Roman GIELERAK, Marek SAWERWAIN, Przemysław RATAJCZAK
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT STEROWANIA I SYSTEMÓW INFORMATYCZNYCH

Nowe funkcjonalności Zielonogórskiego Symulatora Obliczeń Kwantowych

prof. dr hab. Roman GIELERAK

Wszystkie stopnie i tytuły naukowe uzyskał na Uniwersytecie Wrocławskim w okresie od 1976 do 1999(profesura). Brał udział w licznych projektach badawczych we współpracy z wieloma ośrodkami zagranicznymi (ETH Zurych; ZIBJ Dubna; MGU Moskwa; Bochum-Bonn-Bielefeld Uniwersytety; Instytut Galileusza, Paryż; Uniwersytet w Lisbonie). Autor licznych prac i opracowań monograficznych z podstaw teorii kwantowej i jej zastosowań.

e-mail: R.Gielerak@issi.uz.zgora.pl

mgr inż. Przemysław RATAJCZAK

Ukończył studia na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego. W roku 2006 obronił pracę magisterską pt. Protokoły kryptografii kwantowej i ich symulacje komputerowe. Aktualnie pracuje jako asystent na Wydziale Elektrotechniki, Informatyki i Telekomunikacji UZ.

e-mail: P.Ratajczak@issi.uz.zgora.pl

Streszczenie

Przedstawione zostaną nowo skonstruowane biblioteki Zielonogórskiego Symulatora Obliczeń Kwantowych (ZSOK) znajdujące zastosowanie do przeprowadzania symulacji obliczeń kwantowych zaprojektowanych w oparciu o kwantowe unitarne obwody quditowe. Wprowadzone nowe rozwiązania pozwalają na przeprowadzenie symulacji obliczeń nieunitarnych w oparciu o systemy zbudowane z quditów. Możliwym się stało także symulowanie obliczeń opartych o pojęcie uogólnionego pomiaru oraz teleportację. W szczególności przedstawione zostaną konstrukcje uniwersalnych bibliotek dla unitarnych obliczeń oraz implementację procesu pomiaru dowolnego podrejestru rejestru quditowego w dowolnej zadeklarowanej bazie ortonormalnej.

Abstract

In this article certain features of Quantum Computing Simulator (ZSOK) will be presented. In particular, new library that provides simulation of qudit based unitary circuits is shown. Added solutions expand simulator's abilities not only by quantum gates acting on qudits, but also by generalized measurement and teleportation based circuits. We focus in detail on the one-way quantum computations and measurement of any quantum sub-register in any orthonormal basis.

Słowa kluczowe: obliczenia kwantowe, obwody kwantowej, obliczenia jednokierunkowe, symulacja kwantowego modelu obliczeniowego.

Keywords: quantum computation, quantum circuit, one-way quantum computation, simulation of quantum computation model

1. Wstęp -- problem symulowalności układów kwantowych

W ogólnym przypadku złożoność obliczeniowa symulacji obliczeń kwantowych jest określona (z pominięciem nieistotnych w tym momencie stałych) przez następującą zależność wykładniczą:

$$T(n) = d^n,$$

gdzie n oznacza ilość quditów, a d to wymiar kwantowej jednostki logicznej. Zakładając, że chcemy symulować zachowanie układu złożonego ze stu qubitów, potrzebujemy bardzo wielkich macierzy np.: dla opisania zmiany stanu kwantowego takiego układu opisywanego wektorem z przestrzeni $C^{2^{100}}$ potrzebujemy macierzy U o wymiarach $2^{100} \times 2^{100}$! A po uwzględnieniu, że elementy macierzy U , to liczby zespolone, a więc dwie rzeczywiste liczby 8 bajtowe oznacza, że potrzebujemy „tylko":

mgr inż. Marek SAWERWAIN

Ukończył studia na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego. Aktualnie pracuje jako asystent w Instytucie Sterowania I Systemów Informatycznych. Zajmuje się kwantowymi językami programowania oraz pracą nad symulatorem kwantowego modelu obliczeniowego.

e-mail: M.Sawerwain@issi.uz.zgora.pl

*2^100 * 2^100 * 2 * 8 = 514220174162876888173427869549 172032807104958010493707296440 32 bajtów pamięci.*
Warto zatem postawić pytanie, czy kiedykolwiek uda się zbudować pełny system symulacji obliczeń kwantowych na maszynie klasycznej. W tym miejscu stosowne, wydaje się przytoczenie tezy Churcha-Turinga z lat sześćdziesiątych:

Any model of computation can be simulated on a probabilistic Turing machine with at most a polynomial increase (i.e. efficiently) in the number of elementary operations required.

Wniosek z tezy jest następujący: jeśli dany problem jest trudny do rozwiązania, to takim pozostanie dla każdego klasycznego komputera, którego model obliczeniowy można przedstawić za pomocą maszyny Turinga i wielomianowej ilości operacji elementarnych. Jednak komputer kwantowy potrafi realizować trudne problemy, przykładem jest algorytm Shora w czasie krótszym, nieosiągalnym dla komputerów klasycznych. Co więcej, problem Deutscha-Jozsy posiada rozwiązane w czasie wielomianowym na maszynie kwantowej. Ważnym aspektem jest fakt, iż wynik jest zawsze ze stuprocentową pewnością poprawny, a więc jest to rozwiązanie deterministyczne. Te uwagi pozwalają nam, na ostrożne sformułowanie hipotezy, iż teza Churcha-Turinga nie jest prawdziwa.
W pracy [6] R.P. Feynman argumentuje, iż komputer klasyczny nigdy nie pozwoli na efektywną i pełną symulację zachowania się systemu kwantowego w czasie wielomianowym. Uwaga Feynmana może zostać przekształcona do następującego stwierdzenia:

Zakładamy, że istnieje algebra, która pozwala na efektywną reprezentacją stanu rejestru kwantowego wykazującego splątanie dla n qubitów. Symbolem $\hat{\otimes}$ *oznaczamy specjalny operator, który teoretycznie umożliwia złożenie poszczególnych stanów jednqubitowych z zachowaniem splątania pomiędzy qubitami:*

$$|\psi\rangle = |\psi_1\rangle \hat{\otimes} |\psi_2\rangle \hat{\otimes} \ldots \hat{\otimes} |\psi_n\rangle .$$

Pomimo tej efektywnej algebry, wektor nadal zwiera wykładniczą ilość informacji o stanie. Proces pomiaru w najgorszym przypadku potrzebuje $O(2^n)$ *operacji elementarnych.*

Szkic dowodu jest następujący: niech T reprezentuje klasyczną maszynę Turinga. Na taśmie maszyny został zapisany wektor dla n qubitów. W przypadku zapisu bez kompresji należy wykorzystać 2^n wartości zespolonych. Podczas procesu pomiaru całości rejestru, który może być bądź nie musi być w stanie splątanym, rejestr może przejść do jednego z 2^n stanów bazowych. Ponieważ, nie wiadomo, gdzie znajduje się potencjalnie największa amplituda prawdopodobieństwa, w najgorszym przypadku należy wykonać 2^n operacji porównywania. Dlatego, wiele innych podejść do symulacji algorytmów kwantowych zarówno teoretycznych jak i praktycznych poprawia wydajność symulacji, jednak naturalnie nie przełamuje złożoności symulacji obliczeń kwantowych. Warto w tym miejscu dodatkowo podkreślić, iż układy, które nie generują splątania są w pełni symulowane na klasycznych maszynach.

2. Uniwersalne i aproksymatywnie uniwersalne biblioteki unitarne

W przypadku obwodów klasycznych, wiadomo że zbiór bramek zawierający bramki OR, AND oraz NOT stanowi tzw. zbiór uniwersalny. Za pomocą wymienionych bramek można zbudować obwód implementujący dowolną funkcję logiczną. Klasyczna logika umożliwia zastosowanie tylko jednej bramki NAND -- która sama w sobie stanowi zbiór uniwersalny.
Obliczenia kwantowe realizowane np. w postaci obwodów kwantowych nie mają skończonych zbiorów bramek, tzn. aby realizować dowolny typ obliczeń należałoby posiadać nieskończony zbiór bramek. Oznacza to, iż istniejące bramki kwantowe są parametryzowane.
Jak dotąd, wprowadzono kilka typów zbiorów uniwersalnych. Jednym z przykładów dla qubitów jest zbiór złożony z bramki obrotu qubitu R oraz bramki CNOT.

$$R(\Theta)=\begin{bmatrix}1 & 0\\ 0 & e^{2\pi i\Theta}\end{bmatrix},\ CNOT=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix} \quad (1)$$

Analogicznie do bramki NAND, również w obliczeniach kwantowych można wprowadzić jeden typ bramki, jednakże będzie to bramka działająca na trzech qubitach, i będzie to tzw. bramka Deutscha [1]:

$$D(\Theta):|i,j,k\rangle=\begin{cases}i\cos(\Theta)|i,j,k\rangle+\sin(\Theta)|i,j,1-k\rangle & i=j=1\\ |i,j,k\rangle & \begin{matrix}przeciwne\\ przypadki\end{matrix}\end{cases} \quad (2)$$

Bramka ta staje się bramką Toffoliego, gdy $D(\Theta=\pi/2)$, co pociąga za sobą obserwację, iż na komputerze kwantowym można realizować wszystkie boolowskie funkcje logiczne. Możliwe jest także stosowanie bramek dwuqubitowych [2] oraz [3]. Stąd ważnym pojęciem są zbiory bramek aproksymatywnych do zbioru bramek uniwersalnych. Tego typu zbiór pozwala na realizację obwodów z pewnym przybliżeniem. Mówi o tym także twierdzenie Solovaya i Kitaeva [4]. Obwód o n bramkach unitarnych można przybliżyć za pomocą następującej liczby bramek ze zbioru uniwersalnego

$$O(n\log^c(\frac{n}{\varepsilon})), \quad (3)$$

gdzie c jest pewną stałą w przybliżeniu równą cztery, natomiast ε określa dokładność, z jaką chcemy przybliżać działanie wybranej bramki kwantowej.
Uniwersalny zbiory bramek istnieją także dla quditów [5], omawiany symulator obliczeń kwantowych oferuje dostęp do odpowiedników bramek grupy Pauliego dla wyższych jednostek kwantowych.

3. Symulator obliczeń kwantowych

Podstawową zaletą opisywanego systemu (prace na pakietem rozpoczęły się w latach 2004 oraz 2005 [10]), który został oznaczony skrótem QCS (ang. *Quantum Computing Simulator*) jest niespotykana w innych tego typu programach [11, 12, 13] dostępność kilku podstawowych typów obwodów kwantowych. W pakiecie QCS możliwa jest symulacja obwodów typu PQC, CHP jak również symulacja pełnego modelu obwodowego dla qubitów oraz dla quditów.
Wewnętrzna architektura QCS pozwala, nie tylko na przeprowadzanie symulacji, lecz może się stać podstawą języka programowania kwantowego lub innego dowolnego systemu przystosowanego do szczególnych potrzeb użytkownika.
Jest to szczególnie ważny aspekt, ponieważ inne tego typu systemy zazwyczaj są zamkniętymi środowiskami, które trudno rozszerzać i modyfikować.
Symulator obliczeń kwantowych QCS jest projektem ogólnego przeznaczenia. Został opracowany jako biblioteka funkcji zaimplementowana w języku ANSI C, przy wykorzystaniu bibliotek funkcji numerycznych BLAS i LAPACK. Ponieważ dostępna jest wersja tych bibliotek dla systemów równoległych, to istnieje możliwość przeniesienia pakietu QCS na maszyny równoległe.
Symulator pozwala na tworzenia skryptów w kilkunastu różnych językach programowania jak Pyton czy Java.
Z powodu wymienionych uwag oraz mając na uwadze dostępność pakietu i jego rolę edukacyjną, pakiet QCS powstał na bazie darmowego oprogramowania. Zostały zastosowane min. następujące pakiety:

- kompilator GCC v4.x
- system skryptowy SWIG v1.3.xx
- język Python v2.4
- pakiety do algebry liniowej LAPACK, BLAS, ATLAS
- pakiet do obliczeń równoległych MPICH-G2 (Win32, Linux, Clusterix)

Wszystkie wymienione narzędzia są oprogramowaniem darmowym opartym o licencję typu GNU GPL bądź podobną. Umożliwia to obniżenie kosztów tworzenia oprogramowania, jednakże nie ogranicza w żadnym razie funkcjonalności oprogramowania. Zastosowanie przede wszystkim kompilatora GCC, pozwala na uzyskanie bardzo istotnej własności, a mianowicie przenośności oprogramowania pomiędzy różnymi architekturami sprzętowo-programowymi.
Oprogramowanie na obecną chwilę jest dostępne dla najpopularniejszych systemów jak środowisko Windows w wersjach 32 i 64 bitowych oraz analogicznie system Linux. Oprogramowanie było także testowane na platformie IA-64 Itanium II.
Opisany system symulacji kwantowego modelu obliczeń był już stosowany podczas zajęć w ramach przedmiotu „Wstęp do informatyki kwantowej" na czwartym roku studiów magisterskich na kierunku informatyka na Uniwersytecie Zielonogórskim.
System był wykorzystywany przez studentów do realizacji różnych zadań. Zadania te dotyczyły min.: badania algorytmu Grovera, badania jakości kopiowania stanów kwantowych oraz opracowywania obwodów kwantowych realizujących algorytm Shora. Należy podkreślić interesujący aspekt opisywanego pakietu, a mianowicie zastosowania tego systemu do badania wydajności systemu komputerowego. Zarówno pod względem wydajności obliczeniowej procesora oraz wydajności operacji na pamięci.
System QCS podlega także ciągłemu rozwojowi, aktualnie ulepszeniu podlega moduł do równoległych symulacji obliczeń kwantowych. Nowym elementem jest możliwość przeprowadzania symulacji w ramach modelu jednokierunkowego. Jak dotąd, nie istniało oprogramowanie wspomagające symulacje tego rodzaju.

4. Jednokierunkowy schemat obliczeń nieunitarnych

Przykładem nieunitarnego modelu obliczeniowego jest jednokierunkowy kwantowy model obliczeniowy (ang. one-way quantum computation – 1WQC). Istnieje bardzo bogata literatura na ten temat, prace [7, 8] wprowadzają w tą tematykę. W modelu tym, w odróżnieniu od modelu unitarnego, operacja pomiaru jest podstawową operacją obliczeniową. Zastępuje ona bramki unitarne, choć trzeba nadmienić, że 1WQC wykorzystuje bramki Pauliego (oznaczane przez X,Y,Z) do korekcji otrzymanych stanów. Istotna jest także bramka Hadamarda (H):

$$X=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix},\ Y=\begin{bmatrix}0 & -i\\ i & 0\end{bmatrix},\ Z=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix},\ H=\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \quad (4)$$

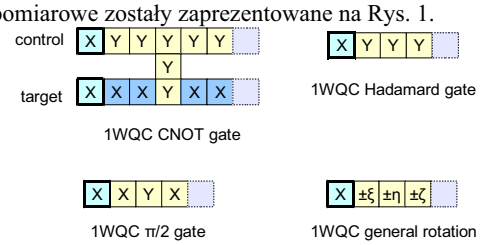
Bramka Hadamarda znajduje zastosowanie na samym początku realizacji obliczeń. W modelu 1WQC istotną rolę stanowi także stan początkowy całego układu nazywanego klastrem. Przed rozpoczęciem obliczeń klaster jest tworzony za pomocą bramki Hadamarda oraz bramki kontrolowanej zmiany fazy.
Podstawowe stany występujące w kontekście 1WQC, to stan zero oraz stan jeden oraz dwa stany stanowiące bazę Hadamarda:

$$P_0=\begin{bmatrix}1\\ 0\end{bmatrix},P_1=\begin{bmatrix}0\\ 1\end{bmatrix}\ oraz\ P_0=\begin{bmatrix}1/\sqrt{2}\\ 1/\sqrt{2}\end{bmatrix},P_1=\begin{bmatrix}1/\sqrt{2}\\ -1/\sqrt{2}\end{bmatrix} \quad (5)$$

Obliczenia 1WQC są wykonywane na prostokątnej siatce qubitów. Elementy tej siatki (albo gridu), czyli poszczególne qubity znajdują się w superpozycji uzyskanej przy pomocy bramki Hadamarda. Poszczególne qubity zostały także splątane za pomocą bramki zmiany fazy (ang. CPhase) o następującej postaci macierzowej

$$CPhase = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \quad (6)$$

Proces obliczania w takim układzie polega na odpowiednim przeprowadzaniu pomiarów poszczególnych qubitów. Ważna jest kolejność przeprowadzanych pomiarów, choć pewne pomiary mogą być wykonywane równolegle. Kluczowym elementem w pomiarze jest naturalnie baza w jakiej jest on dokonywany. Przykładowe wzorce pomiarowe zostały zaprezentowane na Rys. 1.



Rys. 1. Wzorce pomiarowe realizujące bramkę CNOT oraz bramkę Hadamarda, bramkę Pi/2 oraz ogólny wzorzec symulujący dowolną bramkę unitarną
Fig. 1. Example of measurement pattern for a basic set of gates: CNot, Hadamard and Pi/2 phase gate and the pattern for general unitary rotation gate

Można zaproponować różne definicje bazy do wykonywania pomiarów. W literaturze przyjęło się stosować dwa podstawowe typy bazy. Pierwsza jest określona jako

$$B(\varphi) = \left\{ \frac{|0\rangle + e^{i\varphi}|1\rangle}{\sqrt{2}}, \frac{|0\rangle - e^{i\varphi}|1\rangle}{\sqrt{2}} \right\}. \quad (7)$$

Ponieważ do poprawnej realizacji 1WQC wystarczy wykonywać pomiary leżące na jednym z równików sfery Blocha, a dokładnej na przecięciu płaszczyzn X—Y, to można uprościć postać bazy pomiarowej do następującej postaci [8]:

$$M(\varphi) = \left\{ |0\rangle \pm e^{i\varphi}|1\rangle \right\}. \quad (8)$$

Podsumowując, obliczenia w ramach 1WQC składają się z następujących etapów: etap pierwszy to przygotowanie siatki obliczeniowej poprzez wprowadzenie qubitów w superpozycję oraz ich splątanie, drugi etap to wykonanie pomiarów. Wynik każdego pomiaru, bez względu na konkretną postać bazy, zawsze ma przydzielone wartości zero oraz jeden. Po wykonaniu wszystkich pomiarów, w zależności od otrzymanych wartości, stosowane są bramki Pauliego do korekcji błędów, powstałych podczas wykonywania pomiarów. Ostatnia uwaga sugeruje także, iż pomimo stosowania pomiarów, które mają charakter probabilistyczny, mogą istnieć algorytmy realizowane przez 1WQC o charakterze deterministycznym. Tak jest w istocie, i przykładem może być omawiany w następnym punkcie algorytm symulacji bramki unitarnej.

4.1 Algorytm pomiaru w dowolnej bazie ortonormalnej

Zasadniczym problemem w symulacji modelu 1WQC jest realizacja pomiaru w dowolnej bazie. Aktualnie, symulator realizuje pomiar von Neumanna oparty o projektory. Jeśli przyjmiemy, że $\{|\psi_m\rangle\}$ jest zbiorem wektorów bazowych, to odpowiadające mu projektory są określone w następujący sposób:

$$P_m = |\psi_m\rangle\langle\psi_m| \; oraz \; P_{m_i} P_{m_j} = \delta_{i,j} P_m \; oraz \sum_m P_m = I \quad (9)$$

Oznaczając przez p_m prawdopodobieństwo, iż po pomiarze układ będącym w stanie początkowym $|\psi\rangle$ znajdzie się w jednym z możliwych stanów $|\psi_m\rangle$ jest określony przez

$$p_m = \langle\psi|P_m|\psi\rangle.$$

Wtedy,

$$|\psi'\rangle = \frac{P_m|\psi\rangle}{\|P_m|\psi\rangle\|}. \quad (10)$$

Łatwo podać przykłady projektorów dla bazy standardowej oraz dla bazy Hadamarda, które są określone przez następujące relacje:

$$P_0 = |0\rangle\langle 0|, P_1 = |1\rangle\langle 1| \; oraz \; P_0 = |+\rangle\langle +|, P_1 = |-\rangle\langle -| \quad (11)$$

Wartość prawdopodobieństwa w obydwu przypadkach otrzymujemy stosując wyrażenie

$$p_0 = \langle\psi|P_0|\psi\rangle \text{ albo } p_1 = \langle\psi|P_1|\psi\rangle. \quad (12)$$

Pomimo iż stosowane są różne bazy, to wyniki w przypadku 1WQC są zawsze oznaczane przez zero i jeden. Pięć podstawowych kroków składa się na podstawową wersję algorytmu do wykonywania pomiaru jednego qubitu:

(1) wyznaczenie prawdopodobieństw p_0 oraz p_1 przy zastosowania projektorów P_0 i P_1,

(2) jeśli $p_0 > p_1$ to zastosowany zostanie projektor P_0,

(3) jeśli $p_1 > p_0$ to zastosowany zostanie projektor P_1,

(4) jeśli $p_0 = p_1$ to należy w sposób losowy wybrać jeden z projektorów,

(5) normalizacja otrzymanego stanu.

Warto podkreślić możliwość zrównoleglenia operacji wyznaczenia prawdopodobieństwa, zastosowania projektora oraz normalizacji.

5. Przykłady symulacji

Przykład jaki zostanie zaprezentowany, to symulacja zastosowania bramki. Podczas symulacji stosowane będą przede wszystkim pomiary za pomocą bazy określonej wzorem (7). Ponieważ symulowane będzie działanie bramki unitarnej typu Hadamard, to zgodnie ze wzorcem pomiarowym z Rys. 1 zostanie użyta baza oznaczona jako X oraz baza Y. Przy czym, występuje tylko jeden pomiar w bazie X oraz trzy pomiary w bazie Y.

Symulator obliczeń kwantowych pozwala na zapisanie algorytmu symulacji przy zastosowaniu różnych języków programowania. Podstawowym sposobem jest bezpośrednie użycie API w języku C, w którym został opracowany symulator. Możliwe jest korzystanie z dodatkowych portów do innych języków programowania jak Java oraz Python. Nowym elementem wprowadzanym do symulatora jest assembler, który oprócz podobieństwa do istniejących klasycznych assemblerów zawiera także instrukcje obsługujące dane kwantowe. Rys. 2 przedstawia fragment kodu programu realizującego symulację działania bramki unitarnej. Tak znaczące uproszczenie podstawowego zestawu funkcji ułatwia tworzenie symulacji oraz nie nastręcza trudności w używaniu pakietu, podczas przeprowadzania symulacji algorytmów oraz obwodów kwantowych.

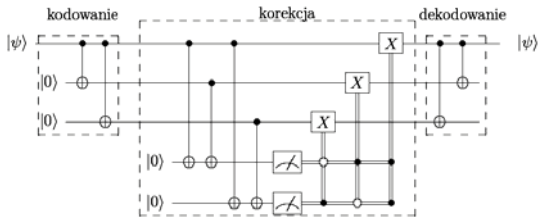
```
float a=QCS_PI / 2, b=QCS_PI / 2, c=QCS_PI / 2;
r = qcs_new_quantum_reg( 5 );
qcs_quantum_reg_reset(r);
qcs_quantum_reg_had_n(r, 1);
qcs_quantum_reg_had_n(r, 2);
qcs_quantum_reg_had_n(r, 3);
qcs_quantum_reg_had_n(r, 4);
qcs_quantum_reg_cphase_f_n(r, 0, 1);
qcs_quantum_reg_cphase_f_n(r, 1, 2);
qcs_quantum_reg_cphase_f_n(r, 2, 3);
qcs_quantum_reg_cphase_f_n(r, 3, 4);
qcs_make_b_base(&b1, 0);
s1=qcs_quantum_reg_measure_one_qubit_in_base(r, 0, &b1);
a=-a * powf(-1, s1); qcs_make_b_base(&b2, a);
s2=qcs_quantum_reg_measure_one_qubit_in_base(r, 1, &b2);
b=-b * powf(-1, s2); qcs_make_b_base(&b3, b);
s3=qcs_quantum_reg_measure_one_qubit_in_base(r, 2, &b3);
c=-c * powf(-1, s1+s3); qcs_make_b_base(&b4, c);
s4=qcs_quantum_reg_measure_one_qubit_in_base(r, 3, &b4);
```

Rys. 2. Fragment funkcji przeprowadzającej symulacje działania bramki unitarnej w modelu 1WQC
Fig. 2. Part of function which simulates the applying of unitary gate with 1WQC

Należy nadmienić, iż przekształcony stan qubitu wejściowego, który znajduje się w ostatnim piątym qubicie, wymaga jeszcze dodatkowych korekcji za pomocą bramek Pauliego Z oraz X. Jednakże, ten fragment programu został pominięty.
Innym przykładem jest symulacja działania kwantowych kodów korekcji (QECC). Rys. 3 prezentuje skrypt w języku Python implementujący kod korekcji zmiany bitu. W tym przypadku chroniony jest pojedynczy stan pierwszego qubitu $|\varphi\rangle$ poprzez zastosowanie kodu powtórzeniowego. Na Rys. 4 zaprezentowano obwód implementujący ten kod korekcji.

```
#! /usr/bin/python
import qcs
import random
random.seed()

q0 = qcs.Qubit()
q0.SetThetaPsi(random.randint(0,180), random.randint(0,180))
q1 = qcs.Qubit(); q1.Set0(); q2 = qcs.Qubit(); q2.Set0()
q_ar = qcs.QubitArray(3)
q_ar.SetQubitN(0, q0); q_ar.SetQubitN(1, q1)
q_ar.SetQubitN(2, q2)
q_reg = qcs.QubitReg(3)
q_reg.SetFromQubitArray(q_ar)

#kodowanie
q_reg.CNot(0,1); q_reg.CNot(0,2)

#wprowadzanie zakłóceń
q_reg.PauliX(random.randint(0,2))

#dekodowanie i korekcja
q_reg.CNot(0,1); q_reg.CNot(0,2)
q_reg.Toffoli(2,1,0)
s1 = q_reg.MeasureOneQubit(1);
s2 = q_reg.MeasureOneQubit(2)
if s1 == s2 == 0: print "Nie wystąpil błąd zmiany bitu"
if s1 == s2 == 1: print "Błąd zmiany bitu na I qubicie"
if s1 == 1 and s2 == 0:print "Błąd zm. bitu na II qubicie"
if s1 == 0 and s2 == 1:print "Błąd zm. bitu na III qubicie"
```

Rys. 3. Skrypt w języku Python dla kodu korekcji zmiany bitu.
Fig. 3. Python script for simulation of bit-flip quantum error correction code.



Rys. 4. Obwód kwantowy dla kodu korekcji zmiany bitu.
Fig. 4. Circuit for simulation of bit-flip quantum error correction code.

6. Podsumowanie

Pakiet QCS na obecną chwilę jest jedynym pakietem, który oferuje z poziomu pojedynczej platformy dostępność do kilku różnych technik symulacji obliczeń kwantowych oraz daje możliwość stosowania różnych modeli kwantowych obliczeń. Walorem pakietu jest także możliwość interaktywnego korzystania z dostępnych funkcji poprzez port do języka Pyton. Ułatwia to tworzenie skryptów, co pozwala na łatwe wykorzystanie QCS do zastosowań edukacyjnych oraz naukowych. Jednym z dalszych celów jest implementacja symulacji innych modeli kwantowych jak np.: model adiabatyczny, czy geometryczny. Również, gotowe już wsparcie dla wyższych jednostek kwantowych, czyli dla quditów, będzie podlegać dalszemu rozwojowi. Obecnie można przeprowadzać symulacje dla obwodów kwantowych zbudowanych z quditów. W najbliższym okresie wprowadzony zostanie odpowiednik obliczeń jednokierunkowych dla quditów.
Interesującym elementem, który również ulegnie dalszej modyfikacji jest możliwość tworzenia programów w języku asemblera. Nowo wprowadzony tryb dla quditów wymaga rozszerzenia tego trybu pracy. Wiąże się to także z możliwością pracy w środowisku równoległym za pomocą pakietu MPI. Wiele przykładowych symulacji przeprowadzonych przy wykorzystaniu QCS, zarówno na pojedynczych maszynach oraz na klastrze zbudowanym na kilku stacjach roboczych PC potwierdzają jego elastyczność. Konieczne trzeba podkreślić, iż system QCS można również wykorzystać w roli testera wydajności zarówno procesora jak również wydajności podsystemu pamięci operacyjnej.

7. Literatura

[1] Deutsch D, Quantum computational networks, Proc. Roy. Soc. London A 425 pp. 73-90, 1989.
[2] DiVincenzo D P, Two-bit gates are universal for quantum computation, *Phys. Rev. A* 51 pp. 1015-1022, 1995.
[3] A. Barenco, A universal two bit gate for quantum computation, Proc. Royal Soc. London A, 449 A995), 679-693.
[4] A.Y.Kitaev, Quantum computation: algorithms and error correction, Russian Mathematical surveys 52:1991 (1997).
[5] J.L. Brylinski and R. Brylinski, Universal quantum gates, In: Mathematics of Quantum Computation, Ranee K. Brylinski and Goong Chen (Eds), Chapman and Hall/CRC, 2002.
[6] R.P.Feynman, Simulating physics with computers, International Journal of Theoretical Physics 21:6/7. 467-488 (1982).
[7] Raussendorf R., Briegel H.J.: *A one-way quantum computer*, Phys. Rev. Lett., 86 (2001), pp. 5188--5191, see arXiv:quant-ph/0010033.
[8] Raussendorf, R., Browne, D.E., Briegel, H.J.: *Measurement-based quantum computation with cluster states*, Phys. Rev. A, 68 (2003), 022312, see arXiv:quant-ph/0301052.
[9] Jozsa R.: *An introduction to measurement based quantum computation*., see arXiv:quant-ph/0508124.
[10] M.Sawerwain, *Symulator obliczeń kwantowych*, Computer Methods and Systems - CMS '05 : V konferencja. Kraków, Polska, 2005 .- Kraków: Oprogramowanie Naukowo-Techniczne, 2005 .- Vol. 2: Regular sessions, s. 185--190 .- ISBN: 83-916420-3-8.
[11] H.Touchette, P.Dumai, *QuCalc - The quantum computation package for Mathematica*, http://crypto.cs.mcgill.ca/QuCalc/, 2000.
[12] P.Gawron, J.A.Miszczak, *Numerical simulations of mixed states quantum computation*, http://www.arxiv.org/quant-ph/0406211
[13] Ogólno dostępny symulator obliczeń kwantowych FQC, dostępny pod adresem WWW: http://www.qc.fraunhofer.de

Title: New features of Quantum Computing Simulator developed at University of Zielona Góra

---

Artykuł recenzowany