PRACA MAGISTERSKA

Splątanie w algorytmach kwantowych ukrytej podgrupy

Autor: Jarosław A. Miszczak

Promotor: dr hab. Sławomir Bugajski

Zakład Fizyki Teoretycznej
Wydział Matematyki, Fizyki i Chemii

Uniwersytet Śląski w Katowicach

12 marca 2003 roku

Pracę tą poświęcam pamięci
dr. hab Sławomira Bugajskiego

Spis treści

Spis rysunków

Spis listingów

Wstęp

„Information is physical"
– Rolph Landauer (1927–1999)

Założenie, że maszyny liczące są opisywane prawami mechaniki klasycznej jest jednym z przemilczanych aksjomatów klasycznego modelu obliczeń. Kwantowa teoria informacji stanowi rozszerzenie klasycznej teorii informacji na układy, które zachowują się zgodnie z prawami mechaniki kwantowej. Potrzeba takiego uogólnienia jest nieunikniona w obliczu postępującej miniaturyzacji systemów informatycznych. Postęp prowadzi do wykorzystania w informatyce obiektów, przy opisie których konieczne będzie uwzględnienie efektów kwantowych.

Nie jest to jednak jedyna przyczyna zainteresowania zastosowaniem praw mechaniki kwantowej do przetwarzania informacji. Okazuje się, iż wykorzystanie tych praw może prowadzić do nowych, nieobecnych w klasycznej teorii informacji efektów. Efekty te dotyczą zarówno przetwarzania informacji, jak i jej przesyłania.

Niniejsze opracowanie poświęcone jest algorytmom kwantowym, czyli algorytmom, które wykorzystują nowe możliwości komputerów kwantowych. Szczególny nacisk położony jest na występowanie i opis stanów splątanych wykorzystywanych w informatyce kwantowej.

Najpopularniejszym i najprostszym modelem obliczeń wykorzystywanym w informatyce klasycznej jest maszyna Turinga. Wynika to z prostego faktu, iż jest ona w stanie symulować każdą inną maszynę. Zatem, pomimo skromnych zasobów, jakie dopuszcza się w tym modelu, pozwala on na opisanie każdego „rozsądnego" modelu obliczeń. Maszyna Turinga może wykonać algorytm zaprojektowany dla dowolnej innej maszyny, przyczym spowolnienie wynikające z symulacji wzrasta jedynie tak jak wielomian czasu potrzebnego na realizację algorytmu na maszynie, na którą był pisany. Okazuje się, że kluczowym założeniem tego stwierdzenia jest „rozsądność" modelu, czyli maszyny jaką chcemy symulować.

Można pokusić się o uogólnienie modelu maszyny Turinga poprzez wprowadzenie do niego niedeterminizmu i prawdopodobieństwa. Model kwantowy pozwala na opis maszyn klasycznych jako przypadków szczególnych.

Jednym z efektów występujących w algorytmach kwantowych jest superpozycja stanów. Ponieważ jedynym sposobem na symulowanie maszyny niedeterministycznej za pomocą maszyny deterministycznej jest sprawdzenie wszystkich dróg ewolucji, nasuwa się analogia do obliczeń równoległych. Jednak tutaj superpozycja jest czymś więcej niż równoległością. Wprowadzając stany maszyny kwantowej w superpozycję zyskujemy jednocześnie możliwość wprowadzenia wzajemnych oddziaływań pomiędzy „drogami ewolucji". Powstaje pytanie, czy można tak wykorzystać to oddziaływanie, aby wydajnie symulować maszyny niedeterministyczne, a co za tym – idzie rozwiązywać problemy z klasy **NP**. Jest to w zasadzie pytanie o podstawy działania algorytmów kwantowych i sposób ich tworzenia. Aby tworzyć nowe algorytmy kwantowe musimy wiedzieć, czym dysponujemy.

Teza Church–Turinga głosi iż:

Każda „funkcja, którą można w sposób naturalny uznać za obliczalną", może być obliczona przez uniwersalną maszynę Turinga.

Nie musi być tak, iż wszystkie fizycznie realizowalne modele obliczeń są równoważne, jeśli chodzi o moc obliczeniową, maszynie Turinga. Nie można – wykorzystując jedynie logikę – wykluczyć układów fizycznych obliczających funkcje nierekurencyjne.

Projektując algorytm na maszynę kwantową należy pamiętać, że przedmiotem manipulacji jest w tym wypadku rozkład prawdopodobieństwa na przestrzeni stanów maszyny. Jest to wpisane w teorię kwantową, którą posługujemy się do opisu takich maszyn. Dostępne nam są jedynie rozkłady prawdopodobieństwa, gdyż one są wynikami pomiarów, jakie przeprowadzamy.

Celem niniejszej pracy jest odpowiedź na pytanie: *„Jakie jest znaczenie splątania w obliczeniach kwantowych?"* Pierwszym problemem, który nasuwa się przy analizie tego zagadnienia, jest brak narzędzi pozwalających badać stopień splątania stanu komputera kwantowego podczas wykonywania algorytmów kwantowych. Splątanie jest jednym z nowych, nieklasycznych elementów wprowadzanych do procesu obróbki informacji przez informatykę kwantową. Zrozumienie i umiejętne wykorzystanie go daje szanse na osiągnięcie wyników nieosiągalnych na gruncie teorii klasycznej i jest kluczowe dla dalszego rozwoju informatyki kwantowej.

W Rozdziale 1 wprowadzony jest formalizm kryteriów splątania i miar splątania rozwinięty w celu jakościowej i ilościowej analizy stopnia splątania układów kwantowych. Omówione są miary splątania, ze szczególnym uwzględnieniem miar wykorzystanych w dalszych częściach pracy.
W Rozdziale 2 omówione jest jedno z najpotężniejszych narzędzi wykorzystywanych w algorytmach kwantowych – kwantowa transformata Fouriera. Okazuje się, że połączenie możliwości szybkiego wykonania dyskretnej transformaty

Fouriera z techniką znajdowania ukrytej podgrupy pozwala na wydajne rozwiązanie wielu problemów uznanych za trudne.
Rozdział 3 omawia klasę algorytmów kwantowych – w obrębie której zanjduje się algorytm szybkiej faktoryzacji liczb – wywodzących się ze sformułowania problemu ukrytej podgrupy.
Proste symulacje algorytmów kwantowych przedstawione w Rozdziale 4 są próbą opisu wpływu splątania na przebieg algorytmów. Rozdział ten ma za zadanie uchwycenie ilościowych aspektów teorii.
W Rozdziale 5 omówione jest znaczenie splątania w kontekście kwantowej teorii obliczeń i kwantowej teorii informacji. Najważniejsze zagadnienia jakie się tu pojawiają to zagadnienie złożoności obliczeń kwantowych i symulacji maszyn probabilistycznych oraz aspekty nielokalności związane z przesyłaniem informacji.
Dwa ostatnie rozdziały pracy zawierają część materiału matematycznego wykorzystanego w pracy – Rozdział 6 – oraz opis oprogramowania – Rozdział 7 – pomocnego w uzyskaniu wyników omawianych w Rozdziale 4.

Jarosław Adam Miszczak
9 listopad 2002

Rozdział 1

Splątanie

„(...)the best possible knowledge of the whole
does not include the best possible knowledge of its parts."
– Erwin Schrödinger (1887-1961)

1.1 Definicja splątania

Splątanie (niem. *Verschränkung*) pojawiło się w teorii kwantowej w roku 1935. Erwin Schrödinger określił w ten sposób pewien typ korelacji kwantowomechnicznych. Wiek XX przyniósł duży postęp wiedzy na temat praw mechaniki kwantowej. Teoria kwantów stała się narzędziem pracy fizyków niemal wszystkich specjalności. Jednakże ten rozwój nie przyniósł zadowalających odpowiedzi na pytanie dotyczące podstaw teorii kwantów: *Czym jest splątanie?*

Do dzisiaj nie podana została akceptowana przez wszystkich definicja splątania [8]. Sporo już wiemy o sposobach oznaczania (kryteria splątania) i mierzenia splątania (miary splątania), nie wiemy natomiast dokładnie co oznaczamy i mierzymy.

Nie znane jest też dokładnie znaczenie splątania. Nielokalność – uznawana czasem za pojęcie równoważne splątaniu – wydaje się być jednym z kluczowych aspektów mechaniki kwantowej odróżniających ją od mechaniki klasycznej. Nie wiadomo dokładnie na czym te różnice polegają i gdzie mogą znaleźć swoje zastosowanie.

Kolejnym krokiem w rozwoju teorii splątania jest opracowanie metod rozpoznawania i sposobów mierzenia splątania w układach wieloskładnikowych. Niestety wyniki w tym kierunku są jak dotychczas znikome.

Najprostszą odpowiedzą na pytanie *Czym jest splątanie?* może być: jest to „pewien rodzaj korelacji". Jednakże stany mieszane zawierają z definicji korelacje statystyczne związane z ich przygotowaniem – są to wszakże mieszanki statystyczne stanów czystych. Można zatem podejrzewać, iż rozpoznawanie

splątania w wypadku stanów mieszanych, ze względu na wzajemny wpływ korelacji, będzie znacznie trudniejsze niż w wypadku stanów czystych. Przypuszczenie to potwierdzają trudności na jakie napotykają badacze w tej dziedzinie. Wiadomo, że korelacje statystyczne oraz korelacje kwantowe mogą na siebie wzajemnie wpływać.

Celem algorytmu kwantowego jest otrzymanie określonego rozkładu prawdopodobieństwa otrzymania w wyniku pomiaru pewnych wyników. Ponieważ splątanie jest typem korelacji, jakie mogą pojawić się między rozkładami, można zapytać jaki jest – jeżeli jest jakikolwiek – wpływ splątania na rozkłady otrzymywane podczas wykonywania algorytmów kwantowych. Inaczej mówiąc – jaki jest związek pomiędzy ewolucją układu kwantowego a występowaniem podczas tej ewolucji stanów splątanych.

Znając odpowiedź na te pytania można pokusić się o próby manipulacji rozkładami prawdopodobieństwa występującymi w przyrodzie. Prowadzi to do możliwości projektowania i wykonywania algorytmów o niespotykanej dotychczas wydajności.

Wiadomo, że wiele zagadnień można rozwiązywać szybciej przy pomocy probabilistycznych maszyn Turinga, czyli wprowadzając do obliczeń element losowości. Kwantowy model obliczeń jest rozszerzeniem modelu probabilistycznego. Wobec powyższego pojawia się także pytanie: *Czy komputery kwantowe mogą efektywnie symulować maszyny probabilistyczne?*

1.2 Stany splątane

Aby obserwować i mierzyć stopień splątania musimy najpierw określić kiedy stan układu jest splątany. Rozpatrując różne rodzaje korelacji, jakie można uzyskać w układach kwantowych, splątanie pojawia się w sposób naturalny.

Poniższe określenie pochodzi z pracy [75]. Należy zaznaczyć, że nie daje ono definicji splątania, a jedynie definicję stanu splątanego.

Oznaczamy przez $\mathcal{S}(\mathcal{H})$ zbiór wszystkich operatorów gęstości (stanów) na przestrzeni Hilberta $\mathcal{H}$. Zbiór $\mathcal{S}(\mathcal{H})$ jest podzbiorem wypukłym zbioru $\mathcal{T}(\mathcal{H})$ operatorów klasy śladowej na $\mathcal{H}$. Zbiór $\mathcal{T}(\mathcal{H})$ z normą $||A||_1 := \sqrt{\mathrm{Tr}(A^\dagger A)}$ jest przestrzenią Banacha.

Definicja 1 *Niech $\mathcal{H}_A$ oraz $\mathcal{H}_B$ będą przestrzeniami Hilberta. Stan ρ na przestrzeni $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$ jest nazywany stanem niesplątanym lub separowalnym (ang.* separable*), jeżeli istnieje ciąg dodatnich liczb rzeczywistych oraz ciągi $\{\rho_i^A \in \mathcal{S}(\mathcal{H}_\mathcal{A})\}$ i $\{\rho_i^B \in \mathcal{S}(\mathcal{H}_\mathcal{B})\}$ operatorów gęstości na podprzestrzeniach $\mathcal{H}_B$ i $\mathcal{H}_A$ takie, że szereg*

$$\rho = \sum_i p_i \rho_i^A \otimes \rho_i^B \tag{1.1}$$

jest zbieżny w normie śladowej. Stan jest splątany, jeżeli taki szereg nie istnieje.

Rozkład z Definicji 1 nie jest jednoznaczny. Nie daje on też możliwości łatwego określenia, czy dany stan jest splątany, gdyż dla zadanego $\rho \in \mathcal{S}(\mathcal{H}_A \otimes \mathcal{H}_B)$ znalezienie takiego rozkładu lub udowodnienie, że rozkład taki nie istnieje, jest zadaniem trudnym.

Zastanawiając się jakiego rodzaju korelacje mogą występować podczas przygotowania stanu kwantowego możemy natknąć się na trzy przypadki.
Dla $\rho^A \in \mathcal{S}(\mathcal{H}_A), \rho^B \in \mathcal{S}(\mathcal{H}_B)$ oraz $\rho \in \mathcal{S}(\mathcal{H}_A \otimes \mathcal{H}_B)$ istnienie rozkładu

$$\rho = \rho^A \otimes \rho^B$$

powoduje, że wartości oczekiwane obserwabli postaci $X_A \otimes X_B$ spełniają warunek

$$\begin{aligned} \mathrm{Tr}(\rho X_A \otimes X_B) &= \mathrm{Tr}(\rho X_A \otimes \mathbb{I}) \mathrm{Tr}(\rho \mathbb{I} \otimes X_B) \\ &= \mathrm{Tr}(\rho_A X_A) \mathrm{Tr}(\rho_B X_B), s \end{aligned}$$

czyli faktoryzują się na rozkłady na przestrzeniach podukładów. Stan ten jest zupełnie nieskorelowany. Przygotowanie takiego stanu może odbyć się za pomocą dwóch, nie zależnych od siebie urządzeń – dla porządku oznaczmy je przez A i B. Dysponując generatorem liczb losowych możemy skorelować te urządzenia, wytwarzając w każdym ρ_k, jeżeli wylosowana jest liczba k. Zakładając generator produkuje liczby $\{1, 2, \ldots, k, \ldots, n\}$ z prawdopodobieństwami $\{p_k | p_k = p(k), k =, 1, \ldots, n\}$ otrzymujemy wówczas stan

$$\rho = \sum_{k=1}^{n} p_k \rho_k^A \otimes \rho_k^B$$

dla którego wartości oczekiwane obserwabli postaci $X_A \otimes X_B$ spełniają zależność

$$\begin{aligned} \mathrm{Tr}(\rho X_A \otimes X_B) &= \sum_{i=1}^{n} p_i \mathrm{Tr}\left(\rho_i^A \otimes \rho_i^B X_A \otimes X_B\right) \\ &= \sum_{i=1}^{n} p_i \mathrm{Tr}\left(\rho_i^A X_A\right) \mathrm{Tr}\left(\rho_i^B X_B\right). \end{aligned}$$

Nie dochodzi tu zatem do prostej faktoryzacji – mówimy, że stan taki jest klasycznie skorelowany.

Wiedząc, iż możliwe jest występowanie korelacji kwantowych – tj. różnych od korelacji klasycznych – dochodzimy do określenia stanu splątanego zawartego w Definicji 1.

Pierwszym ograniczeniem narzuconym na Definicję 1 jest warunek, iż układ dla którego stanu badamy stopień splątania jest układem dwuskładnikowym. Niemal cała zgromadzona dotychczasowa wiedza na temat stanów splątanych ogranicza się do sytuacji, gdy rozpatrywany układ potraktujemy jako układ

dwuskładnikowy. Opis i klasyfikacja stanów splątanych układów trój- i wieloskładnikowych jest znacznie trudniejsza.

Jest to sytuacja analogiczna do spotykanej w teorii prawdopodobieństwa: nie potrafimy wprowadzić miary korelacji, która pozwalałaby na pełną charakterystykę zależności pomiędzy trzema zmiennymi losowymi. Miar splątania dla układów dwuskładnikowych możemy użyć do badania stopnia splątania układów wieloskładnikowych, ale takie podejście nie opisuje wszystkich korelacji w układzie [10].

Ogólna definicja stanu splątanego może zostać uproszczona w szczególnym przypadku stanów czystych układu dwuskładnikowego. Istnieje kryterium, dające warunek konieczny i wystarczający, pozwalające orzec czy dany wektor stanu postaci $|\psi_1\rangle \otimes |\psi_2\rangle$ jest splątany. Opiera się ono na następującym twierdzeniu [60].

Twierdzenie 1 (Rozkład Schmidta) *Niech dany będzie wektor $|\psi\rangle$ w przestrzeni Hilberta $\mathcal{H}_A \otimes \mathcal{H}_B$, reprezentujący stan czysty układu złożonego. Wówczas spełniona jest równość*

$$|\psi\rangle = \sum_i \sqrt{p_i}|i\rangle_A \otimes |i\rangle_B \tag{1.2}$$

gdzie $|i\rangle_A$ i $|i\rangle_B$ to wektory baz ortonormalnych odpowiednio w podprzestrzeniach H_A i H_B

Z istnienia rozkładu Schmidta wynika następujący wniosek, dający warunek konieczny i wystarczający na to, aby stan był separowalny

Wniosek 1.1 *Stan czysty $|\psi\rangle$ układu dwuskładnikowego jest stanem separowalnym wtedy i tylko wtedy, gdy można go zapisać w postaci jednoskładnikowego rozkładu Schmidta.*

$$|\psi\rangle = |x\rangle_A \otimes |y\rangle_B$$

Dowód.

Oznaczmy rozpatrywany stan przez $|\psi\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$.

$\Rightarrow$ Zgodnie z definicją jeżeli $|\psi\rangle$ jest stanem separowalnym to można zapisać odpowiadający mu operator gęstości $|\psi\rangle\langle\psi|$ jako sumę operatorów gęstości dla stanów nieskorelowanych

$$\begin{aligned} |\psi\rangle\langle\psi| &= \sum_i p_i |i\rangle\langle i|_A \otimes |i\rangle\langle i|_B \\ &= \sum_i p_i |i\rangle_A\langle i|_A \otimes |i\rangle_B\langle i|_B \end{aligned}$$

Oznaczając $|a\rangle = \sum_i \frac{1}{\sqrt{p_i}}|i\rangle_A \in \mathcal{H}_A$ oraz $|b\rangle = \sum_i \frac{1}{\sqrt{p_i}}|i\rangle_B \in \mathcal{H}_B$ otrzymujemy, iż

$$|\psi\rangle\langle\psi| = (|a\rangle \otimes |b\rangle)(\langle a| \otimes \langle b|)$$

co dowodzi tezy.

$\Leftarrow$ Jeżeli można wektor stanu zapisać w postaci

$$|\psi\rangle = a_i|a\rangle \otimes |b\rangle, \quad |a\rangle \in \mathcal{H}_a, |b\rangle \in \mathcal{H}_b$$

to odpowiadająca mu operator gęstości odpowiada rzutowaniu na podprzestrzeń rozpiętą przez wektor $|a\rangle \otimes |b\rangle$, czyli operator gęstości dla tego stanu ma postać

$$|\psi\rangle\langle\psi| = (|a\rangle \otimes |b\rangle)(\langle a| \otimes \langle b|)$$

□

Uzyskaliśmy w ten sposób pierwsze kryterium separowalności. W przypadku stanów czystych jest ono równoważne Definicji 1.

Najbardziej znanym przykładem stanów splątanych są tak zwane stany Bella. Oznaczmy przez $\{|0\rangle, |1\rangle\} \subset \mathcal{H}$ bazę w przestrzeni Hilberta opisującej układ o dwóch stanach. Są to stany układu dwuskładnikowego opisanego przestrzenią $\mathcal{H} \otimes \mathcal{H}$, w którym każdy z podukładów może być w dwóch stanach. Mają w oznaczonej powyżej bazie postać

$$|\psi_+\rangle = \frac{1}{\sqrt{2}}(|0\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle) \tag{1.3}$$

$$|\psi_-\rangle = \frac{1}{\sqrt{2}}(|0\rangle \otimes |1\rangle - |1\rangle \otimes |0\rangle) \tag{1.4}$$

$$|\phi_+\rangle = \frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle) \tag{1.5}$$

$$|\phi_-\rangle = \frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle - |1\rangle \otimes |1\rangle) \tag{1.6}$$

Stany te bywają także nazywane „stanami EPR" lub „parami EPR".

Stany splątane układów wieloskładnikowych są słabo poznane. W przypadku układów trójskładnikowych dwa ważniejsze przykłady stanów splątanych to stan $|GHZ\rangle$ oraz stan $|W\rangle$.

W bazie $\{|0\rangle, |1\rangle\}$ mają one postać

$$|GHZ\rangle = \frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle \otimes |0\rangle - |1\rangle \otimes |1\rangle \otimes |1\rangle) \tag{1.7}$$

oraz

$$|W\rangle = \frac{1}{\sqrt{3}}(|0\rangle \otimes |1\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle \otimes |1\rangle + |1\rangle \otimes |1\rangle \otimes |0\rangle) \tag{1.8}$$

Stany te reprezentują dwa nierównoważne rodzaje splątania w układach trójskładnikowych.

Pewnym sposobem obliczania korelacji kwantowych jest wprowadzone w pracy [10] splątanie probabilistyczne i związana z nim funkcja splątania – funkcja

opisująca korelacje zawarte w stanie układu kwantowego. Pozwala ona między innymi wyodrębnienie korelacji kwantowych i klasycznych. Dzięki temu daje ona pełną charakterystykę korelacji pomiędzy stanami podukładów kwantowych. W pracy [11] można znaleźć przykłady obliczeń funkcji splątania dla stanów $|W\rangle$ i $|GHZ\rangle$.

1.3 Kryteria splątania

Ocena, czy dany stan jest stanem splątanym jest łatwa jedynie w przypadku stanów czystych układy dwuskładnikowego. Kolejnym krokiem w rozwoju teorii splątania jest opis stanów mieszanych układów dwuskładnikowych.

Wszystkie poniższe kryteria odnoszą się do stanów tego rodzaju. Większość z nich daje warunek konieczny na to, aby stan układu był separowalny.

Brak też kryteriów oceny stopnia splątania dla układów wieloskładnikowych – dotyczy to zarówno stanów czystych, jak i stanów mieszanych. Ogranicza to możliwość badania efektów kwantowych w przypadku układów wieloskładnikowych, a z takimi mamy do czynienia w kwantowej teorii informacji.

1.3.1 Kryterium Peresa-Horodeckiego

Kryterium to zostało podane w pracach [58, 34] i jest związane z własnościami widma operatora gęstości. Okazuje się, że dla układów opisanych w przestrzeniach o wymiarach 2×2 oraz 2×3 dodatniość częściowej transpozycji operatora gęstości jest warunkiem koniecznym i wystarczającym dla separowalności stanu.

Niech dane będą dwie przestrzenie Hilberta $\mathcal{H}_A$ i $\mathcal{H}_B$ – pierwsza z nich opisuje układ A, druga układ B – oraz ustalone w nich układy wektorów bazowych $\{|a_i\rangle | i = 1, \ldots, \dim \mathcal{H}_A\} \subset \mathcal{H}_A$ i $\{|b_i\rangle | i = 1, \ldots, \dim \mathcal{H}_B\} \subset \mathcal{H}_B$. Jeżeli $\rho \in \mathcal{S}(\mathcal{H}_a \otimes \mathcal{H}_b)$ jest stanem układu $A+B$ powstałego z połączenia układów A i B to jego elementy macierzowe w bazie $\{|a_i\rangle \otimes |b_j\rangle | i, j = 1, \ldots, \dim A \times \dim B\}$ oznaczamy przez

$$\rho_{i,j,k,l} \equiv \langle a_i, b_j | \rho | a_k, b_l \rangle \tag{1.9}$$

Zdefiniujmy operację częściowej transpozycji dla macierzy gęstości układu dwuskładnikowego.

Definicja 2 *Operacja częściowej $\cdot^{T_A}$ transpozycji stanu układu $A+B$ względem stanu układu A na operatorze gęstości $\rho \in \mathcal{S}(\mathcal{H}_a \otimes \mathcal{H}_b)$ daje operator $\rho^{T_A} \in \mathcal{T}(\mathcal{H}_a \otimes \mathcal{H}_b)$ o elementach macierzowych określonych wzorem*

$$\langle a_k, b_j | \rho^{T_A} | a_i, b_l \rangle := \langle a_i, b_j | \rho | a_k, b_l \rangle \tag{1.10}$$

Analogicznie definiujemy operację $\cdot^{T_B}$ transpozycji częściowej względem stanu podukładu B.

Jak wiadomo stan układu kwantowego jest opisany przez operator $\rho : \mathcal{H} \to \mathcal{H}$, który musi spełniać następujące warunki:

1. $\mathrm{Tr}\,(\rho) = 1$
2. $\rho = \rho^\dagger$
3. $\bigwedge_{|\psi\rangle\in\mathcal{H}} \langle\psi|\rho|\psi\rangle \geq 0$

Każda z tych własności może być wyrażona jako własność wartości własnych ρ_n operatora ρ. Mamy odpowiednio:

1. $\mathrm{Tr}\,(\rho) = 1 \Rightarrow \sum_n \rho_n = 1$
2. $\rho = \rho^\dagger \Rightarrow \overline{\rho_n} = \rho_n$
3. $\bigwedge_{|\psi\rangle\in\mathcal{H}} \langle\psi|\rho|\psi\rangle \geq 0 \Rightarrow \rho_n \geq 0$

gdzie $\overline{\rho_n}$ oznacza sprzężenie zespolone liczby ρ_n.

Operator o elementach macierzowych określonych równaniem 1.10 jest operatorem samosprzężonym, spełniającym warunek $\mathrm{Tr}\left(\rho^{T_A}\right) = 1$. Nie musi on jednak spełniać warunku dodatniej określoności

$$\bigwedge_{|\psi\rangle\in\mathcal{H}} \langle\psi|\rho|\psi\rangle \geq 0, \tag{1.11}$$

czyli ρ^{T_A} w ogólności nie określa stanu na $\mathcal{S}\left(\mathcal{H}_A \otimes \mathcal{H}_B\right)$.

Kryterium Peresa-Horodeckiego wykorzystuje dodatnią określoność macierzy gęstości. Oznaczmy przez $\mathcal{SEP}\left(\mathcal{H}\right)$ zbiór stanów separowalnych (niesplątanych) na przestrzeni Hilberta $\mathcal{H}$.

Twierdzenie 2 (Kryterium Peresa-Horodeckiego) *Transpozycja częściowa stany separowalnego $\rho \in \mathcal{SEP}\left(\mathcal{H}_A \otimes \mathcal{H}_B\right)$ jest operatorem dodatnim*

$$\bigwedge_{\rho\in\mathcal{SEP}(\mathcal{H}_A\otimes\mathcal{H}_B)} \rho^{T_A} > 0 \text{ oraz } \rho^{T_B} > 0. \tag{1.12}$$

Na podstawie kryterium Peresa-Horodeckiego możemy wprowadzić użyteczną miarę splątania – ujemność stanu (ang. *negativity*) [76]. Ponieważ pozwala ona na wykonanie obliczeń numerycznych, zostanie ona wykorzystana w dalszej części pracy.

Dla układów niskowymiarowych można wymienić jeszcze dwa użyteczne kryteria, które jednakże są powiązane z kryterium Peresa-Horodeckiego.

1.3.2 Kryterium redukcji

Pierwsze z nich to kryterium redukcji. Zostało on przedstawione w pracy [33].

Twierdzenie 3 (Kryterium redukcji) *Jeżeli stan $\rho \in \mathcal{S}\left(\mathcal{H}_A \otimes \mathcal{H}_B\right)$ jest separowalny to*

$$\rho_A \otimes \mathbb{I} - \rho \geq 0 \; oraz \; \mathbb{I} \otimes \rho_B - \rho \geq 0 \tag{1.13}$$

W przypadku układów opisanych w przestrzeniach o wymiarach 2×2 oraz 2×3 kryterium to jest równoważne z kryterium Peresa-Horodeckiego. Natomiast dla układów opisanych w przestrzeniach o większej liczbie wymiarów, wynika ono z kryterium Peresa-Horodeckiego.

Oba te kryteria są przykładem zastosowania operatorów dodatnich do badania splątania.

1.3.3 Kryterium majoryzacji

Kryterium majoryzacji [54, 8] opiera się również na własnościach spektrum macierzy gęstości. Okazuje się, że stany separowalne są bardziej nieuporządkowane globalnie niż lokalnie.

Oznaczmy przez $\lambda_\rho^\downarrow$ ciąg zbudowany z wartości własnych operatora ρ ułożonych w porządku malejącym.

Definicja 3 *Mówimy, że ciąg $x = \{x_i\}_{i=1}^d$ jest zmajoryzowany przez ciąg $y = \{y_i\}_{i=1}^d$ jeżeli*

$$\bigwedge_{1 \leq k < d} \sum_{j=1}^{k} x_j \leq \sum_{j=1}^{k} y_j \tag{1.14}$$

oraz

$$\sum_{j=1}^{d} x_j = \sum_{j=1}^{d} y_j. \tag{1.15}$$

Piszemy wówczas $x \prec y$.

Określenie powyższe pozwala na wprowadzenie kolejnego kryterium splątania.

Twierdzenie 4 (Kryterium majoryzacji) *Dla stanu separowalnego układ dwusykładnikowego $\rho \in \mathcal{S}(\mathcal{H}_A \otimes \mathcal{H}_B)$ zachodzą relacje:*

$$\lambda_\rho^\downarrow \prec \lambda_{\rho_A}^\downarrow \text{ oraz } \lambda_\rho^\downarrow \prec \lambda_{\rho_B}^\downarrow \tag{1.16}$$

przy czym do ciągów odpowiadającym zredukowanym macierzom gęstości dodajemy na końcu zera, ponieważ ich długości są mniejsze niż ciągu odpowiadającego pełnej macierzy gęstości.

Kryterium majoryzacji zawodzi w większej liczbie przypadków niż kryteria redukcji i Peresa-Horodeckiego. Nie daje ona także prostej metody obliczania stopnia splątania. Dla układów opisanych w przestrzeniach o wymiarach 2×2 oraz 2×3 kryterium to wynika z kryterium Peresa-Horodeckiego [8].

1.3.4 Inne kryteria

Poniżej podane są kryteria, które dają warunki konieczne i wystarczające separowalności dla układów dwuskładnikowych. Ich wadą jest to, iż nie dają prostego sposobu na określenie, czy zadany stan jest splątany.

Odwzorowania dodatnie

W pracy [34] sformułowane zostało kryterium wykorzystujące odwzorowania dodatnie do analizy splątania stanów układów dwuskładnikowych.

Kryteria Peresa-Horodeckiego oraz kryterium redukcji są szczególnymi przypadkami zastosowania odwzorowań dodatnich do badania separowalności stanów. Operacja częściowej transpozycji wykorzystywana w pierwszym z nich oraz operacja $T(\rho) = \mathrm{Tr}\,(\rho)\,\mathbb{I} - \rho$ obecna w sformułowaniu drugiego, są przykładami operacji dodatnich, ale nie całkowicie dodatnich.

Definicja 4 *Mówimy, że operator ρ jest dodatni jeżeli*

$$\bigwedge_{|\phi\rangle\in\mathcal{H}} \langle\phi|\rho|\phi\rangle \geq 0. \tag{1.17}$$

Zbiór wszystkich operatorów dodatnich na przestrzeni Hilberta $\mathcal{H}$ oznaczamy przez $\mathcal{S}_+(\mathcal{H})$.

Niech $\mathcal{A}$ i $\mathcal{B}$ oznaczają odpowiednio zbiory operatorów działających na przestrzeniach $\mathcal{H}_A$ oraz $\mathcal{H}_B$. Zbiór wszystkich odwzorowań liniowych na przestrzeni $\mathcal{A}$ o wartościach w przestrzeni $\mathcal{B}$ oznaczamy przez $\mathcal{L}(\mathcal{A},\mathcal{B})$.

Definicja 5 *Odwzorowanie $F \in \mathcal{L}(\mathcal{A},\mathcal{B})$ nazywamy odwzorowaniem dodatnim jeżeli przeprowadza operatory dodatnie w operatory dodatnie czyli*

$$\bigwedge_{\rho\in\mathcal{S}_+(\mathcal{H}_A)} F(\rho) \in \mathcal{S}_+(\mathcal{H}_B) \tag{1.18}$$

Definicja 6 *Odwzorowanie $F \in \mathcal{L}(\mathcal{A},\mathcal{B})$ nazywamy całkowicie dodatnim jeżeli dla dowolnego $n \in \mathbb{N}$ i dowolnej przestrzeni Hilberta $\mathcal{M}_n$, $\dim \mathcal{M}_n = n$ odwzorowanie*

$$F \otimes \mathbb{I}_n \in \mathcal{L}(\mathcal{H}_A \otimes \mathcal{M}_n, \mathcal{H}_B \otimes \mathcal{M}_n) \tag{1.19}$$

jest dodatnie.

Twierdzenie 5 *Stan $\rho \in \mathcal{S}(\mathcal{H}_A \otimes \mathcal{H}_B)$ układu dwuskładnikowego jest stanem separowalnym wtedy i tylko wtedy, gdy dla dowolnego odwzorowania dodatniego Λ zachodzi warunek*

$$(\mathbb{I} \otimes \Lambda)\rho \geq 0 \tag{1.20}$$

Wykorzystanie tego kryterium utrudnia fakt, iż nie dysponujemy charakterystyką zbioru odwzorowań dodatnich. Naturalnie najbardziej interesujące są odwzorowania dodatnie, ale nie całkowicie dodatnie.

Świadek splątania

Kryterium to zostało podane w pracy[34].

Twierdzenie 6 *Stan $\rho \in \mathcal{S}(\mathcal{H}_A \otimes \mathcal{H}_B)$ układu dwuskładnikowego jest stanem splątanym wtedy i tylko wtedy, gdy istnieje operator hermitowski $\mathcal{W}$ taki, że*

$$\mathrm{Tr}(\mathcal{W}\rho) < 0 \tag{1.21}$$

oraz

$$\bigwedge_{\sigma \in \mathcal{SEP}(\mathcal{H})} \mathrm{Tr}(\mathcal{W}\sigma) \geq 0. \tag{1.22}$$

Można zatem powiedzieć, że operator $\mathcal{W}$ wykrywa splątanie i z tego powodu nazywany jest świadkiem splątania (ang. *entanglement witness*).

Przegląd kryteriów i ich stosunek do możliwości eksperymentalnego wykrywania splątania można znaleźć w [69].

1.4 Miary splątania

Naturalne dążenie do klasyfikacji stanów splątanych prowadzi do pojęcia miar splątania. W przesyłaniu i przetwarzaniu informacji splątanie wydaje się być bardzo cennym zasobem. W przypadku pojedynczego wykonania algorytmu kwantowego interesuje nas ile splątania mamy do dyspozycji i ile zużywamy w trakcie wykonania zadania. Mając dany algorytm chcemy wiedzieć, czy splątane jest konieczne do jego realizacji fizycznej – jeżeli tak jest to implementacja może okazać się trudniejsza.

Określenia występujące w tym rozdziale zostały zaczerpnięte głównie z prac [19, 39, 32]. Głównym obiektem zainteresowania w tej pracy są miary pozwalające wyliczyć stopień splątania dwóch podukładów układu złożonego. W Rozdziale 4 zostaną one wykorzystane przy badaniu zmiany stopnia splątania podczas wykonywania niektórych algorytmów kwantowych.

Najważniejszą grupą miar splątania, która nie jest rozpatrywana w tym rozdziale są miary operacyjne. Zostały one wprowadzona w celu opisu przetwarzania splątania podczas ewolucji układu kwantowego [19]. Są to miary opisujące w języku operacyjnej mechaniki kwantowej procesy takie jak tworzenie i nie dają one możliwości wykonania obliczeń numerycznych.

1.4.1 Wymagania stawiane miarom splątania

Mierząc splątanie możemy zacząć traktować je jako pewną wielkość fizyczną. Analogicznie jak to się ma w przypadku ładunku czy entropii, można postawić pewne warunki dotyczące zachowania się splątania w trakcie ewolucji układu kwantowego. Zestaw warunków nakładanych na miarę splątania zależy od naszego zrozumienia tego, czym jest splątanie.

Każda fizycznie akceptowalna miara splątania powinna spełniać następujące warunki [19, 39, 8]

[S0] Miara splątania jest to funkcja rzeczywista o wartościach dodatnich, określona na zbiorze stanów układu kwantowego $E : S(\mathcal{H}) \rightarrow \mathbb{R}^+$

Warunek ten jest zgodny z przyjętym pojęciem miary. Implikuje on możliwość uporządkowania zbioru stanów ze względu na ich stopień splątania.

[S1] Jeżeli ρ jest stanem separowalnym (niesplątanym) to $E(\rho) = 0$. Jeżeli $\sigma \in \mathcal{S}(\mathcal{H})$ jest stanem maksymalnie splątanym, to $E(\sigma) = \log_2 \dim \mathcal{H}$

Stawiając ten warunek uznajemy, że możemy wprowadzić ograniczenie górne i dolne na ilość splątania obecnego w układzie. Wielkość, która znika na stanach separowalnych, daje nam jedynie kryterium splątania.

[S2] Operacje lokalne na jednym z podukładów układu złożonego nie mogą zwiększyć splątania.

Jest to warunek wynikający z rozumienia splątania jako wielkości dotyczącej układu jako całości. Zwiększyć stopień splątania możemy jedynie poprzez operacje na całym układzie, natomiast operując na jednym z podukładów możemy niszczyć splątania – na przykład poprzez przestrzenne odseparowanie jednego podukładu od drugiego.

[S3] Wypukłość

$$E(\lambda\rho + (1-\lambda)\sigma) \leq \lambda E(\rho) + (1-\lambda)E(\sigma) \tag{1.23}$$

Jest to warunek oznaczający, iż mieszanie stanów nie może zwiększyć splątania. Oznacza, że oddzielamy korelacje związane z mieszaniem (korelacje klasyczne) od korelacji kwantowych.

[S4] Ciągłość
Niech $\{\mathcal{H}_n | n \in \mathbb{N}\}$ oraz $\{\mathcal{K}_n | n \in \mathbb{N}\}$ będą ciągami przestrzeni Hilberta. Dla dowolnych ciągów stanów

$$\{\rho_n \in \mathcal{S}(\mathcal{H}_n \otimes \mathcal{K}_n) | n \in \mathbb{N}\} \text{ oraz } \{\sigma_n \in \mathcal{S}(\mathcal{H}_n \otimes \mathcal{K}_n) | n \in \mathbb{N}\}$$

takich, że

$$||\rho_n - \sigma_n||_{Tr} \overset{n\rightarrow\infty}{\longrightarrow} 0$$

zachodzi

$$\lim_{n\rightarrow\infty} \frac{E(\rho_n) - E(\sigma_n)}{1 + \log_2 dim(\mathcal{H} \otimes \mathcal{K})} = 0. \tag{1.24}$$

Warunek ten wyraża wymaganie, by małe zmiany stanu powodowały małe zmiany miary splątania.

[S5] Addytywność
Warunek addytywności jest jednym z bardziej restrykcyjnych ograniczeń narzucanych na potencjalne miary splątania. Można go sformułować na różne sposoby w zależności od wagi jaką kładziemy na tę własność.

[S5a] Ścisła addytywność

$$E(\rho \otimes \sigma) = E(\rho) + E(\sigma) \tag{1.25}$$

[S5b] Podaddytywność

$$E(\rho \otimes \sigma) \leq E(\rho) + E(\sigma) \tag{1.26}$$

[S5c] Słaba addytywność

$$E(\rho^{\otimes n}) = \frac{1}{n} E(\rho) \tag{1.27}$$

[S5d] Istnienie regularyzacji
Dla każdego $\rho \in \mathcal{S}(\mathcal{H}_A \otimes \mathcal{H}_B)$ istnieje granica

$$E^{\infty}(\rho) := \lim_{n \to \infty} \frac{E(\rho^{\otimes n})}{n} \tag{1.28}$$

Podanie dobrej, spełniającej powyższy zestaw warunków, miary splątania jest trudne, a w wielu przypadkach nie potrafimy odpowiedzieć na pytanie, czy dana miara spełnia powyższe warunki.

1.4.2 Miary abstrakcyjne

Główną motywacją dla wprowadzenia miar określonych mianem abstrakcyjnych [19] są warunki, które powinna spełniać każda użyteczna miara splątania. Miary te – w przeciwieństwie do miar operacyjnych – nie dają bezpośredniego wglądu w proces przetwarzania splątania podczas ewolucji układu kwantowego. Są one jednak użyteczne z dwóch powodów

1. dają możliwość wykonania przy ich pomocy obliczeń stopnia splątania dla niektórych stanów,
2. dzięki twierdzeniu 7 o jednoznaczności upraszają zagadnienie dla stanów czystych.

Splątanie formacji

Najważniejszą miarą z tej grupy jest splątanie formacji (ang. *entanglement of formation*).

Dla stanów czystych jest ona zdefiniowana jako entropia von Neumana dla zredukowanej macierzy gęstości układu.

Niech $|\psi\rangle$ będzie wektorem jednostkowym w przestrzeni Hilberta układu złożonego $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$. Przez $|\psi\rangle\langle\psi|$ oznaczamy operator rzutujący na podprzestrzeń rozpiętą przez wektor $|\psi\rangle$.

Definicja 7 *Zredukowaną entropią von Neumana dla stanu czystego* $|\psi\rangle\langle\psi|$ *nazywamy wielkość* $S_{vN}(|\psi\rangle\langle\psi|)$ *określoną wzorem*

$$S_{vN}(|\psi\rangle\langle\psi|) := S(\mathrm{Tr}_A\left(|\psi\rangle\langle\psi|\right)\log \mathrm{Tr}_A\left(|\psi\rangle\langle\psi|\right)). \tag{1.29}$$

Z rozkładu Schmidta wynika, że wartość zredukowanej entropii von Neumana nie jest zależna od tego, względem którego podukładu dokonujemy operacji śladu częściowego[1].

$$\begin{aligned} S_{vN}(|\psi\rangle\langle\psi|) &= -\mathrm{Tr}_B\left(\mathrm{Tr}_A\left(|\psi\rangle\langle\psi|\right)\log \mathrm{Tr}_A\left(|\psi\rangle\langle\psi|\right)\right) \\ &= -\mathrm{Tr}_A\left(\mathrm{Tr}_B\left(|\psi\rangle\langle\psi|\right)\log \mathrm{Tr}_B\left(|\psi\rangle\langle\psi|\right)\right) \end{aligned}$$

Entropia von Neumana będzie podstawowym narzędziem wykorzystywanym przy obliczeniach numerycznych w tej pracy.

Znając geometrię zbioru stanów układu kwantowego splątanie formacji można intuicyjnie uogólnić na stany mieszane.

Definicja 8 *Splątaniem formacji dla stanu mieszanego* $\rho \in \mathcal{S}\left(\mathcal{H}_A \otimes \mathcal{H}_B\right)$ *określamy jako*

$$E_F(\rho) := \inf \sum_i p_i S_{vN}(|\psi_i\rangle\langle\psi_i|)$$

gdzie infimum jest wzięte po wszystkich rozkładach stanu mieszanego ρ *na wypukłe kombinacje liniowe stanów czystych.*

Uogólnienie to jest proste z matematycznego punktu widzenia, ale znacznie komplikuje obliczenia dla stanów mieszanych.

Miary, które są – jak w przypadku splątania formacji – zdefiniowane na stanach czystych, a następnie uogólniane na stany mieszane $\sum_i p_i|i\rangle\langle i|$ zgodnie z regułą

$$E(\rho) = \inf_{\sum_i p_i|i\rangle\langle i|} \sum_i p_i E(|i\rangle\langle i|) \tag{1.30}$$

nazywamy miarami stromego dachu (ang. *convex roof measure*) [32]. Infimum jest wzięte po wszystkich rozkładach stanu mieszanego ρ na wypukłe kombinacje liniowe stanów czystych.

Miary tego typu spełniają na mocy samej definicji niektóre własności jakich oczekujemy do miar splątania – dotyczy to zwłaszcza to warunku [S3] czyli wypukłości.

W przypadku stanów czystych problem znalezienia odpowiedniej miary splątania rozstrzyga następujący fakt

Twierdzenie 7 (O jednoznaczności miary splątania) *[19, 39] Zredukowana entropia von Neumana* S_{vN} *jest jedyną miarą splątania dla stanów czystych, która spełnia warunki* [S0]-[S4] *oraz warunek* [S5a].

[1] Omówienie najważniejszych własności entropii von Neumana znaleźć można w Dodatku matematycznym.

Wykorzystując to twierdzenie możliwe jest budowanie kwantowej teorii informacji i kodowania opartej na własnościach entropii von Neumana, wzorując się na rezultatach klasycznej teorii informacji opartej na entropii Gibbsa-Shanona.

Jak dotychczas w przypadku stanów mieszanych nie udało się uzyskać podobnego rezultatu. Możliwe jest jedynie podanie ograniczenia górnego i dolnego na miary splątania.

Względna entropia splątania

W celu określenia kolejnej miary splątania zdefiniujemy kwantową entropię względną

Definicja 9 *Kwantową entropią względną nazywamy wielkość*

$$S_{rel}\left(\rho||\sigma\right) = \mathrm{Tr}\left(\rho\right)\log\rho - \mathrm{Tr}\left(\rho\right)\log\sigma \tag{1.31}$$

Podobnie jak w przypadku splątania formacji definiujemy

Definicja 10 (Względna entropia splątania) *Dla stanu mieszanego* $\rho \in \mathcal{S}\left(\mathcal{H}_A \otimes \mathcal{H}_B\right)$

$$E_R(\rho) := \inf_{\sigma} S_{rel}\left(\rho||\sigma\right) = \mathrm{Tr}\left(\rho\right)\log\rho - \mathrm{Tr}\left(\rho\right)\log\sigma$$

gdzie infimum jest wyznaczane po wszystkich stanach czystych $\sigma \in \mathcal{S}\left(\mathcal{H}_A \otimes \mathcal{H}_B\right)$.

Dla stanów czystych względna entropia splątania jest równoważna splątaniu formacji, które (z definicji) w przypadku stanów czystych jest tożsame ze zredukowaną entropią von Neumana [71]. Okazuje się zatem, że Twierdzenie 7 znacznie ułatwia analizę ilościową splątania dla stanów czystych.

W przypadku stanów mieszanych splątanie formacji jest mniejsze od względnej entropii splątania.

1.4.3 Miary oparte na odległości

Jednym z intuicyjnych sposobów obliczania splątania dla danego stanu jest wyznaczanie jego odległości od stanów separowalnych. Wymaga to jednak dokładnego określania, które stany są separowalne. Nieposiadane kryterium splątania które, jest warunkiem koniecznym i wystarczającym, jest tu dużym utrudnieniem.

Oznaczmy przez $\mathcal{SEP}\left(\mathcal{H}\right)$ zbiór stanów separowalnych układu opisanego przestrzenią Hilberta $\mathcal{H}$. Dla danej odległości $D\colon \mathcal{S}\left(\mathcal{H}\right)\times\mathcal{S}\left(\mathcal{H}\right)\to\mathbb{R}_+$ określamy [32]

Definicja 11 *Dla danego* $\rho \in \mathcal{S}\left(\mathcal{H}\right)$

$$E_{D,S}(\rho) := \inf_{\sigma\in\mathcal{SEP}(\mathcal{H})} D(\rho,\sigma) \tag{1.32}$$

Możliwy jest także inny wybór zbioru stanów, względem którego mierzymy odległość – zbiorem tym może być np. zbiór stanów posiadających dodatnią częściową transpozycję, oznaczany przez $\mathcal{PPT}(\mathcal{H})$ (ang. *positive partial transpose*).

Przykładem miary tego typu jest względna entropia splątania, w przypadku której rolę odległości odgrywa entropia względna.

1.4.4 Ujemności

W pracy [76] podany został opis kilku miar splątania opartych na ogólnej definicji $S-$ujemności (ang. *S-negativitv*). Znajdują się wśród nich: odporność splątania, ujemność oraz ujemność logarytmiczna.

Definicja 12 *Niech S będzie zwartym, wypukłym podzbiorem zbioru operatorów hermitowskich o śladzie 1. Definiujemy $S-$normę $||\cdot||_S$ i odpowiadającą jej S-ujemność $\mathcal{N}(\cdot)_S$ jako*

$$||A||_S := \inf\left\{a_+ - a_-|A = a_+\rho^+ + a_-\rho^-, a_\pm \geq 0, \rho^\pm \in S\right\} \quad (1.33)$$

$$\mathcal{N}_S(A) := \inf\left\{a_-|A = a_+\rho^+ - a_-\rho^-, a_\pm \geq 0, \rho^\pm \in S\right\} \quad (1.34)$$

Jeżeli $\mathrm{Tr}(A) = 1$ to otrzymujemy $||A||_S = 1 + 2\mathcal{N}_S(A)$ [76].

Szczególnymi przypadkami są w takim ujęciu dwie miary, które zostaną omówione poniżej – stabilność splątania oraz ujemność.

Stabilność splątania

Kładąc w Definicji 12 jako S zbiór separowalnych operatorów gęstości, otrzymujemy miarę nazwaną stabilnością splątania[2] (ang. *robustness of entanglement*) [72].

Wiąże ona ze sobą korelacje kwantowe i klasyczne, opisując ich wzajemne oddziaływanie. Określa ona minimalną ilość korelacji klasycznych zawartych w stanie separowalnym ρ_{sep}, która wystarcza do zniszczenia korelacji kwantowych zawartych w stanie ρ.

Aby zdefiniować stabilność wprowadza się najpierw stabilność względną, opisującą ilość korelacji statystycznych zawartych w zadanym stanie ρ_{sep}, która niszczy korelacje kwantowe w stanie ρ.

Oznaczmy przez $\mathcal{SEP}(\mathcal{H})$ zbiór stanów separowalnych na przestrzeni Hilberta $\mathcal{H}$.

Definicja 13 *Niech $\rho \in \mathcal{S}(\mathcal{H})$ oraz $\rho_{sep} \in \mathcal{SEP}(\mathcal{H})$. Stabilnością (splątania) stanu ρ względem stanu ρ_{sep}, $R(\rho||\rho_{sep})$ nazywamy najmniejszą liczbę s, dla której stan*

$$\rho(s) = \frac{1}{1+s}(\rho + s\rho_{sep}) \quad (1.35)$$

[2] W kontekście matematycznym tłumaczenie terminu *robustness* podawane przez Słownik Naukowo-Techniczny angielsko-polski WNT to *odporność testu*

jest stanem separowalnym.

Pewną miarą odporności na szumy losowe jest stabilność względem domieszkowania stanem maksymalnie mieszanym.

Definicja 14 *Stabilnością losową (splątania) stanu ρ nazywamy stabilność względem stanu maksymalnie mieszanego $\frac{1}{n}\mathbb{I}$, $R\left(\rho||\frac{1}{n}\mathbb{I}\right)$.*

Natomiast minimalna wartość domieszek niszcząca splątanie daje nam następująca miarę

Definicja 15 *Stabilnością (splątania) – lub stabilnością bezwzględną – stanu ρ nazywamy wielkość*

$$R\left(\rho\right) = \min_{\rho_{sep}\in\mathcal{SEP}(\mathcal{H})} R\left(\rho||\rho_{sep}\right). \tag{1.36}$$

Stabilność splątania jest wielkością dodatnią i pozwala rozróżnić stany separowalne od splątanych – $\rho \in \mathcal{SEP}\left(\mathcal{H}\right)\ R\left(\rho\right) = 0$.

Ponieważ w realnych układach zawsze mamy do czynienia z dekoherencją, musimy uwzględnić ją przy rozpatrywaniu dynamiki splątania podczas ewolucji układu. Odporność splątania pozwala także na oszacowanie wpływu korelacji klasycznych na korelacje kwantowe.

Ujemność i ujemność logarytmiczna

Ujemność stanu (lub ujemność splątania) jest ilościowym odpowiednikiem kryterium Peresa-Horodeckiego. Obie miary omawiane w tym podrozdziale mierzą stopień niedodatniości macierzy gęstości poddanej operacji częściowej transpozycji, czyli stopień, w jakim dany stan nie spełnia kryterium Peresa-Horodeckiego.

Jeżeli w Definicji 12 weźmiemy $S = \left\{A|A = A^{\dagger}, \mathrm{Tr}\left(A\right) = 1, A^{T_A} \geq 0\right\}$ otrzymujemy następująca definicję ujemności

Definicja 16 $\mathcal{N}\left(\rho\right) := \inf\left\{a_{-}|A^{T_A} = a_{-}\rho^{-} + a_{+}\rho^{+}, a_{\pm} \geq 0\right\}$.

Równoważna jej i bardziej użyteczna jest jednak definicja, która może być sformułowana jako twierdzenie

Twierdzenie 8 *Ujemnością macierzy gęstości $\rho \in \mathcal{S}\left(\mathcal{H}_A \otimes \mathcal{H}_B\right)$ nazywamy wielkość*

$$\mathcal{N}\left(\rho\right) = \frac{||\rho^{T_A}||_1 - 1}{2}, \tag{1.37}$$

gdzie $||\cdot||_1$ oznacza normę śladową.

Operator ρ^{T_A} jest operatorem samosprzężonym i $\mathrm{Tr}\left(\rho^{T_A}\right) = 1$. Oznaczmy przez a_i wartości własne ρ^{T_A}. Obliczając normę $||\rho^{T_A}||_1$ otrzymujemy

$$\begin{aligned} ||\rho^{T_A}||_1 &= \mathrm{Tr}\left(\sqrt{(\rho^{T_A})^\dagger \rho^{T_A}}\right) \\ &= \sum_i \mathrm{sgn}(a_i)\ |a_1| \\ &= \sum_i \left|a_i^+\right| - \sum_i \left|a_i^-\right| \end{aligned}$$

gdzie a_i^+ i a_i^- oznaczają odpowiednio dodatnie i ujemne wartości własne operatora ρ^{T_A}. Dodając i odejmując człon $\sum_i \left|a_i^-\right|$ otrzymujemy

$$\begin{aligned} ||\rho^{T_A}||_1 &= \sum_i \left|a_i^+\right| - \sum_i \left|a_i^-\right| + \sum_i \left|a_i^-\right| - \sum_i \left|a_i^-\right| \\ &= 1 - 2\sum_i \left|a_i^-\right| \\ &= 1 - 2\mathcal{N}(\rho) \end{aligned}$$

Miara ta odpowiada sumie wartości bezwzględnych ujemnych wartości własnych operatora ρ^{T_A} oraz znika dla stanów separowalnych. W pracy [76] pokazano, że nie wzrasta ona podczas wykonywania operacji lokalnych i jest monotoniczną miarą splątania (ang. *entanglement monotone*).

Ważną zaletą ujemności jest łatwość, z jaką można użyć jej do wykonania prostych obliczeń stopnia splątania. Pomimo, że w przeciwieństwie do splątania formacji, ujemność można wykorzystać do obliczeń dla stanów mieszanych, przykłady w Rozdziale 4 będą ograniczone do stanów czystych. Ujemność jednoznacznie orzeka, czy dany stan jest splątany w przypadku, gdy kryterium Peresa-Horodeckiego stanowi warunek konieczny i wystarczający dla separowalności stanu. Zatem, poza bramkami dwuqubitowymi i najprostszymi przypadkami algorytmów kwantowych, można je traktować jedynie jako pewien parametr znamionujący występowanie splątania.

Dodatkowo możemy wprowadzić ujemność logarytmiczną

Definicja 17 *Ujemnością logarytmiczną macierzy gęstości* $\rho \in \mathcal{S}(\mathcal{H}_A \otimes \mathcal{H}_B)$ *nazywamy wielkość*

$$\mathcal{N}_{log}(\rho) := \log_2 ||\rho^{T_A}||_1 \tag{1.38}$$

która również nie wzrasta podczas wykonywania operacji lokalnych oraz jest dodatkowa addytywna [76].

Własności ujemności i ujemności logarytmicznej przedstawione są w pracy [76]. Implementacja funkcji obliczającej ujemność dla zadanego stanu jest częścią pakietu `Entanglment` opisanego w Rozdziale 7.1.2.

1.5 Wnioski

Dostępne miary splątania pozwalają dokonać pomiarów jedynie dla układów dwuskładnikowych. Omawiane w kolejnych rozdziałach tej pracy algorytmy kwantowe mogą zostać przedstawione w postaci odpowiadającej temu podejściu – schemat tego podziału zostanie przedstawiony w kolejnym rozdziale. Pozwala to na śledzenie stopnia splątania podczas wykonania tych algorytmów. Przykładem, przy którym to podejście wydaje się być niewystarczające jest algorytm znajdowania logarytmów dyskretnych, gdyż wówczas używane są trzy rejestry kwantowe.

Podział pamięci komputera kwantowego na dwie części jest uzasadniony tym, że interesuje nas rozkład prawdopodobieństwa na zbiorze danych zapisanym w jednym z rejestrów. Nic nie stoi na przeszkodzie, by operować na komputerze opartym na qutritach – układach o trzech stanch. Operowanie na komputerze kwantowym zbudowanym z wielu qubitów można zastąpić operowaniem na maszynie zbudowanej z dwóch podukładów n-wymiarowych, przy n dobranym odpowiednio do zagadnienia. Operowanie na układach o dwóch stanach jest jedynie wygodną – na przykład przy konstrukcji układu realizującego kwantową transformatę Fouriera – analogią do klasycznych układów logicznych[3].

Nie oznacza to bynajmniej, że zagadnienie splątania w układzie wieloskładnikowym można zawsze sprowadzić do zagadnienia splątania dwóch wyodrębnionych podukładów. Zagadnienie to jest znacznie bardziej skomplikowane, a problem opisu splątania w układach wieloskładnikowych nie ma jak dotychczas zadowalającego rozwiązania.

Narzucającym się sposobem badania splątania w układach wieloskładnikowych jest teoretyczna możliwość badania splątania pomiędzy wszystkimi parami qubitów (podukładów). Odpowiada to tworzeniu macierzy kowariancji podczas badania zależności statystycznych. Możliwe jest także uogólnienie podejścia prezentowanego w tej pracy i badanie stopnia splątania pomiędzy stanami podukładów wyodrębnianych arbitralnie poprzez podziały całości na dwie części. Jednak obliczenia wykonane z zastosowaniem któregokolwiek z tych sposobów nie dostarczałyby pełnej wiedzy na temat korelacji występujących w układzie. Jednocześnie wymagałyby przeprowadzenia dużej ilości obliczeń, gdyż zwykle interesują nas układy o bardzo dużej ilości podukładów – w przypadku badania wszystkich par ich liczba rośnie jak n^2, natomiast w przypadku dzielenia układu na dwa podukłady ich liczba rośnie jak 2^n, gdzie n jest liczbą qubitów.

[3]Analogia ta jest także wygodna do rozwiązywania problemu konstrukcji sieci bramek kwantowych, gdyż pozwala na wykorzystanie dobrze znanych klasycznych metod konstrukcji układów logicznych.

Rozdział 2

Kwantowa transformata Fouriera

Obliczanie transformaty Fouriera na komputerze kwantowym stanowi podstawę wszystkich algorytmów rozwiązujących problem ukrytej podgrupy [31, 53, 51, 37]. Możliwość wykonywania tej procedury w czasie wielomianowym względem rozmiaru danych wejściowych pozwala na skonstruowanie efektywnych algorytmów, które zostaną przedstawione w Rozdziale 3. W szczególności jest to podstawą przyśpieszenia uzyskiwanego przez algorytmy opracowane przez Petera Shora.

Ponieważ szybka transformata Fouriera jest wykorzystywana do konstrukcji algorytmu mnożenia o złożoności $O(n \log n)$, można sobie wyobrazić, iż komputer kwantowy pozwoli na znaczne przyśpieszenie wykonywania także takich elementarnych działań.

W tym rozdziale opisane są najważniejsze aspekty związane z konstrukcją i wykonaniem QFT (ang. **Q***uantum* **F***ourier* **T***ransform*) na grupach abelowych. Transformata QFT może być rozpatrywana bez odniesienia do jej roli w kontekście przetwarzania informacji na komputerze kwantowym, podobnie jak to się ma z klasycznym algorytmem dyskretnej transformaty Fouriera. Ponieważ jednak wszystkie algorytmy, które zostaną przedstawione w Rozdziale 3 wykorzystują tą operację, omówione jest także jej znaczenie dla algorytmu jako ciągu operacji. To pozwoli skupić się nad metodą uzyskiwania dzięki QFT odpowiednich rozkładów prawdopodobieństwa.

2.1 Definicja kwantowej transformaty Fouriera

Do reprezentacji danych wejściowych dla algorytmów kwantowych wykorzystujemy stany układów kwantowych – operacja QFT jest przykładem transformaty, która może być wykonana na stanach kwantowych efektywniej niż na stanach układu klasycznego [35].

Poniżej ograniczymy się do przypadku, gdy w zbiorze danych wejściowych

algorytmu możemy określić strukturę grupy abelowej[1]. Uogólnienie na przypadek grup nieabelowych można znaleźć w pracach [62, 45].

W skończeniewymiarowej przestrzeni Hilberta $\mathcal{H}$ odpowiadającej układowi wybieramy bazę ortonormalną $\{|g\rangle | g \in G\}$ ponumerowaną elementami $g \in G$, gdzie G jest pewną skończoną grupą abelową[2]. Wymiar przestrzeni wynosi w takim wypadku $|G|$, gdzie przez $|\cdot|$ oznaczamy rząd grupy. Dowolny wektor stanu układu $|\psi\rangle$ ma rozkład w bazie $\{|g\rangle | g \in G\}$

$$|\psi\rangle = \sum_{i=1}^{|G|} c_i |g_i\rangle \tag{2.1}$$

W zbiorze odwzorowań liniowych $\{f \colon G \to \mathbb{C}\}$ definiujemy iloczyn skalarny

$$\langle f_1 | f_2 \rangle := \sum_{g \in G} \overline{f_1}(g) f_2(g) \tag{2.2}$$

Współczynniki rozkładu 2.1 mogą być potraktowane jako wartości pewnej funkcji $f \colon G \to \mathbb{C}$ określonej wzorem

$$f(g_i) = c_i \tag{2.3}$$

oraz spełniającej warunek

$$||f|| = 1 \tag{2.4}$$

z normą wyznaczaną przez iloczyn skalarny 2.2.

Transformata Fouriera $\hat{f}$ funkcji f określona jest wzorem[3]

$$\hat{f}(g_j) = \frac{1}{|G|} \sum_{k=0}^{|G|-1} e^{\frac{2i\pi g_j g_k}{|G|}} f(g_k). \tag{2.5}$$

Jednocześnie każde odwzorowanie 2.3 w sposób jednoznaczny definiuje stan na $\mathcal{H}$. Dzięki tej jednoznaczności możemy sformułować następującą definicję

Definicja 18 *Kwantowa transformata Fouriera jest to transformata Fouriera funkcji f zadającej stan układu kwantowego.*

Działanie transformaty Fouriera na stan bazowy $|j\rangle \in \mathcal{H}$ w d-wymiarowej przestrzeni wektorowej, określone jest wzorem

$$\mathrm{QFT}|j\rangle = \frac{1}{\sqrt{d}} \sum_{k=0}^{d-1} e^{\frac{2i\pi jk}{d}} |k\rangle \tag{2.6}$$

[1] Jest to założenie słuszne dla wszystkich algorytmów opisanych w Rozdziale 3

[2] W przypadku algorytmu faktoryzacji interesuje nas okresowość funkcji określonej na $\mathbb{Z}$. Jednak wymaganie operowania na skończonej pamięci zmusza nas do operowania tylko na pewnym podzbiorze $\mathbb{Z}_q$

[3] Ogólna definicja transformaty Fouriera na grupie abelowej przedstawiona jest w Rozdziale 6.4

Dzięki liniowości transformaty Fouriera, wzór ten może być rozszerzony na dowolny wektor i przyjęty za definicją transformaty.

Ponieważ dla funkcji f i jej transformaty zachodzi

Twierdzenie 9 (Tożsamość Parsevala)

$$||f|| = ||\hat{f}||$$

kwantowa transformata Fouriera jest operacją unitarną.

Transformata Hadamarda Najprostszym i najczęściej spotykanym przykładem kwantowej transformaty Fouriera jest transformata Hadamarda. W tym wypadku $G = \mathbb{B} = \{0, 1\}$, a działaniem grupowym jest dodawanie modulo 2. Przestrzeń Hilberta odpowiadająca zbiorowi $\mathbb{B}$ ma bazę $\{|0\rangle, |1\rangle\}$.

Transformata Fouriera na $\mathbb{B}$ jest określona jako odwzorowanie

$$\begin{aligned} \hat{f}(x) &= \frac{1}{\sqrt{2}}\left((-1)^{0\cdot x} f(0) + (-1)^{1\cdot x} f(1)\right) \\ &= \frac{1}{\sqrt{2}} f(0) + (-1)^x f(1) \end{aligned}$$

Odpowienio kwantowa transformata Fouriera działa na stany bazowe zgodnie ze wzorem

$$\begin{aligned} QFT_2|x\rangle &= \frac{1}{\sqrt{2}}\left((-1)^{0\cdot x}|0\rangle + (-1)^{1\cdot x}|1\rangle\right) \\ &= \frac{1}{\sqrt{2}}|0\rangle) + (-1)^x|1\rangle \end{aligned}$$

i jest reprezentowana przez macierz unitarną H

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2.7}$$

Bramka Hadamarda jest przydatna przy konstrukcji transformaty Fouriera dla dowolnej ilości qubitów.

2.2 Rola QFT w algorytmach kwantowych

Aby określić cel jakiemu służy kwantowa transformata Fouriera, należy zbudować ogólny schemat algorytmów wykorzystujących tę operację. Nie przywiązując uwagi do szczegółów można go rozpisać na trzy kroki[4].

Nasz komputer kwantowy dzielimy na dwa rejestry – rejestr danych i rejestr wartości. Zakładamy, że stanem początkowym obu rejestrów jest stan bazowy

[4]Dokładniejszy opis tych algorytmów znajduje się w Rozdziale 3

– zwykle stan $|00\ldots00\rangle$. Jeżeli chcemy, aby stan początkowy był innym stanem bazowym, zawsze możemy do tego doprowadzić wykonując operacje na pojedyńczych qubitach. Operacje takie nie wprowadzają splątania do układu.

Pierwszym krokiem algorytmu jest wykonanie QFT na rejestrze danych. Jeżeli stanem początkowym rejestru danych jest stan $|0\ldots0\rangle$, prowadzi to do uzyskania jednorodnej superpozycji wszystkich stanów bazowych tego rejestru. Pozwala to na wykonanie w drugim kroku operacji U_f na wszystkich stanach bazowych jednocześnie, a tym samym na obliczenie wartości funkcji f na wszystkich argumentach jednocześnie. W ten sposób kwantowa transformata Fouriera pozwala na wykorzystanie równoległości wprowadzanej przez obowiązującą w mechanice kwantowej zasadę superpozycji.

Operacja U_f jest kluczowym elementem algorytmu i do jej fizycznej realizacji konieczne jest wprowadzenie oddziaływania pomiędzy rejestrami komputera kwantowego. Jak się okazuje, prowadzi to do pojawienia się splątania między stanem rejestru danych i stanem rejestru wartości.

Ostatnim krokiem jest ponowne wykonanie QFT na pierwszym rejestrze, po czym następuje pomiar stanu całego komputera kwantowego.

Pierwsze wykonanie QFT ma na celu uzyskanie superpozycji wszystkich stanów bazowych pierwszego rejestru. Pomijając czynnik normalizacyjny możemy zapisać

$$\mathrm{QFT} \otimes \mathbb{I}|0\ldots0\rangle \otimes |0\ldots0\rangle = \left(\sum_{i=0}^{n-1} |i\rangle\right) \otimes |0\ldots0\rangle. \tag{2.8}$$

Stan ten nie jest stanem splątanym. Wprowadzając w przestrzeni jednego qubitu bazę

$$\{|0\rangle + |1\rangle, |0\rangle - |1\rangle\}$$

, i korzystając z reprezentacji binarnej, możemy stan 2.8 zapisać jako

$$\Big[(|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle) \otimes \ldots \otimes (|0\rangle + |1\rangle)\Big] \otimes |0\rangle. \tag{2.9}$$

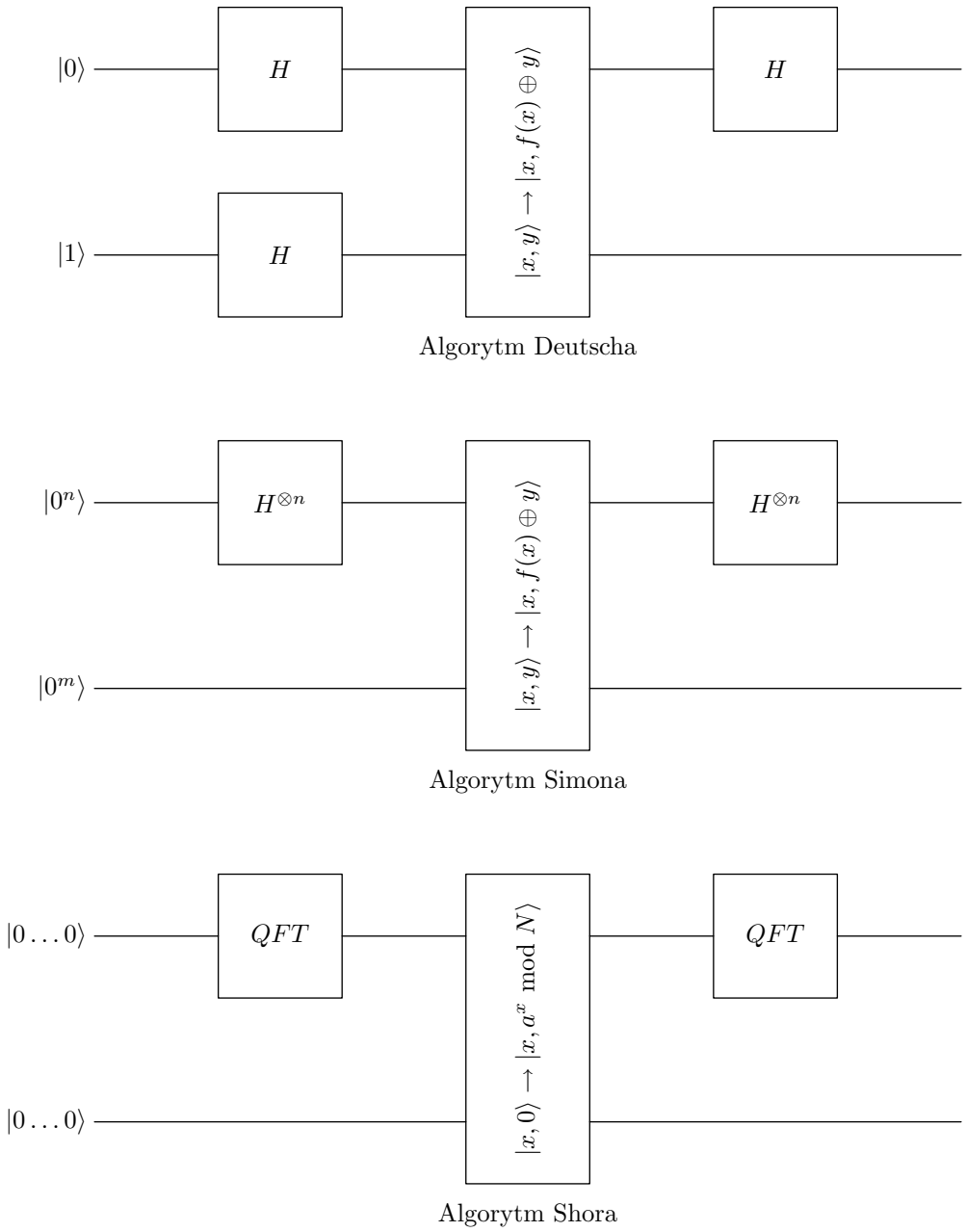
Stan ten jest stanem stanem iloczynowym, a zatem jest stanem separowalnym.

W Rozdziale 4 przekonamy się, że dopiero wykonanie operacji wymagającej oddziaływana pomiędzy podukładami może prowadzić do pojawienia się splątania. Operacje take nie mogą być przedstawione jako iloczyn tensorowy operacji na podukładach[5].

Schematy trzech algorytmów kwantowych ukrytej podgrupy są przedstawione na Rysunku 2.1.

Zgodnie z tym schematem, pomiaru splątania można dokonać jedynie po każdym z kroków algorytmu. W rzeczywistości każdy z kroków w algorytmie musi być rozłożony na szereg operacji elementarnych, zatem można by dokonać pomiaru zmian splątania po każdej operacji elementarnej. Za uproszczeniem wprowadzonym na schemacie 2.1 przemawiają dwa aspekty:

[5]Dokładną charakterystykę bramek wprowadzających splątanie można znaleźć w [7]



Rysunek 2.1: Struktura algorytmów ukrytej podgrupy

1. Jeżeli zaobserwujemy zmianę stopnia splątania po jakimś kroku, to interpretacja występowania splątania i jego wpływu na wykonanie całego algorytmu byłaby niemożliwa, jeżeli skupimy się na operacjach elementarnych.

2. Jeżeli po jakimś kroku algorytmu nie obserwujemy zmiany stopnia splątania stanów podukładów, to możliwa jest taka konstrukcja całego fragmentu algorytmu, która nie wymaga bramek wprowadzających splątanie.

Pierwszy argument jest uzasadniony zależnością rozkładu elementów algorytmu od zupełnego układu bramek który wybierzemy. Dodatkowo każdy element algorytmu może wymagać dużej liczby tych elementarnych operacji.
Skądinąd wiadmono, że bramki nie wprowadzające splątania[6] są albo postaci iloczynu tensorowego bramek działających lokalnie, albo są równoważne bramce SWAP. To uzasadnia drugi argument.

2.3 Złożoność obliczeniowa

Algorytmy ukrytej podgrupy mają złożoność wielomianową dzięki możliwości zaimplementowania szybkiej transformaty Fouriera na komputerze kwantowym. Dokonajmy oszacowania złożoności tej operacji [53]. Poniższe wyprowadzenie daje jednocześnie przepis na konstrukcję układu bramek kwantowych implementującego QFT.

Pierwszym krokiem jest wykonanie kilku prostych przekształceń. Niech j będzie liczbą n-bitową, reprezentowaną przez stan bazowy $|j\rangle \in \mathcal{H}$. W postaci dwójkowej liczbę tę można zapisać jako

$$j = \sum_{k=1}^{n} j_k 2^{n-k}, \tag{2.10}$$

gdzie $j_n \in \{0, 1\}$ oznacza n-ty bit liczby j.

Wychodząc ze wzoru 2.6 możemy zapisać

$$\begin{aligned}
\mathrm{QFT}|j\rangle &= \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{\frac{2i\pi jk}{2^n}} |k\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} e^{\frac{2i\pi j(\sum_{l=1}^{n} k_l 2^l)}{2^n}} |k_1 \ldots k_n\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_{k_1=0}^{1} \cdots \sum_{k_n=0}^{1} \bigotimes_{l=1}^{n} e^{\frac{2i\pi jk_l 2^l}{2^n}} |k_1 \ldots k_n\rangle.
\end{aligned}$$

[6] Bramki takie nazywane są w [9] bramkami prymitywnymi (ang. *primitive*).

Otrzymany stan nie jest stanem splątanym. Można go zapisać w postaci iloczynu tensorowego odpowiednich superpozycji stanów bazowych.

$$\begin{aligned} \mathrm{QFT}|j\rangle &= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left[\sum_{k_l=0}^{1} e^{2i\pi j k_l 2^{-l}} |k_l\rangle \right] \\ &= \frac{1}{\sqrt{2^n}} \bigotimes_{l=1}^{n} \left[|0\rangle + e^{2i\pi j 2^{-l}} |1\rangle \right] \end{aligned}$$

Oznaczając przez $0.j_l \dots j_m$ ułamek binarny o wartości

$$\sum_{k=l}^{m} \frac{j_k}{2^{k-l+1}} \tag{2.11}$$

ostatecznie zapisujemy stan $QFT|j\rangle$ jako

$$\frac{(|0\rangle + e^{2i\pi 0.j_n}|1\rangle)(|0\rangle + e^{2i\pi 0.j_{n-1}j_n}|1\rangle) \dots (|0\rangle + e^{2i\pi 0.j_1 \dots j_n}|1\rangle)}{\sqrt{2^n}} \tag{2.12}$$

Dla przypadku trzech qubitów i $j = 4j_1 + 2j_2 + j_3$ można ostatni krok powyższego wyprowadzenia rozpisać bezpośrednio

$$\begin{aligned} QFT|j\rangle &= \frac{(|0\rangle + e^{2i\pi j/2}|1\rangle)(|0\rangle + e^{2i\pi j/4}|1\rangle)(|0\rangle + e^{2i\pi j/8}|1\rangle)}{\sqrt{8}} \\ &= \frac{(|0\rangle + e^{2i\pi(2j_1+j_2+j_3/2)}|1\rangle)(|0\rangle + e^{2i\pi(j_1+j_2/2+j_3/4)}|1\rangle)(|0\rangle + e^{2i\pi(j_1/2+j_2/4+j_3/8)}|1\rangle)}{\sqrt{8}} \\ &= \frac{(|0\rangle + e^{2i\pi 0.j_3}|1\rangle)(|0\rangle + e^{2i\pi 0.j_2 j_3}|1\rangle)(|0\rangle + e^{2i\pi 0.j_1 j_2 j_3}|1\rangle)}{\sqrt{8}} \end{aligned}$$

gdzie wykorzystana została okresowość funkcji e^x.

Na Rysunku 2.2 przedstawiony jest układ realizujący kwantową transformatę Fouriera dla układu trzech qubitów.

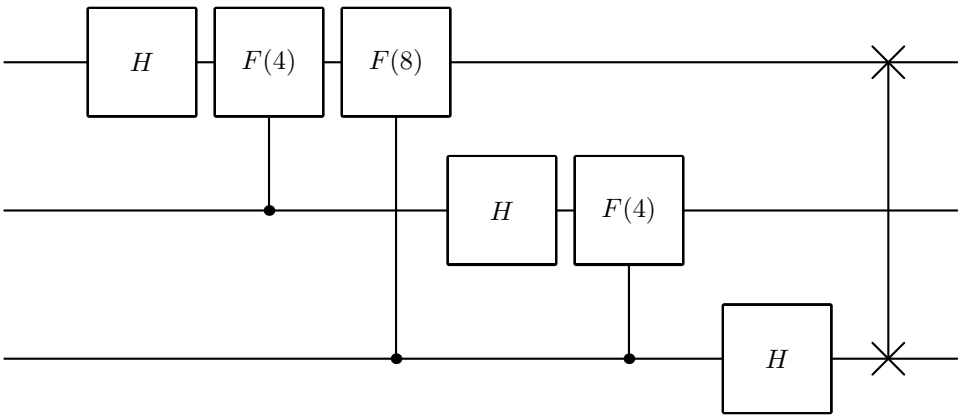
Bramka $F(n)$ jest zdefiniowana dla $n \in \mathbb{N}$ jako operacja

$$F(n) = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{2i\pi}{n}} \end{pmatrix}. \tag{2.13}$$

W tym wypadku mamy

$$F(4) = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, F(8) = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{\pi i}{4}} \end{pmatrix}. \tag{2.14}$$

Ponieważ $e^{2i\pi} = -1$, bramka Hadamarda odpowiada przy takim oznaczeniu bramce $F(2)$.



Rysunek 2.2: Transformata Fouriera dla trzech qubitów.

Ostatnia operacja oznacza bramkę SWAP, powodującą zamianę qubitów. Bramka taka może być zrealizowana za pomocą trzech bramek $CNOT$.

W Rozdziale 7.2 zamieszczony jest program w języku QCL realizujący procedurę QFT.

Do wykonania operacji QFT zgodnie z podanym przepisem trzeba wykonać n-krotnie bramkę Hadamarda oraz dla każdego z qubitów odpowiednią liczbę kontrolowanych bramek $F(x)$. Dla n-tego qubitu wykonujemy bramkę $C - F(\ldots)$, $n-1$-krotnie, dla $n-1$ qubitu – $n-2$ krotnie, a dla drugiego – jednokrotnie. Zamiana kolejności qubitów wymaga $\frac{n}{2}$ bramek SWAP. Łącznie, musimy wykonać $\frac{(n+1)n}{2}$ bramek, co oznacza, iż algorytm QFT ma złożoność $O(n^2)$, gdzie n jest liczbą qubitów.

Warto zauważyć, że wykonywana przez nas operacja jest, ze względu na skończoną pamięć, przybliżeniem transformaty Fouriera. Jest to sytuacja identyczna jak w przypadku komputerów klasycznych, gdyż komputer kwantowy, operuje jedynie na skończonej pamięci (liczbie qubitów). To przybliżenie jest wystarczające do przeprowadzenia obliczeń [5].

Operowanie na skończonych ciągach symboli jest jednym z warunków „rozsądności" modelu obliczeń. Ponieważ stany układu kwantowego i operacje unitarne na nich wykonywane są zadawane przez kombinacje liniowe wektorów oraz macierze nad ciałem $\mathbb{C}$, aby mówić o skończonych obliczeniach musimy się ograniczyć do przybliżonego przeprowadzania ewolucji. Ograniczamy także możliwe amplitudy, przy czym okazuje się, że zestaw amplitud wystarczający do przeprowadzenia obliczeń kwantowych można bardzo ograniczyć [5]. Otrzymujemy wówczas model równoważny pod względem mocy obliczeniowe i bardziej realistyczny w potencjalnych implementacjach.

Rozdział 3

Znajdowanie ukrytej podgrupy

Computer programing is an art form,
like the creation of poetry or music.
– Donald E. Knuth

Rozdział ten poświęcony jest omówieniu grupy algorytmów kwantowych, nazywanych algorytmami ukrytej podgrupy. Znając konstrukcję i działanie kwantowej transformaty Fouriera możemy przekonać się o jej użyteczności do rozwiązywania szerokiej klasy problemów obliczeniowych. Omawiane algorytmy wykorzystują znane własności transformaty Fouriera pozwalające na odtworzenie informacji na temat okresu funkcji. Najbardziej znanym z nich jest algorytm rozwiązujący problem faktoryzacji liczb całkowitych.

3.1 Rodzaje algorytmów kwantowych

Jedną z najważniejszych cech klasycznych maszyn Turinga jest ich uniwersalność. Oznacza ona, że maszyny klasyczne są zdolne do wykonania dowolnego algorytmu, jeżeli tylko udostępni im się program wykonania tego algorytmu. Maszyny kwantowe są uogólnieniem maszyn odwracalnych[1], i jako takie mogą rozwiązać dowolny problem, który może być rozwiązany przez klasyczną maszynę Turinga. Jednakże wykorzystywanie maszyn kwantowych do rozwiązywania dowolnych problemów jest niepraktyczne. Algorytmy kwantowe nie są równoznaczne z przepisanymi w formie równoległej klasycznymi algorytmami odwracalnymi. Są one wynikiem innego sposobu podejścia do problemu i dzięki temu w niektórych przypadkach dają niespodziewane rezultaty. Kosztem, jaki za to musimy zapłacić są trudności związane z realizacją fizyczną[2].

[1] Jeżeli nie istnieje maszyna Turinga rozwiązująca dany problem, to tym samym nie istnieje algorytm rozwiązujący ten problem. Zatem maszyny Turinga można utożsamić z algorytmami.

[2] Dotychczasowe realizacje obliczeń kwantowych ograniczają się do kilku qubitów.

Dotychczas opracowano dwie ogólne metody konstrukcji algorytmów kwantowych dla pewnych klas problemów[3]. Niektóre z nich pozwalają na wykorzystanie efektów kwantowych do uzyskania wykładniczego przyśpieszenia przy rozwiązywaniu niektórych problemów. Wszystkie opracowane algorytmy kwantowe można podzielić na dwie grupy

1. Algorytmy ukrytej podgrupy wykorzystujące kwantową transformatę Fouriera.

   Do tej klasy należą algorytm Deutscha, algorytm Simona oraz algorytmy Shora i Kitaeva [40]. Te dwa ostatnie dotyczą funkcji mających strukturę ukrytej podgupy. Wszystkie pozwalają na wykładnicze – w stosunku do znanych algorytmów klasycznych – zmniejszenie złożoności problemów.

2. Algorytmy wyszukiwania oparte na metodzie Grovera.

   Są to algorytmy rozwiązujące problem przeszukiwania bazy danych N nieuporządkowanych elementów. Algorytm Grovera pozwala na zmniejszenie czasu potrzebnego na znalezienie zaznaczonego elementu z $O(N)$ do $O(\sqrt{N})$. Oryginalna metoda Grovera [28, 29, 36] pozwala także na rozwiązywanie innych zagadnień.

Dodatkowo komputery kwantowe mogą służyć do symulacji złożonych układów fizycznych. Brak możliwości dokonywania takich symulacji na komputerach kwantowych był jednym z bodźców rozwoju informatyki kwantowej. Istnieje więc możliwość, że wraz z rozwojem informatyki kwantowej pojawią się nowe metody symulacji zjawisk fizycznych.

Do algorytmów ukrytej podgrupy zaliczają się wszystkie znane algorytmy kwantowe dające wykładnicze przyśpieszenie w stosunku do algorytmów klasycznych. Przyśpieszenie to sprawia, że algorytmy te są obiektem szczególnego zainteresowania jako potencjalny sposób na:

1. Rozwiązywanie problemów uważanych klasycznie za trudne czyli wymagające wykładniczego w funkcji rozmiaru problemu czasu działania klasycznej maszyny Turinga.

2. Stworzenie algorytmów klasycznych wzorowanych na algorytmach kwantowych.

Do uzyskania pierwszego celu konieczny jest jednak odpowiednio rozbudowany komputer kwantowy, co jest samo w sobie zadaniem trudnym. Natomiast zastanawiając się na drugim sposobem wykorzystania algorytmów kwantowych stajemy przed pytaniem: *Czy możliwe jest dokonanie efektywnej symulacji układu kwantowego na maszynie klasycznej?*

[3]Problem który może być rozwiązany przy pomocy klasycznej maszyny Turinga automatycznie może być rozwiązany przy pomocy maszyny kwantowej, jeżeli skonstruujemy obwód odwracalny dla danego problemu. To zaś jest zawsze możliwe.

3.2 Sformułowanie problemu

Nazwanie grupy algorytmów algorytmami ukrytej podgrupy (ang. *hidden subgroup algorithms*) wynika z faktu, iż można je w prosty sposób uogólnić wykorzystując język teorii grup. Tutaj ograniczymy się do grup abelowych, chociaż możliwe jest uogólnienie na dowolne grupy.

Ścisłe matematycznie sformułowanie problemu ukrytej podgrupy można znaleźć w sekcji 6.5 Dodatku matematycznego.

Odpowiednikiem pojęcia bazy w algebrze liniowej jest w teorii grup pojęcie zbioru generatorów.

Definicja 19 *Zbiór elementów $\{g_1, \ldots, g_n\}$ grupy G nazywamy zbiorem generatorów grupy G jeżeli każdy element $a \in G$ można zapisać jako złożenie tych elementów.*

Problem ukrytej podgrupy można przedstawić w następujący sposób

Załóżmy, że dana jest skończona grupa abelowa G oraz funkcja $f\colon G \to X$, gdzie X jest pewnym skończonym zbiorem. O funkcji f zakładamy, że istnieje nietrywialna podgrupa H grupy G, taka, że f jest różnowartościowa na grupie ilorazowej G/H. Na każdej warstwie podgrupy H funkcja ta jest stała.

Naszym zadaniem jest znalezienie generatorów podgrupy H.

O funkcji spełniającej powyższy warunek mówimy, że spełnia założenie Simona lub, że ma strukturę ukrytej podgrupy.

Warto zauważyć, że dla $h \in H$ i dowolnego $g \in G$ elementy $g+h$ i g należą do tej samej warstwy. Wynika stąd, że funkcja f spełnia warunek $f(g+h) = f(g)$ czyli jest $H-$periodyczna.

Wszystkie kolejno omawiane algorytmy będą przedstawiane w ramach tego ogólnego sformułowania zagadnienia. Na zakończenie tego rozdziału podane są przykłady problemów, które można rozwiązać poprzez zredukowanie do problemu znajdowania ukrytej podgrupy.

3.3 Algorytm Deutscha

Pierwszy algorytm kwantowy, potwierdzający możliwość wykonywania pewnych zadań szybciej na komputerze kwantowym, został opisany przez Davida Deutscha w pracy [17]. Tutaj przedstawiona jest nieco zmodyfikowana wersja [51]. Problem, który algorytm ten ma rozwiązać, jest następujący.

Mając zadaną funkcje $f\colon \mathbb{B} \to \mathbb{B}$ określić czy jest ona stała.

Tutaj i w dalszych częściach pracy oznaczamy przez $\mathbb{B}$ zbiór $\{0,1\}$ i odpowiednio przez $\mathbb{B}^n$ zbiór $\{0,1\} \times \{0,1\} \times \ldots \times \{0,1\}$.

W tym wypadku funkcja jest określona na grupie $(\mathbb{B}, \oplus,)$ i przyjmuje wartości w $\mathbb{B}$. Natomiast ukrytą podgrupą jest $\{0\}$ lub $\{0,1\}$

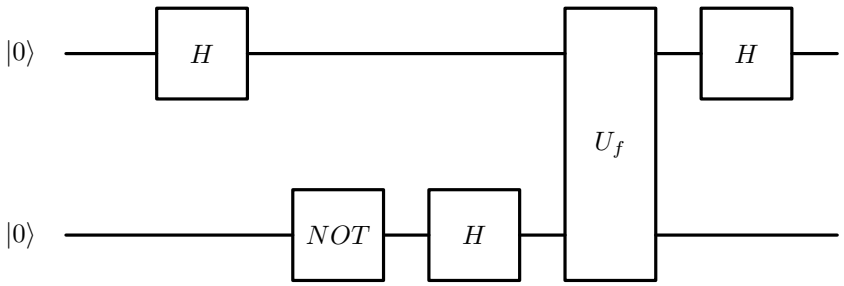
Funkcję $f\colon \mathbb{B} \to \mathbb{B}$ nie będącą funkcją stałą nazywamy zbalansowaną. Ponieważ są dokładnie 4 funkcje $f\colon \mathbb{B} \to \mathbb{B}$

$$\begin{aligned} f_1 &= \{(0,0),(1,0)\} \\ f_2 &= \{(0,1),(1,1)\} \\ f_3 &= \{(0,1),(1,0)\} \\ f_4 &= \{(0,0),(1,1)\} \end{aligned}$$

możemy szczegółowo prześledzić proces obliczeniowy. Funkcje f_1 i f_2 są stałe i odpowiada im ukryta podgrupa $\{0,1\}$, natomiast funkcje f_3 i f_4 są zbalansowane i odpowiada im ukryta podgrupa $\{0\}$.

Dokładny opis kolejnych kroków algorytmu Deutscha, pozwalający na prześledzenie zmiany stanu układu w tracie wykonania algorytmu, znajduje się w podrozdziale 4.3.

Na Rysunku 3.1 przedstawiony jest obwód kwantowy wykonujący ten algorytm.



Rysunek 3.1: Obwód kwantowy dla algorytmu Deutscha

Uogólnieniem algorytmu Deutscha na funkcje $f\colon \mathbb{B}^m \to \mathbb{B}$ jest algorytm Deutscha-Jozsy. Klasyczne rozwiązanie problemu Deutscha dla takiej funkcji wymaga obliczenia wartości funkcji na $n/2+1$ argumentach. Wykorzystując komputer kwantowy mamy możliwość jednoczesnego obliczenia wartości funkcji dla wszystkich argumentów.

W ogólnym przypadku funkcji $f\colon \mathbb{B}^m \to \mathbb{B}$ algorytm Deutsch-Jozsy pozwala na określenie globalnej własności funkcji, która może być bardzo nieregularna. O zadanej funkcji wiemy tylko tyle, że spełnia ona jeden z wzajemnie wykluczających się warunków:

- f jest stała na $\mathbb{B}^m$
- lub f jest zbalansowana na $\mathbb{B}^m$

Nie narzuca to ograniczenia na rozmieszczenie wartości, a jedynie na ich ilość. Algorytm Deutscha-Jozsy nie jest zatem algorytmem ukrytej podgrupy.

Kolejne kroki algorytmu Deutscha-Jozsy są analogiczne jak w przypadku algorytmu Deutscha [53, 51] – zmodyfikowana jest jedynie liczba qubitów potrzebnych na przechowania danych.

3.3.1 Modyfikacja jednoqubitowa

Okazuje się że do rozwiązania problemy Deutscha wystarczy wykorzystać jeden qubit [15]. Rozwiązanie to pozwala na całkowite wyeliminowanie splątania podczas wykonania algorytmu, gdyż mamy wówczas tylko jeden podukład. Uzyskujemy to poprzez inny sposób implementacji funkcji f w postaci bramki kwantowej.

Dla algorytmu Deutscha-Jozsy algorytm dla funkcji o wartościach w $\mathbb{B}^n$ przy $n < 3$ nie wymaga splątania. Natomiast dla $c \geq 3$ splątanie może wystąpić w zależności od tego z jaką funkcją mamy do czynienia.

3.4 Algorytm Simona

Przedstawiony w pracy [67] algorytm Simona jest punktem wyjścia dla uogólnienia metody znajdywania ukrytej podgrupy na komputerze kwantowym.

Algorytm Simona jest uogólnieniem algorytmu Deutscha na funkcje o wartościach w $B^m, m > 1$. Oryginalne sformułowanie problemy, który ma rozwiązywać algorytm można łatwo przepisać, tak by algorytm obejmował szerszą klasę problemów.

3.4.1 Przebieg algorytmu

Użyjemy m qubitów do zakodowania elementów $G = \mathbb{B}^m$, grupy addytywnej przestrzeni wektorowej m-wymiarowej nad $\mathbb{B}$. Ponieważ funkcja spełniająca założenie Simona ma[4] $[G : H] = \frac{|G|}{|H|}$ różnych wartości, a w n qubitach możemy zakodować 2^n wartości, potrzebujemy także $\log_2 \frac{|G|}{|H|} = m - \log_2 |H|$ qubitów do zakodowania wartości funkcji $f \colon G \to R$.

Dla $H < G$ przez $H^\perp$ oznaczamy zbiór

$$H^\perp := \left\{ y \in G | \bigwedge_{h \in H} y \cdot h = 0 \right\} \tag{3.1}$$

Symbol $\cdot$ oznacza tutaj iloczyn skalarny[5] $\cdot \colon G \times G \to \mathbb{R}$ określony wzorem

$$x \cdot y = \sum_{k=1}^{n} x_k y_k \ (\mathrm{mod}\ 2) \tag{3.2}$$

gdzie $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n)$.

[4] Równość ta jest treścią twierdzenia Lagrange'a.

[5] Grupa G jest utożsamiana z przestrzenią liniową G.

Schemat przedstawiony poniżej jest słuszny, jeżeli $\mathbb{B}$ zastąpimy dowolnym ciałem skończonym.

1. Wykonaj transformatę Hadamarda nad $\mathbb{B}^m$ aby otrzymać

$$\frac{1}{\sqrt{2^m}} \sum_{x \in \mathbb{B}^m} |x\rangle|0\rangle \tag{3.3}$$

2. Oblicz wartość funkcji f

$$\frac{1}{\sqrt{2^m}} \sum_{x \in \mathbb{B}^m} |x\rangle|f(x)\rangle = \frac{1}{\sqrt{2^m}} \sum_{x \in H} \sum_{t \in T} |x+t\rangle|f(t)\rangle, \tag{3.4}$$

   gdzie T jest zbiorem zawierającym po jednym reprezentancie z każdej warstwy podgrupy H

3. Wykonaj po raz drugi transformatę Hadamarda nad $\mathbb{B}^m$. Ponieważ dla $y \in \mathbb{B}$, $H < \mathbb{B}$

$$\sum_{h \in H} (-1) = \begin{cases} |H| & y \in H^\perp \\ 0 & y \notin H^\perp \end{cases} \tag{3.5}$$

   otrzymujemy

$$\begin{aligned} QFT \otimes \mathbb{I} \frac{1}{\sqrt{2^m}} \sum_{x \in H} \sum_{t \in T} |x+t\rangle|f(t)\rangle &= \frac{1}{2^m} \sum_{x \in H} \sum_{t \in T} \sum_{y \in G} (-1)^{(x+t)\cdot y} |y\rangle|f(t)\rangle \\ &= \frac{|H|}{2^m} \sum_{t \in T} \sum_{y \in H^\perp} (-1)^{t \cdot y} |y\rangle|f(x)\rangle \end{aligned}$$

4. Ostatnim krokiem algorytmu jest obserwacja stanu układu.

Przy okazji omawiania kolejnych kroków algorytmu Simona przekonamy się, że prawdopodobieństwo otrzymania jakiegokolwiek elementu $y \in H^\perp$ wynosi

$$\mathrm{P}(y) = \frac{1}{|H|}. \tag{3.6}$$

3.4.2 Maska niezmienności względem sumy modulo 2

W oryginalnym sformułowaniu [67] problem Simona dotyczy znajdowania maski niezmienności względem sumy modulo 2 (ang. *xor mask invariance – XMI(f)*), zdefiniowanej w nastepujący sposób:

Definicja 20 *Maską niezmienniczości względem sumy modulo* 2 *dla funkcji* $f\colon \mathbb{B}^n \to \mathbb{B}^m$ *nazywamy słowo* n*-bitowe* s *spełniające warunek: dla dowolnych danych wejściowych* $x, y \in \mathbb{B}^n$ *równość* $f(x) = f(y)$ *jest spełniona tylko wówczas, gdy* $x \oplus s = y$.

Dla problemu znalezienia XMI dla pewnej funkcji $f\colon \mathbb{B}^n \to \mathbb{B}^m, n \leq m$ zachodzi następujące twierdzenie [67]:

Twierdzenie 10 *Istnieje algorytm dla QTM rozwiązujący problem Simona o złożoności $O(nT_f(n) + G(n))$, gdzie $T_f(n)$ to czas potrzebny na obliczenie funkcji na słowie o długości n, a $G(n)$ to czas potrzebny na rozwiązanie układu równań $n \times n$ nad ciałem $\mathbb{Z}_2$*

Dowód.
Algorytm składa się z dwóch części

[A] Procedury obejmującej dwukrotne wykonanie transformaty Fouriera

1. Stanem początkowym jest stan $|0 \ldots 0\rangle \in \mathcal{H}^{2^n}$. Za pomocą bramki $F = H^{\otimes n} \otimes \mathbb{I}^{\otimes n}$ przygotowujemy stan $\frac{1}{\sqrt{2^n}} \sum_{x \in \mathbb{B}^n} |x\rangle|0 \ldots 0\rangle$
2. Obliczamy wartość funkcji f poprzez wykonanie transformacji U_f. W wyniku otrzymujemy stan $\frac{1}{\sqrt{2^n}} \sum_{x \in \mathbb{B}^n} |x\rangle|f(x)\rangle$
3. Wykonujemy po raz drugi transformację F i otrzymujemy w ten sposób stan $\frac{1}{2^n} \sum_{x \in \mathbb{B}^n} \sum_{y \in \mathbb{B}^n} (-1)^{x \cdot y} |y\rangle|0 \ldots 0\rangle$

[B] Rozwiązania układu równań $n \times n$ nad $\mathbb{Z}_2$ na komputerze klasycznym.

Ta część algorytmu nie wymaga maszyny kwantowej. Wykonania $O(n)$ razy procedury [A] pozwala na uzyskanie układu równań nad $\mathbb{Z}_2$

$$\begin{aligned} s_1 \cdot y &= 0 \bmod 2 \\ s_2 \cdot y &= 0 \bmod 2 \\ \ldots \quad &\ldots \quad \ldots \\ s_n \cdot y &= 0 \bmod 2 \end{aligned}$$

Istnieje algorytm oparty na Chińskim twierdzeniu o resztach[6], pozwalający na efektywne rozwiązanie tego problemu na klasycznej maszynie Turinga.

□

Ten szczególny przypadek algorytmu Simona będzie rozpatrywany w Rozdziale 4.

3.5 Algorytmy Shora

Peter Shor w pracach [65] oraz [66] podał algorytm pozwalający na przeprowadzenie z wykorzystaniem komputera kwantowego rozkładu liczby całkowitej

[6] Patrz: sekcja 6.3 Dodatku matematycznego

na czynniki pierwsze. W porównaniu z dotychczas znanymi algorytmami klasycznymi, algorytm Shora odznacza się złożonością wielomianową względem rozmiaru danych wejściowych. W tej samej pracy podany został także algorytm obliczania logarytmów dyskretnych.

Algorytmy faktoryzacji i obliczania logarytmów dyskretnych podane przez Shora stanowią uogólnienie metody wprowadzonej przez Simona.

3.5.1 Znajdowanie rzędu w grupie

Zasadniczą częścią algorytmu faktoryzacji jest, wykonywana na komputerze kwantowym, procedura znajdowania rzędu. Tylko ta część musi być wykonana na maszynie kwantowej – pozostałe części mogą być z powodzeniem wykonane na maszynie klasycznej. W kolejnych paragrafach przedstawię procedurę znajdowania rzędu oraz jej wykorzystanie w problemach faktoryzacji i obliczania logarytmów dyskretnych.

Definicja 21 *Rzędem elementu a w grupie $(\mathbb{Z}_N, \cdot)$ nazywamy najmniejszą liczbę całkowitą r spoełniającą warunek*

$$a^r = 1 \bmod N$$

Liczbę r oznaczamy pisząc $r = \mathrm{ord}_N(a)$.

Stosując zapis symboliczny

$$r = \mathrm{ord}_N(a) = \min\{x \in \mathbb{Z} | m^x = 1 \bmod N\}$$

Jeżeli liczba a jest względnie pierwsza z N to $\mathrm{ord}_N(a)$ istnieje. W przeciwnym wypadku $\gcd(a, N) \neq 1$ więc $a^m - bN$ jest podzielne przez $\gcd(a, N)$ dla dowolnych m i b, gdzie $\gcd(a, b)$ oznacza największy wspólny dzielnik[7] liczb a i b.

Określmy funkcję $f_a(x) := a^x (\bmod N)$, przy założeniu $\gcd(a, N) = 1$. Zgodnie z definicją rzędu

$$f_a(x + r) = a^{x+r} (\bmod N) = a^x a^r (\bmod N) = a^x (\bmod N) = f_a(x),$$

co oznacza, że $r = \mathrm{ord}_N(a)$ jest okresem funkcji f_a. Pozwala to wykorzystać własności transformaty Fouriera do znalezienia liczby r.

Naszym zadaniem jest rozwiązanie następującego problemu

Mając dane liczby N oraz a, takie, że $\gcd(a, N) = 1$, znajdź okres funkcji $f_a : \mathbb{Z} \to \mathbb{Z}_N$ określonej wzorem $f_a(x) = a^x (\bmod N)$

[7] W polskiej literaturze przyjęte jest oznaczenie $\mathrm{nwd}(a, b)$

Problem ten jest bardzo podobny do problemu Simona – jednakże w tym wypadku rozpatrujemy funkcję na zbiorze nieskończonym, podczas gdy w problemie Simona jest mowa o zbiorze skończonym.

Pierwsza przeszkoda jaka się pojawia wynika z ograniczenia reprezentacji liczb. Nie możemy w pamięci komputera kwantowego umieścić wszystkich liczb całkowitych. Musimy zatem rozpatrywać tylko obcięcie funkcji f_a do pewnego skończonego podzbioru $\mathbb{Z}_q \subset \mathbb{Z}$ i na nim badać okresowość funkcji $f_a|_{\mathbb{Z}_q}$.

Przestrzeń Hilberta komputera kwantowego, na którym ma się odbywać algorytm znajdowania okresu, można przedstawić w postaci $\mathcal{H}_1 \otimes \mathcal{H}_2$, przy czym pierwsza podprzestrzeń odpowiada rejestrowi, w którym przechowywane są argumenty funkcji f_a, a druga – rejestrowi zawierającemu wartości. Rejestry te – a co za tym idzie wymiary $q = \dim \mathcal{H}_1$ oraz $d = \dim \mathcal{H}_1$ podprzestrzeni – muszą być dobrane tak aby pomieściły odpowiednio duże liczby. Dobór odpowiednich wymiarów przestrzeni wynika z oszacowania szans powodzenia algorytmu.

W Rozdziale 4 zaprezentowane są obliczenia przeprowadzone dla kilku liczb. Ponieważ złożoność nawet prostej symulacji jest w przypadku algorytmu Shora bardzo duża i czas jej wykonania wzrasta wykładniczo, możliwe jest przeprowadzenie jedynie najprostszych symulacji.

3.5.2 Szybka faktoryzacja

Problem (efektywnego) znajdowania dzielników liczby N można rozwiązać jeżeli potrafimy (efektywnie) obliczać rząd elementu w multiplikatywnej grupie $\mathbb{Z}_N$. Mówiąc precyzyjnie, zastępujemy zagadnienie znalezienia dzielników liczby N, zagadnieniem następującym

Mając daną liczbę N oraz liczbę a, względnie pierwszą z N, znajdź rząd $\mathrm{ord}_N(a)$

Załóżmy, że znamy efektywny sposób rozwiązania tego problemu.

Pierwszą czynnością jaką musimy wykonać jest wybranie losowej liczby a takiej, że $\gcd(a, N) = 1$. Liczba a musi być względnie pierwsza z N aby istniał rząd $\mathrm{ord}_N(a)$.

Okres funkcji $f(x) = a^x (\bmod N)$ jest rzędem elementu a.

$$f(x+r) = a^{x+r} (\bmod N) = a^x a^r (\bmod N) = a^x (\bmod N) = f(x) \qquad (3.7)$$

Ponieważ nie jest możliwa reprezentacja całej dziedziny całkowitej w pamięci (stanach) komputera kwantowego, wybieramy pewną poddzedzinę Z_q funkcji f, z q będącą potęgą dwójki spełniającą warunek $N < q < 2N$.

Algorytm znajdowania rzędu jest zatem algorytmem znajdowania okresu funkcji f. Wymaga on dwóch rejestrów kwantowych i przebiega w następujący sposób:

1. Ustaw w superpozycji wszystkie elementy $\mathbb{Z}_q$ wykonując transformatę Fouriera
$$\frac{1}{\sqrt{q}} \sum_{k=0}^{q-1} |x\rangle|0\rangle \tag{3.8}$$

2. Umieść wartość funkcji $f(x) = a^x (\bmod N)$ w drugim rejestrze
$$\frac{1}{\sqrt{q}} \sum_{k=0}^{q-1} |x\rangle|f(k)\rangle \tag{3.9}$$

3. Wykonaj drugi raz transformatę Fouriera
$$\frac{1}{\sqrt{q}} \sum_{k=0}^{q-1} \sum_{l=0}^{q-1} \exp\left(\frac{2i\pi kl}{q}\right) |l\rangle|f(k)\rangle. \tag{3.10}$$

Analiza rozkładu prawdopodobieństwa otrzymanego w wyniku przeprowadzenia procedury znajdowania rzędu przedstawiona jest w Rozdziale 4.6.

Otrzymane w wyniku pomiarów liczby mogą zostać użyte do uzyskania za pomocą rozwinięć ułamków ciągłych rzędu liczby a. Natomiast znajomość rzędu a pozwala na znalezienie nietrywialnych dzielników N (patrz: Dodatek matematyczny, podrozdział 6.3).

3.5.3 Obliczanie logarytmów dyskretnych

Do obliczania logarytmów dyskretnych na komputerze kwantowym [65] konieczne jest wykorzystanie trzech rejestrów kwantowych. W tym wypadku problem jest postawiony w następujący sposób

Mając daną liczbę $x \in \mathbb{Z}_p$ oraz generator g grupy $\mathbb{Z}_p$, znajdź r takie, że $g^r = x (\bmod p)$.

Dla liczby pierwszej p, grupa $\mathbb{Z}_p$ z mnożeniem modulo p jako działaniem grupowym, jest cykliczna czyli istnieje element $g \in \mathbb{Z}_p$ taki, że kolejne potęgi $g^0, g, \ldots, g^{p-1}$ generują $\mathbb{Z}_p$.

Oznaczmy przez x liczbę, której logarytmu r szukamy. Zgodnie z definicją
$$g^r = x \bmod p \tag{3.11}$$
przy czym $0 \leq r < p - 1$.

Problem obliczenia logarytmu dyskretnego można przedstawić jako problem znalezienia ukrytej podgrupy dla odwzorowania $f\colon \mathbb{Z}_p \times \mathbb{Z}_p \to \mathbb{Z}_p$
$$f(a, b) = g^a x^b \tag{3.12}$$

Ponieważ zachodzi równoważność

$$f(a_1,b_1)=f(a_2,b_2)\Leftrightarrow g^{a_1}x^{b_1}=g^{a_2}x^{b_2}\Leftrightarrow g^{a_1-rb_1}=g^{a_2-rb_2}\Leftrightarrow a_1-rb_1=a_2-rb_2 \tag{3.13}$$

funkcja ta jest stała na warstwach $(a_1,b_1)+H=(a_1,b_2)+H$ dla

$$H=\{(k,rk)|k\in\mathbb{Z}_p\}<\mathbb{Z}_p\rightarrow\mathbb{Z}_p. \tag{3.14}$$

Tak zdefiniowana funkcja spełnia założenie Simona.

Algorytm Shora obliczania logarytmu dyskretnego ma następujący przebieg

1. Ustaw pierwsze dwa rejestry w superpozycji

$$\frac{1}{p-1}\sum_{a=0}^{p-1}\sum_{b=0}^{p-1}|a,b,0\rangle \tag{3.15}$$

2. Oblicz $g^a x^b(\bmod p)$ w trzecim rejestrze

$$\frac{1}{p-1}\sum_{a=0}^{p-1}\sum_{b=0}^{p-1}|a,b,g^a x^b(\bmod p)\rangle \tag{3.16}$$

3. Niech q będzie potęgą dwójki taką, że $p<q<2p$. Wykonujemy na dwóch pierwszych rejestrach transformatę QFT_q i otrzymujemy stan

$$\frac{1}{q(p-1)}\sum_{a,b=0}^{p-1}\sum_{c,d=0}^{p-1}\exp\left(\frac{ac+bd}{q}\right)|c,d,g^a x^b(\bmod p)\rangle \tag{3.17}$$

W ostatnim etapie algorytmu następuje pomiar stanu komputera kwantowego. Otrzymany w dwóch pierwszych rejestrach stan reprezentuje pary (a,b) pozwalające na wydobycie informacji o logarytmie dyskretnym liczby x [53].

3.6 Zagadnienie generacji macierzy U_f

Ważnym elementem składowym wszystkich algorytmów kwantowych jest macierz unitarna odpowiadająca wykonaniu funkcji f, której własności badamy. Pierwszym nasuwającym się tu problemem jest istnienie takiej macierzy unitarnej (obwodu) dla dowolnej funkcji $f\colon\mathbb{B}^m\rightarrow\mathbb{B}^n$. W pracy [4] zostało pokazane, iż jest możliwa konstrukcja klasycznego obwodu odwracalnego dla każdej funkcji obliczalnej na klasycznej maszynie Turinga.

Jednym z podstawowych założeń przyjmowanych przy rozważaniach dotyczących konstrukcji obwodów realizujących procedury kwantowe jest możliwość wykonywania jedynie ograniczonego, funkcjonalnie zupełnego, zestawu operacji kwantowych. Wybór takiego zestawu zależy do wielu czynników, z których najważniejszy jest sposób implementacji algorytmu.

Niestety proces generacji dowolnej macierzy unitarnej z zestawu bramek elementarnych jest bardzo skomplikowany i złożoność jest duża. Stawia to pod znakiem zapytania celowość wykorzystania algorytmów kwantowych w sytuacjach, gdy konieczne jest częste generowanie dużej macierzy. Są jednak sytuacje gdy nie ma alternatywy dla takiego podejścia – najlepszym przykładem jest wyszukiwanie rozwiązania dla algorytmu DES [16].

3.7 Inne zastosowania

Algorytmy ukrytej podgrupy mogą znaleźć zastosowanie do szerokiej gamy problemów matematycznych. Jako problem ukrytej podgrupy można sformułować zagadnienia [51, 53]

- Znajdowanie okresu
  Zastosowanie to jest pokazane w szczególności w algorytmie Shora znajdowania rzędu – wówczas szukamy okresu funkcji $f(x) = a^x \bmod N$.

- Ukryta funkcja liniowa
  Dla funkcji $f(x, y) = x + ay$ określonej na $(\mathbb{Z} \times \mathbb{Z}, +)$ o wartościach w $\mathbb{Z}_N$ i permutacji $\pi \colon \mathbb{Z}_N \to \mathbb{Z}_N$ określmy

  $$h = \pi \circ f$$

  W tym wypadku ukryta podgrupa ma posać:

  $$\{(-ta, t) | t = 1, \ldots, N-1\}.$$

- Rząd permutacji
  Dla $G = (\mathbb{Z}_{2n} \times \mathbb{Z}_{2n}, \oplus)$ i funkcji $f \colon G \to \mathbb{Z}_{2n}$ określonej jako

  $$f(x, y) = \pi^x(y)$$

  gdzie π jest permutacją zbioru $\mathbb{Z}_{2n}$. Szukamy ukrytej podgrupy

  $$\{r \in \mathbb{Z}_{2n} | f(x + r, y) = f(x, y)\}.$$

- Stabilizator grupy abelowej
  Niech elementami grupy abelowej G będą odwzorowania zbioru $X \to X$, ze składaniem odwzorowań

  $$\bigwedge_{a,b \in G} \bigwedge_{x \in X} (ab)(x) = a(b(x))$$

  jako działaniem grupowym. Podzbiór $St \subset G$ złożony z elementów G dla których przy ustalonym $x \in X$

  $$g(x) = x$$

tworzy podgrupę. Podgrupę ta nazywamy stabilizatorem elementu x w G i oznaczamy $St_G(x)$.

Oznaczmy przez f_x odwzorowanie przeprowadzające element grupy G w wartość tego elementu na x , $f_x(g) = g(x)$. Odwzorowaniu temu odpowiada ukryta podgrupa $St_G(x)$.

Stabilizator $St_G(x)$ jest ukryta podgrupą dla odwzorowania $f_x \colon G \to X$ określonego jako

$$f_x(g) = g(x)\ g \in G$$

W pracy [40] został podany sposób znalezienia $St_G(x)$ dla przypadku grup albelowych. Zadanie to zawiera jako przypadki szczególne zadania faktoryzacji i obliczania logarytmów dyskretnych.

Rozdział 4

Splątanie w algorytmach

4.1 Uwagi wstępne

Rozdział ten poświęcony jest obliczeniom obrazującym występowanie splątania w trakcie wykonania algorytmów kwantowych rozwiązujących problem ukrytej podgrupy. Przeprowadzone symulacje wykorzystują model obliczeń kwantowych zaproponowany po raz pierwszy w pracy Davida Deutscha [18] – model sieci kwantowych (ang. *quantum networks*). W Rozdziale 5, przy okazji rozważań dotyczących złożoności, wprowadzony zostanie model kwantowej maszyny Turinga. Okazuje się, że oba te modele są równoważne pod względem mocy obliczeniowej [82], a zatem o ich stosowaniu decydować można na podstawie użyteczności w konkretnej sytuacji.

Model sieci kwantowych niewątpliwie lepiej nadaje się do prezentacji obliczeń wykonywanych podczas pracy komputera kwantowego. Tworzenie programów kwantowych polega na połączeniu w ciąg operacji unitarnych, wykonywanych na wektorach stanu. Pozwala on na pisanie programów kwantowych w oderwaniu od konkretnej realizacji fizycznej[1].

Z numerycznego punktu widzenia istota symulowania komputera kwantowego na komputerze klasycznym tkwi w reprezentacji stanów i rozkazów maszyny kwantowej za pomocą unormowanych wektorów i macierzy unitarnych[2]. Prowadzi to – jak przekonamy się w dalszej części tego rozdziału – do małej efektywności takiej symulacji. Można zatem dojść do tezy, postawionej w 1982 r. przez R. P. Feynmanna [22], i będącej jedną z przyczyn zainteresowania kwantową teorią informacji, iż komputery klasyczne nie są zdolne do efektywnego odtworzenia działania układu kwantowego.

W wielu pracach poświęconych kwantowej teorii informacji i obliczeniom kwantowym pojawia się opinia, że do uzyskania przez komputery kwantowe

[1] Zostało to wykorzystane między innymi przy tworzeniu języka QCL opisanego w Rozdziale 7.2 oraz wielu innych symulatorów maszyn kwantowych [74].

[2] Teoretycznie przy badaniu algorytmów kwantowych można ograniczyć się do stanów czystych. Jednak z praktycznego punktu widzenia opis tak uzyskany jest niewystarczającym przybliżeniem rzeczywistości.

wykładniczego obniżenia złożoności niektórych algorytmów konieczne jest występowanie w trakcie procesu obliczeniowego stanów splątanych. W związku z wieloma problemami teoretycznymi i trudnościami praktycznymi pojawia się problem weryfikacji tej opinii.

Najprostszą metodą przekonania się, czy splątanie występuje w trakcie obliczeń kwantowych, jest przeprowadzenie symulacji numerycznej procesu obliczeniowego. Jednak wraz ze wzrostem rozmiaru danych na których operujemy, czas potrzebny na symulację rośnie bardzo szybko. W związku z tym musimy ograniczyć się do najprostszych przypadków.

Z drugiej strony nie istnieje praktyczne kryterium oceny splątania w układach wieloskładnikowych. Konieczne jest zatem wprowadzenie podziału pamięci kwantowej opartego na przedstawionym w Rozdziale 2 schematycznym opisie algorytmów ukrytej podgrupy. Badanie występowania splątania ogranicza się w takim schemacie do zbadania splątania pomiędzy stanami podukładów po każdym z głównych kroków algorytmu.

Pierwszym z badanych algorytmów jest algorytm Deutscha [17]. Jest to pierwszy i najprostszy z algorytmów rozwiązujących problem ukrytej podgrupy. Przedstawione są też przykłady obliczeń dla algorytmu Deutscha-Jozsy. Jednak najciekawszymi algorytmami ukrytej podgrupy są niewątpliwie algorytm Simona i algorytm Shora. Przykłady realizacji algorytmu Simona pozwalają na potwierdzenie występowania stanów splątanych w trakcie realizacji algorytmów kwantowych. Natomiast algorytm Shora jest przykładem mistrzowskiego operowania rozkładem prawdopodobieństwa, pozwalającego na uzyskania porządanego wyniku.

Złożoność asymptotyczna tych algorytmów bardzo wyraźnie przekłada się na moc obliczeniową wykorzystywaną przy wykonywaniu symulacji na komputerze klasycznym [3]. Dotyczy to szczególnie wymagań związanych z pamięcią – stan n-qubitowej maszyny kwantowej wymaga $O(c2^n)$ bajtów, gdzie c jest stałą zależną od reprezentacji amplitudy.

4.2 Bramki kwantowe wprowadzające splątanie

Bramki kwantowe stanowią podstawowe cegiełki, z których budowane są algorytmy kwantowe. Jest to analogia do sytuacji klasycznej, gdzie operuje się na prostych bramkach logicznych takich jak AND czy XOR. Jednakże bramki kwantowe reprezentują operacje unitarne, a nie operacje logiczne, zaś ich kluczową cechą jest możliwość operowania na superpozycji stanów maszyny.

Zwykle od bramek kwantowych wymaga się, by reprezentowały działania lokalne, wykonywane na niewielkiej liczbie qubitów. Nie oznacza to, że uniemożliwiają one przenoszenie się oddziaływaą pomiędzy różnymi qubitami.

[3] Do obliczeń wykorzystywany był komputer z procesorem AMD ATHLON XP 1800+ 1530 MHz dysponujący 256 MB pamięci operacyjnej oraz komputer AMD K6 350 MHz dysponujący 164 MB pamięci operacyjnej

Zostało udowodnione [3], że do realizacji dowolnego algorytmu wystarczy możliwość realizacji wszystkich bramek jednoqubitowych oraz bramki dwuqubitowej $CNOT$. Bramka ta – jak można się łatwo przekonać – wprowadza splątanie pomiędzy stanami podukładów. Okazuje się, że dowolna bramka wprowadzająca splątanie, może być użyta do konstrukcji bramki $CNOT$ [7], i przez to może służyć jako podstawa działania dowolnego algorytmu kwantowego. Fakt ten świadczy o kluczowej roli splątania w procesach obliczeniowych przeprowadzanych na komputerze kwantowym: aby wykonać dowolną operację, wystarczy dysponować możliwością tworzenia stanów splątanych.

Poniższe obliczenia mają na celu porównanie w przypadkach najprostszych bramek kwantowych rezultatów obliczeń stopnia splątania uzyskanych przy pomocy splątania formacji oraz ujemności.

4.2.1 Bramka Hadamarda

Można spodziewać się, że pewne kroki będą kluczowe dla przebiegu algorytmu kwantowego. Dlatego warto dokładnie przyjrzeć się podstawowym elementom, z których budowane są algorytmy.

Bardzo prosto można się przekonać, że wykonanie bramki Hadamarda

$$H = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{4.1}$$

na jednym z rejestrów komputera kwantowego będącego w stanie bazowym $|00\rangle$ nie wprowadza splątania pomiędzy stanami podukładów[4]. Jeżeli operujemy na dwóch qubitach, to odpowiada to następującemu ciągowi operacji

$$\begin{aligned} |\psi\rangle &= H \otimes \mathbb{I}|00\rangle \\ &= H|0\rangle \otimes |0\rangle \\ &= \frac{1}{\sqrt{2}}(|0\rangle \otimes |0\rangle + |1\rangle \otimes |0\rangle) \\ &= \frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) \end{aligned} \tag{4.2}$$

W tym wypadku po dokonaniu pomiaru stanu drugiego rejestru, stan drugiego rejestru jest jednorodną mieszanką stanów $|0\rangle$ i $|1\rangle$.

Niewiele zmieni się jeżeli wykonamy transformatę $H^{\otimes n} = \underbrace{H \otimes H \ldots \otimes H}_{n}$ na stanie $|\underbrace{0 \ldots 0}_{n}\rangle$ – otrzymamy wówczas jednorodną superpozycję stanów bazowych, a co za tym idzie w wyniku pomiaru jednorodne mieszanki statystyczne.

[4] Żadna bramka postaci $A \otimes B$, gdzie A i B to operatory unitarne działające na stanach podukładów, nie wprowadza splątania.

Zawsze możemy wybrać w przestrzeni H bazę Fouriera $\left\{\frac{|0\rangle+|1\rangle}{\sqrt{2}}, \frac{|0\rangle-|1\rangle}{\sqrt{2}}\right\}$, w której stany superpozycji faktoryzują się.

Entropia von Neumana dla macierzy gęstości otrzymanej po wykonaniu operacji śladu częściowego względem dowolnego zespołu $n-1$ qubitów wynosi 0. Jest to również dowodem, że żaden qubit nie jest splątany z pozostałymi.

Bramka Hadamarda jest jedną z operacji najczęściej pojawiających się w schematach algorytmów kwantowych – reprezentuje ona transformatę Fouriera na $\mathbb{Z}_2$. Pozwala ona także na skonstruowanie obwodu kwantowego dla kwantowej transformaty Fouriera na $\mathbb{Z}_q$. Zwykle jej implementacja fizyczna polega na wytworzeniu superpozycji dwóch, wybranych jako reprezentacja qubitu, stanów układu. Jej znaczenie staje się zrozumiałe, gdy dokonamy połączenia jej z bramką $CNOT$, opisującą oddziaływania pomiędzy qubitami.

4.2.2 Bramka $CNOT$

Bramka $CNOT$ odpowiada klasycznej operacji XOR. Dysponując dwuqubitową bramką $CNOT$ oraz wszystkimi bramkami jednoqubitowymi, jesteśmy w stanie wykonać dowolną sekwencję operacji unitarnych. Ponieważ bramka $CNOT$ pozwala operować na stanach splątanych – tj. tworzyć i niszczyć splątanie, konieczność wykonywania jej wydaje się uzasadniona tylko jeżeli podczas wykonania algorytmu kwantowego maszyna kwantowa miałaby znaleźć się w stanach wykazujących splątanie. Konstrukcja zupełnego układu bramek kwantowych, wśród których żadna nie wprowadzałaby splątania oznaczałoby, iż splątanie nie jest konieczne do wykonywania algorytmów kwantowych.

Macierz reprezentująca bramkę $CNOT$ w bazie $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, gdzie

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

jest stosunkowo prosta:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{4.3}$$

Każdy stan bazowy jest przeprowadzany w stan bazowy zgodnie z regułą

$$\begin{aligned} |00\rangle &\mapsto |00\rangle \\ |01\rangle &\mapsto |01\rangle \\ |10\rangle &\mapsto |11\rangle \\ |11\rangle &\mapsto |10\rangle \end{aligned}$$

Dlatego dla stanów bazowych, np. $|00\rangle$, wykonanie operacji $CNOT$ nie wprowadza splątania

$$E_F(|CNOT|00\rangle\rangle\langle CNOT|00\rangle|) = 0.$$

Natomiast dla superpozycji stanów bazowych $|b\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle)$, otrzymujemy po wykonaniu bramki $CNOT$ stan

$$CNOT\frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix}$$

czyli stan

$$|\phi_+\rangle = \frac{1}{\sqrt{2}}(|11\rangle + |11\rangle). \tag{4.4}$$

Jest to jeden ze stanów Bella 1.5. Splątanie formacji osiąga dla tego stanu (podobnie jak dla pozostałych stanów Bella) wartość $E_F(()|\phi_+\rangle\langle\phi_+|) = 1$, czyli wartość maksymalną na przestrzeni $\mathcal{S}(\mathcal{H}_A \otimes \mathcal{H}_B)$ przy $\dim \mathcal{H}_A = \dim \mathcal{H}_B = 2$. Zatem wykonanie bramki $CNOT$ na stanie separowalnym $|a\rangle$ pozwoliło na uzyskanie stanu maksymalnie splątanego.

Widmo operatora gęstości $|\phi_+\rangle\langle\phi_+|$ poddanego operacji transpozycji względem podukładu A to zbiór

$$\sigma\left(|\phi_+\rangle\langle\phi_+|^{T_A}\right) = \left\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, -\frac{1}{2}\right\} \tag{4.5}$$

czyli ujemność dla stanu $|\phi_+\rangle\langle\phi_+|$ wynosi $\mathcal{N}(|\phi_+\rangle\langle\phi_+|) = \frac{1}{2}$. Podobnie dla wszystkich stanów Bella

$$\mathcal{N}(|\phi_\pm\rangle\langle\phi_\pm|) = \mathcal{N}(|\psi_\pm\rangle\langle\psi_\pm|) = \frac{1}{2}. \tag{4.6}$$

Najczęściej w trakcie wykonania algorytmu kwantowego stanem układu kwantowego jest superpozycja stanów bazowych. Warto zatem prześledzić na przykładzie jak zmienia się splątania w trakcie ewolucji zadanej bramką $CNOT$ w zależności od stanu początkowego.

Na Rysunku 4.1 przedstawiony jest wykres obrazujący jak zmieni się splątanie formacji, jeżeli na wejściu operacji $CNOT$ pojawi się niejednorodna superpozycja stanów bazowych postaci[5]

$$|a\rangle := a|11\rangle + \sqrt{1-a^2}|10\rangle, \tag{4.7}$$

przy $a \in [0, 1]$.

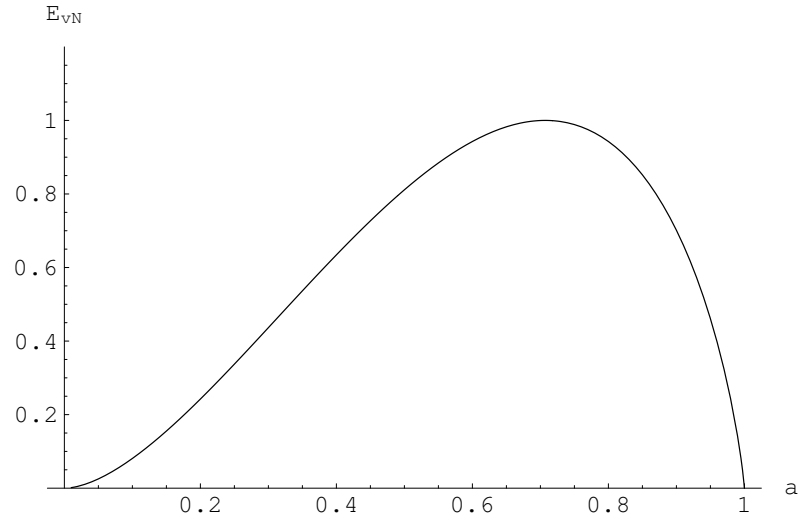
Stan ten ma następujący rozkład Schmidta:

$$|a\rangle = |1\rangle \otimes (a|1\rangle + \sqrt{1-a^2}|0\rangle), \tag{4.8}$$

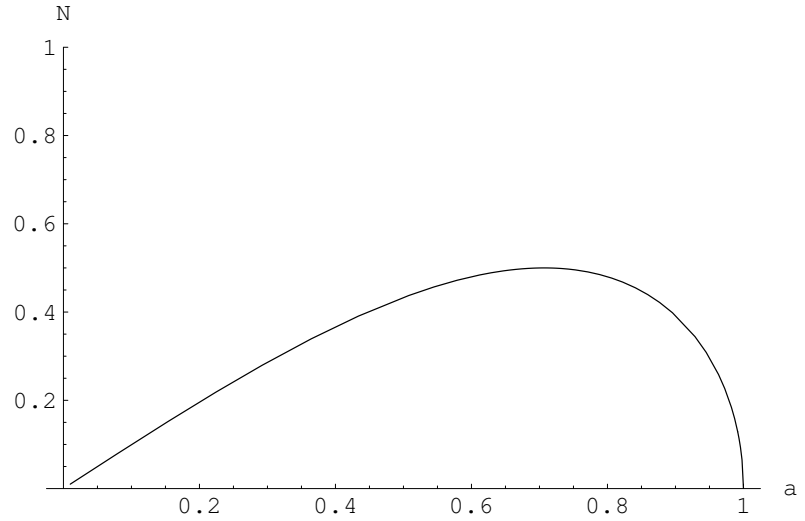
czyli jest stanem separowalnym.

Z kolei Rysunek 4.2 obrazuje zmianę ujemności stanu w trakcie ewolucji. Zachowuje się ona podobnie jak splątanie formacji.

[5]Ponieważ ustalanie amplitud wymaga jedynie zachowania normalizacji, można parametryzować stany na wiele sposobów.



Rysunek 4.1: Splątanie formacji dla stanów $CNOT(a|11\rangle + \sqrt{1-a^2}|01\rangle)$



Rysunek 4.2: Ujemność dla stanów $CNOT(a|11\rangle + \sqrt{1-a^2}|01\rangle)$

Stan początkowy jest tu stanem separowalnym, zatem obie miary zerują się na całym przedziale $a \in [0, 1]$. Natomiast w zależności od superpozycji bramka $CNOT$ powoduje odpowiedni wzrost ilości splątania w układzie.

Ponieważ dwukrotne wykonanie bramki $CNOT$ jest równoważne operacji identyczności na dwóch qubitach, splątanie po operacji $CNOT^2$ powraca do stanu początkowego – ponowne zastosowanie bramki $CNOT$ zniszczy splątanie powstałe w układzie.

Prosty przykład pokazuje, że połączenie superpozycji i oddziaływania prowadzi do uzyskania splątania. Bramka $CNOT(H \otimes \mathbb{I})$ powstała z połączenia $CNOT$ i H, wykonana na stanie $|0\rangle \otimes |0\rangle$ daje stan splątany

$$CNOT(H \otimes \mathbb{I})(|0\rangle \otimes |0\rangle) = \frac{|00\rangle + |11\rangle}{\sqrt{2}}, \tag{4.9}$$

czyli stan Bella $|\psi_+\rangle$.

4.3 Obliczenia dla algorytm Deutscha

Algorytm Deutscha jest najprostszym algorytmem kwantowym, który pozwala zaobserwować najważniejsze cechy algorytmów ukrytej podgrupy. Ponieważ algorytm ten operuje na dwóch qubitach, jest to także jedyny algorytm, w trakcie realizacji którego nie budzi wątpliwości podział układu kwantowego, na którym operujemy na dwie części.

Celem algorytmu jest określenie globalnej własności funkcji $f\colon \mathbb{B}^2 \to \mathbb{B}^2$. Pytamy, czy jest ona stała. Jeżeli funkcja nie jest stała, to mówimy, że jest zbalansowana.

Łatwo zaobserwować, że występowanie splątania nie jest konieczne do rozwiązania na komputerze kwantowym problemu Deutscha [15]. Unikamy go poprzez wprowadzenie modyfikacji omówionej w paragrafie 3.3.1. Jednak pierwotna wersja algorytmu także nie wymaga splątania dla każdej z czterech możliwych funkcji $\mathbb{B} \to \mathbb{B}$.

Jak zawsze podczas wykonania algorytmów ukrytej podgrupy, zakładamy, że dla danej funkcji f potrafimy wykonać operację U_f określoną wzorem:

$$U_f|x\rangle \otimes |y\rangle = |x\rangle \otimes |f(x) \oplus y(\bmod 2)\rangle \tag{4.10}$$

To proste założenie prowadzi do konsekwencji związanych ze złożonością algorytmów kwantowych[6].

Przebieg algorytmu prowadzi w kolejnych krokach do następujących stanów komputera kwantowego:

- 0) Podprocedura przygotowania stanu[7] $\frac{1}{\sqrt{2}}|0\rangle \otimes (|0\rangle - |1\rangle)$

[6] Uwaga ta dotyczy także algorytmu Grovera, szczególnie jeżeli zechcemy go wykorzystać do rozwiązania niektórych problemów.

[7] Podprocedurę tę można zapisać jako przygotowanie stanu $|0\rangle \otimes |1\rangle$, a następnie podziałanie na niego operatorem $\mathbb{I} \otimes H$

1. Przygotuj stan $|\psi_0\rangle = |0\rangle \otimes |0\rangle$
2. Wykonaj bramkę $\mathbb{I} \otimes NOT$
3. Wykonaj bramkę $\mathbb{I} \otimes H$

- 1) Wykonaj bramkę $H \otimes \mathbb{I}$ na $|\psi_0\rangle$ aby otrzymać

$$|\psi_1\rangle = H \otimes \mathbb{I}|0\rangle \otimes |0\rangle = \frac{1}{2}(|0\rangle + |1\rangle) \otimes (|0\rangle - |1\rangle) \tag{4.11}$$

  czyli jednorodną superpozycję wszystkich stanów bazowych.

- 2) Wykonaj operację U_f zdefiniowaną wzorem

$$U_f|x\rangle \otimes |y\rangle = |x\rangle \otimes |f(x) \oplus y\rangle$$

$$\begin{aligned} |\psi_2\rangle = U_f|\psi_1\rangle &= U_f \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes (|0\rangle - |1\rangle) \\ &= \frac{1}{2}(|0\rangle + |1\rangle)(|f(0)\rangle 1 |f(1)\rangle) \end{aligned} \tag{4.12}$$

  W zależności od funkcji, którą rozpatrujemy, otrzymamy

$$|\psi_2\rangle = \begin{cases} \pm\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) & dla\ f_1\ i\ f_2 \\ \pm\frac{1}{2}(|0\rangle + |1\rangle)(|0\rangle - |1\rangle) & dla\ f_3\ i\ f_4 \end{cases} \tag{4.13}$$

- 3) Wykonaj bramkę $H \otimes \mathbb{I}$ na $|\psi_2\rangle$, aby otrzymać stan

$$|\psi_3\rangle = H \otimes \mathbb{I}|\psi_2\rangle. \tag{4.14}$$

Dla funkcji f, która jest zbalansowana, stan $|\psi_3\rangle$ jest stanem Bella. Oznacza to, że stan podczas wykonania algorytmu jest stanem splątanym.

Wszystkie funkcje, jakie wchodzą w grę w naszym przypadku, zostały wypisane na stronie 40. Możemy prześledzić przebieg algorytmu we wszystkich czterech przypadkach – każdej funkcji odpowiada inna bramka U_f.

Funkcja tożsamościowa $f_4 = \{(0,0),(1,1)\}$ jest implementowana na komputerze kwantowym jako bramka $CNOT$ wprowadza więc splątanie. Także funkcja $f_4 = \{(0,1),(1,0)\}$ wprowadza splątanie, jako, iż jest ona reprezentowana przez macierz

$$U_{f_4} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{4.15}$$

W zapisie stosowanym w pakiecie `QuCalc` odpowiada to funkcji

```
In[1]:= gate[2, (ket[{#1, CirclePlus[#2, #1, 1]}])&]
```

Zatem bramka U_{f_4} zmienia stan drugiego qubitu wtedy i tylko wtedy, gdy pierwszy qubit jest w stanie $|0\rangle$.

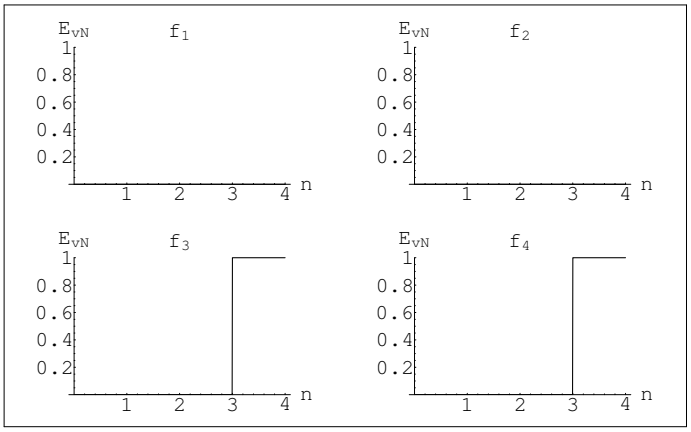
Natomiast bramki U_{f_1} i U_{f_2} są reprezentowane odpowiednio przez macierze

$$U_{f_1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \; U_{f_2} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{4.16}$$

Bramka U_{f_1} odpowiada operacji identyczności na stanie dwóch qubitów, nie wprowadza zatem splątania. Natomiast w przypadku bramki U_{f_2} łatwo zauważyć, że

$$\begin{aligned} U_{f_2} &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\ &= \mathbb{I} \otimes \sigma_x. \end{aligned}$$

Oznacza to, że odpowiada ona jednoczesnemu wykonaniu operacji lokalnych na podukładach i jako taka nie może wprowadzać splątania. Rysunek 4.3 przed-

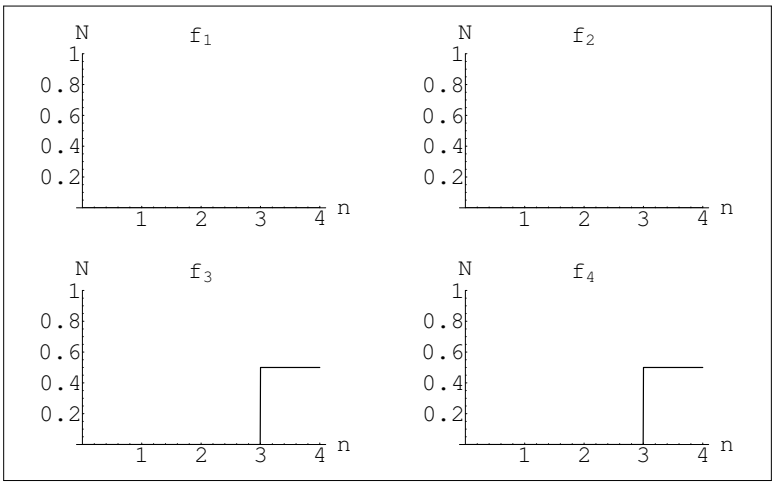


Rysunek 4.3: Splątanie formacji podczas wykonania algorytmu Deutscha.

stawia wartości zredukowanej entropii von Neumana dla stanów komputera kwantowego podczas wykonania algorytmu Deutscha. Na osi poziomej zaznaczone są numery kolejnych kroków algorytmu.

Natomiast Rysunek 4.4 przedstawia ujemność stanów komputera kwantowego podczas wykonania tego algorytmu.

Wyraźnie widać, że obie te miary zgadzają się w tym przypadku – w ostatnim kroku algorytmu Deutscha dla funkcji f_3 i f_4 występuje splątanie. Układ znajduje się wówczas w stanie Bella, co odpowiada wartości ujemności $\mathcal{N}(|a\rangle\langle a|) = \frac{1}{2}$ oraz wartości zredukowanej entropii von Neumana $E_F(|a\rangle\langle a|) = 1$.



Rysunek 4.4: Ujemność podczas wykonania algorytmu Deutscha

4.4 Symulacja algorytmu Simona

W Rozdziale 3.4 opisany został oryginalny problem Simona dotyczący znajdowani pewnej globalnej własności funkcji $f\colon \mathbb{B}^n \to \mathbb{B}^m,\ m \geq n$, zaproponowany w pracy [67]. Badanie splątania podczas wykonania tego algorytmu rozpoczniemy od najprostszych funkcji $f : \mathbb{B}^2 \to \mathbb{B}^2$.

4.4.1 Funkcje $f\colon \mathbb{B}^2 \to \mathbb{B}^2$ oraz $f\colon \mathbb{B}^2 \to \mathbb{B}^3$

W przypadku funkcji $f\colon \mathbb{B}^2 \to \mathbb{B}^2$ możemy rozpatrywać 16 możliwych funkcji. Z punktu widzenia algorytmu Simona interesujące są tylko te, które posiadają określoną strukturę.

Zgodnie z sformułowaniem problemu Simona, funkcja posiadająca maskę niezmienniczości względem sumy modulo 2, ma tę własność, iż jej wartości są stałe na argumentach różniących się o pewien stały ciąg bitów – oznaczmy go przez s.

Proste przykłady takich funkcji otrzymujemy dla $s = (1,1)$ lub $s = (0,1)$. W pierwszym przypadku możemy przyjąć

$$f_{S1}(x,y) \begin{cases} (1,1) & x = y \\ (0,0) & x \neq y \end{cases} \tag{4.17}$$

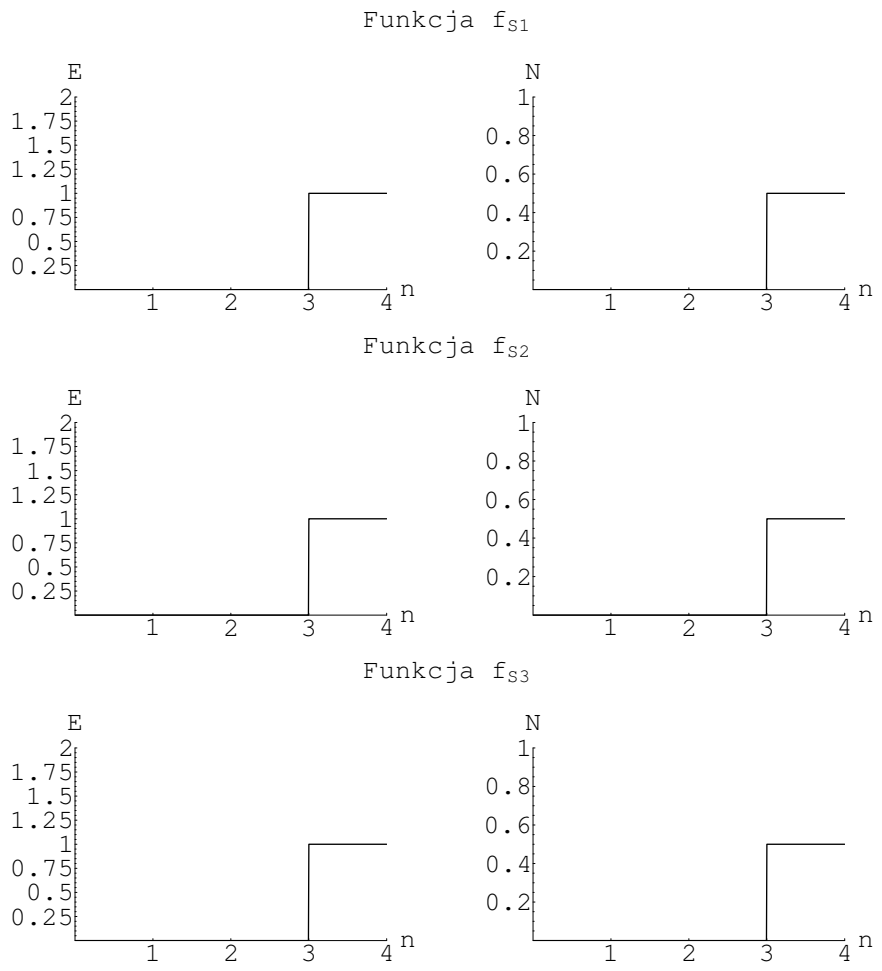
natomiast dla drugiego przypadku możemy wybrać

$$f_{S2}(x,y) \begin{cases} (1,1) & x = 0 \\ (0,0) & x = 1 \end{cases} \tag{4.18}$$

lub

$$f_{S3}(x,y) \begin{cases} (1,0) & x = 0 \\ (0,1) & x = 1 \end{cases} \tag{4.19}$$

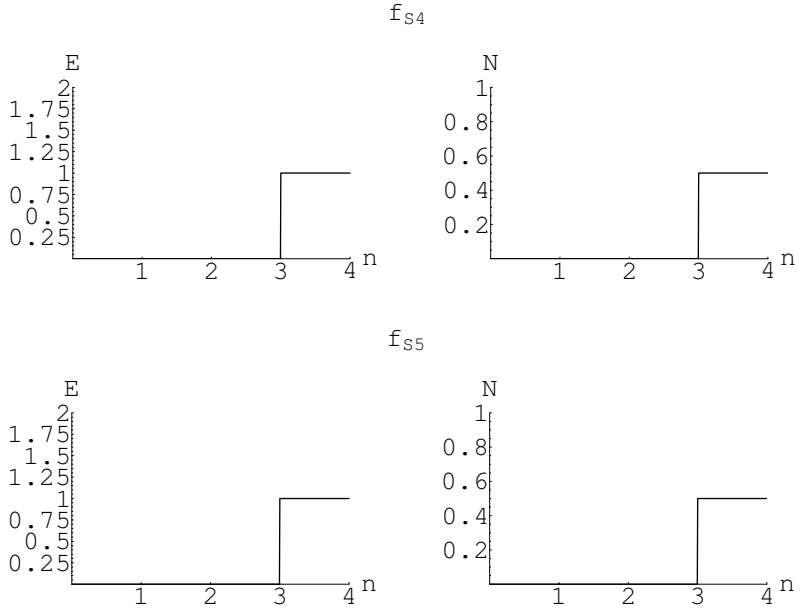
Na rysunku 4.5 przedstawione są wykresy obrazujące zmiany stopnia splątania stanów komputera kwantowego podczas wykonania algorytmu Simona dla funkcji f_{S1}, f_{S2} oraz f_{S3}.



Rysunek 4.5: Splątanie w algorytmie Simona dla $f\colon \mathbb{B}^2 \to \mathbb{B}^2$

Widać wzrost splątania następujący po wykonaniu kroku 3, który odpowiada wykonaniu drugiej transformaty Fouriera w algorytmie. Obie miary wskazują na podobny wzrost splątania. Należy zauważyć jednak, że stan otrzymywany w ostatnim kroku algorytmu Simona nie jest stanem maksymalnie splątanym – zredukowana entropia von Neumana nie osiąga na nim wartości maksymalnej.

Następny przykład prezentuje podobne wyniki w przypadku funkcji $\mathbb{B}^2 \to \mathbb{B}^3$. Rysunek 4.6 obrazuje zredukowaną entropię von Neumana oraz ujem-



Rysunek 4.6: Splatanie dla funkcji $\mathbb{B}^2 \to \mathbb{B}^3$

ność dla dwóch funkcji, f_{S4} oraz f_{S5}, zadanych wzorami:

$$f_{S4}(x,y)\begin{cases}(1,0,1) & y=1\\(0,1,0) & y=0\end{cases} \tag{4.20}$$

oraz

$$f_{S5}(x,y)\begin{cases}(1,0,1) & x=y\\(0,1,0) & x\neq y\end{cases}. \tag{4.21}$$

4.4.2 Funkcje wielobitowe

Bardziej skomplikowane funkcje pozwalają wybierać wśród większej liczby możliwości. Ograniczymy się tylko do funkcji $f\colon \mathbb{B}^2 \to \mathbb{B}^3$ oraz $f\colon \mathbb{B}^3 \to \mathbb{B}^3$. Funkcji pierwszego typu są ciekawe, ponieważ jest możliwość stosowania do układów $\mathcal{H}_A \to \mathcal{H}_B$ przy $\dim \mathcal{H}_A \leq 2, \dim \mathcal{H}_3 \leq 3$ zarówno splątania formacji, jak i ujemności jako wskaźników splątania. Dla funkcji drugiego rodzaju ujemność nie jest dobrym wskaźnikiem splątania.

Dwie funkcje : $\mathbb{B}^2 \to \mathbb{B}^3$ – f_{S4} oraz f_{S5} – są zadane formułami:

$$f_{S4}(x,y,z) = \begin{cases}(0,1,0), & y=1\\(1,0,1), & y=0\end{cases} \tag{4.22}$$

$$f_{S5}(x,y,z) = \begin{cases}(0,1,0), & x=y\\(1,0,1), & x\neq y\end{cases} \tag{4.23}$$

i różnią się maską niezmienniczości. Dla f_{S4} maska nizmienniczości to $s = (0, 1)$, natomiast dla f_{S5} – $s = (1, 1)$.

Kryterium Peresa-Horodeckiego nie pozwala orzec jednoznacznie, czy stan układu opisanego przestrzenią Hilberta $\mathcal{H}_A \otimes \mathcal{H}_B$ dla $\dim \mathcal{H}_A, \dim \mathcal{H}_B = 3$ jest splątany.

Na prawym wykresie na Rysunku 4.7 przedstawiona jest zmiana ujemności podczas wykonania algorytmu Simona dla funkcji $\mathbb{B}^3 \to \mathbb{B}^3$ – f_{S6}, f_{S7} oraz f_{S8}. Pomimo tego, iż entropia von Neumana świadczy o występowaniu splątania, ujemność wskazuje na separowalność stanu.

Dla funkcji f_{S6} i f_{S7} mamy $s = (0, 1, 0)$

$$f_{S6}(x, y, z) = \begin{cases} (0, 0, 0), & x = z = 0 \\ (1, 0, 1), & x = z = 1 \\ (0, 0, 1), & x = 0 \wedge z = 1 \\ (1, 0, 0), & x = 1 \wedge z = 0 \end{cases} \tag{4.24}$$

$$f_{S7}(x, y, z) = \begin{cases} (0, 0, 0, 0), & x = z = 0 \\ (1, 0, 1, 0), & x = z = 1 \\ (0, 0, 1, 1), & x = 0 \wedge z = 1 \\ (1, 0, 0, 0), & x = 1 \wedge z = 0 \end{cases} \tag{4.25}$$

natomiast dla funkcji

$$f_{S8}(x, y, z) = \begin{cases} (0, 0, 0, 0), & x = z = 0 \\ (1, 0, 1, 0), & x = z = 1 \\ (0, 0, 1, 1), & x = 0 \wedge z = 1 \\ (1, 0, 0, 0), & x = 1 \wedge z = 0 \end{cases} \tag{4.26}$$

mamy $s = (0, 1, 1)$.

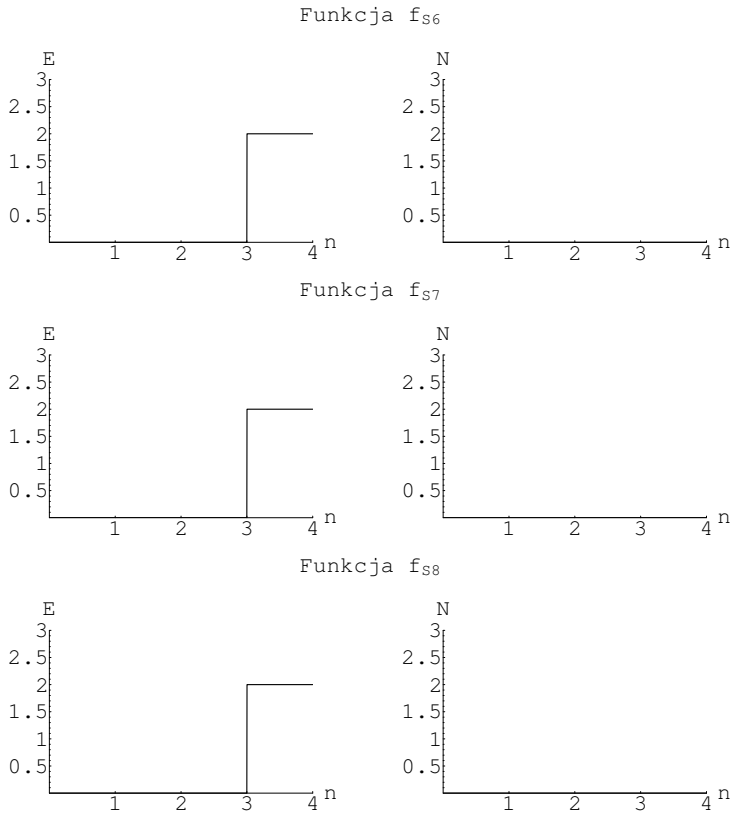
Zatem w tych wypadkach zawodzi kryterium Peresa-Horodeckiego. Jedyną miarą splątania, którą możemy stosować w tym wypadku aby mieć pewność występowania bądź nie splątania, jest zredukowana entropia von Neumana.

4.5 Możliwość symulacji algorytmu Shora

Dotychczasowe obliczenia dotyczyły funkcji określonych na niewielkiej ilości bitów. W przypadku algorytmu Shora problemem staje się czas potrzebny na przeprowadzenie symulacji[8]. Pomimo że schemat algorytmu jest podobny jak w przypadku algorytmu Simona, funkcje które wchodzą w grę określone są na zbiorach $\mathbb{Z}_q$ $q \geq 256$.

Warto zauważyć, iż problem symulacji układów kwantowych w przypadku algorytmu Shora jest związany z rozwiązaniem ważnego zagadnienia praktycznego. Mianowicie możliwość wykonania efektywnej symulacji tego algorytmu daje automatycznie przepis na efektywny algorytm faktoryzacji.

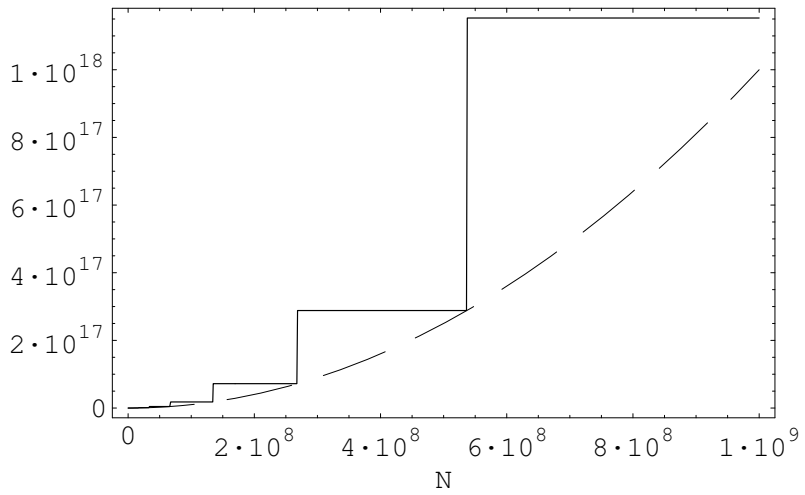
[8]Symulacjie algorytmu Shora dla niewielkiej liczby qubitów są możliwe do wykonania z użyciem języka QCL. Służy temu np. program shor, rozprowadzany razem z językiem QCL (patrz: Rozdział 7), i napisany z użyciem biblioteki libqc, wchodzącej w skład języka QCL.



Rysunek 4.7: Rozbieżność w rozpoznaniu splątania dla funkcji $\mathbb{B}^3 \to \mathbb{B}^3$

4.5.1 Oszacowanie czasu

Algorytm Shora wykorzystuje transformatę Fouriera na grupie $\mathbb{Z}_q$. Liczba q jest dobierana odpowiednio do liczby N, podanej na wejściu: musi ona być potęgą dwójki spełniającą warunek $N^2 < q < 2N^2$. W praktyce dla $N = 15$, $q = 256$, co w wypadku symulacji oznacza wykonywanie (klasycznej) transformaty Fouriera[9] o rozmiarze 256×256. Poniższy wykres obrazuje tempo wzrostu rozmiaru macierzy, którą musimy operować, aby wykonać symulację.



Rysunek 4.8: Tempo wzrostu rozmiaru QFT w trakcie realizacji algorytmu znajdowania rzędu.

Na Rysunku 4.8 jest przedstawione tempo wzrostu rozmiaru QFT w funkcji liczby której podzielników szukamy. Wzrost ten porównać można ze wzrostem funkcji x^2, której wykres jest zaznaczony na Rysunku 4.8 linią przerywaną. Pozwala to oszacować, iż rozmiar transformaty Fouriera rośnie w przybliżeniu jak druga potęga rozmiaru liczby wyrażonego w bitach. Zatem wraz z rozmiarem danych wejściowych, wzrost rozmiaru operacji koniecznej do wykonania jest wykładniczy. Pomimo, że algorytm FFT pozwala na znaczne zmniejszenie czasu wykonania dyskretnej transformaty Fouriera, nie wystarcza to na efektywną symulację algorytmu.

[9] Jeżeli nie stosujemy algorytmu FFT, to jest to równoważne pomnożeniu przez macierz kwadratową $q \times q$

4.6 Manipulacja prawdopodobieństwem

W algorytmach kwantowych mechanika kwantowa pozwala nam na operowanie prawdopodobieństwami z jakim otrzymamy pewne wyniki. Prawdopodobieństwo gra kluczową rolę w projektowaniu algorytmów kwantowych – uzyskanie odpowiedniego stanu końcowego jest równoważne otrzymaniu odpowiedniego rozkładu prawdopodobieństwa na pełnej grupie.

Na algorytmy kwantowe można zatem spojrzeć jako na sposób generowania rozkładów prawdopodobieństwa, pozwalający na uzyskanie z jak największym prawdopodobieństwem informacji na temat interesujących nas wielkości. Podejście to stawia nas w roli nie tyle obserwatora, który wyciąga wnioski z uzyskanego wyniku, ale w roli projektanta ewolucji układu.

Wspomniany wcześniej algorytm Grovera wykorzystuje prawdopodobieństwo otrzymania pewnego rezultatu do sprawdzania, czy w danym zbiorze pojawia się element dający pozytywny przykład problemu decyzyjnego.

Tutaj przyjrzyjmy się jak zmienia się rozkład prawdopodobieństwa podczas wykonania algorytmu kwantowego ukrytej podgrupy.

4.6.1 Kolejne kroki algorytmu Simona

Ostateczny rozkład prawdopodobieństwa w algorytmie Shora jest doskonałą ilustracją wyników jakie daje algorytm kwantowy. Aby zobaczyć, jak zmienia się rozkład prawdopodobieństwa po wykonaniu każdego kroku posłużmy się algorytmem dla oryginalnego problemu Simona[10]. Algorytmy te nie różnią się w sposobie manipulowania prawdopodobieństwem i ich konstrukcja oparta jest na tym samym, ogólnym schemacie.

Algorytm Simona jest prostszy i bardziej przejrzysty, a jednocześnie pozwala na lepsze zrozumienie działania algorytmów ukrytej podgrupy. Wynika to między innymi z tego, iż zaprojektowano go do wykorzystania na zbiorach skończonych i operuje on na funkcjach boolowskich $f\colon \mathbb{B}^n \to \mathbb{B}^m$. Aby zbadać, co dzieje się z rozkładem prawdopodobieństwa podczas wykonywania algorytmu prześledźmy jego kolejne kroki, dokonując po każdym z nich pomiaru[11].

Oznaczmy przez $\mathrm{P}(|x\rangle)$ prawdopodobieństwo, iż układ jest w stanie $|x\rangle$. Chcemy rozwiązać problem XMI dla funkcji $f\colon \mathbb{B}^n \to \mathbb{B}^m$.

1. Przygotuj stan początkowy

$$|\phi_0\rangle = |0^n\rangle|0^m\rangle \tag{4.27}$$

[10] Opis algorytmu znajduje się w Rozdziale 3.4.2

[11] Prosta realizacja fizyczna takiego eksperymentu myślowego polegałaby na wykonaniu serii przebiegów algorytmu Simona dla ustalonej funkcji, przy czym przerywalibyśmy wykonanie algorytmu po kolejnych krokach. Eksperyment taki różni się od wykonania algorytmu raz z pomiarem po każdym kroku.

2. Wykonaj transformatę Hadamarda na pierwszym qubicie

$$\begin{aligned} |\phi_1\rangle &= H^n \otimes \mathbb{I}_m |0^n\rangle|0^m\rangle \\ &= \sum_{x=0}^{2^m} |x\rangle|0^m\rangle. \end{aligned} \tag{4.28}$$

$\mathbb{I}_m$ oznacza tu operację identyczności na przestrzeni m-wymiarowej, która służy nam do opisu zbioru wartości. Po wykonaniu tego kroku otrzymujemy

$$\mathrm{P}(|a\rangle|b\rangle) = \begin{cases} \frac{1}{2^n} & b = 0 \\ 0 & b \neq 0 \end{cases} \tag{4.29}$$

czyli rozkład jest jednorodny na całym zbiorze $\mathbb{B}^n$. Uzyskujemy w ten sposób generator liczb losowych o rozkładzie jednorodnym na $\mathbb{B}^n$.

3. Zastosuj transformację U_f

$$\begin{aligned} |\phi_2\rangle &= U_f |\phi_1\rangle \\ &= \sum_{x=0}^{2^n} |x\rangle|f(x)\rangle. \end{aligned} \tag{4.30}$$

Prowadzi to do następującego rozkładu prawdopodobieństwa

$$\mathrm{P}(|a\rangle|b\rangle) = \begin{cases} \frac{2}{2^n} & \bigvee_x b = f(x) \\ 0 & \neg \bigvee_x b = f(x) \end{cases} . \tag{4.31}$$

4. Wykonaj drugi raz opeację $H^n \otimes \mathbb{I}_m$

$$\begin{aligned} |\phi_3\rangle &= H^n \otimes \mathbb{I}_m |\phi_2\rangle = \\ &= \sum_{x=0}^{2^n} \sum_{y=0}^{2^m} (-1)^{x \cdot y} |y\rangle|f(x)\rangle \end{aligned}$$

otrzymując w ten sposób rozkład prawdopodobieństwa zadany wzorem

$$\mathrm{P}(|a\rangle) = \begin{cases} \frac{1}{2^m} & a \cdot s = 0 \\ 0 & a \cdot s = 1 \end{cases} . \tag{4.32}$$

Zatem w wyniku dokonania pomiaru z równym prawdopodobieństwem uzyskamy jeden z elementów prostopadłych do s czyli spełniających warunek

$$a \cdot s = 0. \tag{4.33}$$

Wystarczająca duża liczbe przebiegów algorytmu pozwala na znalezienie elementu s.

4.6.2 Pomiar końcowy w algorytmie Shora

W przypadku algorytmu Shora, w wyniku wykonania procedury znajdowania rzędu, otrzymujemy rozkład prawdopodobieństwa na pewnej grupie, której elementy są reprezentowane za pomocą stanów bazowych komputera kwantowego. Przed pomiarem końcowym[12] stan układu jest opisany wektorem

$$|\phi\rangle = \frac{1}{q}\sum_{x=0}^{q-1}\sum_{y=0}^{q-1}\omega^{xy}|y\rangle|a^x \bmod N\rangle, \tag{4.34}$$

gdzie $\omega = \exp\left(\frac{2i\pi}{q}\right)$.

Interesuje nas informacja zawarta w pierwszym rejestrze, gdyż odzwierciedla ona okresowość funkcji $f_a(x) = a^x \bmod N$.

Prawdopodobieństwo znalezienia układu w stanie $|y\rangle \otimes |dowolny\_stan\rangle$ wynosi

$$\mathrm{P}(|y\rangle) = \sum_{i=1}^{q-1}\mathrm{P}(|y\rangle|i\rangle) \tag{4.35}$$

Ponieważ dla $u, v \in \mathbb{Z}_q$

$$\begin{aligned}(\langle u|\langle v|)|\phi\rangle &= \sum_{x,y=0}^{q-1}\omega^{xy}\langle u|y\rangle\langle v|a^x \bmod N\rangle \\ &= \sum_{x=0}^{q-1}\omega^{xu}\langle b|a^x \bmod N\rangle.\end{aligned}$$

Rozpisując wyrażenie 4.35 otrzymujemy

$$\begin{aligned}\mathrm{P}(|y\rangle) &= \sum_{c=1}^{q-1}\mathrm{P}(|y\rangle|c\rangle) \\ &= \frac{1}{q^2}\sum_{c=1}^{q-1}\left|\sum_{x=0}^{q-1}\omega^{xu}\langle c|a^x \bmod N\rangle\right|^2\end{aligned}$$

Ponieważ r jest okresem funkcji

$$\begin{aligned}\langle m^z \bmod N|m^x \bmod N\rangle \neq 0 &\Leftrightarrow m^z = m^x \bmod N \\ &\Leftrightarrow z = kr + x,\end{aligned}$$

[12]Można przyjąć, że pomiar jest ostatnim krokiem algorytmu.

gdzie k jest liczbą całkowitą z zakresu od 0 do $\lfloor \frac{q}{r} \rfloor$, ostatecznie otrzymujemy

$$\mathrm{P}(|y\rangle) = \frac{1}{q^2} \sum_{i=1}^{q-1} \left| \sum_{k=0}^{r} \sum_{\{l \in \mathbb{Z} | k+rl \leq q\}} \omega^{(k+rl)u} \langle c | a^k \bmod N \rangle \right|^2 \tag{4.36}$$

$$= \frac{1}{q^2} \sum_{i=1}^{q-1} \left| \sum_{k=0}^{r} \sum_{l=0}^{\lfloor (q-k)/r \rfloor} \omega^{(k+rl)u} \langle c | a^k \bmod N \rangle \right|^2 \tag{4.37}$$

Należy zwrócić uwagę na to, że po pierwszym kroku procedury znajdowania rzędu rozkład prawdopodobieństwa na przestrzeni stanów komputera kwantowego jest rozkładem jednostajnym[13]. Zastosowanie operacji unitarnej U_{f_a} oraz transformaty Fouriera powoduje taką zmianę amplitudy prawdopodobieństwa, iż w efekcie stan układu przejdzie do któregoś ze stanów uprzywilejowanych przez periodyczność funkcji f_a. Podczas wykonania tych dwóch operacji stan układu kwantowego jest stanem splątanym.

Innymi słowy: periodyczność funkcji f_a przejawi się w periodyczności funkcji gęstości prawdopodobieństwa na przestrzeni wyników otrzymywanych w eksperymencie.

Na poniższych rysunkach przedstawione są rozkłady prawdopodobieństwa uzyskane w wyniku symulacji[14] przebiegu procedury znajdowania rzędu dla wybranych liczb.

Poniżej zamieszczone są wyniki dla liczby 15. Warto zwrócić uwagę na zmiany jakie następują w rozkładzie prawdopodobieństwa gdy zmienimy q dla jakiego wykonywane są obliczenia. Zwiększenie q powoduje, że trudniej trafić w wartość dającą informację o rzędzie. Natomiast zmniejszenie tego parametru powoduje rozmycie rozkładu prawdopodobieństwa. Zatem w tym drugim przypadku zwiększa się prawdopodobieństwo otrzymania przypadkowej wartości, a co za tym idzie zmniejsza się wydajność algorytmu.

Przykładowe obliczenia

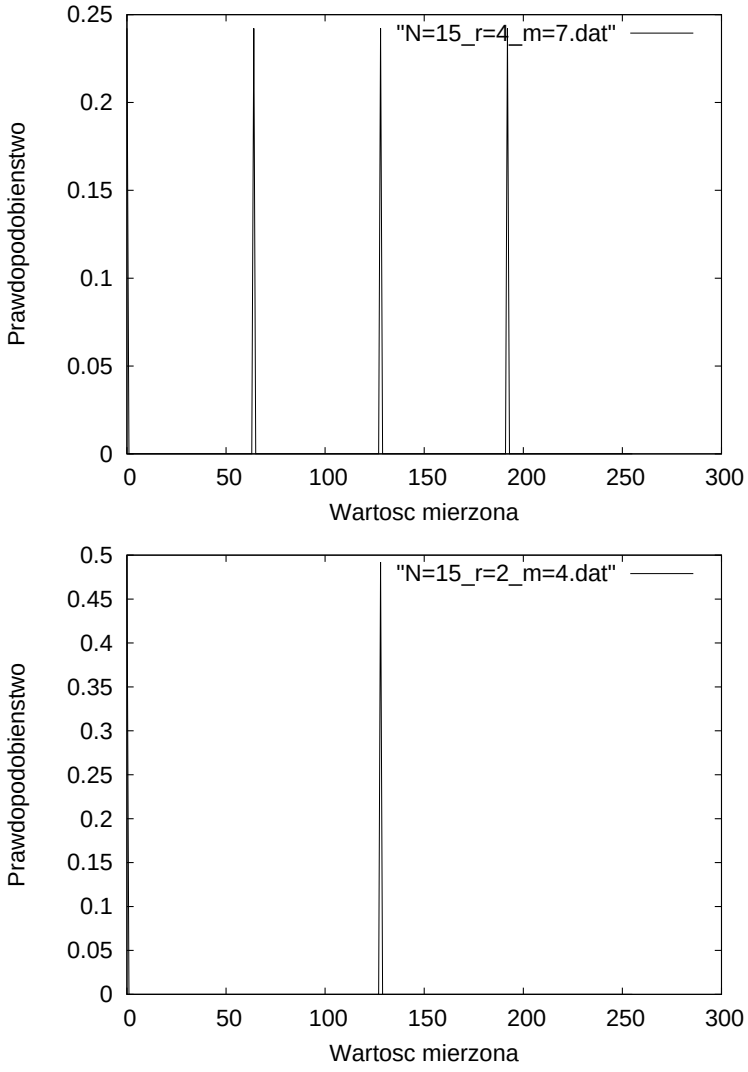
Najmniejszą liczbą, jaka może zostać poddana kwantowej procedurze znajdowania rzędu jest liczba 15. Poniżej przedstawione są wyniki obliczeń wykonanych przy użyciu programu `ShorProb`.
Pomimo tego, że liczba jest mała, cały algorytm jest równie skomplikowany jak w przypadku większych liczb.

Pierwsza seria dotyczy $N = 15$ i $r = 4$ oraz $r = 2$ przy czym obliczenia przeprowadzane są na $\mathbb{Z}_{256}$.

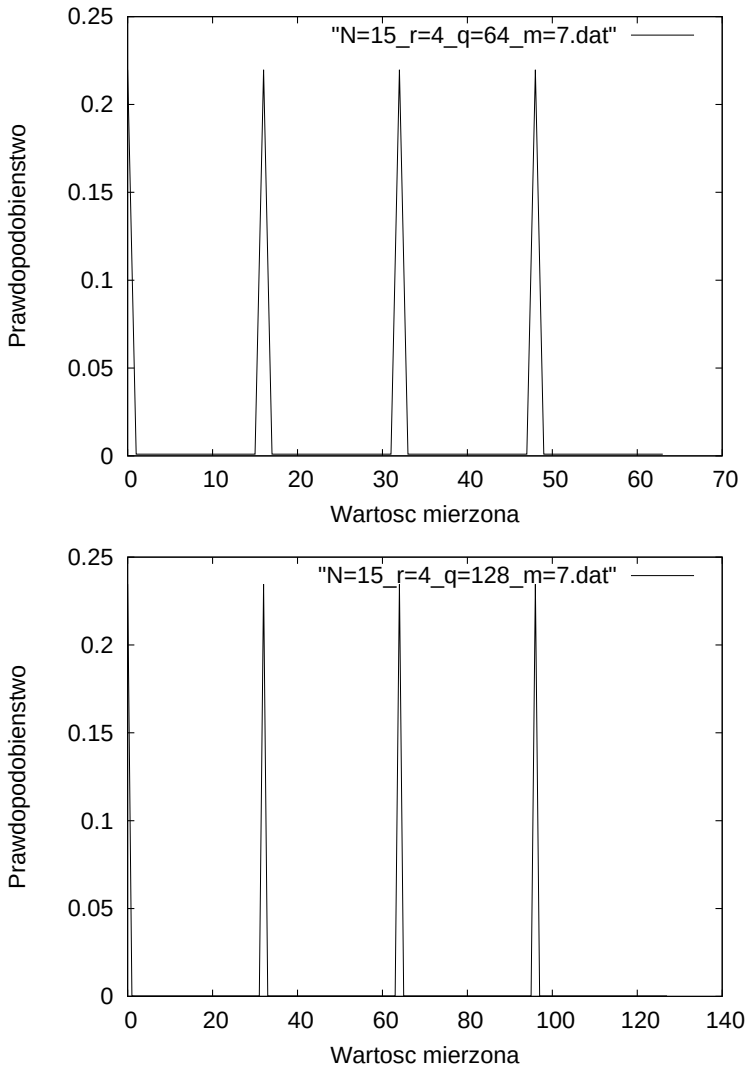
Obliczenia, których wynikiem jest rozkład prawdopodobieństwa przedstawiony na Rysunku 4.10, zostały przeprowadzone na $\mathbb{Z}_q$ przy q mniejszym od 256. W tym wypadku $q = 64$ na górnym wykresie, oraz $q = 128$ na dolnym wykresie.

[13] W pierwszym rejestrze mamy superpozycję wszystkich stanów bazowych.

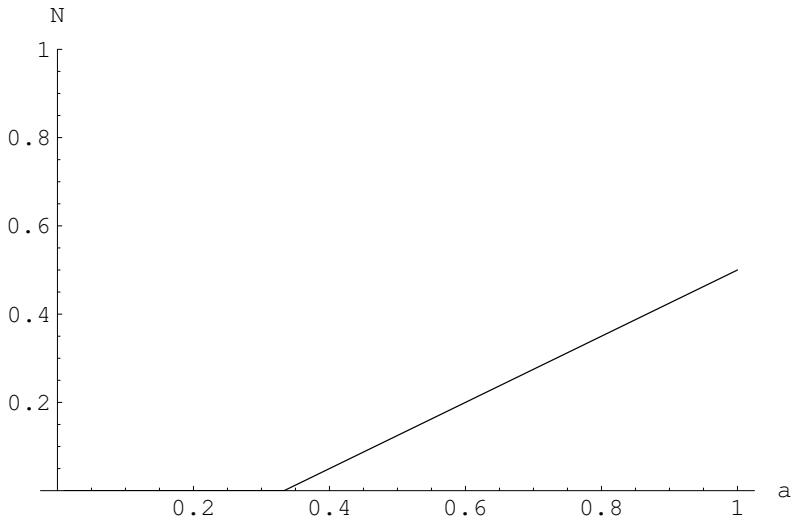
[14] Do zrealizowania takiej symulacji potrzebujemy znajomości rzędu liczby a. Implementacja w języku `C++` funkcji realizującej to zadanie jest zamieszczona w dodatku.



Rysunek 4.9: Rozkład prawdopodobieństwa dla algorytmu znajdowania rzędu dla $N = 15$



Rysunek 4.10: Wpływ zmiany q na rozkład prawdopodobieństwa w algorytmie Shora.



Rysunek 4.11: Zależność ujemności od składu mieszanki statystycznej.

4.7 Stany mieszane

W układach fizycznych rzadko mamy możliwość operowania bezpośrednio na stanach czystych. Dużym wyzwaniem dla rozwoju algorytmów kwantowych jest opracowane metod operowania na stanach mieszanych. Ważnym zagadnieniem jest też zbadanie związku i wzajemnego wpływu korelacji klasycznych i kwantowych podczas ewolucji stanu komputera kwantowego.

Prostym przykładem wpływu tych dwóch rodzajów korelacji jest wykonanie bramki $CNOT$ na stanie $\rho(a) = \frac{a}{\sqrt{2}}(|00\rangle + |10\rangle)(\langle 00| + \langle 10|) + (1-a)M(2)$, gdzie $M(2)$ jest stanem maksymalnie mieszanym układu dwóch qubitów.

Zmiana stopnia splątania układu w stanie $CNOT\rho(a)CNOT^{\dagger}$ w zależności od składu mieszanki statystycznej opisanego parametrem a pokazana jest na Rysunku 4.11.

Stan początkowy, jako mieszanka stanów separowalnych, jest stanem separowalnym. Wyraźnie widać, że dodanie stanu maksymalnie mieszanego, który odpowiada tu szumowi, powoduje obniżenie stopnia splątania.

Ujemność pozwala na obliczenia stopnia splątania stanów mieszanych układów dwuskładnikowych. W przypadku stanów czystych zredukowana entropia von Neumana jednoznacznie określa, czy dany stan jest stanem separowalnym – wówczas zredukowana macierz gęstości reprezentuje stan czysty i entropia przyjmuje wartość zero. Natomiast dla stanów mieszanych jedynie w przypadku układów niskowymiarowych kryterium Peresa-Horodeckiego daje jednoznaczną odpowiedź na pytanie czy stan jest splątany. W związku z tym ujemność

pozwala ocenić, czy dany stan jest splątany jedynie dla tych układów. Natomiast splątanie formacji nie może być obliczone dla ogólnego przypadku stanu mieszanego – znana jest jedynie formuła dla stanów mieszanych układu dwóch qubitów [79].

Rozdział 5

Znaczenie splątania

Splątanie jest obiektem zainteresowania ze względu na konsekwencje, jakie niesie ze sobą włączenie tego typowo kwantowego fenomenu do zagadnień rozpatrywanych dotychczas bez odniesienia do ich fizycznego podłoża.

Wiadomo, iż nabycie umiejętności, operowania splątaniem może pozwolić na zwiększenie wydajność przesyłania informacji. Jest to także podstawa do wykorzystania zjawiska teleportacji.

5.1 Modele maszyn kwantowych

Aby mówić o złożoności algorytmów kwantowych konieczne jest wprowadzenie modelu komputera, który może je wykonywać. Dotychczas powstały dwa takie modele:

- kwantowa maszyna Turinga
- obwody kwantowe

Oba te sformułowania pochodzą z pracy [17] Davida Deutscha. W pracy [82] pokazana została równoważność mocy obliczeniowej tych dwóch modeli.

Definicja kwantowej maszyny Turinga omówiona jest w Rozdziale 6.1.4, natomiast obwody kwantowe zostały częściowo omówione w Rozdziale 4.1 przy okazji opisu symulacji algorytmów kwantowych.

Mówiąc o przewadze komputerów kwantowych nad komputerami klasycznymi nie należy zapominać, że problem $\mathbf{P} \stackrel{?}{\neq} \mathbf{NP}$ nie został dotychczas rozstrzygnięty. Nie wiadomo zatem czy nie istnieją algorytmy rozwiązujące problemy „trudne" efektywnie. Zagadnienie $\mathbf{P} \stackrel{?}{\neq} \mathbf{NP}$ jest równoważne zagadnieniu znalezienia algorytmu symulowania niedeterministycznej maszyny Turinga na maszynie deterministycznej, który pracowałby w czasie wielomianowym. Nie wiemy także, czy nie istnieją efektywne sposoby symulowania efektów kwantowych na maszynach klasycznych. Jednakże wszystko wskazuje, iż algorytmy takie nie istnieją i, że $\mathbf{P} \neq \mathbf{NP}$.

Operacje kwantowej maszyny Turinga odbywają się na ogół na superpozycji stanów bazowych. Jest to zasadnicza różnica w stosunku do modelu klasycznej maszyny Turinga. Jeżeli nałożymy na kwantową maszynę Turinga ograniczenie, aby na każdym etapie jej ewolucji jej stan był stanem bazowym, to otrzymamy klasyczną maszynę Turinga działającą w sposób odwracalny.

Maszyny kwantowe mogą być przedstawione jako maszyny probabilistyczne. Konstrukcja algorytmu kwantowego jest równoważna konstrukcji probabilistycznej maszyny Turinga (algorytmu probabilistycznego). Ponieważ stany maszyny kwantowej są zazwyczaj numerowane elementami pewnej grupy G, algorytm probabilistyczny musi odtwarzać rozkład prawdopodobieństwa na stanach przyporządkowując prawdopodobieństwa elementom grupy G.

Rozpatrywanie kwantowych maszyn które nie są probabilistyczne – jest tak wówczas gdy w algorytmie nie występuje superpozycja stanów – powoduje, że model kwantowy sprowadza się do klasycznej maszyny Turinga.

Widać zatem, że dwiema zasadniczymi cechami wyróżniającymi maszyny kwantowe, są:

1. działanie na superpozycji stanów,
2. wytwarzanie pewnego rozkładu prawdopodobieństwa.

Ich połączenie pozwala na duże możliwości operowania prawdopodobieństwem poprzez wykorzystanie nielokalności operacji na superpozycji stanów, czyli splątania.

5.2 Kwantowa teoria złożoności

Algorytmy kwantowe są przykładem tego, że model obliczeń, który wykorzystuje kwantową naturę zjawisk przyrodniczych, wnosi nowe elementy do teorii obliczeń. Zadaniem kwantowej teorii złożoność jest odpowiedź na pytanie, czy elementy te są jakościowo nowe, i czy – co za tym idzie – komputery kwantowe są modelem o mocy obliczeniowej przekraczającej możliwości maszyn klasycznych.

Teoria złożoności stara się wyjaśnić związki pomiędzy klasami problemów, które można rozwiązać w sposób algorytmiczny. Próbuje ona podać wystarczająco ogólne prawa dotyczące wydajności obliczeń tak, aby obowiązywały dla jak najszerszej klasy problemów.

Najlepszym sposobem zrozumienia czym są algorytmy, jest utożsamienie ich z maszynami Turinga. Dany problem decyzyjny ma rozwiązanie, jeżeli istnieje maszyna Turinga, która potrafi odpowiedzieć na postawione w nim pytanie „Tak” lub „Nie”. W tym ujęciu pojawiają się w sposób naturalny problemy które nie mają rozwiązania, czyli problemy nierozstrzygalne. Możemy również dokonać podziału wszystkich możliwych algorytmów (problemów)[1] na zbiory

[1] Określenia „wszystkie możliwe” nie należy traktować jako definicji, lecz jako próbę ujęcia jak najszerszej klasy problemów.

według kryterium takiego, iż do jednego zbioru należą te problemy które mogą być rozstrzygnięte w przybliżeniu w takim samym czasie.

Omawiany w Rozdziale 3.5 algorytm Shora jest bezpośrednim dowodem na to, że komputer kwantowy może pozwolić na efektywne rozwiązywanie problemów uważanych za trudne w modelu klasycznym. W związku z tym jest konieczne zrewidowania dotychczasowych poglądów na problem złożoności algorytmów i przebudowanie teorii złożoności z uwzględnieniem informatyki kwantowej [13, 5].

Formalnie wynik uzyskany przez Petera Shora może być sformułowany w postaci następującego twierdzenia

Twierdzenie 11 *Dla danej liczby naturalnej N, $n = \log N$, istnieje obwód kwantowy zbudowany z $O(n^2 \log^d(\frac{n}{\epsilon}))$ bramek rozwiązujący problem faktoryzacji z dokładnością ϵ.*

Dotychczas najbardziej wydajny znany algorytm probabilistyczny rozwiązujący problem faktoryzacji wymagał użycia $O(2^{d\sqrt{n \log n}})$ bramek.

Wykładniczy zysk wydajności osiągany na maszynie kwantowej nie może być jednak zwiększony. Mówi o tym następujące

Twierdzenie 12 *Dla każdego obwodu $S(n)$-qubitowego złożonego z $T(n)$-bramek, istnieje klasyczny probabilistyczny obwód złożony z $O(2^{S(n)T(n)^3 \log^2(\frac{1}{\epsilon})})$ symulujący obwód kwantowy z dokładnością ϵ.*

Określenie „symulacja z dokładnością ϵ oznacza, iż po dokonaniu pomiaru w algorytmie kwantowym prawdopodobieństwo uzyskania każdego z możliwych wyników różni się od prawdopodobieństwa uzyskiwanego na klasycznej maszynie probabilistycznej o co najwyżej ϵ.

Podobnie, wynik uzyskiwany przez algorytm Grovera możemy sformułować w postaci twierdzenia

Twierdzenie 13 *Problem wyszukiwania wśród n elementów może być rozwiązany na komputerze kwantowym przy użyciu obwodu złożonego z $O(\sqrt{n \log(1/\epsilon)})$*

Także w tym wypadku okazuje się, iż rezultat uzyskany przez algorytm jest optymalny [83]. Nie jest więc możliwe podanie algorytmu wyszukiwania działającego w czasie wielomianowym względem rozmiaru przeszukiwanej bazy danych.

Oznacza to, że algorytmy kwantowe mają wyraźnie ograniczone możliwości zwiększania wydajności. Niemniej w obrębie problemów, które mogą być rozwiązane przez komputer kwantowy, leży wiele interesujących problemów.

Dodatkowym argumentem przemawiającym za wykorzystaniem maszyn kwantowych jest możliwość przeprowadzania na nich efektywnych symulacji układów kwantowych. Jest to rezultat analogiczny do klasycznego twierdzenia o istnieniu uniwersalnej maszyny Turinga. Konstrukcja kwantowej uniwersalnej maszyny Turinga została przedstawiona w pracy [5].

Mając do dyspozycji klasyczną maszynę Turinga mamy tylko jeden sposób aby dokonać symulacji maszyny probabilistycznej. Odtworzenie działania probabilistycznej maszyny Turinga jest możliwe poprzez wykonanie próbkowania po przestrzeni realizacji procesu obliczeniowego takiej maszyny.

Komputery kwantowe są tworami – jak dotychczas teoretycznymi – które dysponują nadzwyczajnymi możliwościami w zakresie manipulacji rozkładem prawdopodobieństwa. Cech ta jest widoczna już w przypadku najprostszej bramki kwantowej – bramki Hadamarda. Pozwala ona na wytworzenie jednorodnego rozkładu prawdopodobieństwa na przestrzeni stanów bazowych. Wykonując jedną operację maszynową uzyskujemy rozkład prawdopodobieństwa. Aby rozkład ten był różny od rozkładu jednorodnego możemy posłużyć się innymi bramkami.

Wyobraźmy sobie klasyczną maszynę probabilistyczną która zwraca z równym prawdopodobieństwem wartość pewnej funkcji na 1 lub 0. Aby zbadać wartości tej funkcji na maszynie klasycznej musimy wykonać dwie próby. Natomiast na maszynie kwantowej możemy posłużyć się bramką Hadamarda aby uzyskać rozkład prawdopodobieństwa, a następnie wykonać operację unitarną odpowiadając obliczeniu tej funkcji.

5.3 Kwantowe przesyłanie informacji

Kolejnym obszarem, na który wpływ może mieć zastosowanie w informatyce praw mechaniki kwantowej, jest komunikacja i przesyłanie informacji. Nowe spojrzenie na przesyłanie i kodowanie informacji daje nadzieje praktycznego zastosowania w krótkim czasie protokołów kwantowych.
Znaczenie splątania najwyraźniej widać w dwóch przykładach pokazujących, że wykorzystanie splątania może zwiększyć wydajność przesyłania informacji.

Zagadnienie teleportacji stanu – czy w ogólności zagadnienie przesyłania informacji – można sformułować jako zagadnienie obliczeniowe. W języku klasycznych maszyn Turinga z wieloma ciągami przesyłanie danych to kopiowanie z jednej taśmy – wyróżnionej jako wejściowa – na drugą taśmę – wyróżnioną jako wyjściowa.

5.3.1 Gęste kodowanie

Gęste kodowanie (ang. *dense coding*) jest prostym przykładem potencjalnego wykorzystania stanów splątanych do efektywniejszego (niż w przypadku klasycznym) przesyłania informacji. Jak sama nazwa wskazuje, w tym wypadku splątanie pozwala na wydajniejsze kodowanie symboli podczas przesyłania informacji.

Załóżmy, że Ania i Bartek mają do dyspozycji stan splątany $|\phi_+\rangle$. Ania może wykonać na swoim podukładzie jedną z operacji unitarnych reprezentowanych przez macierze Pauliego

1. $\mathbb{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$,
2. $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$,
3. $\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$,
4. $\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$.

Odpowiada to odpowiednio przejściu w jeden ze stanów:

1. $|\phi_+\rangle = \mathbb{I} \otimes \mathbb{I}|\phi_+\rangle$,
2. $|\psi_+\rangle = \sigma_x \otimes \mathbb{I}|\phi_+\rangle$,
3. $|\psi_-\rangle = \sigma_y \otimes \mathbb{I}|\phi_+\rangle$,
4. $|\phi_-\rangle = \sigma_z \otimes \mathbb{I}|\phi_+\rangle$.

Jeżeli teraz Bartek dokona pomiaru stanu całości w bazie $\{|\phi_\pm\rangle, |\psi_\pm\rangle\}$ jednoznacznie rozróżni operację jaką wykonała Ania. Cała informacji jest tu zawarta w korelacjach pomiędzy stanami qubitów Ani i Bartka. Zatem przesłanie od Ani do Bartka jednego qubitu umożliwia przesłane dwóch bitów informacji.

5.3.2 Teleportacja kwantowa

Mechanika kwantowa nie zezwala na dokonanie kopii nieznanego stanu – jest to treścią twierdzenie o nie-klonowaniu (ang. *no-cloning theorem*) [2]. Możliwe jest jednak przesyłanie na odległość dokładnej kopii nieznanego stanu kwantowego $|x\rangle$, o ile dysponujemy kanałem kwantowym.

Załóżmy, że Ania i Bartek mają do dyspozycji stan splątany $|\phi_+\rangle$. Aby przesłać dokładną kopię stanu $|x\rangle = \alpha|0\rangle + \beta|1\rangle$ Ania musi wykonać następujące kroki:

1. Utworzyć układ w stanie $|x\rangle|\phi^+\rangle$.
2. Dokonać pomiaru stany podukładu dwóch pierwszych qubitów w bazie $\{|\phi_\pm\rangle, |\psi_\pm\rangle\}$.
3. Przesłać do Bartka kanałem klasycznym dwa bity opisujące jej pomiar poprzez numer wektora bazowego.

Z kolei Bartek, aby odtworzyć stan $|x\rangle$, wykonuje następującą operację na swoim podukładzie, w zależności od otrzymanej wiadomości klasycznej:

1. $|\phi_+\rangle \rightarrow \mathbb{I}$,

2. $|\psi_+\rangle \to \sigma_x$,
3. $|\psi_-\rangle \to \sigma_y$,
4. $|\phi_-\rangle \to \sigma_z$.

Przykładowo, jeżeli Ania wybierze pomiar na stan $|\phi^+\rangle$, musi wykonać następujący ciąg operacji:

$$\begin{aligned} & (|\phi^+\rangle\langle\phi^+| \otimes \mathbb{I})|x\rangle|\phi^+\rangle = \\ = \; & \frac{1}{2}\big(|00\rangle\langle 00| \otimes \mathbb{I} + |00\rangle\langle 11| \otimes \mathbb{I} + |11\rangle\langle 00| \otimes \mathbb{I} + |11\rangle\langle 11| \otimes \mathbb{I}\big)|x\rangle|\phi^+\rangle \\ = \; & \frac{1}{2\sqrt{2}}\big(|00\rangle\langle 00| \otimes \mathbb{I}|x\rangle|00\rangle + |00\rangle\langle 11| \otimes \mathbb{I}|x\rangle|11\rangle + |11\rangle\langle 00| \otimes \mathbb{I}|x\rangle|00\rangle + |11\rangle\langle 11| \otimes \mathbb{I}|x\rangle|11\rangle\big) \end{aligned}$$

W ten sposób Bartek otrzymuje stan

$$\frac{1}{2\sqrt{2}}\big(\beta|001\rangle + \alpha|000\rangle + \alpha|000\rangle + \beta|111\rangle\big) = \frac{|00\rangle|x\rangle + |11\rangle|x\rangle}{2\sqrt{2}}$$

czyli stan zostanie przeniesiony do rejestru Bartka. Przykład obwodu kwantowego realizującego procedurę teleportacji można znaleźć w [20].

Teleportacja i gęste kodowanie są wzajemnie dopełniającymi się procedurami. W pierwszym przypadku korzystamy ze splątania do przenoszenia stanów przy wykorzystaniu informacji klasycznej, natomiast w drugim wykorzystujemy splątanie do przesyłania informacji klasycznej.

Rozdział 6

Dodatek matematyczny

6.1 Maszyny Turinga

Wiadomości na temat maszyn Turinga oraz teorii złożoności można znaleźć w [43, 14]. W [57] omawiane są szczegółowo zagadnienia związane z niedeterminizmem. Temat obliczeń odwracalnych podejmowany jest w [4, 23, 70].

6.1.1 Podstawy modelu

Maszyna Turinga dysponuje bardzo skromnymi środkami obliczeniowymi, lecz pomimo to potrafi wykonać wiele skomplikowanych zadań. Wszystkie operacje maszyny wykonywane są na taśmie, na której znajdują się symbole z pewnego skończonego zbioru Σ. Maszyna w jednym kroku może pobrać za pomocą czytnika jeden symbol $x \in \Sigma$ z taśmy, wykonać (lub nie) ruch nad taśmą i zmienić stan wewnętrzny.

Definicja 22 *Deterministyczną maszyną Turinga nazywamy czwórkę uporządkowaną* $M = (K, \Sigma, \delta, \mathfrak{s})$ *złożoną z:*

- *skończonego zbioru* K *stanów zawierającego stan początkowy* $\mathfrak{s} \in K$
- *skończonego zbioru* Σ *symboli (alfabetu) zawierającego symbol pusty* $\sqcup$ *i symbol końcowy* $\lhd$*.*
- *Funkcji* $\delta\colon K \times \Sigma \to (K \cup \{$ *„stop”*, *„tak”*, *„nie”*$\}) \times \Sigma \times \{\leftarrow, -, \rightarrow\}$

gdzie $\leftarrow, \rightarrow, -$ *oznaczają odpowiednio ruch w prawo, w lewo i pozostanie na miejscu.*

Najprostszym rozszerzeniem tego modelu jest dodanie możliwości operowania na kilku ciągach danych – otrzymujemy wówczas maszynę Turinga z wieloma taśmami.

Ponieważ maszyna Turinga operuje na symbolach, najlepiej zadawać jej pytania dotyczące ciągów symboli. Oznaczmy przez $(\Sigma \setminus \{\sqcup\})^*$ zbiór wszystkich

słów nad $\Sigma \setminus \{\sqcup\}$, czyli skończonych ciągów elementów zbioru Σ. Dowolny jego podzbiór $L \subset (\Sigma \setminus \{\sqcup\})^*$ nazywamy językiem.

Definicja 23 *Mówimy, że maszyna M rozstrzyga język L, jeżeli na każdym słowie $x \in L$ maszyna zatrzymuje się w stanie „tak", a na każdym słowie $y \notin L$ maszyna zatrzymuje się w stanie „nie". Jeżeli jest spełniony tylko ten pierwszy warunek to mówimy, że maszyna rozpoznaje język L.*

Mając dany jakiś problem decyzyjny chcemy wiedzieć jaki jest spodziewany czas otrzymania rozwiązania. Aby wprowadzić porządek do zbioru wszystkich problemów decyzyjnych wprowadza się klasy złożoności. Podział ten zdumiewająco dobrze oddaje złożoność problemów obliczeniowych, z którymi spotykamy się w praktyce.

Dla $x \in L$ oznaczmy przez $|x|$ ilość elementów ciągu x. Mówimy, że maszyna M działa w czasie $f(n)$, jeżeli dla dowolnego słowa wejściowego x czas wymagany przez M jest równy co najwyżej $f(|x|)$. Zbiór $\mathbf{TIME}(f(x))$ zawiera te języki, które są rozstrzygane przez deterministyczną maszynę Turinga z wieloma ciągami[1] w czasie ograniczonym przez $f(x)$. W szczególności

Definicja 24 *Klasa* $\mathbf{P}$ *jest to zbiór języków rozpoznawalnych przez deterministyczną maszynę Turinga w czasie wielomianowym względem rozmiaru słowa wejściowego.*

$$\mathbf{P} = \bigcup_{c \in \mathbb{N}} \mathbf{TIME}(x^c). \tag{6.1}$$

Problemy z klasy $\mathbf{P}$ (ang. *polynomial time*) są zwykle łatwe do rozwiązania, gdyż wymagają niewielkiego nakładu środków obliczeniowych. Oczywiście może zdarzyć się przypadek algorytmu o złożoności wielomianowej, który pomimo to ma w praktyce szybko rosnące wymagania[2], ale takie przypadki są rzadkie.

Przy rozpatrywaniu obliczeń kwantowych przydatna jest jeszcze jedna definicja

Definicja 25 *Odwracalna maszyna Turinga jest to deterministyczna maszyna Turinga, dla której funkcja przejścia δ jest różnowartościowa.*

6.1.2 Niedeterminizm

Aby ująć ważną klasę problemów decyzyjnych wprowadza się do modelu maszyny Turinga uogólnienie związane z niedeterminizmem.

Definicja 26 *Niedeterministyczną maszyną Turinga nazywamy czwórkę uporządkowaną $(K, \Sigma, \gamma, \mathrm{s})$ określoną tak jak w Definicji 22, z tą różnicą, że od γ wymagamy jedynie by była relacją.*

[1] Ponieważ maszyna Turinga z wieloma ciągami może być symulowana przez maszynę z jednym ciągiem z wielomianową stratą wydajności, przyjęcie konkretnego modelu do określenia złożoności nie wpływa na jakościowy podział problemów.

[2] Może się zdarzyć, że algorytm ma złożoność $O(n^{100})$ i wówczas także należy do $\mathbf{P}$.

Od maszyny niedeterministycznej wymagamy mniejszej precyzji niż od maszyny deterministycznej.

Definicja 27 *Mówimy, że maszyna niedeterministyczna M rozstrzyga język L, jeżeli istnieją takie słowa $u, w \in L$, że na każdym słowie $x \in L$ maszyna zatrzymuje się w konfiguracji ($\text{„tak”}, u, w$).*

Niedeterminizm jest obiektem dużego zainteresowania w teorii złożoności. Jedną z najważniejszych klas złożoności jest klasa **NP** (ang. *nondeterministic polynomial time*).

Definicja 28 *Klasa* **NP** *jest to zbiór języków rozpoznawalnych przez niedeterministyczną maszynę Turinga w czasie wielomianowym względem rozmiaru słowa wejściowego. Definując zbiór* $\mathbf{TIME}(f(x))$ *języków, które są rozstrzygane przez niedeterministyczną maszynę Turinga z wieloma ciągami w czasie ograniczonym przez $f(x)$, możemy zapisać*

$$\mathbf{NP} = \bigcup_{c \in \mathbb{N}} \mathbf{NTIME}(x^c). \tag{6.2}$$

Nieformalnie klasę **NP** określa się czasem jako zbiór problemów decyzyjnych dla których odpowiedź „tak” może być sprawdzona na deterministycznej maszynie Turinga w czasie wielomianowym, pod warunkiem dostarczenia pewnej dodatkowej informacji.

Największym, wciąż nie rozstrzygnionym problemem teorii złożoności jest pytanie czy $\mathbf{P} = \mathbf{NP}$.

6.1.3 Algorytmy losowe

Algorytmy losowe można opisać modyfikując sposób w jaki słowo wejściowe jest akceptowane przez niedeterministyczna maszynę Turinga.

Niedeterministyczna maszyna Turinga M jest dokładna jeżeli wszystkie jej obliczenia na słowie $x \in L$ zatrzymują się po takiej samej, wielomianowej liczbie kroków. Załóżmy, że maszyna ta wykonuje w każdym kroku wyboru jednej z dwóch możliwości ruchu.

Definicja 29 *Maszyna M jest wielomianową maszyną Turinga typu Monte Carlo dla języka L, jeżeli na słowie wejściowym $x \in L$ wykonuje ona zawsze $poly(|x|)$ kroków, oraz conajmniej połowa obliczeń maszyny zatrzymuje się w stanie „tak”, o ile $x \in L$ i wszystkie obliczenia zatrzymują się w stanie „nie”, o ile $x \notin l$.*

Oznacza to, iż maszyna Turinga typu Monte Carlo nigdy nie daje złej odpowiedzi pozytywnej, ale z pewnym prawdopodobieństwem może udzielić fałszywej odpowiedzi negatywnej.

Definicja 30 *Klasa złożoności* **RP** *(ang.* randomized polynomial time*) zawiera wszystkie języki, dla których istnieje wielomianowa maszyna Turinga typu Monte Carlo.*

Klasę **RP** $\cap$ **coRP** oznaczamy przez **ZPP** (ang. *zero probability of error*). Algorytmy tej klasy nazywane są algorytmami Las Vegas. Do czasu publikacji [1] uważano, że problem $PRIME$ należy do **ZPP**.

Najszersze realistyczne ujęcie pojęcia obliczana daje nam jednak klasa **BPP** .

Definicja 31 *Klasa* **BPP** *zawiera wszystkie te języki L dla których istnieje niedeterministyczna wielomianowa maszyna Turinga taka, że dla dowolnego słowa wejściowego x przynajmniej $\frac{3}{4}$ ścieżek*[3] *obliczeń akceptuje x jeżeli $x \in L$ i co najmniej $\frac{3}{4}$ ścieżek obliczeń odrzuca x jeżeli $x \notin L$.*

6.1.4 Kwantowa maszyna Turinga

Model kwantowej maszyny Turinga został po raz pierwszy przedstawiony w pracy [17]. Kwantowa maszyna Turinga składa się z:

- Procesora: M 2-stanowych obserwabli $\{n_i | i \in_{\mathbb{Z}_M}\}$.
- Pamięci: nieskończonego ciągu 2-stanowych obserwabli $\{m_i | i \in \mathbb{Z}\}$.
- Obserwabli x – adresu bieżącej pozycji kursora.

Stan maszyny opisany jest wektorem $|\psi(t)\rangle = |x; n_0, n_1, \ldots; m\rangle$ w pewnej przestrzeni Hilberta $\mathcal{H}$.

W chwili $t = 0$ opisany jest on wektorem $|\psi(0)\rangle = \sum_m a_m |0; 0, \ldots, 0; \ldots, 0, 0, 0, \ldots\rangle$ spełniającym warunek

$$\sum_i |a_i|^2 = 1.$$

Ewolucja kwantowej maszyny Turinga zadana jest poprzez operator unitarny U działający na przestrzeni Hilberta $\mathcal{H}$.

Klasyczna niedeterministyczna maszyna Turinga jest to maszyna kwantowa której stan w trakcie przebiegu ewolucji jest zawsze jednym ze stanów bazowych.

Bardziej formalna definicja maszyny Turinga została podana w [5].

Definicja 32 *Oznaczmy przez $\widetilde{\mathbb{C}}$ zbiór tych liczb zespolonych $c \in \mathbb{C}$, dla których istnieje algorytm na deterministyczną maszynę Turinga, pozwalający obliczyć ich* $\mathrm{Re}\,(c)$ *i* $\mathrm{Im}\,(c)$ *z dokładnością $\frac{1}{2^n}$ w czasie wielomianowym względem n.*

Kwantową maszyną Turinga nazywamy trójkę uporządkowaną (K, Σ, δ), gdzie K oraz Σ są określone identycznie jak w Definicji 22, a δ jest odwzorowaniem

$$\delta \colon K \times \Sigma \to \widetilde{\mathbb{C}}^{\Sigma \times K \times \{\leftarrow, -, \rightarrow\}}. \tag{6.3}$$

[3] Liczba $\frac{3}{4}$ może być tu zastąpiona dowolną liczbą z przedziału $(\frac{1}{2}, 1)$

Odwracalne klasyczne maszyny Turinga są szczególnym przypadkiem maszyn kwantowych. Dlatego też kwantowa maszyna Turinga jest zdolna do symulacji klasycznej maszyny Turinga.

Analogicznie do klas złożoności dla maszyn klasycznych można zdefiniować klasy dla maszyn kwantowych. Najważniejsza z nich to **BQP**.

Definicja 33 *Klasa* **BQP** *zawiera wszystkie te języki L dla których istnieje kwantowa wielomianowa maszyna Turinga taka, że dla dowolnego słow wejściowego x przynajmniej $\frac{3}{4}$ ścieżek obliczeń akceptuje $x$jeżeli $x \in L$ i co najmniej $\frac{3}{4}$ ścieżek obliczeń odrzuca c jeżeli $x \notin L$.*

Zależność klasycznych i kwantowych klas złożoność wyraża następujące

Twierdzenie 14 *Pomiędzy klasami probabilistycznymi zachodzi relacja.*

$$\mathbf{BPP} \subseteq \mathbf{BPP}.$$

Dowód tego fakty można znaleźć w [5].

6.2 Entropia i informacja

Bazując na formalizmie macierzy gęstości można zbudować analogiczną do klasycznej kwantową teorię informacji [12]. W teorii informacji sformułowanej przez Claude'a Shanona podstawowym pojęciem jest entropia Gibbsa[4] określona na przestrzeni rozkładów prawdopodobieństwa. W sformułowaniu kwantowym jej miejsce zajmuje entropia von Neumana na przestrzeni stanów układu kwantowego.

6.2.1 Entropia Gibbsa-Shanona

Miarą informacji przekazywanej przez sygnał, w którym z prawdopodobieństwami $p_1, \ldots, p_n$ występują znaki $A_1, \ldots, A_n$, jest entropia informacyjna [68]

Definicja 34 (Entropia informacyjna Gibbsa-Shanona)

$$H(p) = -\sum_{i=1}^{n} p_i \log p_i. \tag{6.4}$$

6.2.2 Entropia von Neumana

Odpowiednikiem entropii informacyjnej w fizyce kwantowej jest entropia von Neumana

[4] W kontekście teorii informacji entropia Gibbsa bywa nazywana entropią Gibbsa-Shanona

Definicja 35 (Entropia von Neumana)

$$S(\rho) = -\mathrm{Tr}\,(\rho)\log\rho. \tag{6.5}$$

Jeżeli rozpatrzymy diagonalną macierz gęstości, to definicja ta sprowadza się do wyrażenia klasycznego.

Entropia von Neumana posiada kilka ważnych własności:

1. Entropia jest nieujemna i jest równa zeru wtedy i tylko wtedy, gdy stan jest czysty.

2. W n-wymiarowej przestrzeni Hilberta entropia osiąga maksimum dla stanu maksymalnie mieszanego $\frac{1}{n}\mathbb{I}$ i wynosi wówczas $\log n$.

3. Jeżeli stan ρ układu złożonego $A+B$ jest stanem czystym, to $S(\mathrm{Tr}_A\,(\rho)) = S(\mathrm{Tr}_B\,(\rho))$.

Dowód.

Własność 1 wynika z określenia funkcji S.

Własność 2 udowadnia się wykorzystując kwantową entropię względną [5], dla której słuszne jest

Twierdzenie 15 (Nierówność Kleina) *Kwantowa entropia względna jest nieujemna*

$$S_{rel}\,(\rho||\sigma) \geq 0, \tag{6.6}$$

przy czym równość zachodzi wtedy i tylko wtedy, gdy $\rho = \sigma$

Na podstawie nierówności 6.6 możemy dla dowolnego stanu $\rho \in \mathcal{S}\,(\mathcal{H})$ zapisać:

$$\begin{aligned} 0 \leq S_{rel}(\rho||\frac{1}{n}\mathbb{I}) &= \mathrm{Tr}\,(()\,\rho\log\rho) - \mathrm{Tr}\,(()\,\rho\log\mathbb{I}n) \\ &= -S(\rho) - \log n, \end{aligned}$$

co oznacza, że

$$\bigwedge_{\rho\in\mathcal{S}(\mathcal{H})} S(\rho) \leq \log n, \tag{6.7}$$

gdzie $n = \dim\mathcal{H}$.

Z rozkładu Shmidta wynika, że operatory zredukowane mają takie same wartości własne. Ponieważ entropia von Neumana jest określona poprzez widmo operatora gęstości, otrzymujemy własność 3.

[5] Definicja 1.31 na stronie 24

6.3 Wiadomości z teorii liczb

Zebrane tutaj wiadomości na temat teorii liczb i związanych z nią algorytmów można znaleźć w [52, 73]. Najlepiej znane algorytmy faktoryzacji są przedstawione w artykule przeglądowym [50]. Dyskusja wyników z teorii liczb wykorzystanych w algorytmie szybkiej faktoryzacji znajduje się w [21].

6.3.1 Podstawowe twierdzenia

Twierdzenie 16 (Podstawowe Twierdzenie Arytmetyki) *Każda liczba naturalna n może być przedstawiona w postaci iloczynu potęg liczb pierwszych*

$$n = p_1^{k_1} p_2^{k_2} \dots p_l^{k_l},$$

i rozkład ten jest jednoznaczny z dokładnością do porządku czynników.

Twierdzenie 17 (Małe twierdzenie Fermata) *Jeżeli p jest liczbą pierwszą oraz $\gcd(a,p) = 1$, to*

$$a^{p-1} = 1 (\bmod p).$$

Określamy funkcję ϕ-Eulera $\phi(a)$ jako moc zbioru liczb całkowitych dodatnich mniejszych od a, względnie pierwszych z a.

Definicja 36

$$\phi(a) \quad = \quad \{x \in \mathbb{N} | \gcd(x,a) = 1 x < a\}.$$

Poniższe twierdzenie stanowi uogólnienie małego twierdzenia Fermata.

Twierdzenie 18 (Eulera) *Jeżeli $\gcd(a,b) = 1$ to*

$$a^{\phi(b)} = 1 (\bmod p). \tag{6.8}$$

Wykonanie algorytmu Simona dla maski niezmienniczości względem sumy modulo 2 wymaga rozwiązania układu równań.

Twierdzenie 19 (Chińskie twierdzenie o resztach) *Jeżeli liczby $m_1, m_2, \dots, m_n$ są parami względnie pierwsze, to układ równań*

$$\begin{aligned} x &= a_1 \bmod m_1 \\ x &= a_2 \bmod m_2 \\ \dots \quad &\dots \quad \dots \\ x &= a_n \bmod m_n \end{aligned}$$

ma wówczas jednoznaczne rozwiązanie zadane wzorem

$$x = (a_1 N_1 M_1 + a_2 N_2 M_2 + \ldots + a_2 N_n M_n) \bmod M, \tag{6.9}$$

gdzie $M = m_1 m_2 \dots m_n$, $M_i = \frac{M}{m_i}$, $N_i = M_i^{-1} \bmod m_i$.

6.3.2 Algorytm Euklidesa

Znany od przeszło 2000 lat algorytm Euklidesa pozwala na sprawdzenie czy dane dwie liczby całkowite są względnie pierwsze.

Poniższa implementacja w języku `C` jest oparta na fakcie, iż $\gcd(a, b) = \gcd(a - b, b)$. Pozwala to na rekurencyjne wywołanie funkcji.

Listing 6.1: Przykład implementacji algorytmu Euklidesa

```
#include <stdio.h>

int gcd(int a, int b);

main(int argc,char *argv[]){

    int x;
    int a,b;
    if(argc!=3) {
        printf("Za malo argumentow");
        return(1);
    }

    a=atoi(argv[1]);
    b=atoi(argv[2]);

    x=gcd(a,b);
    printf("%d\n",x);
    return(0);
}

int gcd(int a, int b){
    int wynik;

    if(b==0) wynik=a;
    else if (b>a)
        wynik=gcd(b,a);
    else wynik=gcd(a-b,b);
    return wynik;
}
```

6.3.3 Efektywne testowanie liczb pierwszych

W roku 2002 jednym z najciekawszych doniesień z zakresu teorii liczb było opublikowanie w sierpniu deterministycznego, efektywnego algorytmu pozwalającego na rozstrzygnięcie, czy zadana liczba jest liczbą pierwszą czy złożoną [1].

Dotychczas najbardziej efektywne algorytmy testowania pierwszości były algorytmami probabilistycznymi. Przykładowo w programie *Mathematica* od wersji 2.2, funkcja `PrimeQ` wykorzystuje efektywny, probabilistyczny algorytm Rabina-Millera.

Udowodniono, że złożoność nowego algorytmu wynosi $O(ln^{12}n)$ przy testowaniu liczby n. Jednak pewne przypuszczenia pozwalają na oszacowanie tego ograniczenia na $O(ln^6 n)$, i prawdopodobnie uda się je zredukować do $O(ln^3 n)$.

W związku z możliwością efektywnego testowania, zaczęto oczekiwać efektywnego algorytmu pozwalającego na faktoryzację.

6.3.4 Ułamki łańcuchowe

Ułamki łańcuchowe [52] pozwalają na przybliżanie liczb niewymiernych za pomocą liczb wymiernych.

Definicja 37 *Jeżeli* $a_1, \ldots, a_n > 0$ *to skończony ciąg liczb rzeczywistych* $\langle a_0; a_1, \ldots, a_n \rangle$ *nazywamy ułamkiem łańcuchowym.*

Liczbę

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots}}} \tag{6.10}$$

nazywamy wartością ułamka $\langle a_0; a_1, \ldots, a_n \rangle$ i oznaczamy ją przez $[a_0; a_1, \ldots, a_n]$

Twierdzenie 20 *Każda liczba wymierna a jest wartością dwóch ułamków łańcuchowych skończonych* $\langle a_0; a_1, \ldots, a_n \rangle$ *oraz* $\langle a_0; a_1, \ldots, a_n \rangle$*. Do obliczenia rozwinięcia liczby a w postaci ułamka ciągłego służy następująca relacja rekurencyjna*

$$\begin{aligned} \alpha_0 &= a, \alpha_{k+1} = \frac{1}{\alpha_k - [\alpha_k]} \\ a_k &= [\alpha_k]\,. \end{aligned}$$

Program na Listingu 6.2 pozwala na obliczenie rozwinięcia zadanej liczby na ułamek łańcuchowy.

Listing 6.2: Program obliczający rozwinięcie liczby na ułamek łańcuchowy

```
#include <cmath>
#include <iostream>
using namespace std;

int main(int argc, char *argv[]){
  float war;
  int dl;
```

```
  float *alpha;
  long *wynik;
  cout << "Podaj wartosc:\t";
  cin >> war;
  cout << "Podaj dlugosc rozwiniecia:\t";
  cin >> dl;
  alpha = new float[dl];
  wynik = new long[dl];
  alpha[0] = war;
  wynik[0] = (long) floor(alpha[0]);

  for(int i=1;i<dl;i++){
    alpha[i] = 1/(alpha[i-1]-wynik[i-1]);
    wynik[i] = (long) floor(alpha[i]);
  }
  cout << war << " = [";
  for(int i=0;i<dl;i++){
    cout << wynik[i] << " ";
  }
  cout << "]";

  delete alpha;
  delete wynik;
  return(0);
}
```

6.3.5 Faktoryzacja a znajdowanie rzędu

Wiadomo, że zagadnienie znalezienia dzielnika liczby N można rozwiązać, jeżeli można rozwiązać zagadnienie znajdowania rzędu elementu m spełniającego warunek $\gcd(m, N) = 1$ (tj. względnie pierwszego z N) w grupie $(Z_N, \cdot)$. Fakt ten jest wykorzystywany w konstrukcji kwantowego algorytmu Shora.

Oznaczmy $\mathrm{ord}_N(a) = r$. Jeżeli r jest liczbą parzystą, to możemy zapisać

$$(a^r - 1) = (a^{\frac{r}{2}} + 1)(a^{\frac{r}{2}} - 1) = 0 (\mathrm{mod}\ N). \tag{6.11}$$

Ponieważ liczba N dzieli $a^r - 1$, musi ona mieć wspólne dzielniki bądź z $a^{\frac{r}{2}} + 1$, bądź z $a^{\frac{r}{2}} - 1$. Wykorzystując algorytm Euklidesa możemy otrzymać te dzielniki, o ile N nie dzieli obu tych liczb.

N nie może dzielić $a^{\frac{r}{2}} - 1$ na mocy definicji rzędu. Okazuje się, że dla losowo dobranego a prawdopodobieństwo koniunkcji

$$2 \not| r = \mathrm{ord} \wedge (a) a^{\frac{r}{2}} \neq -1 (\mathrm{mod}\ N) \tag{6.12}$$

wynosi co najmniej $\frac{9}{16}$ [31].

6.3.6 Logarytmy dyskretne

Logarytmy dyskretne stanowią podstawę algorytmu Diffie-Hellman wymiany klucza. Algorytm ten jest algorytmem publicznym i bazuje on na założeniu, że logarytmy dyskretne są trudne do obliczenia, szczególnie dla pewnych grup. Sposób wymiany klucza za pomocą tego protokołu został przedstawiony w Rozdziale 6.6.2

Definicja 38 *Niech G będzie grupą. Dla elementu $g \in G$ n-tą potęgą dyskretną g nazywamy element* $\underbrace{g \cdot g \cdot \ldots \cdot g}_{n-\text{razy}}$.

Operacja odwrotna polega na znalezieniu wykładnika n, jeżeli znamy $h = g^n$ oraz g.

Definicja 39 *Logarytmem dyskretnym liczby h w grupie G przy podstawie g nazywamy liczbę n taką, że*

$$g^n = h(\bmod p).$$

Zazwyczaj n spełnia warunek $0 \leq n \leq |\langle g \rangle|$. Jeżeli grupa jest cykliczna, a g jest jej generatorem, to oznaczając $n = \log_g h$ otrzymujemy tożsamość

$$\log_g(h_1 + h_2) = \log_g h_1 + \log_g h_2 \pmod{|G|}.$$

Przykładem grup cyklicznych są grupy $(\mathbb{Z}_p, \cdot)$ dla p będącej liczbą pierwszą.

6.4 Transformata Fouriera

6.4.1 Konstrukcja transformaty Fouriera

Przedstawioną tutaj konstrukcję transformaty Fouriera na grupie abelowej można znaleźć w [31] oraz w [44, 37].

Niech G będzie skończoną grupą abelową. Szukamy zupełnego ortogonalnego układu w przestrzeni funkcji zespolonych na G. Możemy go uzyskać wykorzystując charaktery reprezentacji nieredukowalnych. Ponieważ dla grup abelowych wszystkie reprezentacje nieredukowalne są jednowymiarowe, możemy przyjąć następującą definicję:

Definicja 40 *Charakterem na grupie G nazywamy funkcję $\chi : G \to \mathbb{C} \setminus \{0\}$ spełniającą warunek $\chi(g_1 + g_2) = \chi(g_1)\chi(g_2)$*

Można zanotować następujące własności charakterów

1. $\chi(0) = 1$.
2. $\chi(g)^n = \chi(ng)$ gdzie $n = |G|$ to rząd grupy. Zatem wartości charakterów są pierwiastkami z jedności.

3. Charaktery tworzą grupę względem działania $\chi_1\chi_2(g) = \chi_1(g)\chi_2(g)$. Grupę tę nazywamy grupą dualną i oznaczamy przez $\widehat{G}$.

Dla dowolnej grupy abelowej zachodzi następujące twierdzenie

Twierdzenie 21 *Niech $G = G_1 \oplus \ldots \oplus G_m$ będzie rozkładem grupy abelowej G na sumę prostą jej podgrup cyklicznych. Wówczas $G \simeq \widehat{G}$ oraz $\chi(g) = \chi_1(g_1)\ldots\chi_m(g_m)$.*

Przykład 1 *Dla grupy $\mathbb{Z}_n$ definiujemy $\chi_y(x) = e^{\frac{2i\pi xy}{n}}$.*

W przestrzeni funkcji zespolonych na skończonej grupie abelowej G charaktery tworzą bazę ortogonalną. Możemy łatwo wprowadzić bazę ortonormalną

$$B_i = \frac{1}{\sqrt{n}}\chi_i$$

Wówczas dowolna funkcja $f\colon G \to \mathbb{C}$ ma rozwinięcie w tej bazie

$$f = \frac{1}{\sqrt{n}}\sum_{i=1}^{n}\widehat{f}_i\chi_i$$

To prowadzi do następującej definicji transformaty Fouriera

Definicja 41 *Transformatą Fouriera funkcji $f : G \to \mathbb{C}$ nazywamy odwzorowanie $\widehat{f}\colon G \to \mathbb{C}$ zdefiniowane wzorami*

$$\widehat{f}(g_i) = \widehat{f}_i$$

Ponieważ $\langle B_i|f\rangle = \widehat{f}_i$ możemy zapisać

$$\widehat{f}(g_i) = \frac{1}{\sqrt{n}}\sum_{k=1}^{n} f_i(g_k)\chi_i^*(g_k) \tag{6.13}$$

Przykład 2 *Charaktery grupy $\mathbb{B}^m$ mają postać $\chi_y(x) = (-1)^{x\cdot y}$. Zatem transformata Fouriera na tej grupie to*

$$\widehat{f}(x) = \frac{1}{\sqrt{2^m}}\sum_{y\in\mathbb{B}^m}(-1)^{x\cdot y}f(y)$$

Transformata Fouriera na $\mathbb{B}^m$ jest nazywana transformatą Walsha-Hadamarda.

6.4.2 Periodyczność

Poniższy przykład ilustruje, w jaki sposób transformata Fouriera pozwala na wydobycie informacji na temat periodyczności funkcji. Własność ta jest wykorzystywana przy konstrukcji kwantowych algorytmów ukrytej podgrupy.

Niech $f\colon G \to \mathbb{C}$ spełnia warunek $\bigvee_{p\in G} \bigwedge_{g\in G} f(g+p) = f(g)$. Wówczas

$$\begin{aligned}
\widehat{f}(g_i) &= \frac{1}{\sqrt{n}} \sum_{k=1}^{n} f(g_k)\chi_i^*(g_i) \\
&= \frac{1}{\sqrt{n}} \sum_{k=1}^{n} f(g_k+p)\chi_i^*(g_i+p-p) \\
&= \chi_i^*(-p)\frac{1}{\sqrt{n}} \sum_{k=1}^{n} f(g_k+p)\chi_i^*(g_i+p) \\
&= \chi_i^*(-p)\widehat{f}(g_i).
\end{aligned}$$

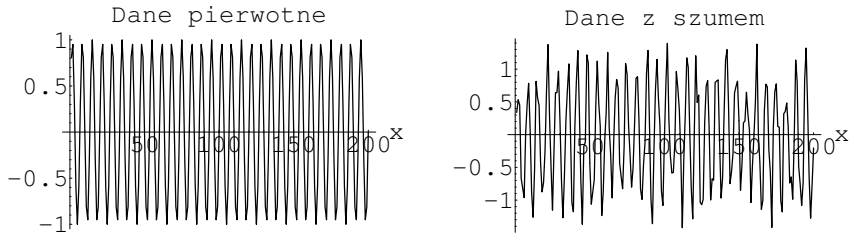
Ponieważ $\chi(-g) = \chi^*(g)$ implikuje to, iż $\widehat{f}(g_i) = 0$, o ile $\chi_i(p) \neq 1$.

Przykład 3 *Powyższe rozważania dla grupy $\mathbb{Z}_n$ prowadzą do równości*

$$\widehat{f}(x) = e^{-\frac{2i\pi xp}{n}}\widehat{f}(x)$$

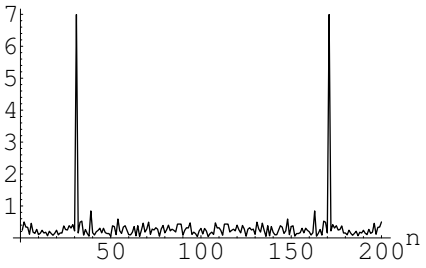
Oznacza to iż $\widehat{f}(x) = 0$ o ile $e^{-\frac{2i\pi xp}{n}} \neq 1$. Ta sytuacja zachodzi, gdy xp nie jest podzielne prze n. Jeżeli dodatkowo $\gcd(p,n) = 1$*, to jedyna możliwość, żeby n dzieliło xp zachodzi, gdy x jest podzielne przez n.*

Praktycznym zastosowaniem właściwości transformaty Fouriera jest odzyskanie informacji na temat okresu z zakłóconych danych [78].
Niech dana będzie tablica danych `data` zawierająca wartości funkcji okresowej z dodanym szumem losowym, generowanym w tym przypadku przy użyciu funkcji `Random[]` dostępnej w pakiecie *Mathematica*. Oryginalny sygnał zadany



Rysunek 6.1: Działanie DFT. Dane oryginalne i dane z szumem

funkcją $\sin(\frac{30*2*\pi}{200}n)$ oraz dane zniekształcone zobrazowane są na Rysunku 6.1.



Rysunek 6.2: Odczytanie periodyczności przy pomocy DFT

Dyskretna transformata Fouriera sygnału zakłóconego przedstawiona jest na Rysunku 6.2.

Silne maksima dla $n = 31$ oraz dla $n = 201 - 30$ na Rysunku 6.2 pozwalają odczytać częstotliwość oryginalnego sygnału.

6.4.3 Algorytm FFT

Algorytm szybkiej transformaty Fouriera (ang. *fast Fourier transform*) został rozpowszechniony w połowie lat sześćdziesiątych dzięki pracom J. W. Cooleya i J. W. Tukeya, wiadomo jednak, że początki tego algorytmu sięgają prac C. F. Gaussa z roku 1805. Konstrukcję algorytmu FFT można znaleźć w [61], a omówienie znaczenia w kontekście teorii grup w [46].

Dyskretna transformata Fouriera wektora danych $\underline{a} = (a_0, \ldots, a_{n-1}) \in \mathbb{C}^n$ to wektor $\hat{\underline{a}} \in \mathbb{C}^n$ o współrzędnych $(\hat{a}_0, \ldots, \hat{a}_{n-1})$, zadanych wzorami

$$\hat{a}_k = \sum_{j=0}^{n-1} \omega^{jk} a_j, \ k = 0, \ldots, n-1, \tag{6.14}$$

gdzie $\omega = \exp\left(\frac{2i\pi}{n}\right)$, a czynnik $\frac{1}{\sqrt{n}}$ został pominięty[6].

Zatem otrzymanie wektora $\hat{\underline{a}}$ odpowiada wymnożeniu wektora wejściowego przez macierz zespoloną $n \times n$ o elementach $M_{k,j} = \omega^{kj}$. Mnożenie takie wymaga n^2 operacji na liczbach zespolonych, czyli złożoność czasowa algorytmu wynosi $O(n^2)$ i jest wykładnicza względem rozmiaru danych.

Przyjmijmy[7], że $n = 2^m$ dla pewnego $m \in \mathbb{N}$. Możemy wówczas zastosować procedurę polegającą na podzieleniu szeregu 6.14 na dwa szeregi złożone

[6] W zależności od zastosowań istnieją różne konwencje dotyczące czynnika normalizacyjnego [6].

[7] Możliwe jest zdefiniowanie algorytmu FFT dla innych rozmiarów danych, ale najlepiej sprawdza się on dla liczb będących potęgą dwójki.

z wyrazów o numerach parzystych i nieparzystych.

$$\begin{aligned}
\sum_{j=0}^{n-1} \omega^{jk} a_j &= \sum_{j=0}^{n/2-1} \omega^{2jk} a_{2j} + \sum_{j=0}^{n/2-1} \omega^{k(2j+1)} a_{2j+1} \\
&= \sum_{j=0}^{n/2-1} \exp\left(\frac{2\pi i 2jk}{n}\right) a_{2j} + \sum_{j=0}^{n/2-1} \exp\left(\frac{2\pi i (2j+1)k}{n}\right) a_{2j+1} \\
&= \sum_{j=0}^{n/2-1} \exp\left(\frac{2\pi i jk}{n/2}\right) a_{2j} + \omega^k \sum_{j=0}^{n/2-1} \exp\left(\frac{2\pi i jk}{n/2}\right) a_{2j+1} \\
&= \hat{a}_k^e + \omega^k \hat{a}_k^o
\end{aligned}$$

przy czym $\hat{a}_k^e$ i $\hat{a}_k^o$ oznaczają k-te składowe wektorów otrzymanych poprzez przetransformowanie wektorów o długości $n/2$, złożonych odpowiednio ze składowych wektora $\underline{a}$ o numerach parzystych (ang. *even*) i nieparzystych (ang. *odd*) odpowiednio.

Stosując ten podział rekurencyjnie otrzymujemy w ostatnim kroku transformatę jednopunktową

$$\hat{a}_k^{ooeo\ldots ooeo} = f_x,\ k = 0, \ldots, n-1$$

dla pewnego $x \in \{0, \ldots, n-1\}$. Wartość x otrzymujemy podstawiając odpowiednio $e = 0$ oraz $o = 1$ i odwracając kolejność cyfr binarnych w otrzymanym ciągu.

Przykład 4 *Niech* $\underline{a} = (a_0, a_1, a_2, a_3)$. *Wówczas* $\omega = i$ *oraz*

$$\begin{aligned}
\hat{a}_k &= \hat{a}_k^e + \omega^k \hat{a}_k^o \\
&= \hat{a}_k^e e + \omega^k \hat{a}_k^e o + \omega^k \hat{a}_k^o e + \omega^{2k} \hat{a}_k^o o \\
&= a_0 + \omega^k a_2 + \omega^k a_1 + \omega^{2k} a_3
\end{aligned}$$

Dla $\underline{a} = (1, 0, 0, 0)$ *dostajemy*

$$\hat{a} = (1, 1, 1, 1)$$

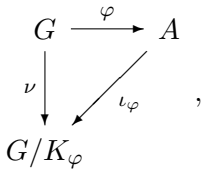
Przykładową implementację opisanej procedury w języku `C` można znaleźć w [61].

Wykonanie szybkiej transformaty Fouriera jest zagadnieniem kluczowym dla wielu zagadnień [6] z dziedzin takich jak cyfrowa obróbka dźwięku czy przetwarzanie obrazów. Zadanie stworzenia biblioteki funkcji potrzebnych do jak najlepszego wykorzystania potencjału maszyny podczas wykonywania tej transformaty wymaga indywidualnego podejścia dla różnych typów procesorów [24].

6.5 Problem ukrytej podgrupy

Opis algorytmów Simona i Shora w języku teorii grup można znaleźć w [44].

Definicja 42 *Mówimy, że odwzorowanie $\varphi\colon G \to A$ ma strukturę ukrytej podgrupy, jeżeli istnieje podgrupa K_φ grupy G (nazywana ukrytą podgrupą) oraz injekcja $\iota_\varphi\colon G/K_\varphi \to A$ (nazywana ukrytą injekcją) taką, że poniższy diagram komutuje.*



gdzie $\nu\colon G \to G/K_\varphi$ jest naturalnym odwzorowaniem grupy G na zbiór warstw prawostronnych.

Problem ukrytej podgrupy formułuje się w następujący sposób:

Mając dane odwzorowanie $\varphi\colon G \to A$ o strukturze ukrytej podgrupy, znajdź ukrytą podgrupę K_φ grupy G.

Algorytm rozwiązujący ten problem nazywamy algorytmem ukrytej podgrupy (ang. *hidden subgroup algorithm* – ***HSA***)

Realizacją algorytmu ukrytej podgrupy na komputerze kwantowym identyfikujemy elementy grupy G oraz zbioru A z elementami baz ortogonalnych w przestrzeniach Hilberta $\mathcal{H}_G$ oraz $\mathcal{H}$. Funkcja φ jest przedstawiana za pomocą transformacji unitarnej U_φ działającej na przestrzeni $\mathcal{H}_\mathcal{G} \otimes \mathcal{H}_\mathcal{A}$ w następujący sposób

$$U_\varphi|a\rangle|e\rangle = |a\rangle|\varphi(a)\rangle$$

gdzie e jest elementem neutralnym grupy G.

6.6 Wybrane protokoły kryptograficzne

Największe zainteresowanie algorytmami kwantowymi związane jest z potencjalną możliwością łamania za ich pomocą szyfrów. Poniżej przedstawione są dwa protokoły kryptograficzne często spotykane przy okazji omawiania potencjalnych zastosowań algorytmu Shora. Więcej wiadomości na temat zastosowań kryptografii i jej podstaw matematycznych można znaleźć w [42]. W [64] można znaleźć przykłady implementacji niektórych z metod kryptograficznych.

6.6.1 Algorytm RSA

W tym podrozdziale opisany jest algorytm tworzenia klucza, szyfrowania oraz deszyfrowania w systemie RSA. Procedura tworzenia podpisu cyfrowego jest

identyczna z deszyfrowaniem, a weryfikacja jest identyczna z szyfrowaniem. Prostota i przejrzystość tych funkcji są niewątpliwymi zaletami tego systemu.

System RSA został opracowany w 1977 w Massachusetts Institute of Technology. Nazwa pochodzi od nazwisk jego twórców: Rona Rivesta, Adi Shamira oraz Lena Adlemana. Jest on podstawą wielu systemów bezpiecznej komunikacji – między innymi opisanych w [59, 56, 26]. Dostępny jest pakiet funkcji dla programu *Mathematica* [38] pozwalający na operowanie algorytmem RSA.

Pierwszy etap algorytmu kończy się wygenerowaniem pary kluczy – klucza publicznego i klucza prywatnego.

1. Tworzenie klucza
   (a) Wygeneruj dwie (różne) liczby pierwsze p oraz q i oblicz ich iloraz $N = pq$.
   (b) Oblicz funkcję Eulera $\phi(N)$, $m = \phi(N) = \phi(pq) = (p-1)(q-1)$
   (c) Znajdź losową liczbę e spełniającą warunek $\gcd(e, m) = 1$
   (d) Oblicz d taką, że $ed = 1 (\bmod m)$ czyli $d = e^{-1} (\bmod m)$
   (e) Para (d, N) stanowi klucz prywatny, natomiast para (e, N) może być publicznie ogłoszona i użyta do szyfrowania.
2. Szyfrwanie polega na wykonaniu działania
   (a) Daną wiadomość M podziel na bloki M_i mające jednoznaczną reprezentację $\bmod N$
   (b) Oblicz $C_i = M_i^e (\bmod N)$
3. Deszyfrowanie
   (a) Oblicz $D_i = C_i^d (\bmod N)$.
   Ponieważ $D_i = C_i^d = (M_i^e)^d = M_i^{de} (\bmod N) = M_i^{1+k\phi(n)}$ dla pewnej liczby $k \in \mathbb{N}$. Na podstawie twierdzenia Eulera wnioskujemy o poprawności deszyfrowania.

6.6.2 Protokół Diffiego-Hellmana ustalania klucza

Protokół Diffiego-Hellmana ustalania klucza został opublikowany w 1976 roku. Może on służyć do transmisji klucza przez niezabezpieczony kanał bez jakiejkolwiek wcześniejszej wymiany tajnych danych.

Powiedzmy że Ania i Bartek chcą ustalić klucz którego będą używali do szyfrowania transmisji pomiędzy sobą. Aby wykorzystać do tego protokół Diffiego-Hellmana muszą oni wykonać następujące kroki:

1. Ustalić liczbę pierwszą p oraz generator g grupy $\mathbb{Z}_p$.
2. Wygenerować losowe liczby a dla Ani i b dla Bartka, przy czym $1 \leq a, b \leq p-2$. Są to ich klucze prywatne.

3. Wygenerować klucze publiczne $g^a (\bmod p)$ dla Ani i $g^b (\bmod p)$ dla Bartka.

4. Przesłać sobie nawzajem wartości kluczy publicznych.

5. Obie strony uzyskują w ten sposób liczbę $k = (g^a)^b = (g^b)^a = (g^{ab} (\bmod p))$.

Protokół ten bazuje na założeniu, iż nie jest zadaniem łatwym obliczenie g^{ab} gdy znane są jedynie liczby g^a oraz g^b. Jeżeli logarytmy dyskretne mogą być łatwo obliczane, to założenie to nie jest spełnione. Otwartym problemem jest natomiast pytanie, czy prawdziwa jest implikacja w druga stronę.

Rozdział 7

Oprogramowanie

W ciągu ostatnich lat powstało wiele programów służących do symulacji pracy maszyn kwantowych. Poniżej opisane są dwa przykłady – język programowania QCL oraz pakiet `QuCalc`. Więcej informacji na temat dostępnego oprogramowania można znaleźć w [74].

7.1 Pakiet QuCalc

Obliczenia wykonane w ramach tej pracy zostały wykonane z wykorzystaniem pakietu `QuCalc`. Jest to biblioteka funkcji przeznaczonych dla programu *Mathematica* w wersji 4 lub wyższej. Pakiet został stworzony przez Paula Dumais'a w Laboratorium Informatyki Teoretycznej i Kwantowej Uniwersytetu w Montrealu. Informacje o nowych wersjach oraz przykłady wykorzystania pakietu znaleźć można w [20]

7.1.1 Typy danych i funkcje dostępne w pakiecie

Poniżej wymienione są najistotniejsze funkcje udostępniane przez pakiet `QuCalc`. Pozwalają one na operowanie abstrakcyjnymi typami danych wykorzystywanymi w mechanice kwantowej.

Typy danych

1. `ens` - zespół statystyczny
2. `ket` - stan czysty
3. `schmidt` - rozkład Schmidta stanu czystego układu dwuskładnikowego
4. `state` - macierz gęstości,
5. `supop` - superooperator,
6. `unit` - operacja unitarna,

7. `sqo` - (ang. *selective quantum operation*) struktura danych pozwalająca na reprezentowanie superoperatorów oraz miar spektralnych.

Wszystkim wymienionym powyżej typom danych, za wyjątkiem typu `schmidt`, odpowiadają funkcje *nazwa-typu*`Q` sprawdzające czy dany obiekt jest odpowiedniego typu.

Funkcje

1. `entropy[q]` - oblicza entropię von Neumana stanu q
2. `trout[k, l, m]` - operacja wykonania śladu częściowego względem przestrzeni Hilberta o wymiarze `l` pomiędzy dwiema przestrzeniami Hilberta o wymiarach `k` i `m`. Działanie tej funkcji ilustruje poniższy przykład:

   ```
   In[1]:= trout[2, 2, 1].ket["00"]
   Out[1]:= {{1,0},{0,0}}
   ```

3. `fgate[m, n, f]` - zwraca obiekt typu `unit` odpowiadający transformacji unitarnej na $m + n$ qubitach, która jest równoważna funkcji $f\colon 2^m \to 2^n$
4. `gate[m, f]` - zwraca obiekt typu `unit` odpowiadający transformacji unitarnej odpowiadającej funkcji `m`-bitowej `f`.

agi

Pozostałe funkcje pakietu to między innymi

```
anc             band            bits            block
bnot            bor             bscal           bxor
circuit         cycle           ctrl            dag
dotexp          eigen           eigenVal        eigenVect
fidelity        fourier unvec           vec
id              kron            krondiv swap
kronexp         ktrl            lvNorm          lvProd
maxmix          phase           randomUnit      rotx
roty            rotz
```

Pozwalają one na zapis dowolnego układu bramek.

Stałe Wśród predefiniowanych stałych pakietu `QuCal` znaleźć można najczęściej wykorzystywane w mechanice kwantowej stany oraz operacje.

1. Stałe typu `ket`
   (a) `phim`, `phip`, `psim`, `psip` - stany Bella
2. Stałe typu `unit`

(a) `cnot` - bramka $CNOT$

(b) `knot` - bramka $CNOT^{-1}$

(c) `not` - bramka NOT;

(d) `sigx`, `sigy`, `sigz` - macierze Pauliego

(e) `wh` - transformata Walsha-Hadamarda

3. Inne stałe

(a) `mm` - stała typu `supop` odpowiadająca pomiarowi w bazie $\{|0\rangle, |1\rangle\}$

(b) `mmm` - stała typu `sqo` odpowiadająca pomiarowi w bazie $\{|0\rangle, |1\rangle\}$

7.1.2 Funkcje dodatkowe

Na stronie internetowej [48] znajduje się pakiet zawierający zaimplementowane w języku programu *Mathematica* funkcje pozwalające na obliczanie dwóch miar splątania – ujemności oraz zredukowanej entropii von Neumana. Zostały one oparte na funkcjach pakietu `QuCalc` oraz standardowych funkcjach dostępnych w programie *Mathematica*. Wykorzystanie tych funkcji jest możliwe poprzez załadowanie pakietu `Entanglement`.

```
In[1]:= <<Entanglement`
```

Pakiet dostarcza jedynie podstawowych definicji pomocniczych

```
baseKet[x_, d_]
stateToMatrix[x_, w_]
partTransA[x_state, d1_Integer, d2_Integer]
pureEntanglement[w1_Integer, w2_Integer, ket_ket]
```

oraz dwóch definicji pozwalających na obliczanie miar splątania

```
pureEntanglement[w1_Integer, w2_Integer, ket_ket]
negativ[x_state, d1_Integer, d2_Integer]
```

Parametry całkowite zaczynające się od w odpowiadają wymiarowi przestrzeni, a zaczynające się od d – liczbie qubitów. Więcej informacji można uzyskać wpisując polecenie

```
In[2]:= ?Entanglement
```

7.2 Język programowania QCL

Język programowania komputerów kwantowych (ang. *Quantum Computer Language*) został stworzony przez Bernarda Ömera [55] na Politechnice Wiedeńskiej. W skład pakietu wchodzi symulator komputera kwantowego (biblioteka libqc) oraz interpreter języka QCL. Całość dostępna jest w postaci źródłowej

lub jako pakiet binarny. Powstały także wersje interpretera działające w systemie Microsoft Windows [80] oraz [47] oraz symulator współpracujący za pośrednictwem sieci z interpreterem QCL i pozwalający na wizualną generację programów w języku QCL [25].

Język QCL pozwala na zapis operacji kwantowych w postaci zaczerpniętej z języka `C`. Posiada on takie znane z `C` elementy jak: instrukcja warunkowa `if...then` oraz pętla `for`. Umożliwia wykonywanie procedur klasycznych (dodawanie, potęgowanie) oraz kwantowych (bramka CNOT, bramka Hadamarda). Dostępna jest napisana w QCL-u biblioteka obejmująca między innymi algorytm faktoryzacji oraz algorytm wyszukiwania.

Na Listingu 7.1 zamieszczony jest fragment kodu źródłowego w języku QCL. Program ten realizuje kwantową transformatę Fouriera. Jest to wersja procedury QFT rozprowadzana wraz z pakietem QCL.

Listing 7.1: Przykład programu w języku QCL

```
// Author: Bernhard mer
// Reindorfgasse 35/1/5
// A-1150 Wien
// AUSTRIA
// e-mail: oemer@tph.tuwien.ac.at

// pseudo classic operator to swap bit order

cond qufunct flip(qureg q) {
  int i;                  // declare loop counter
  for i=0 to #q/2-1 {     // swap 2 symetric bits
    Swap(q[i],q[#q-i-1]);
  }
}

// discrete Fourier transform (Coppersmith)

operator dft(qureg q) { // main operator
  const n=#q;             // set n to length of input
  int i; int j;           // declare loop counters
  for i=1 to n {
    for j=1 to i-1 {      // apply conditional phase gates
      V(pi/2^(i-j),q[n-i] & q[n-j]);
//      if q[n-i] and q[n-j] { Phase(pi/2^(i-j)); }
    }
    H(q[n-i]);            // qubit rotation
  }
  flip(q);                // swap bit order of the output
}
```

Przedstawiony program wykorzystuje funkcję `H()` odpowiadającą wykonaniu bramki Walsha-Hadamarda[1] na jednym lub wielu qubitach.

7.3 Program ShorProb

W Rozdziale 4.6 przedstawione są rozkłady prawdopodobieństwa uzyskiwane w trakcie procedury znajdowania rzędu. Zostały one uzyskane przy użyciu programu `ShorProb`, którego źródła zamieszczone są na stronie internetowej [48].

Na Listingu 7.2 zamieszczony jest kod źródłowy programu. Implementuje on funkcję pozwalającą na numeryczne obliczanie wyrażenia na rozkład prawdopodobieństwa otrzymanego po wykonaniu pomiaru końcowego w algorytmie Shora.

Listing 7.2: Program ShorProb

```
#include <string.h>
#include <fstream.h>
#include <iostream.h>
#include <stdio.h>
#include <unistd.h>

#include <cln/cln.h>

using namespace cln;

// stałe potrzebne do wszystkich obliczeń
const cl_F Pi = pi();
const cl_N I = complex(0,1.);

int main(int argc, char* argv[]){

    int scan;
    cl_I r,q;
    char* file = NULL;

    opterr = 0;

    while((scan=getopt(argc,argv,"q:r:f:"))!=-1)
        switch (scan)
        {
        case 'q':
            q = (cl_I)atoi(optarg);
            cout << "q=" << q << endl;
```

[1] W nowej wersji języka funkcja `H()` zastąpiła funkcję `Mix()`.

```
            break;
        case 'r':
            r = (cl_I)atoi(optarg);
            cout << "r=" << r << endl;
            break;
        case 'f':
            file = optarg;
            cout << "plik=" << optarg << endl;
            break;
        case '?':
            cout << "Nieznana opcja..." << endl;
        default:
            break;
        }
    if(file==NULL){
        cout << "Wyniki zostaną zapisane do "
             << "pliku dane.dat"
             << endl;
        file = "dane.dat";
    }
    if(q==0|r==0){
        cout << "Musisz podać r i q" << endl;
        return(1);
    }

    ofstream dopliku(file);

    dopliku << "#Wartość mierzona" << "\t"
            << "Prawdopodobieństwo"
            << endl;

    // parametry symulacji
    // omega to pierwiastek z jedynki
    // zależny od okresu
    const cl_N omega = exp((2*Pi*I*r)/q);
    // K określa ilość składników które sumujemy
    // zależy on od okresowości funkcji
    const cl_I K = floor1(q,r)+1;

    cl_N podstawa;
    cl_N temp;
    cl_N wynik;
    cl_N amp;
    cl_I normal = q*q;
```

```
    for(int x=0;x<q;x++){
        // resetuj wynik
        amp = 0.0;
        // to potęgujemy
        podstawa = expt(omega,x);

        for(int j=0;j<q;j++){
            wynik = 0.0;
            temp = 1.0;
            for(int i=0;i<K;i++){
                wynik = wynik+temp;
                temp = temp*podstawa;
            }
            amp=amp+wynik;
        }

        dopliku << x << "\t"
            << abs(amp*conjugate(amp))/normal
            << endl;
    }

    return(0);
}
```

Program został napisany z wykorzystaniem biblioteki CLN [30], dostarczającej klas pozwalających na przejrzystą manipulację na wielu rodzajach liczb. Program został skompilowany przy użyciu kompilatora `gcc` w wersji 2.95 rozprowadzanego na warunkach licencji GPL.

Skorowidz

Bibliografia

[1] Manindra Agrawal, Neeraj Kayal, Nitin Saxena. PRIMES in P. http://www.cse.iitk.ac.in/news/primality.html.

[2] Leslie E. Ballenetiene. *Quantum Mechanics. A Modern Development.* World Scientific, 1998.

[3] Adriano Barenco, Charles H. Bennet, Richard Cleve, David P. DiVicenzo, Norman Margolus, Peter Shor, Tycho Sleator, John Smolin, Harald Weinfurter. Elementary gates for quantum computation. *Phys. Rev. A*, 52:3457, 1995. arXiv:quant-ph/9503016.

[4] Charles H. Bennet. Logic reversibility of computation. *IBM J. Res. Devel*, 17:525–532, 1973.

Artykuł dostępny na stronie http://kh.bu.edu/qcl/.

[5] Ethan Bernstein, Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997.

Praca dostępna na stronie http://www.cs.berkeley.edu/~vazirani/.

[6] Ronald Newbold Bracewell. *The Fourier Transform & Its Applications.* McGraw-Hill, 1999.

[7] Michael J. Bremner, Christopher M. Dawson, Jennifer L. Dodd, Alexei Gilchrist, Aram W. Harrow, Duncan Mortimer, Michael A. Nielsen, Tobias J. Osborne. A practical scheme for quantum computation with any two-qubit entangling gate. *Phys. Rev. Lett.*, 89:247902, 2002. arXiv:quant-ph/0207072.

[8] Dagmar Bruß. Characterizing entanglement. *J. Math. Phys*, 43:4327, 2002. arXiv:quant-ph/0110078.

[9] Jean-Luc Brylinski, Ranee K. Brylinski. Universal quantum gates. Ranee K. Brylinski, Goong Chen, redaktorzy, *Mathematics of Quantum Computation.* CRC Press, 2002. arXiv:quant-ph/0108062.

[10] Sławomir Bugajski, Enrico G. Beltrametti. Correlations and entanglement in probability theory. arXiv:quant-ph/0211083, 2002.

[11] Sławomir Bugajski, Jarosław A. Miszczak. On entanglement at tripartie states. Praca w przygotowaniu, Luty 2003.

[12] Nicolas J. Cerf, Chris Adami. Quantum information theory of entanglement and measurement. *Physica D*, 120:62–81, 1998. Artykuł dostęny na stronie arXiv:quant-ph/0201095.

[13] Richard Cleve. An introduction to quantum complexity theory. C. Macchiavello, G.M. Palma, A. Zeilinger, redaktorzy, *Quantum Computation and Quantum Information Theory*, strony 103–127. World Scientific, 2000. arXiv:quant-ph/9906111.

[14] Peter Clote, Evangelos Kranakis. *Boolean Functions and Computation Models*. Springer, 2002.

[15] David Collins, K. W. Kim, W. C. Holton. Deutsch-Jozsa algorithm as a test of quantum computation. *Phys. Rev. A*, 58:R1633R1636, 1998.

[16] Data encryption standard (DES). Raport instytutowy FIPS PUB 46-2, Information Technology Laboratory, 1993.

Dokument dostępny w formie elektronicznej na stronie http://www.itl.nist.gov/fipspubs/fip46-2.htm.

[17] David Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proc. Roy. Soc. Lond.*, A 400:97, 1985. http://www.qubit.org/people/david/David.html.

[18] David Deutsch. Quantum computational networks. *Proc. Roy. Soc. Lond.*, A 425:73, 1989.

[19] Matthew J. Donald, Michał Horodecki, Oliver Rudolph. The uniquness theorem for entanglement measures. *J. Math. Phys.*, 43:4252–4272, 2002. arXiv:quant-ph/0105017.

[20] Paul Dumais, Hugo Touchette. QuCalc – Quantum Calculator, 2000. Witryna internetowa http://crypto.cs.mcgill.ca/QuCalc/.

[21] Artur Ekert, Richard Jozsa. Quantum computation and Shor's factoring algorithm. *Reviews of Modern Physics*, 68(3), 1996.

[22] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6/7):467–488, 1982.

[23] Edward Fredkin, Tommaso Toffoli. Conservative logic. *Int. J. Theor. Phys.*, 21:219–253, 1982.

Praca dostępna na stronie http://kh.bu.edu/qcl/.

[24] Matteo Frigo, Steven G. Johnson. FFTW – The Fastest Fourier Transform in the West. Strona internetowa http://www.fftw.org/.

[25] Piotr Gawron. Symulacje komputerów kwantowych. Praca magisterska, Politechnika Śląska w Gliwicach, Luty 2003. Praca w przygotowaniu.

[26] The GNU Privacy Guard - GnuPG.org. Strona internetowa http://www.gnupg.org/, 2003.

[27] Marian Grabowski, Roman S. Ingarden. *Mechanika Kwantowa*. Państwowe Wydawnictwo Naukowe, 1989.

[28] Lov K. Grover. Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.*, 79:325, 1997. arXiv:quant-ph/9706033.

[29] Lov K. Grover. A framework for fast quantum mechanical algorithms. *Proceedings of 30th Annual ACM Symposium on Theory of Computing (STOC)*, strony 53–62, 1998. arXiv:quant-ph/9711043.

[30] Bruno Haible, Richy Kreckel. CLN – Class Library for Numbers, 2002.
Kod źródłowy dostępny na stronie http://www.ginac.de/CLN/.

[31] Mika Hirvensalo. *Quantum computing*. Springer-Verlag, 2001.

[32] Michał Horodecki. Entanglement measures. *Quantum Information and Computation*, 1(1):3–26, 2001. http://www.rintonpress.com/journals/qic-1-1/miarprz3.ps.

[33] Michał Horodecki, Paweł Horodecki. Reduction criterion of separability and limits for a class of protocols of entanglement distillation. *Physical Review A*, 59:4206–4216, 1999. arXiv:quant-ph/9708015.

[34] Michał Horodecki, Paweł Horodecki, Ryszard Horodecki. Separability of mixed states: Necessary and sufficient conditions. *Phys. Lett. A*, 223(1-2):1–131, 1996. arXiv:quant-ph/9605038.

[35] Peter Hoyer. Efficient quantum transforms. arXiv:quant-ph/9702028, 1997.

[36] Richard Jozsa. Searching in Grover's search algorithm. arXiv:quant-ph/9706033.

[37] Richard Jozsa. Quantum algorithms and the Fourier transform. *Proceedings of Santa Barbara Conference on Quantum Coherence and Decoherence*, strony 323–337, 1998. arXiv:quant-ph/9707033.

[38] Stephan Kaufmann. RSA public-key encryption, 2003. Witryna internetowa http://library.wolfram.com/database/MathSource/1966/RSA-Info.txt.

[39] Michael Keyl. Fundamentals of quantum information theory. *Phys. Rep.*, 5(369):431–548, 2002. arXiv:quant-ph/0202122.

[40] Alexei Kitaev. Quantum measurements and the abelian stabilizer problem. *Electronic Colloquium on Computational Complexity (ECCC)*, 3(3), 1996. arXiv:quant-ph/9511026.

[41] Donald E. Knuth. *Sztuka programowania. Tom 2. Algorytmy Seminumeryczne*. Klasyka Informatyki. Wydawnictwa Naukowo-Techniczne, 2002.

[42] Neal Koblitz. *Wykłady z teorii liczb i kryptografii*. Wydawnictwa Naukowo-Techniczne, 1995.

[43] Antoni Kościelski. *Teoria Obliczeń. Wykłady z matematycznych podstaw informatyki*. Wydawnictwo Uniwersytetu Wrocławskiego, 1997.

[44] Samuel J. Lomonaco, Louis H. Kauffman. Quantum hidden subgroup algorithms: A mathematical perspective. *AMS CONM/305*, 2002. arXiv:quant-ph/0201095.

[45] David K. Maslen, Daniel N. Rockmore. Generalized FFTS - A survey of some recent results. Raport instytutowy TR96-281, Mathematics Department, Dartmouth College, Hanover, 1996.

Raport dostępny na stronie http://www.cs.dartmouth.edu/~maslen/.

[46] David K Maslen, Daniel N. Rockmore. The Cooley-Tukey FFT and group theory. *Notices of the AMS*, 48(10), 2001.

Publikacja elektroniczna dostępna na stronie Notices of the AMS http://www.ams.org/notices/.

[47] Jarosław A. Miszczak. Interpreter języka QCL dla systemu Windows. Strona internetowa http://www.jarekadam.prv.pl/fizyka/qcl.

[48] Jarosław A. Miszczak. Pakiet Entanglement. Strona internetowa http://www.jarekadam.prv.pl/fizyka/.

[49] Jarosław A. Miszczak. Efficient quantum algorithm for factorization. *Archiwum Informatyki Teoretycznej i Stosowanej*, 2:117–129, 2002.

Artykuł dostępny na stronie http://www.iitis.gliwice.pl/zksi/.

[50] Peter L. Montgomery. A survey of modern integer factorization algorithms. *CWI Quarterly*, 7(4):337–365, 1994. http://www.crypto-world.com/.

[51] Michele Mosca. *Quantumu Computer Algorithms*. Praca doktorska, University of Oxford, 1999.

Praca dostępna na stronie http://www.cacr.math.uwaterloo.ca/~mmosca/.

[52] Władysław Narkiewicz. *Teoria Liczb*, wolumen 50 serii *Biblioteka Matematyczna*. Państwowe Wydawnictwa Naukowe, 1977.

[53] M. A. Nielsen, Isaac L. Chuang. *Quantum Computation and Quantum information*. Cambridge University Press, 2000.

[54] Michael A. Nielsen, Julia Kempe. Separable states are more disordered globally than locally. *Phys. Rev. Lett.*, 86:5184–5187, 2001. arXiv:quant-ph/0011117.

[55] Berbar Ömer. Quantum programming in QCL, 2000.

Kod źródłowy oraz publikacje związane z językiem QCL dostępne są na stronie http://tph.tuwien.ac.at/˜oemer.

[56] OpenSSH. Strona internetowa http://www.openssh.org/, 2003.

[57] Christos H. Papadimitriou. *Złożoność obliczeniowa*. Klasyka Informatyki. Wydawnictwa Naukowo-Techniczne, 2002.

[58] Asher Peres. Separability criterion for density matrices. *Phys. Rev. Lett*, 77:1413–1415, 1996. arXiv:quant-ph/9604005.

[59] The International PGP Home Page. Strona internetowa http://www.pgpi.org/, 2003.

[60] John Preskill. Lecture notes on physics: Quantum computation. Wykłady dostępne na stronie http://www.theory.caltech.edu/people/preskill/ph229/.

[61] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery. *Numerical Recipes in C. The Art of Scientific Computing*. Cambridge University Press, wydanie 2, 1992.

Materiały dostępne na stronie http://www.nr.com/.

[62] Markus Pueschel, Martin Roetteler, Thomas Beth. Fast quantum Fourier transforms for a class of non-abelian groups. arXiv:quant-ph/9807064, 1998.

[63] Walter Rudin. *Analiza funkcjonalna*. Wydawnictwo Naukowe PWN, 2001.

[64] Bruce Schneier. *Kryptografia dla praktyków. Protokoły, algorytmy i programy źródłowe w języku C*. Wydawnictwa Naukowo-Techniczne, 1995.

[65] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. *IEEE Symposium on Foundations of Computer Science*, strony 124–134, 1994.

Artukuł dostępny na stronie http://www.research.att.com/˜shor/papers/.

[66] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal of Computing*, strony 1484–1509, 1997.

Artukuł dostępny na stronie http://www.research.att.com/~shor/papers/.

[67] David R. Simon. On the power of quantum computation. *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, strony 116–123, Los Alamitos, CA, 1994. Institute of Electrical and Electronic Engineers Computer Society Press.

Artykuł dostępny na stronie http://research.microsoft.com/crypto/dansimon/me.htm.

[68] Wojciech Sobczak. *Elementy teorii informacji*. Wiedza Powszechna, 1973.

[69] Barbara M. Terhal. Detecting quantum entanglement. *Journal of Theoretical Computer Science*, 1(287):313–335, 2002. arXiv:quant-ph/0101032.

[70] Tommaso Toffoli. Reversible computing. Raport instytutowy, MIT Laboratory for Computer Science, 1981.

Praca dostępna na stronie http://pm1.bu.edu/~tt/publ.html.

[71] Vlatko Vedral, Martin B. Plenio. Entanglement measures and purification procedures. *Phys. Rev. A*, 57(3):1619–1633, 1998.

Artykuł dostępny na stronie http://www.lsr.ph.ic.ac.uk/~plenio/pub.html.

[72] Guifre Vidal, Rolf Tarrach. Robustness of entanglement. *Phys.Rev. A*, 59:141–155, 1999. arXiv:quant-ph/9806094.

[73] Joachim von zur Gathen, Jürgen Gerhard. *Modern Computer Algebra*. Cambridge Unversity Press, 1999.

[74] Julia Wallace. Quantum computer simulators, 2002. Strona internetowa http://www.dcs.ex.ac.uk/~wallace/simtable.htm.

[75] Reinhard F. Werner. Quantum states with Einstein-Rosen-Podolsky correlations admitting a hidden-variable model. *Phys. Rev. A*, 40:4277–4281, 1989.

Artykuł dostepny na stronie http://www.imaph.tu-bs.de/qi/papers.html.

[76] Reinhard F. Werner, Guifre Vidal. Computable measure of entanglement. *Phys. Rev. A*, 65:032314, 2002. arXiv:quant-ph/0102117.

[77] Robert Wieczorkowski, Ryszard Zieliński. *Komputerowe generatory liczb losowych*. Wydawnictwa Naukowo-Techniczne, 1997.

[78] Stephen Wolfram. *The Mathematica book*. Cambridge University Press and Wolfram Media, wydanie 4, 1999.

[79] William K. Wootters. Entanglement of formation of an arbitrary state of two qubits. *Phys.Rev.Lett.*, 80:2245–2248, 1998. arXiv:quant-ph/9709029.

[80] Rafał Wysocki. Implementacja kompilatora języka QCL w środowisku MS Windows. Praca magisterska, Politechnika Śląska w Gliwicach, 2002.

[81] Daoqi Yang. *C++ and object oriented numeric computing for scientists and engineers*. Springer, 2001.

[82] Andrew Chi-Chih Yao. Quantum circuit complexity. *Proceedings of the 34th Annual Symposium on Foundations of Computer Science*, strony 352–361, Los Alamitos, CA, 1993. Institute of Electrical and Electronic Engineers Computer Society Press.

Artykuł dostępny na stronie http://citeseer.nj.nec.com/yao93quantum.html.

[83] Christof Zalka. Grover's quantum searching algorithm is optimal. *Phys.Rev. A*, 60:2746–2751, 1999. arXiv:quant-ph/9711070.

Przedrostek „arXiv:" można zastąpić adresem jednego z serwerów lustrzanych archiwum internetowego arXiv, np: http://www.arxiv.org/abs/