum_{x\in\{0,1\}^n}|x\rangle|f(x)\rangle$ întoarce numai $f(x)$ pentru o singură valoare a lui x. Bineînţeles că un calculator clasic poate face acelaşi lucru foarte uşor. Calculul cuantic are nevoie de ceva mai mult decât simplul paralelism cuantic pentru a putea fi folositor; are nevoie în plus si de posibilitatea de a *extrage informaţia* referitoare la mai multe valori ale lui $f(x)$ din stările compuse ca $\sum_{x\in\{0,1\}^n}|x\rangle|f(x)\rangle$.

2.2. Algoritmul lui Deutsch

Acesta demonstrează cum circuitele cuantice pot depăşi în performanţe circuitele clasice. Algoritmul lui Deutsch combină paralelismul cuantic cu o altă proprietate a mecanicii cuantice, cunoscută sub numele de *interferenţă*. Circuitul care implementează acest algoritm:

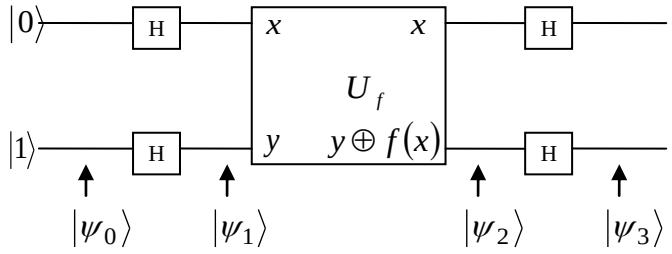


Figura 4. Circuit cuantic care implementează algoritmul lui Deutsch

Succesiunea stărilor începe cu starea computaţională de bază pe 2 qubiţi: $|\psi_0\rangle = |0\rangle|1\rangle$. Porţile finale conduc la $|\psi_3\rangle = \pm|f(0)\oplus f(1)\rangle|1\rangle$. Prin măsurarea primului qubit se poate determina direct suma modulo 2, $f(0)\oplus f(1)$. Deci, acest circuit cuantic oferă posibilitatea de a determina o *proprietate globală* a funcţiei $f(x)$, $f:\{0,1\}\mapsto\{0,1\}$, şi anume $f(0)\oplus f(1)$, prin efectuarea *unei singure evaluări* $f(x)$. Asta este de două ori mai rapid decât este posibil cu o maşină clasică, care ar avea nevoie de cel puţin două evaluări: $f(0)$ şi $f(1)$.

2.3. Algoritmul Deutsch-Jozsa

Algoritmul anterior este un caz simplu al unui algoritm cuantic mai general, algoritmul Deutsch-Jozsa care rezolvă următoarea problemă. Se consideră o funcţie $f:\{0,1\}^n \mapsto \{0,1\}$ despre care se ştie că este ori constantă, ori balansată. Se caută un algoritm determinist cât mai eficient care să decidă tipul funcţiei. Schema algoritmului clasic determinist este:

1. alege $x\in\{0,1\}^n$
2. iniţializează $A=\{0,1\}^n-\{x\}$; iniţializează $B=\{x\}$; iniţializează $y_0 = f(x)$
3. dacă $|B|>2^{n-1}$ atunci întoarce „ f constantă”
4. alege $x\in A$ şi calculează $y=f(x)$
5. dacă $y\neq y_0$ atunci întoarce „ f balansată”
6. $y_0 \leftarrow y$; $A\leftarrow A-\{x\}$; $B\leftarrow B\cup\{x\}$
7. reia de la pasul 3.

În cazul cel mai nefavorabil (f este constantă) algoritmul are complexitatea $O(2^{n-1}+1)$.

2.3.1. Problema Deutsch-Jozsa probabilistă

Se consideră o funcţie $f:\{0,1\}^n \mapsto \{0,1\}$ despre care se ştie că este ori constantă, ori balansată. $\forall \varepsilon > 0$, probabilitate de eroare, se caută un algoritm probabilist cât mai eficient care să decidă tipul funcţiei cu probabilitatea $1-\varepsilon$.
Algoritmul clasic probabilist care rezolvă această problemă este asemănător cu corespondentul său clasic determinist. O diferenţă importantă este modul în care se aleg elementele $x \in A$:

- în cazul determinist, elementele se aleg la
- în cazul probabilistic, elementele se aleg în mod pur aleator

De asemenea, în cazul algoritmului clasic probabilist, deoarece nu se necesită un răspuns exact, ne este necesar a se calcula $2^{n-1}+1$ valori pentru f, ci un număr mai mic, dependent de $\varepsilon : M_\varepsilon < 2^{n-1}+1$. Astfel, pasul 3. se înlocuieşte în cazul algoritmului clasic probabilist cu:

3. dacă $|B| > M_\varepsilon - 1$ atunci întoarce „ f constantă”

Se observă că acest algoritm probabilist nu greşeşte niciodată când funcţia este constantă, adică nu va cataloga niciodată drept balansată o funcţie care este de fapt constantă. Dar, dacă funcţia este de fapt balansată, există riscul ca acest algoritm să o catalogheze drept constantă. Asta se poate întâmpla când în cei M_ε paşi s-a obţinut mereu aceeaşi valoare pentru f.
După pasul M_ε, probabilitatea de eroare poate fi scrisă aşadar ca:

$$\varepsilon = p_{M_\varepsilon - 1} = \frac{2^{n-1}}{2^n} \times \frac{2^{n-1}-1}{2^n-1} \times \ldots \times \frac{2^{n-1}-M_\varepsilon+1}{2^n-M_\varepsilon+1} \le \frac{1}{2} \times \frac{1}{2} \times \ldots \frac{1}{2} = \frac{1}{2^{M_\varepsilon}}$$

Se observă deci că: $M_\varepsilon \le \log_2 \frac{1}{\varepsilon}$. Aşadar, rulând algoritmul pentru $\log_2 \frac{1}{\varepsilon}$ paşi, obţinem un răspuns corect cu probabilitatea de eroare aleasă. Complexitatea algoritmului clasic probabilist este deci $O\left(\log_2 \frac{1}{\varepsilon}\right)$ – nu depinde de dimensiunea problemei ci numai de probabilitatea de eroare aleasă.

2.3.2. Circuitul cuantic Deutsch-Jozsa

Revenind la problema Deutsch-Jozsa generală (deterministă), ea poate fi rezolvată folosind un algoritm cuantic al cărui circuit este prezentat mai jos.

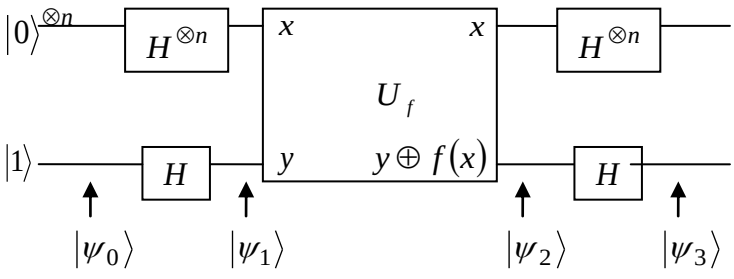


Figura 5. Circuit cuantic care implementează algoritmul Deutsch-Jozsa general

Problema Deutsch-Jozsa generală poate fi rezolvată prin măsurarea primilor n qubiţi din starea $|\psi_3\rangle$. Dacă fiecare qubit este 0 atunci funcţia f este constantă, dacă cel puţin un qubit este 1 atunci funcţia f este balansată. În cazul algoritmului cuantic, evaluarea funcţiei f s-a

făcut o singură dată. Asta înseamnă o îmbunătăţire de ordin exponenţial faţă de algoritmul clasic determinist (care, în cazul cel mai defavorabil, evaluează f de $2^{n-1}+1$ ori).

2.4. Codificarea supra-densă

Se presupune că Alice este în posesia unei informaţii clasice pe doi biţi (codificată în mod standard astfel $00, 01, 10, 11$), pe care vrea să o transmită lui Bob folosind un canal de comunicaţie la distanţă. Se poate arăta că această transmisie poate fi efectuată prin transferul unui singur qubit. Astfel, trebuie presupus iniţial că Alice şi Bob împărtăşesc iniţial o pereche de qubiţi aflaţi în starea EPR: $|\psi_0\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$. Alice este în posesia primului qubit, în timp ce Bob se află în posesia celui de-al doilea. Aceasta este o stare iniţială fixată, nu este nevoie de nici un transfer de qubiţi pentru a pregăti această stare. Se poate presupune de exemplu că această stare a fost pregătită anterior de cineva care a transmis apoi un qubit lui Bob, iar pe celălalt lui Alice. În continuare, în funcţie de informaţia clasică pe care vrea să o transmită, Alice aplică unul din operatorii Pauli asupra qubitului aflat în posesia sa.

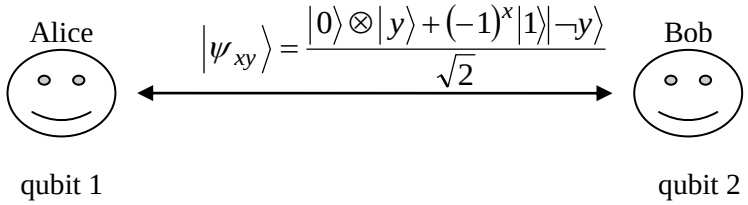


Figura 6. Starea de după aplicarea operatorului Pauli

Alice trimite acum qubitul său lui Bob. Aflându-se in posesia ambilor qubiţi, Bob poate efectua o măsurătoare de proiecţie în baza Bell. El defineşte în acest sens patru operatori de măsurare (proiecţii) şi poate afla deci cu certitudine valoarea xy care reprezintă exact informaţia trimisă de Alice. O generalizare pentru mai mulţi biţi este posibilă in felul următor: pentru a transmite $2n$ biţi de informaţie, Alice şi Bob trebuie să împărtăşească o stare „entangled" (complicată) formată din $2n$ qubiţi: n qubiţi se găsesc în posesia lui Alice iar ceilalţi n în posesia lui Bob. De fiecare dată când vrea să transmită 2 biţi, Alice aplică un operator Pauli asupra unui qubit, şi trimite qubitul rezultat lui Bob. La rândul său, Bob aplică o măsurătoare de proiecţie asupra stării formate din qubitul primit şi perechea sa entangled. O altă caracteristică utilă a acestui protocol se desprinde din următoarea observaţie. Se presupune existenţa unui al treilea personaj, Eve, cu rol negativ, care *ascultă* canalul de transmisie folosit de Alice şi Bob. Eve interceptează aşadar qubitul trimis de Alice şi doreşte să intre în posesia informaţiei transmise. În acest scop, Eve va trebui să efectueze o măsurătoare asupra qubitului interceptat. În cazul unei măsurători generale, ea defineşte o serie de operatori pozitivi E_m pe care îi va aplica asupra qubitului 1. Dar, probabilitatea de a obţine unul din rezultate este aceeaşi, indiferent de starea reală de dinaintea efectuării măsurătorii. În concluzie, deşi Eve poate intra în posesia qubitului trimis de Alice, Eve nu poate afla informaţia transmisă de Alice. Acest protocol este garantat a fi sigur, cu condiţia suficientă ca qubitul aflat în posesia lui Bob să fie păstrat strict secret.

2.5. Teleportarea cuantică

Teleportarea cuantică este o tehnică de transmisie a stărilor cuantice, putând fi folosită chiar în absenţa unui canal de comunicare cuantică care să lege transmiţătorul stării cuantice de

receptor. Misiunea lui Alice este de a transmite lui Bob un qubit aflat într-o stare oarecare $|\psi\rangle$. Restricţiile la care Alice trebuie să se supună sunt:

- Alice nu cunoaşte starea pe care trebuie să o transmită. Deoarece nu deţine decât o copie a qubitului ce se doreşte transmis, Alice nici măcar nu poate afla care este starea respectivă. Pentru a afla starea $|\psi\rangle$ în care se află qubitul, Alice ar trebui să efectueze o măsurare; dar prin asta s-ar distruge starea ce se doreşte transmisă.
- Între Alice si Bob nu există decât un canal de transmisie digital, clasic. Acest fapt complică situaţia şi mai mult, deoarece chiar dacă ar cunoaşte starea $|\psi\rangle$ pe care doreşte să o transmită, ca s-o transmită Alice ar avea nevoie de un număr infinit de biţi clasici pentru că $|\psi\rangle$ ia valori în spectrul continuu al numerelor complexe.

Soluţia se bazează pe presupunerea că Alice şi Bob împărtăşesc iniţial (înainte de a fi separaţi prin canalul de transmisie clasic) o pereche de qubiţi aflaţi în starea EPR: $|\beta_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$

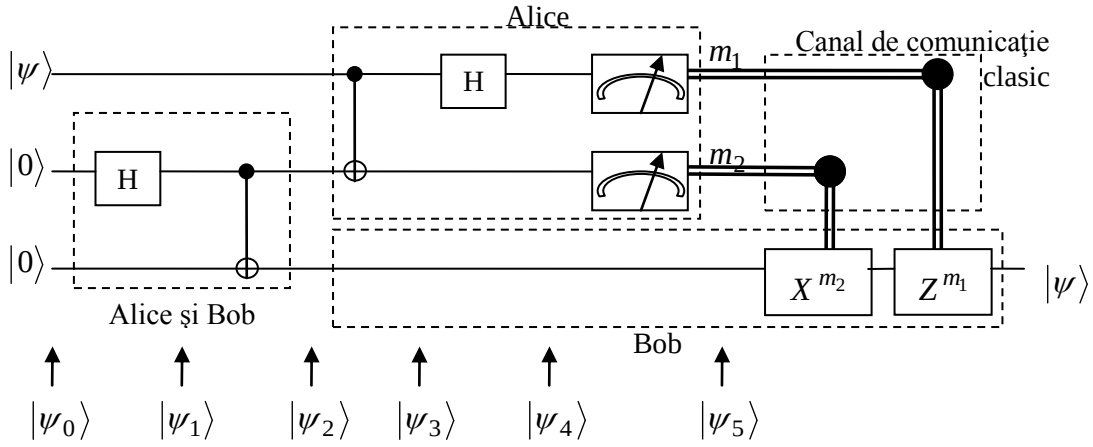


Figura 7. Circuitul cuantic pentru teleportarea unui qubit

Trebuie accentuat faptul că teleportarea cuantică nu contrazice în nici un fel postulatul teoriei relativităţii restrânse, conform căruia informaţia nu poate fi transmisă cu o viteză mai mare ca viteza luminii în vid. Bob nu poate reproduce starea $|\psi\rangle$ decât după ce primeşte rezultatul măsurătorii efectuate de Alice, rezultat care se transmite pe cale clasică, deci cu viteză limitată. Trebuie remarcat de asemenea că teleportarea cuantică nu produce nici o copie a stării de teleportat. La un moment dat, numai un qubit se află în starea $|\psi\rangle$. La sfârşitul procesului, qubitul aflat iniţial în starea de teleportat $|\psi\rangle$ se va afla în final într-una din stările computaţionale de bază $|0\rangle$ sau $|1\rangle$, pentru că asupra lui se efectuează o măsurătoare.

Teleportarea cuantică arată posibilitatea de inter-schimbare a unor resurse diferite, demonstrând că o pereche EPR împreună cu doi biţi de comunicaţie clasici este o resursă cel puţin identică cu un qubit de comunicaţie. Acest fapt se foloseşte în construcţia unor porţi cuantice rezistente la zgomot şi în corectarea erorilor de transmisie cuantice.

3. Reprezentare grafică

3.1. Urma unui operator

Oricărui operator A peste spaţiul Hilbert V de dimensiune n i se poate asocia o matrice pătratică complexă $a_{ij}[n\times n]$, definită pentru o bază ortonormată $|v_1\rangle...|v_n\rangle$ astfel [26]:

$$a_{ij}=\langle v_i|A|v_j\rangle, \forall i,j=\overline{1,n}.$$

Deoarece prin schimbarea de bază se obţine o matrice asociată similară cu matricea originală, şi deoarece urma este invariantă la transformarea de similaritate, rezultă că se poate defini urma unui operator peste spaţiul vectorial V ca fiind urma oricărei matrice asociate corespunzătoare unei baze ortonormate. Dacă $|j\rangle, j=\overline{1,n}$ este o bază ortonormată în V, se observă că: $\forall j,k=\overline{1,n}$ şi $j\neq k$: $\mathrm{tr}(|j\rangle\langle j|)=1$ şi $\mathrm{tr}(|j\rangle\langle k|)=0$ şi $\mathrm{tr}(I_n)=n$

3.2. Spaţiul vectorial al operatorilor liniari

Mulţimea L_V, formată din operatorii liniari care se pot defini pe un spaţiu Hilbert V, este la rândul său un spaţiu vectorial peste mulţimea numerelor complexe. Pe spaţiul L_V se poate defini astfel un produs scalar: $(A,B)\overset{\Delta}{=}\mathrm{tr}(A^\dagger B)$. Următorul set de n^2 operatori formează o bază ortonormată în L_V:

$$A_{jk}=\begin{cases}\dfrac{|k\rangle\langle j|+|j\rangle\langle k|}{\sqrt{2}}, & \forall j,k=\overline{1,n} \text{ şi } k>j\\ \dfrac{i|j\rangle\langle k|-i|k\rangle\langle j|}{\sqrt{2}}, & \forall j,k=\overline{1,n} \text{ şi } j>k\\ |j\rangle\langle j|, & \forall j=\overline{1,n}\end{cases}$$

3.3. Matricele Pauli

Cu notaţiile de mai sus, dacă V_2 este un spaţiu vectorial bidimensional iar ca bază ortonormată [27] se consideră vectorii corespunzători stărilor computaţionale de bază şi renunţând la condiţia de normalizare, se obţin operatorii Pauli [10]:

$$\sigma_0\equiv I_2\equiv|0\rangle\langle 0|+|1\rangle\langle 1| \quad \sigma_1\equiv\sigma_x\equiv X\equiv|1\rangle\langle 0|+|0\rangle\langle 1|$$
$$\sigma_2\equiv\sigma_y\equiv Y\equiv i|1\rangle\langle 0|-i|0\rangle\langle 1| \quad \sigma_3\equiv\sigma_z\equiv Z\equiv|0\rangle\langle 0|-|1\rangle\langle 1|$$

Matricele asociate operatorilor Pauli în baza formată din vectorii $|0\rangle$ şi $|1\rangle$ au proprietăţile:

$$\sigma_0\equiv I_2\equiv\begin{bmatrix}1&0\\0&1\end{bmatrix}\ \sigma_1\equiv\sigma_x\equiv X\equiv\begin{bmatrix}0&1\\1&0\end{bmatrix}\ \sigma_2\equiv\sigma_y\equiv Y\equiv\begin{bmatrix}0&-i\\i&0\end{bmatrix}\ \sigma_3\equiv\sigma_z\equiv Z\equiv\begin{bmatrix}1&0\\0&-1\end{bmatrix}$$

- sunt Hermite: $\sigma_k^\dagger=\sigma_k, \forall k=\overline{0,3}$ şi unitare: $\sigma_k^\dagger\sigma_k=\sigma_k\sigma_k^\dagger=\sigma_k^2=I_2, \forall k=\overline{0,3}$
- $\forall k=\overline{1,3}$ notaţiile: $\sigma_j\sigma_k=\delta_{jk}I_2+i\sum_{l=1}^{3}\varepsilon_{jkl}\sigma_l$
- $XY=iZ \quad YZ=iX \quad ZX=iY \quad YX=-iZ \quad ZY=-iX \quad XZ=-iY$
- sunt anti-comutative: $\{\sigma_j,\sigma_k\}=\sigma_j\sigma_k+\sigma_k\sigma_j=0_2, \quad \forall j,k=\overline{1,3}$ şi $j\neq k$

- satisfac relaţiile de comutare: $\left[\sigma_j, \sigma_k\right] = 2i\sum_{l=1}^{3} \varepsilon_{jkl}\sigma_l$
- $\mathrm{tr}(\sigma_0) = 2$ şi $\mathrm{tr}(\sigma_k) = 0$, $\forall k = \overline{1,3}$

3.4. Reprezentarea geometrică a qubiţilor

3.4.1. Qubiţi în stare pură – sfera Bloch

Pentru reprezentarea geometrică a qubiţilor, se consideră reprezentarea în coordonate polare a numerelor complexe. Din principiul de măsurare din mecanica cuantică rezultă că măsurând două stări cuantice care diferă numai printr-o fază globală se obţin întotdeauna aceleaşi rezultate. $|\psi\rangle \cong \mathrm{e}^{i\phi}|\psi\rangle$, $\forall \phi \in \Re$. Starea qubitului devine astfel : $|\psi\rangle = \cos\frac{\gamma}{2}|0\rangle + \sin\frac{\gamma}{2}\mathrm{e}^{i\varphi}|1\rangle$ cu numerele reale unic determinate $\gamma \in [0,\pi]$ şi $\varphi \in [0,2\pi)$.

Există o bijecţie între mulţimea stărilor măsurabile ale unui qubit şi mulţimea punctelor de pe sfera unitate în spaţiul clasic Euclidian. Fiecărei stări $|\psi\rangle = \cos\frac{\gamma}{2}|0\rangle + \sin\frac{\gamma}{2}\mathrm{e}^{i\varphi}|1\rangle$ îi este asociat un punct având coordonatele sferice $P(\varphi,\gamma)$. Vectorul care uneşte centrul sferei cu P se numeşte vector Bloch şi are coordonatele $(p_x, p_y, p_z) = (\cos\varphi\sin\gamma, \sin\varphi\sin\gamma, \cos\gamma)$.

3.4.2. Qubiţi în stare mixtă – bila Bloch

Pentru a reprezenta sistemele cuantice a căror stare nu este complet cunoscută la un moment dat, se foloseşte operatorul densitate. Presupunând că un sistem cuantic este într-una din stările $|\psi_i\rangle$ cu probabilitatea p_i, operatorul densitate al sistemului se defineşte prin ecuaţia:

$$\rho \equiv \sum_i p_i |\psi_i\rangle\langle\psi_i|,\ 0 \le p_i \le 1 \text{ şi } \sum_i p_i = 1.$$

Operatorul densitate este operator Hermit. Operatorul densitate pentru o stare pură $|\psi\rangle$ se defineşte: $\rho \equiv |\psi\rangle\langle\psi|$. Urma operatorului densitate compus cu el însuşi satisface relaţia: $\mathrm{tr}(\rho^2) \le 1$, cu egalitate dacă şi numai dacă operatorul densitate ρ reprezintă o stare pură.

Dacă se consideră că sistemul cuantic este format dintr-un qubit, operatorul densitate al sistemului aparţine spaţiului vectorial al operatorilor generaţi de operatorii Pauli. Deci: $\rho = aI_2 + b\sigma_x + c\sigma_y + d\sigma_z$, cu a, b, c şi d numere complexe. Operatorul densitate se poate scrie aşadar în funcţie de operatorii Pauli astfel, unde $\bar{r} = (r_x, r_y, r_z)$ este un vector real tridimensional: $\rho = \frac{1}{2}\left(I_2 + r_x\sigma_x + r_y\sigma_y + r_z\sigma_z\right) = \frac{1}{2}\left(I_2 + \bar{r}\cdot\bar{\sigma}\right)$. Considerând baza computaţională, matricea asociată operatorului densitate trebuie să aibă valori proprii pozitive: $\lambda_{1,2} = \frac{1 \pm \sqrt{{r_x}^2 + {r_y}^2 + {r_z}^2}}{2} \ge 0 \Rightarrow {r_x}^2 + {r_y}^2 + {r_z}^2 \le 1 \Leftrightarrow \|\bar{r}\| \le 1$

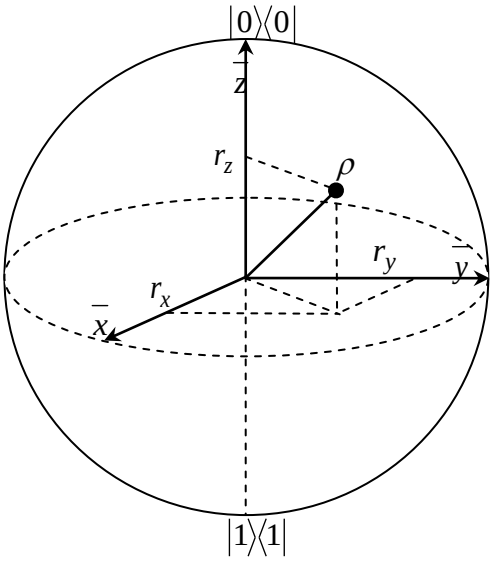


Figura 8. Bila Bloch

În concluzie, orice operator densitate pentru sisteme formate dintr-un qubit este unic identificat prin vectorul real $\bar{r} = (r_x, r_y, r_z)$, aflat în bila de rază 1, cu centrul în origine, situată în spaţiul real tridimensional. Dacă punctul reprezentând starea ρ este chiar pe sfera unitate, adică dacă $\|\bar{r}\| = 1$, atunci acest operator densitate reprezintă o stare pură.

3.5. Operatorii de rotaţie

Considerând proprietatea matricelor Pauli $\sigma_k^2 = I_2$, se pot defini următorii operatori în spaţiul bidimensional complex, care se dovedesc a fi chiar rotaţii în spaţiul tridimensional, în jurul axelor de coordonate: $R_k(\theta) \equiv \exp\left(\frac{-i\theta\sigma_k}{2}\right) = \cos\frac{\theta}{2} I_2 - i\sin\frac{\theta}{2}\sigma_k, \forall k = \overline{0,3}$

Considerându-se un vector real unitar $\bar{n} = (n_x, n_y, n_z)$, se poate defini operatorul de rotaţie generalizat: $R_n(\theta) \equiv \exp\left(-i\frac{\theta}{2}\bar{n}\cdot\bar{\sigma}\right) = \cos\frac{\theta}{2} I_2 - i\sin\frac{\theta}{2}\bar{n}\cdot\bar{\sigma}$ având următoarea interpretare geometrică. Dacă se consideră un qubit în starea $|\psi_0\rangle = \cos\frac{\gamma_0}{2}|0\rangle + \sin\frac{\gamma_0}{2}e^{i\varphi_0}|1\rangle$, asupra căruia acţionează operatorul $R_n(\theta)$, astfel încât $|\psi_1\rangle = R_n(\theta)|\psi_0\rangle = \cos\frac{\gamma_1}{2}|0\rangle + \sin\frac{\gamma_1}{2}e^{i\varphi_1}|1\rangle$, şi dacă P_0 şi P_1 sunt punctele de pe sfera Bloch corespunzătoare stărilor $|\psi_0\rangle$ respectiv $|\psi_1\rangle$, atunci P_1 este obţinut prin rotaţia lui P_0, cu unghiul θ în jurul axei $\bar{n}$.

3.6. Descompunerea operatorilor unitari pe un qubit

Deoarece setul de operatori Pauli I_2, X, Y, Z formează o bază ortonormată în spaţiul operatorilor liniari peste spaţiul complex bidimensional, orice operator liniar unitar pe un qubit U se poate descompune în: $U = aI_2 + bX + cY + dZ$ cu a, b, c, d numere complexe unic determinate, sau ca: $U = e^{i\alpha} R_n(\theta)$, unde vectorul unitate $\bar{n} = (n_x, n_y, n_z)$ şi $\theta \in [0, \pi]$.

3.6.1. Descompunerea Z-Y a operatorilor unitari pe un qubit

Dacă U este un operator liniar peste un spaţiu bidimensional complex, atunci orice matrice a sa asociată de forma $U=\begin{bmatrix} a & b \\ c & d \end{bmatrix} \Rightarrow U^{\dagger}=\begin{bmatrix} a^* & c^* \\ b^* & d^* \end{bmatrix}$ se poate descompune ca:

- dacă $|b|=0 \vee |c|=0 \Rightarrow |a|=|d|=1$, atunci $U=\mathrm{e}^{i\frac{\alpha_a+\alpha_d}{2}} R_z(-\alpha_a)R_y(0)R_z(\alpha_d)$

- dacă $|a|=0 \vee |d|=0 \Rightarrow |b|=|c|=1$, atunci $U=\mathrm{e}^{i\frac{\alpha_b+\alpha_c+\pi}{2}} R_z(-\alpha_b+\alpha_c-\pi)R_y(\pi)R_z(0)$

- dacă $|a||b||c||d|\neq 0$, atunci $U=\mathrm{e}^{i\left(\frac{\alpha_a+\alpha_d}{2}\right)} R_z(-\alpha_a+\alpha_c)R_y(2\arccos|a|)R_z(-\alpha_c+\alpha_d)$

3.6.2. Descompunerea X-Y a operatorilor unitari pe un qubit

Considerând din nou U ca fiind un operator liniar unitar peste un spaţiu bidimensional complex, se poate descompuner în rotaţii X-Y: $U=\mathrm{e}^{i\alpha} R_x(\beta)R_y(\gamma)R_x(\delta)$, unde unghiurile de rotație se calculează din:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}=\mathrm{e}^{i\alpha}\begin{bmatrix} \cos\frac{\gamma}{2}\cos\frac{\beta+\delta}{2}-i\sin\frac{\gamma}{2}\sin\frac{\beta-\delta}{2} & -\sin\frac{\gamma}{2}\cos\frac{\beta-\delta}{2}-i\cos\frac{\gamma}{2}\sin\frac{\beta+\delta}{2} \\ \sin\frac{\gamma}{2}\cos\frac{\beta-\delta}{2}-i\cos\frac{\gamma}{2}\sin\frac{\beta+\delta}{2} & \cos\frac{\gamma}{2}\cos\frac{\beta+\delta}{2}+i\sin\frac{\gamma}{2}\sin\frac{\beta-\delta}{2} \end{bmatrix}$$

4. Circuite cuantice controlate

Folosind descompunerea operatorilor unitari pe un qubit în produs de rotaţii se poate demonstra un rezultat important în construcţia operatorilor unitari pe mai mulţi qubiţi.
Corolar: Se presupune că U este un operator unitar pe un qubit. Atunci există operatorii unitari A, B, C pe un singur qubit astfel încât $ABC=I_2$ şi $U=e^{i\alpha}AXBXC$ unde α este un factor generic de fază.

4.1. Operatorul U-Controlat de un qubit

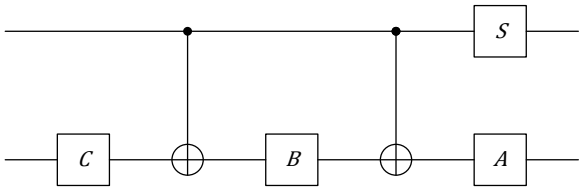
4.1.1. Definiţie şi notaţii

Se consideră un operator unitar pe un singur qubit U. Un operator $U-Controlat$ este prin definiţie un operator pe doi qubiţi – un qubit de control şi un qubit de date [34]:
- qubitul de control rămâne mereu neschimbat,
- dacă qubitul de control este setat atunci operatorul U acţionează asupra qubitul de date,
- dacă qubitul de control este resetat atunci qubitul de date rămâne neschimbat.

$$|c\rangle|d\rangle \xrightarrow{U-Controlat} |c\rangle U^c|d\rangle$$

4.1.2. Implementarea operatorului U-Controlat de un qubit

Folosind descompunerea $U=e^{i\alpha}AXBXC$, se demonstrează faptul că:
Teoremă: Circuitul $U-Controlat$ poate fi implementat folosind numai porţi care acţionează pe un singur qubit şi poarta CNOT.
Demonstraţie: Circuitul este următorul, unde acțiunea operatorului de schimbare de fază este definită prin: $|0\rangle \xrightarrow{S} S|0\rangle=|0\rangle$, $|1\rangle \xrightarrow{S} S|1\rangle=e^{i\alpha}|1\rangle$



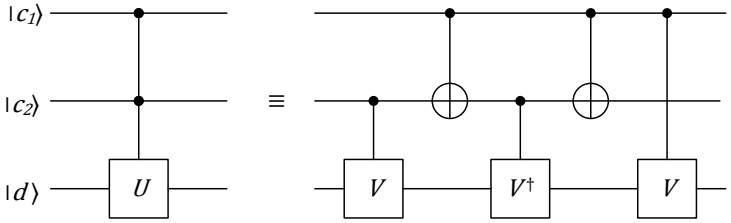
4.2. Operatorul U-Controlat pe mai mulţi qubiţi

Un exemplu particular de poartă controlată pe mai mulţi qubiţi este poarta Toffoli [7]. Considerând starea computaţională de bază, în cazul general, presupunând un operator pe $n + k$ qubiţi în care operatorul U acţionează pe k qubiţi, operatorul U controlat de n qubiţi se defineşte ca: $C_k^n(U)|c_1c_2 \dots c_n\rangle|d_1d_2 \dots d_k\rangle \stackrel{\text{def}}{=} |c_1c_2 \dots c_n\rangle U^{c_1c_2\dots c_n}|d_1d_2 \dots d_k\rangle$,

4.3. Operatorul U-Controlat de doi qubiţi

4.3.1. Implementare folosind porţi controlate de 1 qubit

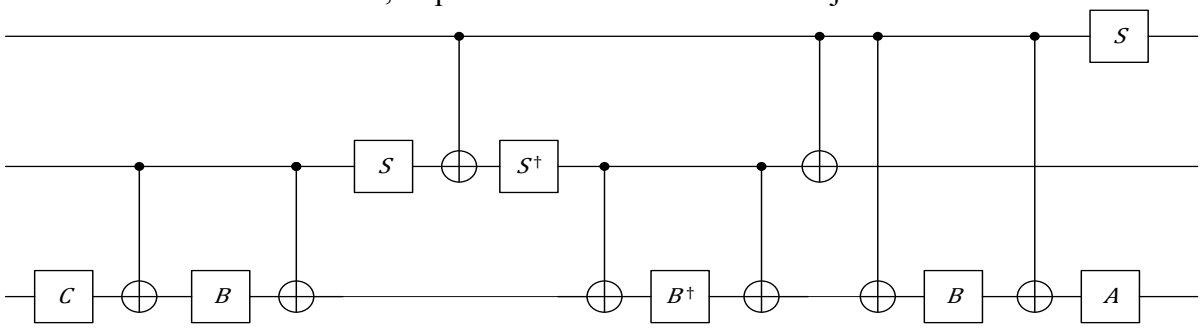
Teoremă: $C_1^2(U)$ poate fi descompus în produs de operatori $C_1^1(V)$, unde $V^2 = U$.
Pentru poarta Toffoli: $U \equiv X$ şi $V \equiv \frac{1-i}{2}(I_2 + iX)$, deci porţi pe un qubit împreună cu porţi pe doi qubiţi sunt suficiente pentru a implementa poarta reversibilă Toffoli. În calculul clasic, această observaţie nu este aplicabilă.



4.3.2. Implementare folosind numai CNOT şi porţi simple pe un qubit

Teoremă: Orice operator $C_1^2(U)$ poate fi implementat folosind cel mult 8 porţi pe un singur qubit şi 6 porţi CNOT.

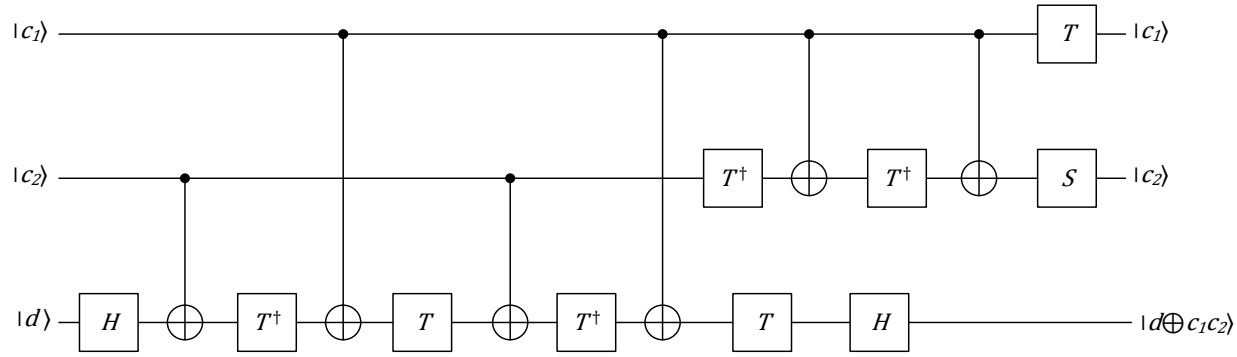
Demonstraţie: Folosind descompunerea $V = e^{i\alpha}AXBXC \Rightarrow V^\dagger = e^{-i\alpha}C^\dagger XB^\dagger XA^\dagger$, conform celor două teoreme de mai sus, se poate construi circuitul de mai jos.



4.4. Implementarea cuantică a porţilor clasice universale reversibile

4.4.1. Implementarea porţii Toffoli

Poarta Toffoli poate fi implementată folosind numai următoarele tipuri de porţi: Hadamard, fază, CNOT şi T. Circuitul care implementează poarta Toffoli este următorul:



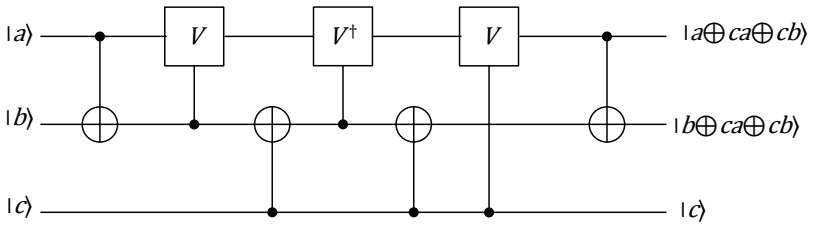
4.4.2. Implementarea porţii Fredkin folosind Toffoli

Poarta Fredkin (sau poarta de inversare controlată), având o semnificaţie aparte în teoria calculului reversibil, se defineşte ca fiind o poartă pe trei biţi, doi biţi de date şi un bit de control, care satisfac următoarele condiţii: bitul de control rămâne mereu neschimbat și biţii de date sunt interschimbaţi dacă şi numai dacă bitul de control este setat. Poarta Fredkin este:

- *reversibilă*: aplicând de două ori în serie poarta Fredkin se obţin biţii în starea originală.
- *conservativă*: numărul de biţi setaţi se conservă la trecerea prin poartă.

Din definiţia de mai sus se obţine descrierea algebrică a porţii Fredkin, unde a, b, c sunt numere binare pe 1 bit: $|abc\rangle \xrightarrow{Fredkin} |a \oplus ca \oplus cb\rangle|b \oplus ca \oplus cb\rangle|c\rangle$

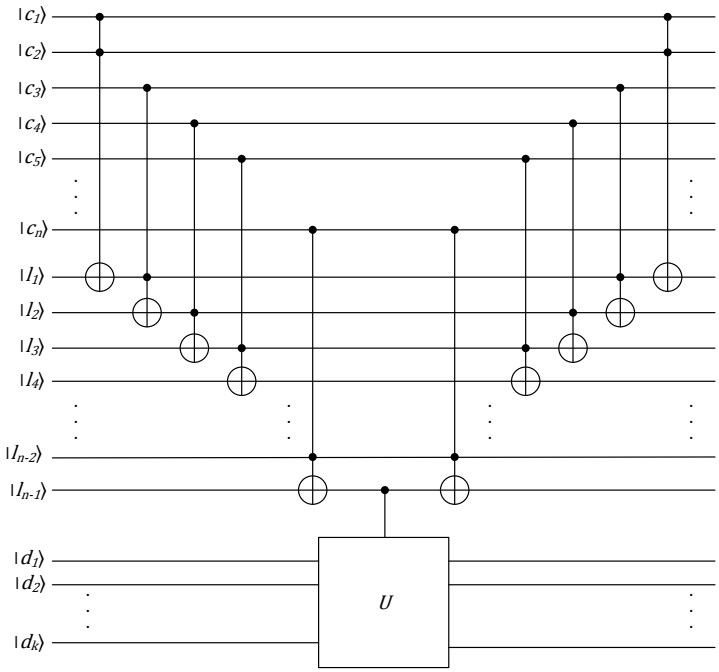
Poarta Fredkin se poate implementa folosind 6 porți pe doi qubiți, unde $V \equiv \frac{1-i}{2}(I_2 + iX)$:



5. Implementarea operatorilor controlaţi

5.1. Implementarea liniară a operatorilor controlaţi

Un circuit foarte simplu care realizează implementarea operatorilor controlaţi $C_k^n(U)$ în cazul general este prezentat mai jos. Circuitul este împărţit din punct de vedere logic din trei etape şi foloseşte $n-1$ qubiţi de lucru, setaţi toţi iniţial în starea computaţională de bază $|0\rangle$.



Considerând qubiţii de control în starea computaţională de bază $|c_1 c_2 \dots c_n\rangle$, circuitul implementează reversibil în prima etapă operaţia de produs logic între toţi biţii de control: $c_1 \cdot c_2 \cdot \dots \cdot c_n$. Sunt folosite în acest scop $n-1$ porţi Toffoli şi cei $n-1$ qubiţi de lucru. În a doua etapă, circuitul implementează operaţia căutată $C_k^n(U)$ folosind o simplă poartă condiţionată de un singur qubit $C_k^1(U)$. În cea de-a treia şi ultima etapă, circuitul implementează operaţia inversă corespunzătoare operaţiei din prima etapă pentru a reseta toţi qubiţii de lucru la starea lor computaţională de bază iniţială $|0\rangle$.

5.2. Implementarea exponenţială a operatorilor controlaţi

5.2.1. Implementarea operatorilor controlaţi de 3 qubiţi

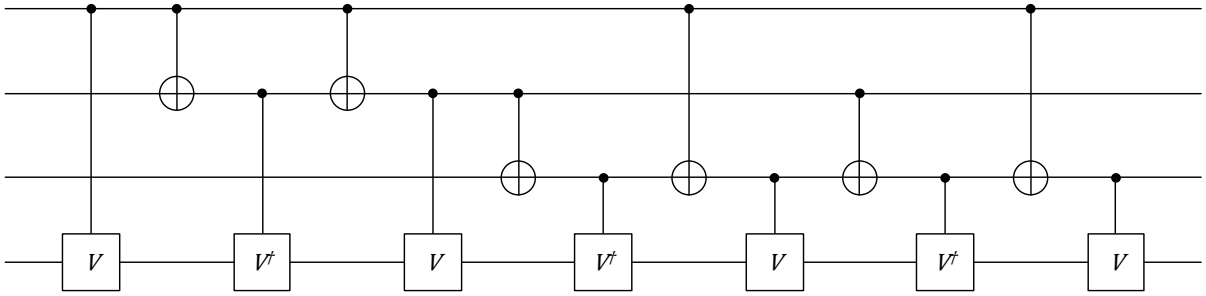
Implementarea operatorilor $C_k^n(U)$ poate fi realizată folosind numai operatori $C_k^1(V)$, fără a folosi nici un qubit de lucru. Metoda prezentată mai sus pentru implementarea operatorului condiţionat de 2 qubiţi $C_1^2(U)$ poate fi generalizată. În mod analog, pentru implementarea $C_1^3(U)$ se determină operatorul V astfel încât $V^4 = U$. Secvenţa ce operaţii efectuate asupra celui de-al patrulea qubit este:

- (100) V dacă şi numai dacă $c_1 = 1$
- (110) $V^\dagger$ dacă şi numai dacă $c_1 \oplus c_2 = 1$
- (010) V dacă şi numai dacă $c_2 = 1$
- (011) $V^\dagger$ dacă şi numai dacă $c_2 \oplus c_3 = 1$
- (111) V dacă şi numai dacă $c_1 \oplus c_2 \oplus c_3 = 1$
- (101) $V^\dagger$ dacă şi numai dacă $c_1 \oplus c_3 = 1$
- (001) V dacă şi numai dacă $c_3 = 1$

Pentru fiecare operaţie de mai sus, şirul de biţi din stânga indică asupra căror qubiţi se aplică condiţia de setare ($= 1$). Paritatea fiecărui şir de bit indică tipul operatorului aplicat: pentru şiruri cu număr impar de biţi setaţi se aplică V, în timp ce pentru şirurile cu număr par de biţi se aplică $V^\dagger$. Prin compararea acestei secvenţe de operaţii cu termenii din ecuaţia:

$$c_1 - c_1 \oplus c_2 + c_2 - c_2 \oplus c_3 + c_1 \oplus c_2 \oplus c_3 - c_1 \oplus c_3 + c_3 = 4 \cdot (c_1 \wedge c_2 \wedge c_3)$$

se poate verifica faptul că secvenţa de operaţii de mai sus este echivalentă cu aplicarea operatorului V^4 asupra celui de al patrulea qubit dacă şi numai dacă $c_1 \wedge c_2 \wedge c_3 = 1$, adică exact definiţia operatorului condiţionat: $C_1^3(U)$.
Circuitul care implementează secvenţa de operaţii de mai sus este următorul. Pentru o implementare eficientă, şirurile de biţi de mai sus trebuie să formeze o secvenţa de cod Gray.



5.2.2. Implementarea operatorilor controlaţi. Generalizare

Analog cu construcţia circuitului de mai sus, se poate ajunge prin inducţie la următoarea generalizare:
Teoremă: Pentru orice $n \geq 2$, şi orice operator unitar U pe k qubiţi, poarta condiţionată $C_k^n(U)$ poate fi implementată de un circuit pe $n + k$ qubiţi care este format din $2^n - 1$ porţi $C_k^1(V)$ şi $C_k^1(V^\dagger)$, la care se adaugă $2^n - 2$ porţi CNOT, unde $V^{2^{n-1}} = U$ este un operator unitar. Circuitul este obţinut prin transpunerea următoarei ecuaţii:

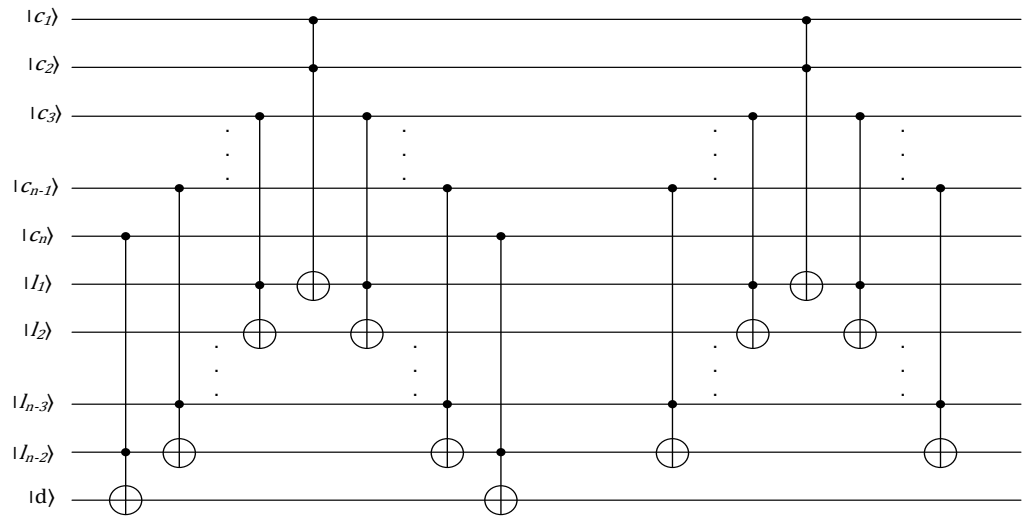
$$\sum_{i_1=1}^{n} c_i - \sum_{i_1<i_2}^{n} \left(c_{i_1} \oplus c_{i_2}\right) + \sum_{i_1<i_2<i_3}^{n} \left(c_{i_1} \oplus c_{i_2} \oplus c_{i_3}\right) - \ \ldots + (-1)^{n-1}(c_1 \oplus c_2 \oplus \ldots \oplus c_n)$$
$$= 2^{n-1} \cdot (c_1 \wedge c_2 \wedge \ldots \wedge c_n)$$

Se observă că, deşi această variantă de implementare a operatorilor condiţionaţi generali are avantajul că nu necesită nici un fel de qubiţi de lucru, numărul de porţi necesare creşte exponenţial cu numărul de qubiţi de control. În contrast, în varianta care necesită qubiţi de lucru, numărul de porţi necesare creşte numai liniar cu numărul qubiţilor de control.
Avem de a face aşadar şi în acest caz cu bine cunoscutul compromis între viteza de procesare (i.e. numărul de porţi conectate în cascadă) şi mărimea memoriei de lucru necesare (i.e. numărul qubiţilor de lucru).

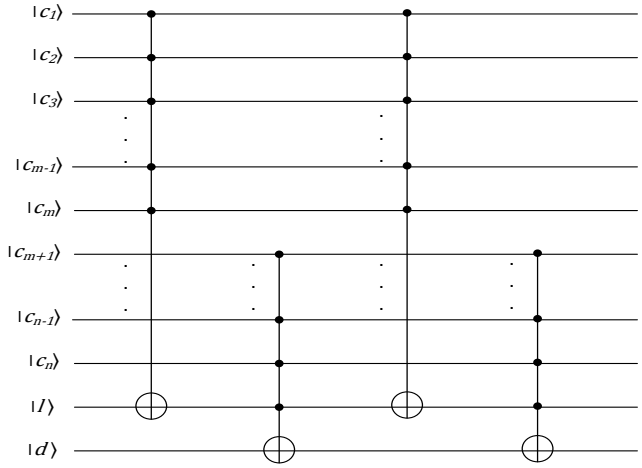
5.3. Implementarea pătratică a operatorilor controlaţi

5.3.1. Implementarea porţii CNOT generalizată folosind porţi Toffoli

Ca o soluţie de compromis între numărul exponenţial al porţilor necesare pentru simularea fără qubiţi de lucru şi numărul liniar al porţilor necesare pentru simularea cu qubiţi de lucru, este prezentat un circuit care implementează operatorul condiţionat $C_k^n(U)$ folosind un singur qubit de control.
Lema: Pentru $n \geq 3$, operatorul $C^n(X)$ poate fi implementat folosind $2n - 3$ porţi Toffoli şi $n - 2$ qubiţi de lucru care nu trebuie setaţi la o valoare iniţială anume. Adăugând încă $2n - 5$ porţi Toffoli, circuitul păstrează valoarea iniţială a qubiţilor de lucru.
Demonstraţie: prin inducţie după n. Circuitul în cazul general este următorul:



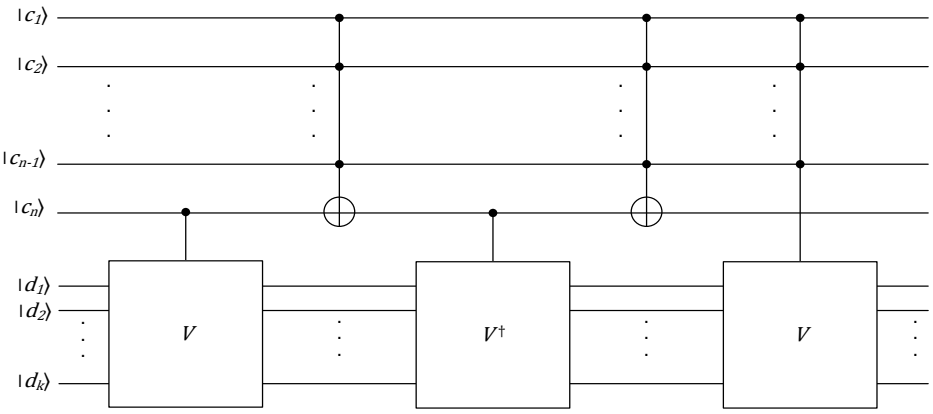
Lema: Pentru orice $n \geq 3$ şi $1 \leq m \leq n-1$ operatorul $C^n(X)$ poate fi implementat folosind un circuit format din două porţi $C^m(X)$, două porţi $C^{n-m+1}(X)$ şi un singur qubit de lucru.



Corolar: Pentru orice $n \geq 5$, operatorul $C^n(X)$ poate fi implementat folosind un circuit format din $8n - 24$ porţi Toffoli şi un singur qubit de lucru. Deci, complexitatea temporală este liniară iar complexitatea spaţială este constantă.

5.3.2. Implementarea operatorilor controlaţi, fără qubiţi de lucru

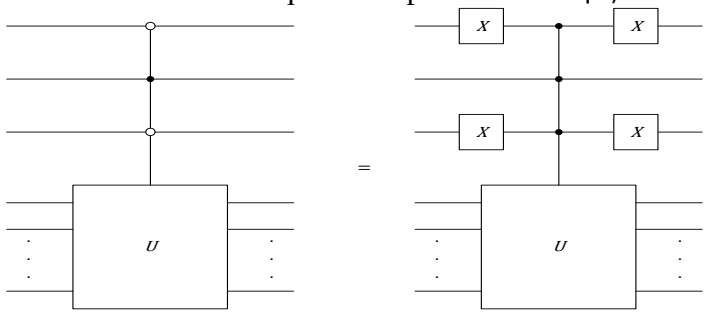
Lema: Pentru orice operator U, operatorul condiţionat $C_k^n(U)$ poate fi implementat de circuitul de mai jos, unde $V^2 = U$.
Teorema: Orice operator $C^n(U)$ poate fi implementat folosind $O(n^2)$ porţi elementare: porți care acţionează pe un singur qubit împreună cu porţi Toffoli şi CNOT.



6. Porţi cuantice universale

6.1. Porţi controlate prin valoarea 0

În circuitele prezentate anterior, porţile controlate sunt activate când qubiţii de control sunt setaţi la $|1\rangle$, şi sunt dezactivate când cel puţin un qubit de control este resetat la $|0\rangle$. Bineînţeles că nu e nimic special cu valoarea 1 în comparaţie cu valoarea 0. Se pot construi porţi cuantice controlate care sunt activate de qubiţi de control resetaţi la $|0\rangle$ [44]. Orice poartă pe un număr oarecare de qubiţi poate fi controlată de o combinaţie de qubiţi de control în aşa fel încât poarta respectivă este activată dacă şi numai dacă unii qubiţi de control sunt setaţi la $|1\rangle$ iar restul sunt resetaţi la $|0\rangle$. Acest tip de circuit poate fi implementat folosind porţi X şi o poartă U-controlată numai de qubiţi setaţi la $|1\rangle$. Porţile X pe un qubit acţionează asupra qubiţilor de control care activează poarta U prin valoarea $|0\rangle$.



6.2. Mulţimi continue de porţi cuantice universale

În calculul clasic, orice operaţie poate fi implementată folosind numai porţi din mulţimea $\{AND, NOT, FANOUT\}$. Alte asemenea de mulţimi universale pentru calculul clasic sunt mulţimea $\{Toffoli\}$ şi mulţimea $\{Fredkin\}$**Error! Reference source not found.**. Asta înseamnă că, aşa cum s-a arătat anterior, deoarece operatorul Toffoli poate fi implementat folosind un set limitat de porţi cuantice, rezultă că mulţimea tuturor circuitelor clasice formează o submulţime în mulţimea tuturor circuitelor cuantice. Se ajunge la exact aceeaşi concluzie folosind operatorul Fredkin.

O mulţime de porţi cuantice se defineşte ca fiind *exact universală pentru calculul cuantic* dacă şi numai dacă orice operator unitar, acţionând asupra unui număr oarecare finit de qubiţi, poate fi implementat exact de un circuit cuantic compus numai din tipurile de porţi respective. Pentru acest tip de universalitate s-au descoperit numai mulţimi infinite de porţi.

6.2.1. Matrice de nivel 2

Se consideră o matrice unitară pătratică U, de dimensiune $d \times d$, care acţionează asupra unui spaţiu Hilbert d-dimensional. Prin definiţie, o matrice U de ordin d care acţionează asupra unui vector v cu d componente, se numeşte *matrice de nivel 2* dacă şi numai dacă ea modifică numai două dintre componentele vectorului respectiv, lăsând toate celelalte componente neschimbate.

6.2.2. Descompunerea in matrice de nivel 2

Teoremă: Orice matrice unitară U de ordin d se poate descompune în produs finit de matrice unitare, fiecare de dimensiune $d \times d$ şi de nivel 2.
Numărul total de matrice unitare de ordin d şi nivel 2 necesare pentru descompunerea unei matrice unitare de ordin d este astfel cel mult: $nr_d \leq \frac{d(d-1)}{2}$. Ca urmare, deoarece matricea reprezentând un operator corespunzător unui circuit cuantic pe n qubiţi are ordinul $d = 2^n$, această matrice va fi descompusă într-un număr exponenţial de matrice unitare de nivel 2: $2^{n-1}(2^n - 1)$. Pentru unele matrice speciale totuşi se pot construi descompuneri mult mai eficiente. Este de asemenea important de constatat că există matrice unitare U de ordin d care nu pot fi descompuse într-un produs de matrice unitare de nivel 2 conţinând un număr de termeni mai mic decât $d - 1$. Deci, deşi există matrice corespunzătoare unor circuite a căror implementare poate fi eficientizată la un număr polinomial de termeni, există de asemenea şi matrice a căror descompunere nu poate fi mai eficientă decât exponenţialul numărului de qubiţi din circuit.

6.2.3. Implementarea matricelor unitare de nivel 2

Presupunând U este o matrice unitară de ordin $d = 2^n$ şi de nivel 2. Se doreşte implementarea acestei matrice folosind un circuit cuantic pe n qubiţi. Considerând stările computaţionale de bază $|b^i\rangle$, cu $i \in \{1, \dots, d\}$, un vector oarecare se descompune în: $|v\rangle = \sum_{i=1}^{d} \langle b^i|v\rangle |b^i\rangle$. Aşadar, dacă matricea U_{ij} acţionează numai asupra componentelor i şi j, ea va avea efect numai asupra vectorilor care fac parte din sub-spaţiul vectorial generat de vectorii din starea computaţională de bază $|b^i\rangle = |b_1^i b_2^i \dots b_n^i\rangle$ şi $|b^j\rangle = |b_1^j b_2^j \dots b_n^j\rangle$.
Etapele necesare construirii circuitului căutat sunt:

$$|b^i\rangle = |g^1\rangle \xrightarrow{Etapa\ 2} |g^{m-1}\rangle \xrightarrow{Etapa\ 3} U_{ij}|g^{m-1}\rangle \xrightarrow{Etapa\ 4} U_{ij}|b^i\rangle$$
$$|g^2\rangle \xrightarrow{Etapa\ 2} |g^1\rangle \xrightarrow{Etapa\ 3} |g^1\rangle \xrightarrow{Etapa\ 4} |g^2\rangle$$
$$|g^3\rangle \xrightarrow{Etapa\ 2} |g^2\rangle \xrightarrow{Etapa\ 3} |g^2\rangle \xrightarrow{Etapa\ 4} |g^3\rangle$$
$$\vdots$$
$$|g^{m-1}\rangle \xrightarrow{Etapa\ 2} |g^{m-2}\rangle \xrightarrow{Etapa\ 3} |g^{m-2}\rangle \xrightarrow{Etapa\ 4} |g^{m-1}\rangle$$
$$|b^j\rangle = |g^m\rangle \xrightarrow{Etapa\ 2} |b^j\rangle \xrightarrow{Etapa\ 3} U_{ij}|b^j\rangle \xrightarrow{Etapa\ 4} U_{ij}|b^j\rangle$$

6.2.4. Calculul complexităţii

Numărul maxim de porţi necesare implementării matricei de ordin 2^n şi nivel 2 este

$$cost(U_{ij}) = (n-1)cost(C^{n-1}(X)) + cost\left(C^{n-1}(\widetilde{U}_{ij})\right) + (n-1)cost(C^{n-1}(X))$$

În continuare, deoarece un operator general U pe n qubiţi poate fi descompus în produs de matrice de nivel 2, produsul respectiv având cel mult $2^{n-1}(2^n - 1)$ termeni, rezultă că U

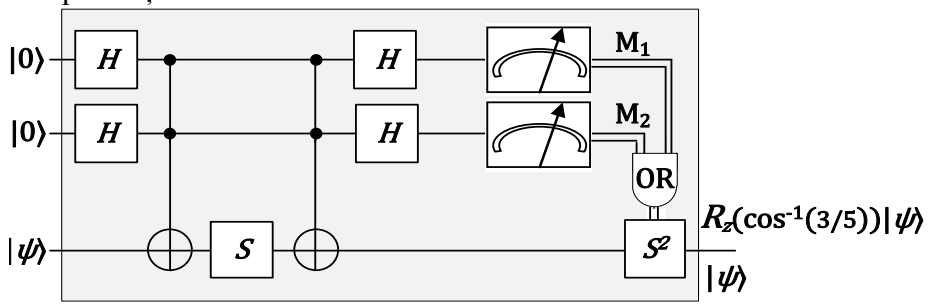
poate fi implementat folosind numai porţi care acţionează pe un qubit şi porţi CNOT, numărul total de porţi necesare fiind: $cost(U) = 2^{n-1}(2^n - 1)O(n^3) = O(n^3 4^n)$
Bineînţeles că această construcţie nu este dintre cele mai eficiente, deoarece necesită un număr exponenţial de porţi. De aceea, pentru găsirea unor algoritmi cuantici eficienţi este necesar a se urma o altfel de construcţie.

6.3. Mulțimi universale discrete de porți cuantice

Înainte ca modelul de calcul cuantic să poată fi aplicat în practică, câteva probleme au trebuit să fie adresate mai întâi. Cea mai importantă problemă este datorată susceptibilității sporite la interferențe externe nedorite (i.e. zgomot) a proceselor fizice care au loc la nivel cuantic. De aceea, este necesar a se demonstra că operatorii unitari (care sunt folosiți în modelarea operațiilor de calcul cuantice) pot avea implementări care sunt bazate numai pe porți cuantice robuste, rezistente la zgomot extern. Există deja câteva rezultate bine cunoscute în această direcție, a universalității porților cuantice de bază, care se bazează în principal pe folosirea unei porți ne-elementare, i.e. o poartă care implementează o rotație pe un qubit cu un unghi care este un multiplu irațional al lui 2π. Totuși, o implementare directă, rezistentă la zgomot a unei asemenea porți nu este posibilă în realitate; de aceea, aceste porți nu pot fi integrate cu ușurință în medii susceptibile la zgomot extern.
Există modalități de codificare a informației cuantice care pot fi folosite pentru a demonstra că o submulțime restrânsă de porți cuantice elementare: Hadamard, schimbare de fază, CNOT – numită grupul normalizator – poate fi implementată într-o manieră robustă, tolerantă la defecte. Dar această submulțime nu este suficientă pentru universalitate pentru că nu poate genera întreaga mulțime de operatori unitari. Acest fapt a condus la sugestia de a adăuga o nouă poartă elementară la grupul normalizator: Toffoli, o poartă care poate fi și ea implementată într-o manieră tolerantă la defecte.

6.3.1. Circuit de bază pentru rotații ne-elementare

Circuitul cuantic de mai jos implementează operatorul de rotație de bază, în jurul axei z, $R_z(\theta)$, cu un unghi specific: θ, unde $\cos\theta = 3/5$. Acest unghi θ a fost ales în așa fel încât să fie un multiplu irațional a lui 2π.



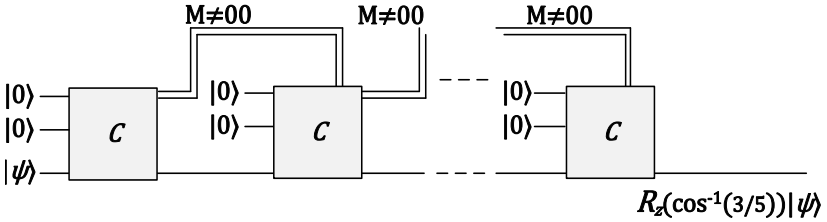
Starea circuitului de dinaintea operațiilor de măsurare devine atunci:

$$\longrightarrow \frac{\sqrt{10}}{4} e^{i\frac{\pi}{4}} |00\rangle R_z(\theta) + \frac{1-i}{4}(|01\rangle + |10\rangle - |11\rangle) S^2 |\psi\rangle$$

6.3.2. Circuit pentru rotații elementare, cu probabilitate unitară

Circuitul cuantic de mai sus implementează operatorul de rotație $R_z(\theta)$ asupra qubitului țintă, dacă rezultatele măsurătorilor efectuate asupra qubiților de control sunt amândouă 0. Altfel, dacă cel puțin o operație de măsurare întoarce rezultatul 1, qubitul țintă rămâne neschimbat,

în starea inițial. Probabilitățile acestor patru rezultate diferite, date de cei doi qubiți de control, pot fi calculate ca fiind: $P_{00} = \left|\frac{\sqrt{10}}{4} e^{i\frac{\pi}{4}}\right|^2 = \frac{5}{8}$ $P_{01} \equiv P_{10} \equiv P_{11} = \left|\frac{1-i}{4}\right|^2 = \frac{1}{8}$
Așa cum cele două ecuații de mai sus arată, probabilitatea ca circuitul cuantic de mai sus să implementeze intr-adevăr operatorul de rotație dorit este mult mai mare decât probabilitatea ca circuitul să implementeze o „no-op". Totuși, este posibil ca această distribuție de probabilitate să fie îmbunătățită mai departe, în așa fel încât probabilitatea cazului favorabil să tindă asimptotic către valoarea de certitudine 1. Aceasta poate fi realizat prin aplicarea succesivă a aceluiași circuit cuantic, până când operatorul de rotație este într-adevăr aplicat.



Circuitul de mai sus implementează operatorul de rotație elementară, cu o probabilitate care tinde asimptotic către 1. Acest proces rulează astfel: dacă, la pasul curent, rezultatul a cel puțin unei măsurători asupra unui qubit de control este 1, atunci aplică circuitul din nou folosind doi qubiți de control noi resetați la starea computațională de bază $|0\rangle$ și același qubit țintă ca cel rezultat prin aplicarea circuitului. Altfel, dacă la orice pas intermediar *n* rezultatele celor două măsurători asupra qubiților de control sunt amândouă 0, atunci qubitul țintă a fost transformat cu $R_z(\theta)$, și procesul se oprește. Probabilitatea ca procesul să se oprească la pasul *n* este așadar: $P(n) = P_{00}[\sum_{k=0}^{n-1}(P_{01} + P_{10} + P_{11})^k] = \frac{5}{8}\sum_{k=0}^{n-1}\left(\frac{3}{8}\right)^k$, $\lim_{n\to\infty}\big(P(n)\big) = \frac{5}{8}\frac{1}{1-\frac{3}{8}} = 1.$

6.3.3. Aproximarea operatorilor unitari

Deoarece mulțimea operatorilor unitari este continuă, o mulțime discretă de porți nu este suficientă pentru a implementa orice operator unitar arbitrar. În schimb, o mulțime discretă poate fi folosită numai pentru a aproxima orice operator unitar arbitrar. Dacă *U* și *V* sunt doi operatori unitari pe același spațiu al stărilor, *U* fiind operatorul dorit a se implementa, *V* este operatorul implementat de fapt, eroarea de aproximare a lui *U* prin *V* este: $E(U,V) \equiv \max_{|\psi\rangle}\|(U-V)|\psi\rangle\|$. Această definire garantează că dacă eroarea de aproximare respectivă este mică, atunci orice operație de măsurare efectuată asupra stării modificate de operatorul implementat de fapt *V*, folosind orice stare inițială și orice operator de măsurare, dă o distribuție de probabilitate similară ca și când aceeași operație de măsurare ar fi fost efectuată asupra stării transformate de operatorul dorit a se implementa *U*. În plus, dacă o succesiune de porți cuantice este folosită pentru a aproxima o altă succesiune de porți cuantice, erorile se acumulează într-un mod cel mult liniar: $E(U_m U_{m-1} \dots U_1, V_m V_{m-1} \dots V_1) \leq \sum_{j=1}^{m} E\big(U_j, V_j\big)$.

6.3.4. Aproximarea operatorului de rotație

Considerând cele două relații anterioare, dacă operatorii în cauză sunt operatori de rotație, eroarea poate fi exprimată în termenii unghiurilor de rotație. În ecuațiile de mai jos, axa *z* poate fi înlocuită cu orice axă de rotație arbitrară: $E\big(R_z(\alpha), R_z(\alpha+\beta)\big) = 2\left(1-\cos\frac{\beta}{2}\right)$
Această relație poate fi generalizată la aproximații prin rotații succesive identice:

$$E(R_z(\alpha), R_z(\theta)^n) = 2\left(1-\cos\frac{\big((n\theta)\bmod 2\pi\big)-\alpha}{2}\right)$$

Principiul „pigeonhole” implică faptul că dacă θ este un multiplu irational al lui 2π, atunci, pentru orice α și orice acuratețe dorită $\delta > 0$ este posibil a se găsi n, astfel încât $\big((n\theta)\mathrm{mod}2\pi\big) - \alpha < \delta$. Dar, deoarece $\big((n\theta)\mathrm{mod}2\pi\big)$ tinde să se apropie de α, în același timp $\cos\frac{((n\theta)\mathrm{mod}2\pi)-\alpha}{2}$ tinde către 1. Și ca urmare, $E(R_z(\alpha), R_z(\theta)^n)$ în ecuația de mai sus tinde să se apropie de 0. În concluzie, pentru orice α și orice acuratețe dorită $\epsilon > 0$, există un număr natural n_ε, care depinde de α și de acuratețea dorită: $E(R_z(\alpha), R_z(\theta)^{n_{\alpha,\varepsilon}}) < \frac{\epsilon}{3}$

Mai departe, poate fi demonstrat că mulțimea discretă de porți cuantice alcătuită din grupul normalizator, la care se adaugă poarta Toffoli este universală pentru calculul cuantic; mai precis, o operație arbitrară unitară pe *d* qubiți poate fi aproximată cu o acuratețe arbitrar de mare folosind un circuit compus numai din aceste porți cuantice. Circuitul obținut va trebui în cele mai multe cazuri să fie aplicat de mai multe ori, numărul aplicărilor fiind direct proporțional cu acuratețea de aproximare dorită. Mai întâi, deoarece operatorii Pauli satisfac $HZH = X$, orice operator unitar pe un singur qubit poate fi descompus într-un produs de rotații în jurul axei și operatori Hadamard:

$$U \cong R_z(\alpha)R_x(\beta)R_z(\gamma) = R_z(\alpha)HR_z(\beta)HR_z(\gamma)$$

Apoi, din ultimele două ecuații, rezultă că circuitul de mai sus, la care se adaugă două sau mai multe porți Hadamard, poate fi folosit în a aproxima cu succes orice operator unitar pe un singur qubit: $E(U, R_z(\theta)^{n_\alpha}HR_z(\theta)^{n_\beta}HR_z(\theta)^{n_\gamma}) < 3\frac{\epsilon}{3} + 2E(H,H) = \epsilon$

Mai mult, deoarece orice operator unitar pe un număr arbitrar de qubiți *d* poate fi descompus într-un produs de operatori unitari de nivel doi pe *d* qubiți, și deoarece acești operatori unitari de nivel doi pe *d* qubiți pot la rândul lor să fie implementați exact (i.e. nici un fel de aproximație necesară) folosind numai porți pe un singur qubit și porți CNOT, rezultă că orice operator unitar pe un număr arbitrar de qubiți *d* poate fi implementat cu aproximație, cu o acuratețe arbitrară ϵ, folosind numai porți Hadamard, schimbare de fază, CNOT și Toffoli, adică numai porți din baza Shor.

Considerente de performanță

Demonstrația directă prezentată anterior pentru universalitatea bazei Shor ridică anumite întrebări în legătură cu eficiența modelelor bazate pe circuite cuantice și cu cantitatea de resurse de calcul necesare pentru aproximarea operațiilor unitare. Din păcate, nu este posibil a se aproxima operatori unitari generici pe *d* qubiți folosind un circuit a cărui mărime (adică număr de porți conținute) este polinomială în *d*.

7. Transformarea Fourier

În modelul de calcul cuantic se pot rezolva unele probleme de calcul pentru care nu s-a găsit încă nici o soluţie eficientă de implementare în modelul de calcul clasic (incluzând modelul probabilistic). Cel mai cunoscut exemplu în acest sens este problema determinării factorilor primi ai unui număr natural stocat pe n biţi. Până în prezent, cei mai buni algoritmi clasici au nevoie de un număr paşi care creşte aproape liniar cu valoarea numărului de factorizat, ceea ce înseamnă că este o creşte exponenţială în funcţie de n. În contrast, există un algoritm cuantic poate efectua acelaşi calcul folosind numai $O(n^2 \log n \log\log n)$ operaţii. Astfel, un calculator cuantic poate factoriza un număr natural cu un câştig de performanţă exponenţială faţă de un calculator clasic. Se poate ridica următoarea întrebare: ce alt tip (or tipuri) de probleme pot fi rezolvate de un calculator cuantic cu un câştig de performanţă exponenţial faţă de calculatoarele clasice? [3] [4]

Unul din ingredientele cheie care sugerează un posibil răspuns la această întrebare este transformarea Fourier cuantică, transformare folosită atât în rezolvarea problemei factorizării unui număr natural, cât şi a multor altor probleme interesante. Transformarea Fourier

cuantică este un algoritm cuantic eficient pentru calcularea transformatei Fourier a unei mulţimi de amplitudini cuantice. Algoritmul nu aduce nici o îmbunătăţire în performanţa de calcul a transformatei Fourier pentru mulţimi de date clasice. Dar acest algoritm permite estimarea fazei, aproximarea valorilor singulare ale unui operator unitar. Astfel acest algoritm poate fi folosit în rezolvarea unor altor probleme interesante cum ar fi de exemplu problema factorizării unui număr natural şi problema găsirii ordinului unui element dintr-un grup finit (aceste două probleme fiind de fapt echivalente). Algoritmul de estimare a fazei poate fi de asemenea combinat cu algoritmul cuantic de căutare într-o mulţime nesortată pentru a rezolva problema numărării soluţiilor unei probleme de căutare. Transformarea Fourier cuantică poate fi în plus folosită pentru a rezolva problema subgrupurilor ascunse, o generalizare a problemelor de estimare a fazei şi de găsire a ordinului, care are printre cazurile sale speciale un algoritm cuantic eficient pentru rezolvarea problemei discrete a logaritmilor, o altă problemă considerată greu tractabilă pentru un calculator clasic.

7.1. Transformarea Fourier cuantică

O descoperire foarte importantă în calculul cuantic a fost că unele dintre aceste transformări pot fi implementate mult mai eficient folosind un calculator cuantic în comparaţie cu un calculator clasic. Transformarea Fourier cuantică (QFT) este similară ca semnificaţie cu transformarea Fourier discretă clasică (DFT), doar că notaţia şi semnificaţia datelor este oarecum diferită. Astfel, transformarea Fourier cuantică [30] este prin definiţie un operator liniar pe un spaţiu vectorial de dimensiune N care transformă mulţimea de N vectori ortonormaţi în starea computaţională de bază $|0\rangle, |1\rangle, \dots, |N-1\rangle$ conform relaţiei:

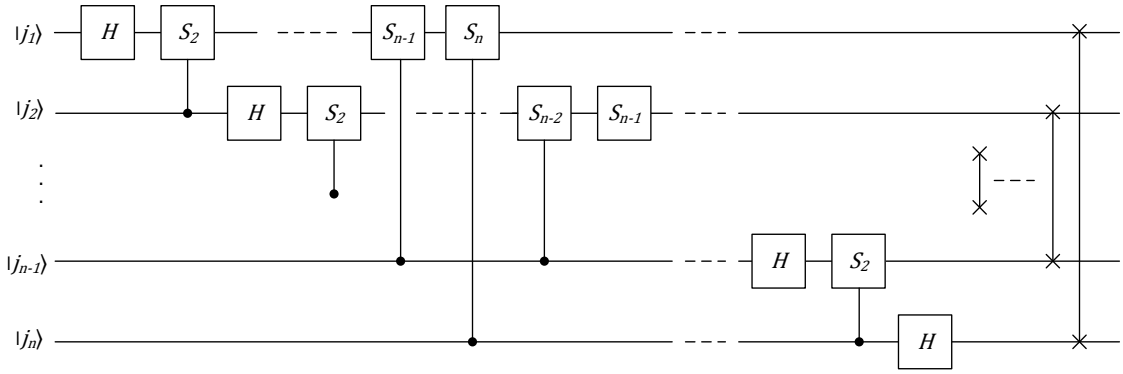
$$|j\rangle \xrightarrow{QFT} \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{\frac{2\pi i j k}{N}} |k\rangle$$

Astfel, orice vector $|x\rangle$ din spaţiul respectiv, descompus conform bazei formate de vectorii în stare computaţională de bază, poate fi transformat: $|x\rangle = \sum_{j=0}^{N-1} x_j |j\rangle \xrightarrow{QFT} |y\rangle = \sum_{k=0}^{N-1} y_k |k\rangle$ unde mulţimea de numere complexe y_k este DFT-ul mulţimii x_j.

7.2. Implementarea transformării Fourier cuantice

Folosind aceste formule se poate construi un circuit eficient pentru implementarea transformării Fourier cuantice, alcătuit numai din porţi Hadamard şi din porţi condiţionate S_l - transformare de fază: $|j_l\rangle|\psi\rangle = |j_l\rangle(a|0\rangle + b|1\rangle) \xrightarrow{C^1(S_l)} |j_l\rangle\left(a|0\rangle + be^{2\pi i j_l 2^{-l}}|1\rangle\right)$

Adăugarea porţilor de inversare a qubiţilor de la capătul circuitului este de fapt opţională deoarece la citirea rezultatului se pot citi qubiţii în ordine inversă.



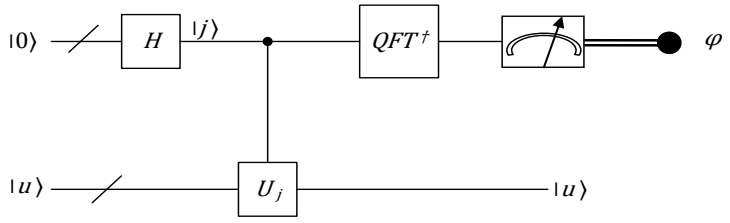
7.3. Calculul complexităţii

$$C\big(QFT(n)\big) = n + (n-1) + \cdots + 2 + 1 + \left\lfloor \frac{n}{2} \right\rfloor = \frac{n(n+1)}{2} + \left\lfloor \frac{n}{2} \right\rfloor \leq \frac{n^2}{2} + n.$$

Complexitatea algoritmului de calcul al transformatei Fourier cuantice este aşadar $O\left(\frac{n^2}{2}\right)$. Aceasta reprezintă o îmbunătăţire exponenţială în performanţă, în comparaţie cu cel mai bun algoritm clasic cunoscut – Fast Fourier Transform (FFT) a cărui complexitate este exponenţială: $O(n2^n)$. Transformata Fourier este foarte des întâlnită într-o mulţime de aplicaţii incluzând domenii ca prelucrarea semnalelor, recunoaşterea vorbirii – în care primul pas este de a transforma Fourier sunetul digitizat, şi multe altele. Din păcate problema principală este că amplitudinile semnalelor nu pot fi accesate direct prin măsurătoare într-un calculator cuantic. Astfel nu se cunoaşte nici o modalitate de a determina amplitudinile transformate Fourier ale stării originale. Şi mai rău, nu există nici o modalitate generică eficientă de a pregăti starea originală cuantică pentru a fi transformată Fourier. De aceea, găsirea unor posibilităţi de folosire practică a performanţelor teoretice ale transformatei Fourier cuantice este mult mai subtilă decât pare.
Construcţia circuitului cuantic care implementează transformarea Fourier cuantică nu necesită porţi a căror precizie ar trebui să crească exponenţial în funcţie de numărul de qubiţi. Totuşi acest tip de precizie exponenţială nu este niciodată necesară într-un circuit cuantic cu un număr polinomial de porţi. De exemplu, dacă se consideră U este QFT ideală pe n qubiţi şi V este transformarea reală care rezultă dacă porţile controlate $C^1(S_k)$ din circuitul de mai sus ar fi implementate cu o precizie finită polinomială $\Delta = \frac{1}{p(n)}$. Atunci marja de eroare este o funcţie $\Theta\left(\frac{n^2}{p(n)}\right)$. Astfel precizie de tip polinomial în implementarea fiecărei porţi este suficientă pentru a garanta o acurateţe polinomială a întregului circuit.

8. Estimarea fazei

Dacă se consideră un operator unitar U care are un vector singular $|u\rangle$ şi valoarea singulară corespunzătoare $e^{2\pi i\varphi}$: $U|u\rangle = e^{2\pi i\varphi}|u\rangle$, se doreşte estimarea fazei necunoscute φ, care este un număr real subunitar. Algoritmul de estimare a fazei, are patru etape principale.



În mod intuitiv, presupunând că faza ce se doreşte calculată este reprezentată în mod exact printr-o fracţie binară $\varphi = \frac{1}{2^t}\sum_{l=1}^{t} \varphi_l 2^{t-l}$, cu $\varphi_l \in \{0,1\}$. O măsurătoare în starea computaţională de bază va întoarce cu exactitate (i.e. probabilitate 1) valoarea lui φ. În realitate, dacă φ este un număr irațional, algoritmul produce o aproximare:

$$\frac{1}{2^{\frac{t}{2}}}\left(\sum_{k=0}^{2^t-1} e^{2\pi ik\varphi}|k\rangle\right)|u\rangle \xrightarrow{QFT^\dagger} |\tilde{\varphi}\rangle|u\rangle$$

Unde $|\tilde{\varphi}\rangle$ este o stare care prin măsurătoare dă o bună estimare a fazei φ.
În cazul ideal studiat anterior, s-a considerat că faza de calculat φ poate fi reprezentată în mod exact pe un număr de t biţi. În cazul general, un număr real poate fi reprezentat pe un număr fix de biţi numai cu o anume marjă de eroare. Algoritmul de estimare a fazei prezentat

anterior produce în cazul real o bună aproximaţie a valorii reale, cu o probabilitate înaltă. Fie b cea mai bună aproximare maximală întreagă a lui φ pe t biţi. Deci b este un număr întreg astfel încât $b/2^t$ este cea mai bună aproximare a lui φ:

$$0 \le b \le 2^t - 1, \qquad \varphi - \frac{1}{2^t} \le \frac{b}{2^t} \le \varphi$$

Şi dacă se defineşte eroarea de aproximare $\delta \equiv \varphi - b/2^t$, ea satisface: $0 \le \delta \le \frac{1}{2^t}$.

Se demonstrează că algoritmul de estimare a fazei pentru cazul real produce un δ mic cu probabilitate mare. Pentru a obţine φ aproximat pe n biţi cu o probabilitate de succes de cel puţin $1 - \varepsilon$, trebuie folosit cel puţin un număr de qubiţi egal cu: $t = n + \left\lceil log\left(2 + \frac{1}{2\varepsilon}\right)\right\rceil$.

Algoritm: Estimarea cuantică a fazei

Intrare:

1. O cutie neagră care implementează operatorul controlat $C^1(U^j)$, cu j număr întreg
2. Un registru pe $t = n + \left\lceil log\left(2 + \frac{1}{2\varepsilon}\right)\right\rceil$ qubiţi iniţializaţi la $|0\rangle$
3. Un registru pregătit în starea $|u\rangle$, unde $|u\rangle$ este o stare singulară a operatorului U, cu valoarea singulară corespunzătoare $e^{2\pi i\varphi_u}$

Ieşire:

1. Numărul întreg pe n biţi $\tilde{\varphi}_u$, care este o aproximare pe n biţi a lui φ_u

Timp de rulare:

1. $O(t^2)$ operaţii unitare şi câte o invocare pentru fiecare $C^1(U^j)$ cutie neagră.
2. Probabilitatea de a obţine rezultatul dorit este cel puţin $1 - \varepsilon$.

Procedură:

1. $|0\rangle|u\rangle$ — starea iniţială
2. $\longrightarrow \frac{1}{2^{\frac{t}{2}}} \sum_{k=0}^{2^t-1} |k\rangle|u\rangle$ — crearea superpoziţiei
3. $\longrightarrow \frac{1}{2^{\frac{t}{2}}} \sum_{k=0}^{2^t-1} |k\rangle U^k|u\rangle$ — aplicarea cutiilor negre
4. $\longrightarrow \frac{1}{2^{\frac{t}{2}}} \sum_{k=0}^{2^t-1} e^{2\pi ik\varphi}|k\rangle|u\rangle$ — rezultatul aplicării cutiilor negre
5. $\longrightarrow |\tilde{\varphi}_u\rangle|u\rangle$ — aplicarea transformării Fourier cuantice
6. $\longrightarrow \tilde{\varphi}_u$ — măsurarea registrului de control

9. Aplicarea algoritmilor cuantici la probleme concrete

9.1. Determinarea ordinului

În limbajul teoriei numerelor naturale, problema se formulează astfel: pentru numere întregi pozitive x şi N, unde $x < N$ care nu au factori comuni (sunt co-prime), i.e. $\text{cmmdc}(x, N) = 1$, ordinul lui x modulo N se defineşte ca fiind cel mai mic număr întreg pozitiv, r, astfel încât $x^r = 1 (\text{mod } N)$. Se poate demonstra în teoria numerelor că r există întotdeauna şi că $r \le N$. Problema poate fi enunţată mai general în teoria grupurilor ciclice finite.

Determinarea ordinului este considerată ca fiind o problemă dificilă de rezolvat pe un calculator clasic, în sensul că nu se cunoaşte nici un algoritm pentru rezolvarea acestei probleme care să folosească resurse polinomiale în $O(L)$, unde L este numărul necesar de biţi pentru exprimarea problemei, în acest caz $L \equiv \lceil \log_2(N) \rceil$. În linii generale, algoritmul cuantic

de rezolvare a problemei determinării ordinului este chiar algoritmul de estimare a fazei, aplicat următorului operatorului U care înlocuieşte cutia neagră:

$$\begin{cases} U|y\rangle \equiv |xy(\text{mod } N)\rangle & \forall y \in \{0,1\}^L, \quad 0 \le y \le N-1 \\ U|y\rangle \equiv |y\rangle & \forall y \in \{0,1\}^L, \quad N \le y < 2^L - 1 \end{cases}$$

Algoritmul de estimare a fazei necesită pe lângă specificarea operatorului unitar şi specificarea unei stări singulare a operatorului respectiv. Stările singulare corespunzătoare:

$$|u_s\rangle \equiv \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} \left[e^{\frac{-2\pi i s k}{r}} |x^k (\text{mod } N)\rangle \right] \quad \forall s \in \mathbb{N}, \ 0 \le s \le r-1$$

Pentru a putea aplica algoritmul de estimare cuantică a fazei mai trebuie satisfăcute următoarele două condiţii:

1. Găsirea unei proceduri eficiente pentru a implementa $C^1(U^{2^j})$, pentru orice întreg j.
2. Găsirea unei modalităţi eficiente de a prepara o stare singulară $|u_s\rangle$, cu o valoare singulară netrivială, or o superpoziţie de astfel de stări singulare.

Dacă aceste condiţii sunt îndeplinite, se poate apoi calcula r conform relaţiei $r = \frac{s}{\tilde{\varphi}_s}$.

Probabilitatea de a obţine ordinul r căutat este de cel puţin $\frac{1}{4}$. Calculul complexităţii de timp se poate face considerând fiecare etapă a algoritmului. Pentru efectuarea transformării Hadamard sunt necesare $O(L)$ porţi; QFT necesită $O(L^2)$ porţi. Costul cel mai mare al algoritmului este cel introdus de etapa exponenţierii modulare care foloseşte $O(L^3)$ porţi, pentru a obţine aşadar un total de $O(L^3)$ porţi pentru întregul circuit cuantic. Algoritmul fracţiilor continue adaugă încă $O(L^3)$ porţi pentru un total de $O(L^3)$ porţi necesare pentru obţinerea lui r'. Este necesar a se repeta această procedură de un număr constant de ori.

Algoritm: Algoritm cuantic de determinare a ordinului

Intrare:

(1) O cutie neagră $U_{x,N}$ care implementează transformarea $|z\rangle|y\rangle \longrightarrow |z\rangle|x^z y(\text{mod } N)\rangle$ unde x este co-prim cu N, exprimat pe L biţi.

(2) $t = 2L + 1 + \left\lceil log\left(2 + \frac{1}{2\varepsilon}\right)\right\rceil$ qubiţi iniţializaţi la $|0\rangle$

(3) L qubiţi iniţializaţi la $|1\rangle$.

Ieşire:

Cel mai mic întreg $r > 0$ astfel încât $x^r = 1 \ (\text{mod } N)$

Timp de execuţie:

$O(L)$ operaţii. Reuşeşte cu probabilitate $O(1)$.

Procedura:

1. $|0\rangle|1\rangle$ — Starea iniţială
2. $\longrightarrow \frac{1}{2^{\frac{t}{2}}} \sum_{j=0}^{2^t-1} |j\rangle|1\rangle$ — Crearea superpoziţiei
3. $\longrightarrow \frac{1}{2^{\frac{t}{2}}} \sum_{j=0}^{2^t-1} |j\rangle|x^j (\text{mod } N)\rangle \cong \frac{1}{\sqrt{r}2^{\frac{t}{2}}} \sum_{s=0}^{r-1} \sum_{j=0}^{2^t-1} e^{\frac{2\pi i s j}{r}} |j\rangle|u_s\rangle$ — Aplicarea operatorului $U_{x,N}$
4. $\longrightarrow \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} \left|\frac{\tilde{s}}{r}\right\rangle |u_s\rangle$ — Aplicarea QFT inverse asupra registrului de control
5. $\longrightarrow \frac{\tilde{s}}{r}$ — Măsurarea registrului de control
6. $\longrightarrow r$ — Aplicarea algoritmului fracţiilor continue

9.2. Aplicaţie: factorizarea numerelor naturale

Problema factorizării: dându-se un număr întreg pozitiv N se doreşte descompunerea sa ca produs de factori primi. Această problemă a factorizării se dovedeşte a fi de fapt echivalentă cu problema determinării ordinului, în sensul că algoritmul pentru găsirea ordinului poate fi transformat într-un algoritm eficient pentru factorizare. Aproape toţi paşii algoritmului pot fi implementaţi eficient chiar şi pe un calculator clasic. Singura excepţie este o subrutină eficientă care să determine ordinul unui număr întreg modulo N. Prin repetarea algoritmului se poate determina în totalitate descompunerea lui N în factori primi.

Algoritm: Reducerea problemei factorizării la problema determinării ordinului

Intrare:

Numărul natural de descompus N

Ieşire:

Un factor ne-trivial al lui N

Timp de execuţie:

$O((\log_2 N)^3)$. Reuşeşte cu probabilitate $O(1)$.

Procedura:

1. Dacă N este par, întoarce factorul 2
2. Pentru fiecare b astfel încât $2 \leq b \leq \lceil \log_2 N \rceil$
   2.1. Dacă $N = a^b$, întoarce a
3. Alege numărul aleator $1 \leq x \leq N - 1$
4. Dacă $f = \text{cmmdc}(x, N) > 1$ atunci întoarce f
5. Apelează sub-rutina de calcul a ordinului r al lui x, modulo N
6. Dacă r este par şi $x^{\frac{r}{2}} \neq -1 (\text{mod } N)$ atunci calculează $f_1 = \text{cmmdc}(x^{\frac{r}{2}} - 1, N)$ şi $f_2 = \text{cmmdc}(x^{\frac{r}{2}} + 1, N)$
7. Dacă $f_1 \neq 1$ return f_1
8. Dacă $f_2 \neq 1$ return f_2
9. Altfel algoritmul eşuează. Întoarce eroare.

Primii doi paşi ai algoritmului întorc un factor sau se asigură că N este un număr impar cu cel puţin doi factori. Aceşti paşi sunt efectuaţi folosind $O(L^3)$ operaţii. Următorii doi paşi întorc un factor sau întorc un element aleator din $\mathbb{Z}_N^*$ cu complexitate $O(L^2)$. Ultimii trei paşi determină un factor şi reuşesc cu o probabilitate de cel puţin $\frac{1}{2}$.

Acestea sunt aplicaţiile principale cu cea mai mare posibilitate de aplicare, din punct de vedere practic. Dar prin folosirea transformării Fourier cuantice se poate ataca o gamă de probleme mult mai largă [6]. Problema cea mai generală care cuprinde ca nişte cazuri particulare ale sale toate aplicaţiile exponenţial eficiente ale transformării Fourier cuantice, este problema subgrupului ascuns. Această problemă poate fi gândită ca fiind o generalizare a problemei de găsire a perioadei unei funcţii periodice, în contextul în care structura domeniului şi a codomeniului funcţiei respective sunt foarte complicate. În limbajul algebric al teoriei grupurilor, această problemă poate fi exprimată în cazul cel mai general astfel [29]: fie f o funcţie definită pe un grup G finit generat, cu valori într-o mulţime finită X astfel încât f este constantă pe orice coset definit de un subgrup necunoscut K, valorile constante respective fiind diferite. Dacă se dă o cutie neagră cuantică care să implementeze transformarea unitară $U|g\rangle|h\rangle = |g\rangle|h \oplus f(g)\rangle$, unde $g \in G$, $h \in X$ şi $\oplus$ este o operaţie binară în X aleasă corespunzător; să se găsească o mulţime generatoare pentru K.

9.3. Limbaje de programare pentru calculul cuantic

Experimentarea cu algoritmii cuantici cunoscuți și, mai ales, proiectarea unor algoritmi cuantici noi, sunt domenii a căror progres este mult îngreunat de lipsa unor limbaje de

programare și a unor platforme de dezvoltare software adecvate. Modelul bazat pe circuite cuantice prezentat anterior reprezintă un prim pas important în această direcție, acest model cuprinzând în detaliu cele mai importante aspecte ale algoritmilor cuantici. În plus, circuitele cuantice pot fi ușor implementate prin hardware specific calculului cuantic, după ce sunt reduse la mulțimea respectivă de porți cuantice elementare. Dar, pentru a putea dezvolta cu ușurință programe de calcul cuantic mai complicate, un nivel de abstracție mai înalt este necesar [24].

9.3.1. Programe cuantice

Progrese importante au fost făcute în ultima vreme, în abstractizarea componentelor hardware cuantice prin folosirea unei metode în multe privințe asemănătoare cu metoda care a fost folosită în calculul clasic acum mai multe decade [38]. Făcând abstracție momentan de toate detaliile complexe specifice diferitelor arhitecturi de calcul și a diferitelor paradigme de programare, un calculator clasic poate fi redus din punct de vedere conceptual la [54]:

PROGRAM = DATE + INSTRUCȚIUNI

Această schemă de bază a fost generalizată pentru a cuprinde programarea cuantică [9], prin adăugarea unor elemente specifice calculului cuantic:

PROGRAM CUANTIC = DATE CUANTICE +
OPERAȚII CUANTICE +
INSTRUCȚIUNI

În relația conceptuală de mai sus, se presupune că un calculator cuantic, care rulează un asemenea program cuantic, este compus dintr-un dispozitiv cuantic, capabil să manipuleze datele cuantice – reprezentate prin qubiți care pot fi adresați. Acest dispozitiv execută un set predefinit de operații cuantice, care pot fi împărțite în două categorii:

- operații unitare: transformă datele cuantice prin menținerea proprietăților lor cuantice
- operații de măsurare: inspectează datele cuantice transformându-le în date clasice

În implementarea practică, mulțimea predefinită de operatori unitari trebuie:

- să fie finită, pentru a putea fi implementată în hardware
- să fie universală, pentru ca toate programele cuantice să poată fi create
- fiecare operator să acționeze asupra unui spațiu Hilbert finit dimensional

Un exemplu de asemenea mulțime este formată din porțile cuantice din baza Shor: Hadamard, schimbarea de fază, CNOT, Toffoli. Trebuie menționat faptul că această mulțime predefinită de operatori nu este neapărat necesar să fie minimală din punct de vedere teoretic. Dar este evident că din considerente de natură practică sau economică, această mulțime va conține întotdeauna un număr relativ mic de porți cuantice. Această mulțime predefinită de operatori unitari poate fi folosită pentru a defini operatori din ce în ce mi complicați, în același fel cum porțile cuantice sunt compuse împreună pentru a forma circuite cuantice. Deoarece toate mulțimile cunoscute de operatori unitari care sunt finite și universale pot numai aproxima unii operații cuantice mai complicate, rezultă că anumiți operatori nu vor avea o implementare exactă, ci numai una aproximativă. Măsurătorile sunt considerate ca fiind reprezentate prin operatori de proiecție construiți folosind stările computaționale de bază. Rezultatele măsurătorilor sunt exprimate prin biți clasici care pot fi citiți din dispozitivul cuantic.

9.3.2. Limbaje pentru programarea cuantică

În mod asemănător cu calculul clasic, primul nivel de abstracție folosit în programarea cuantică este limbajul de asamblare [40]. Și, din nou ca în cazul clasic, modelul de calcul cuantic va trebui construit pe baza unei arhitecturi specifice a mașinii cuantice respective. Totuși nu este necesar a se specifica detaliile de implementare hardware ale dispozitivului de calcul cuantic. În același mod în care limbajul de asamblare clasic (și unele limbaje de nivel

mai înalt) sunt construite pa baza unor concepte abstracte cum ar fi: stive, memorie heap, etc., limbajul de asamblare cuantic trebuie să aibă în vedere un set de concepte abstracte.

Circuite cuantice

Circuitele cuantice au fost primul model de calcul cuantic introdus. Acest model presupune existența următoarelor ingrediente folosite în calcul:

- un dispozitiv de intrare care prepară o mulțime inițială de qubiți, care pornește calculul
- o mulțime finită de porți cuantice, fiecare acționând asupra unui număr finit de qubiți, conectate secvențial și în paralel, formând un graf direcționat aciclic – un circuit cuantic
- un dispozitiv de măsurare care oferă la ieșire o mulțime de biți clasici care pot fi citiți. Aceste operații de măsurare pot fi întotdeauna efectuate la ieșirea circuitului cuantic, după ce toate porțile cuantice au terminat procesarea qubiților.

Mașina Turing cuantică

Acest model, în care se adaugă elementele specifice calculului cuantic la modelul standard clasic al mașinii Turing probabiliste este foarte folositor în studiul claselor de complexitate de calcul, dar este dificil de folosit în construcția programelor cuantice mai mare sau în dezvoltarea de algoritmi cuantici. Acest model este bazat pe bine cunoscuta arhitectură a mașinii Turing care conține o bandă infinită unidimensională pe care pot fi înscrise caractere dintr-o mulțime finită. Această bandă se poate mișca spre stânga sau spre dreapta, și se presupune existența unui dispozitiv de citire – scriere, care poate acționa numai asupra caracterului curent. Diferențele principale dintre mașina Turing clasică probabilistă și mașina Turing cuantică sunt:

- funcția de tranziție pentru mașina probabilistă întoarce probabilități (i.e. numere reale ne-negative subunitare), în timp ce funcția de tranziție pentru mașina cuantică întoarce numere complexe, a căror modul ridicat la pătrat reprezintă probabilități.
- la fiecare pas de execuție, mașina probabilistă alege o anume tranziție în mod aleator, în timp ce mașina cuantică execută toate tranzițiile posibile în paralel.

Modelul cuantic de memorie cu acces aleator (QRAM)

Aceasta este modelul de calul cuantic cel mai potrivit programatorilor obișnuiți deja să programeze clasic, deoarece oferă o paradigmă de programare apropiată de cea clasică permițând în același timp atât specificarea relativ ușoară a algoritmilor cuantici cât și chiar a limbajelor de programare cuantică de nivel mai înalt. Acest model presupune o arhitectură hardware care este de fapt doar o extensie cuantică a celei clasice. Acest hardware este compus din punct de vedere conceptual dintr-un un dispozitiv clasic și un dispozitiv cuantic. Se definesc tipuri de instrucțiuni cuantice pe care acest model de calcul trebuie să-l ofere:

- inițializează registrul cuantic de qubiți cu o stare computațională de bază.
- selectează o submulțime de qubiți din registrul cuantic cu scopul de a-i folosi în procesări ulterioare, fie transformări unitare sau operații de măsurare
- aplică o transformare unitară asupra registrului cuantic (sau a unei submulțimi de qubiți)
- compune două transformări unitare, adică execută-le în mod secvențial
- aplică produsul tensorial asupra a două transformări unitare, adică execută-le în paralel
- măsoară registrul cuantic (sau o submulțime de qubiți) și transferă rezultatul într-un registru clasic de biți

De exemplu, următorul set de instrucțiuni construiește o pereche EPR și măsoară primul qubit din pereche, obținând 0 sau 1, cu probabilitate egală:

```
qbit q[ 2 ] = {0, 0};              // inițializare
let U1 = tensor( H, I2 );          // produs tensorial
let Epr = concat( U1, CNOT );      // compunere de operatori
Epr( q );                          // aplicarea operatorilor
bit r[ 1 ] = measure( q[ 0 ] );    // măsurare
```

De asemenea, nu este necesar să se ofere un set special de instrucțiuni cuantice pentru controlul fluxului de execuție în dispozitivul cuantic, cum ar fi de exemplu instrucțiuni de tipul if-then-else. Acest fapt este susținut din două motive:

- orice astfel de instrucțiuni pentru execuție condiționată pot fi implementate în modelul de calcul cuantic folosind operatori condiționați
- instrucțiunile de execuție condiționată pot fi implementate folosind numai biți clasici obținuți ca urmare a efectuării unor operații de măsurare, de fiecare dată când se impune.

Limbaj de asamblare bazat pe manipularea șirurilor

Acest limbaj de nivel scăzut oferă un compromis între modelul bazat pe mașina Turing cuantică și QRAM. Este ușor de folosit în implementarea multor algoritmi reali, și abstractizează resursele hardware interne destul de bine pentru a-l face util în analiza claselor de complexitate de calcul [23]. În acest model, se definește o mulțime finită de caractere C. Se presupune deasemenea că există un număr (posibil infinit) de locații de memorie, fiecare din aceste locații fiind adresabilă printr-un șir de caractere peste C și fiecare locație conținând cel mul un șir de caractere peste C. Inițial, toate aceste locații sunt resetate la șiruri vide. Dacă s este un șir de caractere peste C, se folosește notația uzuală din C++: *s pentru a reprezenta șirul de la adresa s. Așadar șirul *s este și el alcătuit tot din caractere peste C. Fiecare operator (fie el unitar sau de proiecție) este indexat printr-un șir de caractere peste C. În orice moment al executării programului, starea sistemului cuantic este reprezentat prin starea $|\psi\rangle$ în $H_{s_1} \otimes \ldots \otimes H_{s_k}$. Un limbaj cuantic de asamblare poate fi așadar definit prin următoarele comenzi: - Instrucțiuni pentru operațiile clasice:

- InputTo *s
- OutputFrom *s
- AppendTo x, *s
- DeleteLast *s
- ConditionalJump x, *s, n

- Comenzi pentru operațiile cuantice:

- Apply s, s1, s2, … sk
- Observe s, s1, s2, … sk, *s'

În acest model, se consideră că un program calculează o problemă π dacă, pentru orice șir de caractere de intrare s, programul procesează acest șir și probabilitatea de a obține șirul de ieșire corect este mai mare decât probabilitatea de a obține orice alt șir – incorect. Mai mult, probabilitatea ca programul să nu se oprească niciodată trebuie să fie 0.

9.3.3. Limbaje de programare cuantică de nivel înalt

Modelele de calcul cuantic prezentate anterior reprezintă un prim nivel de abstracție care poate fi folosit în dezvoltarea de algoritmi cuantici or în simularea lor pe mașini de calcul clasice [42]. Totuși, aceste limbaje au capabilități oarecum limitate. De exemplu, modelul QRAM operează numai cu șiruri de qubiți, în timp ce modelul bazat pe manipularea șirurilor de caractere operează numai cu astfel de șiruri. Pentru a defini limbaje de programare cuantică care operează la un nivel mai înalt, trebuie avut în vedere:

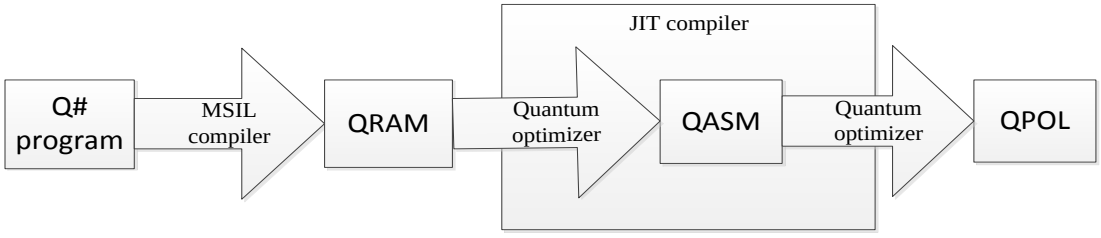
- alegerea uneia dintre următoarele opțiuni:
  - un limbaj care este bazat pe un limbaj de programare clasică, cel mai probabil unul existent. Acest limbaj clasic trebuie dezvoltat să cuprindă cerințele calculului cuantic.
  - un limbaj de programare cuantic de sine stătător, un limbaj care conține numai operații de calcul cuantice. Într-un astfel de limbaj de programare, programele trebuie implementate folosind numai funcțiuni de programare reversibilă. Va fi astfel necesar a se apela la „scratch pad" cuantic pentru a implementa funcțiile clasice ireversibile.
- alegerea unei paradigme de programare: imperativă, funcțională, logică.

Au fost definite deja mai multe astfel de limbaje de programare de nivel înalt pentru calculul cuantic. Unele dintre ele folosesc paradigma de programare imperativă, cum este de exemplu QCL – care este de asemenea un limbaj de sine stătător, sau Q – care a fost definit ca o extensie a lui C++ [37]. Altele folosesc paradigma de programare funcțională, cum ar fi de exemplu QFC – care folosește numai instrucțiuni clasice de control al fluxului, sau QML – în care atât datele cât și instrucțiunile de control sunt cuantice. Dar, deja, în limbajele de programare clasică, paradigmele de programare încep să se contopească împreună. Un foarte bun exemplu în acest sens este oferit de evoluțiile recente din platforma .Net. C# a pornit ca un limbaj prin definiție imperativ, dar apoi, de la versiunea 3.0 încolo, prin adăugarea extensiei LINQ, a început să ofere facilități specifice programării funcționale, ca de exemplu închideri funcționale, șabloane de meta-programare (i.e. programe care manipulează alte programe), etc. În plus de aceasta, un limbaj de programare funcțională de sine stătător a fost adăugat la platforma .Net: F#. În acest fel, prin folosirea interoperabilității dintre C# și F# oferite de platforma .Net, paradigmele de programare funcțională și imperativă pot fi contopite în cadrul aceluiași program, care trebuie împărțit în componentele corespunzătoare. Considerând aceste aspecte, o abordare promițătoare pentru limbajele de calcul cuantic ar fi definirea unui nou limbaj, Q#, bazat pe C#. Acest limbaj:

- ar folosi toate capabilitățile curente oferite de C# pentru a efectua operațiile clasice, făcând apel atât la paradigma imperativă cât și la cea funcțională
- ar oferi un nivel extra de funcționalități pentru programarea cuantică. Deoarece aceste funcționalități sunt bazate în principiu pe aplicarea de operatori, calea cea mai convenabilă de urmat este de a folosi un fel de sintaxă de programare funcțională

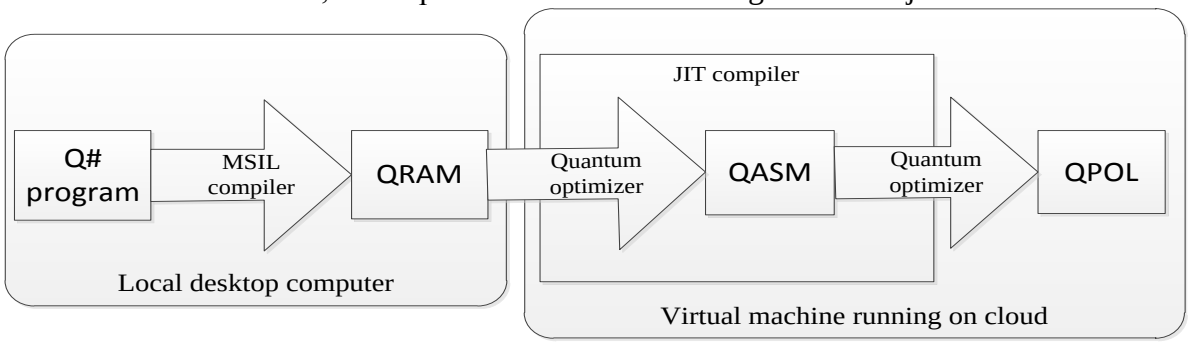
Un alt avantaj oferit de această nouă abordare ar fi oferit de felul în care limbajele .Net sunt compilate. Ele nu sunt compilate direct, ci în două etape: în prima etapă, programul originar de compilat este translatat în cod MSIL (Microsoft independent language). Apoi, la pasul al doilea, care de cele mai multe ori se poate petrece chiar în timpul rulării, prin funcționalitatea oferită de compilatorul JIT (just-in-time), codul MSIL este convertit în cod mașină nativ, executabil. Un limbaj de programare de nivel înalt pentru calculul cuantic bazat pe o extensie a limbajului C# s-ar potrivi foarte bine din această perspectivă. Aspectele de calcul cuantic ale programului vor deveni cod de tip MSIL cuantic – spre exemplu, o variantă bazată pe modelul QRAM. Apoi compilatorul JIT va trebui să convertească aceste instrucțiuni MSIL cuantice în cod mașină specific pentru acel hardware cuantic folosit pe calculatorul respectiv, în timpul acestui proces de conversie urmând a se opera optimizările necesare.
Procesul de compilare este reprezentat în mod schematic mai jos, unde:

- QRAM: limbaj cuantic intermediar, de nivel scăzut, asemănător cu cel descris anterior
- QASM (Quantum Assembly Language) [50] este o reprezentare bazată pe circuite cuantice; mai precis, un circuit cuantic optimizat compus numai din porți făcând parte din mulțimea universală aleasă de porți cuantice elementare.
- QPOL (Quantum Physical Operations Language) este un limbaj de reprezentare la nivel fizic, folosind parametrii tehnologici specifici.



Dar mai este și un alt avantaj în folosirea platformei .Net pentru definirea unui limbaj de programare pentru calculul cuantic: disponibilitatea platformelor de calcul distribuit (cloud

computing), cum este cea definită de Azure pentru C#. Când se dorește utilizarea programelor de calcul cuantic pentru a efectua simulări pe calculatoare clasice sau când se dorește utilizarea programelor de calcul cuantic pentru a le rula pe dispozitive cuantice, în ambele astfel de cazuri, mașinile de calcul propriu zise necesare ar avea nevoie să ofere capabilități de calcul impresionante, privind viteza și capacitatea de stocare (pentru primul caz) sau costul de întreținere și operare (pentru cazul al doilea) [53]. De aceea este foarte probabil ca aceste calculatoare clasice să fie din categoria super-calculatoarelor sau să fie conectate în rețele de foarte mare viteză. Este recomandat deci ca aceste calculatoare să fie expuse programatorilor din afară, care doresc să programeze folosind calculul cuantic, prin perspectiva platformelor de „cloud computing". Folosind arhitectura .Net Azure, compilarea în cele două faze ale sale, este reprezentată schematic în figura de mai jos.



Spre exemplu, considerând aceeași problemă ca mai sus de generare a unei perechi EPR, un astfel de program scris în Q# ar arăta foarte similar cu un program clasic C#, care folosește în plus extensia LINQ:

```
qbit q[ 2 ] = {0, 0};                          // inițializare
QApply( q => CNOT( H( q[ 0 ] ), q[ 1 ] ) )     // definirea și aplicarea operatorilor unitari
bit r[ 1 ] = QObserve( q[ 0 ] );               // aplicarea operatorului de măsurare
```

10. Contribuții și concluzii

10.1. Contribuțiile autorului

În capitolul 2. autorul acestei teze a conceput câteva demonstrații pentru câteva exemple care dovedesc puterea calculului cuantic și a procesării cuantice a informației, explicând totodată cum sunt ele conectate cu procesele de natură fizică.
Astfel, autorul a analizat în detaliu problema Deutsch-Jozsa, extinsă la funcții booleene in spații finite n - dimensionale. Pentru rezolvarea acestei probleme, autorul a conceput un algoritm probabilistic care este analizat din punct de vedere al performanței de execuție, în comparație cu algoritmul determinist și cu cel bazat pe circuite de calcul. S-a demonstrat astfel eficiența sporită în cazul utilizării circuitelor cuantice de calcul.
De asemenea, autorul a conceput și prezentat o demonstrație a faptului că protocolul de codificare super-densă este sigur din punct de vedere a securității: dacă o entitate externă interceptează qubitul care este transmis nu poate deduce nimic în legătură cu informația transmisă.
În capitolul 3. au fost analizate o serie de demonstrații concepute de autorul acestei teze care justifică reprezentarea grafică a qubiților prin sfere unitare tridimensionale și permit modelarea funcționării calculatoarelor cuantice prin rotații tridimensionale, doar în jurul axelor de coordonate [20]. Demonstrațiile respective sunt bazate pe construcție, și ca urmare, folosind formulele demonstrate, operatorii cuantici respectivi pot fi simulați în mod direct printr-un circuit (sau program) de prelucrare grafică. Reprezentările grafice și modelarea

bazată pe transformări geometrice constituie o metodă foarte expresivă de simulare a operațiilor exercitate de către sistemele de prelucrare a informației cuantice.
S-a dovedit aşadar că există o legătura profundă conceptuală între rotaţiile sferice complexe şi operaţiile efectuate asupra unui qubit. Operațiile de calcul cuantic pe un qubit sunt exprimate prin exponențierea operatorilor Pauli, în timp ce transformările geometrice corespunzătoare sunt rotații pe sfera Bloch, în jurul axelor de coordonate.
Datorită acestei legături între prelucrarea grafică şi calculul bazat pe evoluţia qubiţilor, este posibil chiar ca domeniul procesărilor grafice şi al aplicaţiilor multimedia să aibă de beneficiat din dezvoltarea unor algoritmi de calcul cuantic cu aplicaţie directă în respectivul domeniu.
În capitolele 4. și 5. sunt prezentate câteva circuite cuantice concepute de autorul acestei teze:

- un circuit care oferă o implementare minimală a operatorului generic controlat de doi qubiți, implementare care folosește numai porți pe un qubit și porți CNOT.
- un circuit care implementează poarta Fredkin folosind numai o poartă Toffoli și două porți CNOT.
- un circuit care implementează poarta Fredkin folosind numai 6 porți pe doi qubiți [19].
- un circuit care implementează poarta Toffoli generalizată, fără a folosi qubiți de lucru. Acest circuit are o complexitate polinomială de gradul 2.
- un circuit care implementează un operator generic controlat pe un număr oarecare de qubiți, fără a folosi qubiți de lucru. Și acest circuit are tot o complexitate polinomială de gradul 2.

În capitolul 6. este prezentată o analiză riguroasă a universalității porților cuantice. Autorul analizează universalitatea exactă bazată pe mulțimi infinite de porți cuantice pe un qubit, și descompunerea circuitelor complexe în circuite elementare. Acest capitol cuprinde si analiza și calculul complexității, concepute de autor [21].
În continuarea capitolului, este prezentată o demonstrație concepută de autor a universalității aproximative, cu eroare care devine asimptotic neglijabilă, bazată pe o mulțime discretă de porți cuantice: Hadamard, schimbare de fază, CNOT și Toffoli. Această mulțime de porți cuantice elementare (numită bază Shor) este necesară pentru aproximarea operatorilor unitari generali, pe un număr oarecare de qubiți. Deoarece demonstrația este bazată pe construcție, ea include și câteva circuite necesare pentru implementarea unor operatori unitari pe un qubit.
În ultimele trei capitole sunt analizaţi câţiva algoritmi cunoscuți, dintr-o perspectivă nouă, care demonstrează eficienţa crescută a calculatoarelor cuantice, în ceea ce priveşte complexitatea temporală în comparaţie cu algoritmii corespondenți din calculul clasic:

- transformarea Fourier cuantică
- estimarea fazei
- determinarea ordinului
- factorizarea numerelor naturale

În ultimul capitol, autorul analizează limbajele curente de programare cuantică, atât limbaje de bază cât și limbaje de nivel înalt. Autorul propune apoi un astfel de limbaj nou, de nivel înalt, bazat pe arhitectura platformei de dezvoltare .Net. Acest nou limbaj de programare cuantică extinde capabilitățile limbajului C# și folosește atât paradigma de programare imperativă cât și pe paradigma de programare funcțională. Autorul descrie arhitectura compilatorului pentru acest limbaj nou, atât din perspectiva calculului local cât și din punctul de vedere al calculului distribuit, bazat pe ”cloud computing”.

10.2. Concluzii și dezvoltări ulterioare

Calculul cuantic este un domeniu relativ nou, în comparaţie cu metoda de calcul analogic sau cu metoda bazată pe maşina Turing. Cele mai dificile probleme în acest domeniu sunt datorate naturii fizice a proceselor cuantice, care nu este uşor de controlat şi investigat, cât şi

de limitările de natură tehnologică care apar în realizarea componentelor hardware necesare. Din ce în ce mai numeroasele încercări de realizare experimentală a unor maşini de calcul bazate pe principiile mecanicii cuantice se lovesc în principal de probleme datorate greutăţii manipulării materiei la scară cuantică. Deşi experimentele care implementează fizic unii algoritmi cuantici cu dimensiuni mici ale datelor de intrare (doar câţiva qubiţi) au fost încununate de succes, maşinile respective trebuie operate în condiţii foarte restrictive, oferite numai de laboratoare foarte sofisticate de cercetare. În plus, trebuie accentuat că mărimea datelor de intrare folosite este foarte mică, în comparație cu cantitățile de date care în prezent pot fi procesate de un calculator personal obișnuit.
Există şi alt gen de limitări – datorate în principal slabelor resurse „software" de care calculul cuantic dispune în prezent. Proiectarea algoritmilor bazați pa paradigma de calcul cuantic se bazează încă în cea mai mare măsură pe inspirația celor implicaţi [48]. Totuși, programele utilitare care vin în sprijinul dezvoltării de algoritmi, ca de exemplu compilatoarele, interpretoarele sau limbajele de nivel înalt pentru astfel de maşini de calcul cuantic sunt deja în faza cercetării. Ceea ce este mai grav, dar totodată care face acest subiect şi mai interesant, este faptul că datorită schimbării profunde a paradigmei teoretice, cele mai multe dintre aceste resurse software vor trebui probabil revăzute și cel mai probabil schimbate chiar radical. Dacă ele se vor baza pe una din paradigmele actuale de programare (procedurală, funcţională, declarativă, orientată obiect, etc.) rămâne de văzut. Primii paşi în această direcţie au fost deja făcuţi, folosind atât paradigma de programare procedurală cât şi pe cea funcțională. S-ar putea însă ca programarea funcțională să ofere o alternativă mai bună. Asta deoarece prelucrarea în paralel a qubiților, adică transformarea lor prin trecerea lor prin porți cuantice, este exprimată formal folosind teoria operatorilor, care din punct de vedere conceptual sunt funcții pe spațiul stărilor. Există deja platforme de programe de simulare a unor maşini de calcul cuantic, ceea ce înseamnă că, pentru investigarea aspectelor teoretice ale unor algoritmi adresaţi maşinilor cuantice, sunt suficiente mașini Turing clasice; ceea ce, din punct de vedere tehnic, înseamnă că este suficient un PC.
A fost demonstrat deja că un astfel de calculator cuantic, construit la scala calculatoarelor personale actuale, va fi capabil să spargă orice cod de criptare cu chei publice bazat pe factorizarea numerelor naturale. Dar, este evident că acesta nu este un argument prea convingător pentru cei care ar vrea să investească în domenii noi de cercetare, cu aplicabilitate în industrie. Există totuşi o speranţă: în domeniul căutărilor pe baze de date nestructurate, algoritmii de calcul cuantic oferă de asemenea o îmbunătăţire [55]. În plus, poate chiar mai mult ca oricare din beneficiile anterioare, în domeniul aplicaţiilor ştiinţifice, de simulare a proceselor fizice care au loc la scară cuantică, calculul cuantic oferă de asemenea un avantaj evident.
Este așadar de așteptat ca viitoarele calculatoare cuantice să conțină atât subansamble care operează la nivel de calcul cuantic, dar și circuite clasice de calcul, dar comunicarea între cele două tipuri de componente va trebui proiectată în așa fel încât să fie asigurată păstrarea coerenței qubiților aflați în procesare [52].
Modelul de calcul bazat pe circuite cuantice este echivalent cu multe alte modele de calcul propuse anterior, în sensul că alte modele necesită aceleaşi resurse esenţiale pentru aceleaşi tipuri de probleme. Se poate pune întrebarea dacă folosind un model de calcul bazat pe triplete de sisteme cuantice (qutriţi), în loc de sisteme cuantice binare (qubiţi) ar conferi vreun avantaj din punct de vedere computaţional. Deşi din punct de vedere practic se poate ca astfel de avantaje să existe, din punct de vedere teoretic, diferenţa dintre aceste cele două modele este esenţial neglijabilă. Aceasta deoarece, modelul de calcul bazat pe maşinile Turing cuantice, o generalizare a modelului maşinii Turing clasice universale, s-a demonstrat a fiind echivalent cu modelul bazat pe circuite cuantice.

Nu este deocamdată deloc evident dacă presupunerile făcute în teoria circuitelor cuantice sunt total justificate din punct de vedere fizic. Spre exemplu, presupunerile făcute relativ la spaţiul stărilor şi la alegerea stărilor iniţiale sunt doar o simplă alegere. În acest model, spaţiul stărilor este considerat finit dimensional. Şi se poate pune întrebarea dacă prin trecerea la spaţii vectoriale infinite dimensional nu se poate obţine vreun avantaj oarecare.
De asemenea, stările iniţiale ale qubiţilor din circuit sunt considerate a fiind stări computaţionale de bază. Şi se ştie că multe sisteme fizice în natură există în stări puternic entangled [22]. Deci o a doua întrebare care s-ar putea pune este dacă acest tip de stări ar putea fi folosit în obţinerea vreunui avantaj computaţional. Toate acestea ridică semne de întrebare asupra completitudinii modelului de calcul bazat pe circuite cuantice, şi implicit asupra modelului clasic corespunzător.
Domeniul criptografiei cuantice, în care canale de comunicaţie cuantice sunt folosite pentru distribuţia cheilor private folosite în criptografia cu chei publice sau private, este probabil primul în care aplicaţiile comerciale şi-au făcut deja apariţia.
O altă arie de cercetare cu perspective promițătoare, care este deja investigată pe mai multe nivele, este legată de reprezentările transformărilor grafice. Datorită legăturii profunde dintre transformările calculului cuantic pe un qubit și rotațiile grafice, sunt indicații că algoritmii de procesare grafică, și în mod mai general chiar aplicațiile multimedia, sunt candidați cu perspective bune de a fi transformați folosind paradigma calculului cuantic. Dar pentru ca aceasta să se întâmple cu adevărat, corespondența dintre operatorii pe un qubit și rotațiile sferice în spațiul tridimensional trebuie să fie extinsă la transformările pe mai mulți qubiți, poate chiar prin considerarea unor transformări geometrice în spații multidimensionale. Principala problemă este ca reprezentarea grafică să poată simula și qubiți în stări „entangled”.
De aceea este foarte important a se avea întotdeauna în vedere aspectul fizic al prelucrării informaţiei, şi adaptarea modelelor folosite la legile fizice fundamentale.

11. Bibliografie

[1] Aaronson S., Gottesman D.: „*Improved Simulation of Stabilizer Circuits*”, Physical Rev. A, vol. 70, no. 5, 2004

[2] Aharonov Y., Rohrlich D.: „*Quantum Paradoxes: Quantum Theory for the Perplexed*”, Wiley-VCH, Weinheim, 2005

[3] Ambainis A.: „*Quantum walk algorithm for element distinctness*”, SIAM J. Comput. 37/210, 2007

[4] Ambainis A., Kempe J., Rivosh A.: „*Coins make quantum walks faster uantum walk algorithm for element distinctness*”, ACM SIAM Symp. on Discrete Algorithms, 2005

[5] Aspuru-Guzik A., Dutoi A., Love P.J., Head-Gordon M.: „*Simulated quantum computation of molecular energies*”, Science 309/5741, 2005

[6] Bacon D., Dam W.V.: „*Recent Progress in Quantum Algorithms*”, Communications of the ACM, Vol. 53, No. 02, 2010

[7] Barenco A., Bennet C. H., Cleve R., DiVincenzo D. P., Margolus N., Shor P., Sleator T., Smolin J., Weinfurter H.: „*Elementary Gates for Quantum Computation*”, Physical Review Letters 52, 1995

[8] Bennett C. H., DiVincenzo D. P.: „*Quantum Information and Computation*”, Nature, 404, 2000

[9] Bettelli S., Calarco T., Serafini L.: „*Toward an Architecture for Quantum Programming*”, The European Physics J. D, vol. 25, no. 2, pp. 181-200, 2003

[10] Buhrman H., Špalek R.: „*Quantum verification of matrix products*", in Proceedings of the 17th Annual ACMSIAM Symposium on Discrete Algorithms, 2006
[11] Chester M.: „*Primer on Quantum Mechanics*", Dover Publications, 2003
[12] Childs A.M., Cleve R., Deotto E., Farhi E., Gutmann S., Spielman D.A.: „*Exponential algorithmic speedup by quantum walk*", in Proceedings of the 35th ACM Symposium on Theory of Computing, 59–68, 2003
[13] Cormen T. H.: „*Introduction to Algorithms*", Second Edition, MIT Press, 2001
[14] Dasgupta S., Papadimitriou C. H., Vazirani U.: „*Algorithms*", McGraw-Hill, 2006
[15] Deutsch D.: „*Quantum computational networks*", Proc. R. Soc. Lond. A 425, 1989
[16] Deutsch D., Barenco A., Ekert A.: „*Universality in Quantum Computation*", Proc. R. Soc. Lond. A, 1995
[17] Dragne L., Moldoveanu F., Soceanu A.: „*An Object Oriented Framework For Network Management*", 13th International Conference On Control Systems And Computer Science, Bucharest – Romania, 2001
[18] Dragne L.: „*A Component Based Hardware Abstraction Layer For Multimedia Home Platforms*", 14th International Conference On Control Systems And Computer Science, Bucharest – Romania, 2003
[19] Dragne L.: „*Modelling the Controlled Swap Gate with Quantum Circuits*", Annals of DAAAM & Proceedings of the 20th International DAAAM Symposium, 2009
[20] Dragne L.: „*Geometrical Representation of Quantum Bit Operations*", U.P.B. Scientific Bulletin, Bucharest – Romania, 2010
[21] Dragne L.: „*Elementary Gates for Fault-Tolerant Quantum Computing*", Annals of DAAAM & Proceedings of the 21th International DAAAM Symposium, 2010
[22] Ekert A., Jozsa R.: „*Quantum Algorithms: Entanglement Enhanced Information Processing*", Proc. R. Soc. Lond. A, 356(1743), 1998
[23] Fortnow L.: „*One Complexity Theorist's View of Quantum Computing*", Theoretical Computer Science, 292(3), 2003
Farhi E., Goldstone J., Gutmann S.: „*A quantum algorithm for the hamiltonian NAND tree*", Eprint arXiv:quant-ph/0702144, 2007
[24] Gay S. J.: „*Quantum Programming Languages: Survey and bibliography*", Bulletin of the EATCS, 86, 2005
[25] Geroch R.: „*Perspectives in Computation*", The University of Chicago Press, 2009
[26] Gilbert J., Gilbert L.: „*Linear Algebra and Matrix Theory*", Thomson Brooks, 2004
[27] Grimaldi R. P.: „*Discrete and Combinatorial Mathematics: An Applied Introduction*", Addison-Wesley, 2003
[28] Grover L.: „*A Fast Quantum-Mechanical Algorithm for Database Search*", ACM Symposium on Theory of Computing, ACM, 1996
[29] Hallgren S.: „*Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem*", in Proceedings of the 34th Annual ACM Symposium on the Theory of Computation, New York, 2002
[30] Jozsa R.: „*Quantum Algorithms and the Fourier transform*", arXive quant-ph, 1997
[31] Kitaev A. Yu., Shen A. H., Vyalyi M. N.: „*Classical and Quantum Computation (Graduate Studies in Mathematics)*", American Mathematical Society, 2002
[32] Lee C. F., Johnson N. F.: „*Let the quantum games begin*", Physics World, 2002

[33] Magniez F., Santha M., Szegedy M.: „*Quantum algorithms for the triangle problem*", Proc. 16th Annual ACM SIAM Symposium on Discrete Algorithms, 2005
[34] Nielsen M. A., Chuang I. L.: „*Quantum Computation and Quantum Information*", Cambridge University Press, 2004
[35] Shannon C. E., Weaver W.: „*The Mathematical Theory of Communication*", University of Illinois Press, 1998
[36] Pittenger A. O.: „*An introduction to Quantum Computing Algorithms*", Progress in Computer Science and Applied Logic, Vol. 19, Birkhauser, Boston, 2001
[37] Ömer B.: „*A Procedural Formalism for Quantum Computing*", doctoral dissertation, Dept. Theoretical Physics, Technical Univ. of Vienna, 1998
[38] Raussendorf R., Briegel H. J.: „*A One Way Quantum Computer*", Ph. Rev., 86, 2001
[39] Rieffel E.: „*An Introduction to Quantum Computing for Non-Psysicists*", ACM Computing Surveys, Vol. 32., 2000
[40] Ruediger R.: „*Quantum Programming Languages: An Introductory Overview*", The Computer Journal, 50(2), 2007
[41] Rosen K. H.: „*Discrete Mathematics and Its Applications*", McGraw-Hill, 2003
[42] Selinger P.: „*Towards a Quantum Programming Language*", Mathematical Structures in Computer Science, 2004
[43] Shende V.V., Markov I.L., Bullock S.S.: „*Syntesis of Quantum Logic Circuits*", IEEE Trans. Computer-Aided Design of Integrated circuits, 2006
[44] Shende V.V., Markov I.L., Bullock S.S.: „*Finding Small Two-Qubit Circuits*", Proc. SPIE, vol. 5436, 2004
[45] Shor P. W.: „*Algorithms for Quantum Computation: Discrete Logarithms and Factoring*", IEEE Press, 1994
[46] Shor P. W.: „*Polynomial-time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*", SIAM, 26(5), 1997
[47] Shor P. W.: „*Introduction to Quantum Algorithms*", Proceedings of the Symposium in Applied Mathematics, 58, 2002
[48] Shor P. W.: „*Why Haven't More Quantum Algorithms Been Found?*", ACM 50, 2003
[49] Sipser M.: „*Introduction to the Theory of Computation*", Thomson Course Tech, 2005
[50] Svore K. M., Aho A. V., Cross A. W., Chuang I., Markov I. L.: „*A Layered Software Architecture for Quantum Computing Design Tools*", Computer, January 2006
[51] Svore K.M., Terhal B.M., DiVincenzo D.P.: „*Local Fault-Tolerant Quantum Computation*", Physical Rev. A, vol. 72, no. 5, 2005
[52] Unruh W. G.: „*Maintaining Coherence in Quantum Computers*", Ph. Rev. A 51, 2001
[53] Viamontes G.F., Markov I.L., Hayes J.P.: „*Graph-Based Simulation of Quantum Computation in the State-Vector and Density-Matrix Representation*", Quantum Information and Computation, vol. 5, no. 2, 2005
[54] Yanofsky N. S., Mannucci M. A.: „*Quantum Computing for Computer Scientists*", Cambridge University Press, 2008
[55] Zalka C.: „*Grover's Quantum Searching Algorithm Is Optimal*", Physical Review Letters A 60(4), 1999
[56] Zhou X., Leung D. W., Chuang I. L.: „*Quantum Logic Gate Constructions with one-bit Teleportation*", arXive e-print quant-ph/0002039, 2000