

# Inicjalizacja rejestrów w kwantowym języku programowania

Jarosław Miszczak

Instytut Informatyki Teoretycznej i Stosowanej PAN

16 Luty 2007

# Plan wystąpienia

Motywacja

Algorytmy kwantowe

Typy kwantowe

Obecny stan wiedzy

Cel pracy

Przygotowanie superpozycji

Przypadek 1-qubitowy

Rejestry wieloqubitowe

Przykład

Podsumowanie

Dalsze prace

# Motywacja

- ▶ Do tworzenia nowych algorytmów potrzebne jest abstrakcyjne postrzeganie obiektów którymi się posługujemy

# Motywacja

- ▶ Do tworzenia nowych algorytmów potrzebne jest abstrakcyjne postrzeganie obiektów którymi się posługujemy
- ▶ Struktury danych muszą być rozpatrywane w połączeniu z operacjami które są na nich wykonywane

# Motywacja

- ▶ Do tworzenia nowych algorytmów potrzebne jest abstrakcyjne postrzeganie obiektów którymi się posługujemy
- ▶ Struktury danych muszą być rozpatrywane w połączeniu z operacjami które są na nich wykonywane
- ▶ Algorytm kwantowy to sposób uzyskania pożądanego rozkładu prawdopodobieństwa z wykorzystaniem reguł mechaniki kwantowej

## Działanie algorytmów kwantowych

Algorytm (protokół) kwantowy jest opisany operacją unitarną  $U$  działająca na przestrzeni  $\mathbb{C}^{2^n}$  zgodnie z regułami

# Działanie algorytmów kwantowych

Algorytm (protokół) kwantowy jest opisany operacją unitarną  $U$  działająca na przestrzeni  $\mathbb{C}^{2^n}$  zgodnie z regułami

- ▶ stanem początkowym jest stan podstawowy  $|0 \dots 0\rangle \in \mathbb{C}^{2^n}$ ,
- ▶ operacja  $U$  przeprowadza stan początkowy w stan  $|\psi\rangle$ ,
- ▶ wynikiem jest rozkład prawdopodobieństwa

$$P(|i\rangle) = |\langle i|\psi\rangle|^2, \quad i = 0, \dots, 2^n - 1. \quad (1)$$

## Obecny stan wiedzy

- ▶ Większość języków kwantowych (QCL, Q) operuje jedynie na tablicach qubitów

## Obecny stan wiedzy

- ▶ Większość języków kwantowych (QCL, Q) operuje jedynie na tablicach qubitów
- ▶ Język cQPL pozwala zadeklarować rejestr kwantowy jako `qshort` lub `qint`, ale oznacza to jedynie że reprezentuje on 8 lub 16 qubitów

## Obecny stan wiedzy

- ▶ Większość języków kwantowych (QCL, Q) operuje jedynie na tablicach qubitów
- ▶ Język cQPL pozwala zadeklarować rejestr kwantowy jako `qshort` lub `qint`, ale oznacza to jedynie że reprezentuje on 8 lub 16 qubitów
- ▶ Perl6 wprowadza tzw. połączenia (ang. *junction*) wzorowane na superpozycji stanów

```
pugs> my $x = 0|1|2|3;  
(0 | 1 | 2 | 3)  
pugs> if $x == 0|2 { say "Wow!"; }  
Wow!
```

# Poszerzenie możliwości języków kwantowych

1. Dodanie wbudowanego typu danych dla rejestrów kwantowych reprezentujących liczby całkowite i ich superpozycje (podobnie jak w Perl6)

# Poszerzenie możliwości języków kwantowych

1. Dodanie wbudowanego typu danych dla rejestrów kwantowych reprezentujących liczby całkowite i ich superpozycje (podobnie jak w Perl6)
2. Określenie operacji dla tego typu danych

## Poszerzenie możliwości języków kwantowych

1. Dodanie wbudowanego typu danych dla rejestrów kwantowych reprezentujących liczby całkowite i ich superpozycje (podobnie jak w Perl6)
2. Określenie operacji dla tego typu danych
3. Zdefiniowane reguł rzutowanie do typów klasycznych

# Przypadek 1-qubitowy

## Zadanie

Wychodząc ze stanu  $|0\rangle$  chcemy otrzymać stan  $\sqrt{1-p}|0\rangle + \sqrt{p}|1\rangle$ .  
Parametr  $p$  to prawdopodobieństwo otrzymania stanu  $|1\rangle$ .

## Przypadek 1-qubitowy

### Zadanie

Wychodząc ze stanu  $|0\rangle$  chcemy otrzymać stan  $\sqrt{1-p}|0\rangle + \sqrt{p}|1\rangle$ .  
Parametr  $p$  to prawdopodobieństwo otrzymania stanu  $|1\rangle$ .

Odpowiednia operacja to

$$G(p) = \begin{cases} R_y \left( 2 \arctan \sqrt{\frac{p}{1-p}} \right), & 0 \leq p < 1 \\ \sigma_x, & p = 1 \end{cases}, \quad (2)$$

## Przypadek 1-qubitowy

### Zadanie

Wychodząc ze stanu  $|0\rangle$  chcemy otrzymać stan  $\sqrt{1-p}|0\rangle + \sqrt{p}|1\rangle$ .  
 Parametr  $p$  to prawdopodobieństwo otrzymania stanu  $|1\rangle$ .

Odpowiednia operacja to

$$G(p) = \begin{cases} R_y \left( 2 \arctan \sqrt{\frac{p}{1-p}} \right), & 0 \leq p < 1 \\ \sigma_x, & p = 1 \end{cases}, \quad (2)$$

co jest równoważne

$$G(p) = \begin{pmatrix} \sqrt{1-p} & \sqrt{p} \\ -\sqrt{p} & \sqrt{1-p} \end{pmatrix}. \quad (3)$$

## Wykorzystywane oznaczenia

- ▶ Przez kontrolę dla bitu  $k$  rozumiemy ciąg binarny  $\alpha$  złożony z bitów  $1, \dots, k - 1$ , który określa jednoznacznie podprzestrzeń rozpiętą przez wektor  $|\alpha\rangle$

## Wykorzystywane oznaczenia

- ▶ Przez kontrolę dla bitu  $k$  rozumiemy ciąg binarny  $\alpha$  złożony z bitów  $1, \dots, k - 1$ , który określa jednoznacznie podprzestrzeń rozpiętą przez wektor  $|\alpha\rangle$
- ▶  $count[i, \alpha_j]$  – tablica zawierająca liczbę wystąpień kontroli  $\alpha_j$  dla bitu  $i$

## Wykorzystywane oznaczenia

- ▶ Przez kontrolę dla bitu  $k$  rozumiemy ciąg binarny  $\alpha$  złożony z bitów  $1, \dots, k - 1$ , który określa jednoznacznie podprzestrzeń rozpiętą przez wektor  $|\alpha\rangle$
- ▶  $count[i, \alpha_j]$  – tablica zawierająca liczbę wystąpień kontroli  $\alpha_j$  dla bitu  $i$
- ▶  $count\_1[i, \alpha_j]$  – tablica zawierająca liczbę przypadków dla których bit numer  $i$  dla kontroli  $\alpha_j$  jest ustawiony na 1

## Generacja obwodu

**Wejście:** lista  $A = (a^1, \dots, a^K)$  liczb  $N$ -bitowych.

**Wyjście:** operacja unitarna  $R$  taka, że

$$R|0 \dots 0\rangle = \sum_{l=1}^K |a^l\rangle. \quad (4)$$

# Generacja obwodu – krok 1

Ustawienie pierwszego bitu zależy tylko od informacji o nim.

- ▶  $x$  = ilość wystąpień jedynki na pierwszym bicie
- ▶  $R_1 = G\left(\frac{x}{K}\right) \otimes \mathbb{I}^{N-1}$

## Generacja obwodu – kroki $2, \dots, N$

Ustawienie bitu  $n$  zależy od ustawień pozycji  $1, \dots, n - 1$ .

- ▶ dla  $n$  od 2 do  $N$  :

## Generacja obwodu – kroki $2, \dots, N$

Ustawienie bitu  $n$  zależy od ustawień pozycji  $1, \dots, n - 1$ .

- ▶ dla  $n$  od 2 do  $N$  :
  - ▶ oblicz  $count[n, \alpha_j]$  oraz  $count\_1[n, \alpha_j]$  dla wszystkich  $\alpha_j$ ,

## Generacja obwodu – kroki $2, \dots, N$

Ustawienie bitu  $n$  zależy od ustawień pozycji  $1, \dots, n-1$ .

▶ dla  $n$  od 2 do  $N$  :

▶ oblicz  $count[n, \alpha_j]$  oraz  $count\_1[n, \alpha_j]$  dla wszystkich  $\alpha_j$ ,

▶  $R_n = \sum_{\alpha_j} |\alpha_j\rangle\langle\alpha_j| \otimes G \left( \frac{count\_1[n, \alpha_j]}{count[n, \alpha_j]} \right) \otimes \mathbb{I}^{N-1-n}$

## Generacja obwodu – kroki $2, \dots, N$

Ustawienie bitu  $n$  zależy od ustawień pozycji  $1, \dots, n-1$ .

▶ dla  $n$  od 2 do  $N$  :

▶ oblicz  $count[n, \alpha_j]$  oraz  $count\_1[n, \alpha_j]$  dla wszystkich  $\alpha_j$ ,

▶ 
$$R_n = \sum_{\alpha_j} |\alpha_j\rangle\langle\alpha_j| \otimes G \left( \frac{count\_1[n, \alpha_j]}{count[n, \alpha_j]} \right) \otimes \mathbb{I}^{N-1-n} \\ + \overline{\sum_{\alpha_j} |\alpha_j\rangle\langle\alpha_j|} \otimes \mathbb{I}^{N-n}.$$

## Generacja obwodu – kroki $2, \dots, N$

Ustawienie bitu  $n$  zależy od ustawień pozycji  $1, \dots, n-1$ .

▶ dla  $n$  od 2 do  $N$  :

▶ oblicz  $count[n, \alpha_j]$  oraz  $count\_1[n, \alpha_j]$  dla wszystkich  $\alpha_j$ ,

▶  $R_n = \sum_{\alpha_j} |\alpha_j\rangle\langle\alpha_j| \otimes G\left(\frac{count\_1[n, \alpha_j]}{count[n, \alpha_j]}\right) \otimes \mathbb{I}^{N-1-n}$   
 $+ \overline{\sum_{\alpha_j} |\alpha_j\rangle\langle\alpha_j|} \otimes \mathbb{I}^{N-n}$ .

Wynikowy operator to

$$R = \prod_{k=1}^N R_k. \quad (5)$$

**Wejście:**  $A = (001, 101, 110)$

**Wejście:**  $A = (001, 101, 110)$

1.  $R_1 = G(2/3) \otimes \mathbb{I}^2$

**Wejście:**  $A = (001, 101, 110)$

1.  $R_1 = G(2/3) \otimes \mathbb{I}^2$

2.  $R_2 =$

$$(|1\rangle\langle 1| \otimes G(1/2) + |0\rangle\langle 0| \otimes G(0)) \otimes \mathbb{I} + (\mathbb{I} - (|0\rangle\langle 0| + |1\rangle\langle 1|)) \otimes \mathbb{I}^2$$

**Wejście:**  $A = (001, 101, 110)$

1.  $R_1 = G(2/3) \otimes \mathbb{I}^2$

2.  $R_2 =$

$$(|1\rangle\langle 1| \otimes G(1/2) + |0\rangle\langle 0| \otimes G(0)) \otimes \mathbb{I} + (\mathbb{I} - (|0\rangle\langle 0| + |1\rangle\langle 1|)) \otimes \mathbb{I}^2$$

3.  $R_3 =$

$$(|00\rangle\langle 00| \otimes G(1) + |11\rangle\langle 11| \otimes G(0) + |10\rangle\langle 10| \otimes G(1)) + (\mathbb{I}^2 - (|00\rangle\langle 00| + |11\rangle\langle 11| + |10\rangle\langle 10|)) \otimes \mathbb{I}$$

## Podsumowanie

- ▶ Przedstawiony algorytm pozwala na wygenerowanie obwodu kwantowego który może być wykorzystany jako wejście dla symulatora – układ kontrolujący maszynę kwantową wymaga bardziej elementarnych operacji

# Podsumowanie

- ▶ Przedstawiony algorytm pozwala na wygenerowanie obwodu kwantowego który może być wykorzystany jako wejście dla symulatora – układ kontrolujący maszynę kwantową wymaga bardziej elementarnych operacji
- ▶ Odpowiednia inicjalizacja stanu jest rozwinięciem idei algorytmu Grovera, gdzie mamy możliwość jedynie podbicia określonych prawdopodobieństw

## Dalsze prace

- ▶ Generacja optymalnego kodu – np. przez eliminowanie operacji na niezależnych bitach

## Dalsze prace

- ▶ Generacja optymalnego kodu – np. przez eliminowanie operacji na niezależnych bitach
- ▶ Tworzenie dowolnego rozkładu prawdopodobieństwa

## Dalsze prace

- ▶ Generacja optymalnego kodu – np. przez eliminowanie operacji na niezależnych bitach
- ▶ Tworzenie dowolnego rozkładu prawdopodobieństwa
- ▶ Zastosowanie operacji na superpozycjach do opisu użytecznych algorytmów

Proszę o pytania