Jihočeská univerzita v Českých Budějovicích

Ekonomická fakulta

Katedra aplikované matematiky a informatiky

Bakalářská práce

Kvantové aplikace založené na platformě IBM QX

Vypracoval: Josef Polák

Vedoucí práce: Ladislav Beránek, doc. Ing. CSc., MBA

České Budějovice 2023

Prohlašuji, že svou bakalářskou práci jsem vypracoval/a samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury. Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to – v nezkrácené podobě/v úpravě vzniklé vypuštěním vyznačených částí archivovaných Ekonomickou fakultou - elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

Datum: Podpis studenta:

Poděkování:

Rád bych poděkoval vedoucímu své práce, kterým byl Ladislav Beránek doc. Ing. CSc., MBA. Dále děkuji své rodině a přítelkyni za podporu a trpělivost.

Obsah

Úvod

Cíle

Z počátku bylo cílem práce vysvětlit a objasnit matematické principy a základy kvantových výpočtů v jednoduché aplikaci, a to zejména v oblasti kryptologie. Po konzultaci s vedoucím práce bylo dovoleno se od oblasti kryptografie vzdálit. Dále ukázat některé možnosti využití a práce s platformou IBM QX. Také poukázat na rozdíl klasického a kvantového přístupu k výpočtům.

Struktura práce

První část této práce bude obsahovat základní informace o kvantové mechanice, abychom se seznámili s fungováním kvantového světa a jeho pravidly. Následovat bude nahlédnutí do historie kvantové informatiky a kvantových počítačů.
Část číslo dvě už se bude soustředit na vysvětlení samotné kvantové informatiky. Jako například co je to qubit a jeho zobrazování nebo zápis, jaké existují brány a jakým způsobem fungují, fungování celých kvantových obvodů a na závěr příklad známých kvantových algoritmů. Další bude shrnutí výhod a nevýhod kvantový počítačů, jejich porovnání s klasickými a využití v reálných problémech ve světě.
Třetí část osvětlí, co to je společnost IBM a k čemu slouží jejich platforma IBM Q Experience, jak se platforma vyvíjí a jaké zařízení má k dispozici.
Čtvrtá část se zaměří na definování a popsání problému, který bude řešen danou aplikací, a jaké algoritmy jsou obvykle používány na řešení tohoto problému prostřednictvím klasických počítačů.
Pátá a zároveň předposlední kapitola popisuje postup řešení problému. Problémy, které se vyskytly na cestě za úspěšným dokončením a doladěním aplikace, a jejich postupné řešení.
Poslední kapitola bude obsahovat shrnutí a zhodnocení výsledku celé práce, problémy při řešení a na kolik se povedlo dosáhnout stanovených cílů. Součástí bude i uvedení příkladů, kde by výsledná aplikace mohla být prospěšná a náměty na zlepšení a zdokonalení práce.

1 Kvantová mechanika

Kvantová mechanika je vědní obor, který se zabývá studiem mikroskopických objektů, jako jsou například atomy, elektrony, neutrony, molekuly, fotony a mnohé další. Zákony, kterými se řídí tyto objekty se poněkud odlišují od těch, kterými se řídí běžná tělesa. Chování těchto částic je natolik zvláštní, že se s nimi nemohl ztotožnit ani Albert Einstein (teorie relativity) nebo Richard Phillips Feynman (teorii kvantové elektrodynamiky). R. P. Freynman dokonce řekl: *„Myslím, že mohu klidně prohlásit, že kvantové teorii nerozumí nikdo.“*

Zde jsou uvedeny příklady chování těchto mikroskopických částic, které jsou třeba pochopit k dalšímu chápání kvantové informatiky. Jak píše ve své práci (Tyc, 2006) jedná se zejména o:

a) Princip superpozice – kvantový objekt může nabývat několika hodnot najednou. To je první podstatný rozdíl mezi kvantovou informatikou a tou klasickou, kde jednotky bit může nabývat jen jedné hodnoty v jeden okamžik.
b) Diskrétní spektrum – určité veličiny u těchto objektů nabývají pouze hodnoty z diskrétní množiny hodnot. Například energie elektronů, které jsou vázány v atomovém obalu.
c) Vliv měření na pozorovaný objekt – při měření dochází k takzvanému kolapsu stavu u měřeného objektu a ten spadne do jedné z hodnot které nabýval před měřením A). S tím souvisí i to, že nejsme schopni přesně určit, jaký výsledek měření bude mít, ale jsme schopni určit u jednotlivých hodnot pravděpodobnosti, že po měření skončí jako výsledná hodnota.
d) Tunelový jev – jedná se o jev, kdy částice pronikají předměty, kterými by pronikat neměli a obráceně, kdy se odrážejí od objektů pro ně obvykle průchozích.
e) Dualismus vln a částic – tento jev poukazuje na to, že částice mikrosvěta se mohou chovat jako vlny, ale také mít vlastnosti jako částice. S touto hypotézou přišel Louis de Broglie (Kulhánek, 2016).

2 Historie kvantové informatiky

První teorie o kvantovém světě se začali objevovat prvně na přelomu 19. a 20. století, kdy Max Planck uhodl správnou formuli pro Teorii záření absolutně černého tělesa (rok 1900) a tu později i teoreticky odvodil (Kulhánek, 2016). Na tuto práci navázal dánský vědec Niels Henrik David Bohr, který jako první aplikoval kvantový koncept, omezující energii systému na určité diskrétní hodnoty a na problematiku atomové a molekulární struktury (Aaserud, 2023). Tento objev vedl k dalším průlomům ve kvantové mechanice, o které se postarali Werner Karl Heisenberg (sestavení matematického modelu atomu pomocí maticového výpočtu), Erwin Rudolf Josef Alexander Schrödinger (Schrödingerova rovnice, Schrödingerova kočka), Paul Adrien Maurice Dirac (Diracova rovnice) a také Wolfgang Pauli (Pauliho vylučovací princip). Tato jména během práce ještě zazní. Všechny tyto objevy byly natolik důležité, že každý z těchto teoretických fyziků obdržel za svou práci Nobelovu cenu. Toť něco málo k počátkům historie kvantové teorie a mechaniky, díky kterým bylo možné nejen položit základy kvantové informatiky, ale také pochopit mnoho dalšího, co klasická fyzika nebyla schopná objasnit.

Historie kvantové informatiky začíná o pár desítek let později, v 70. a 80. letech. Postarali se o ní především Charles Bennett, Paul Benioff, David Deutsch a Richard Feynman. Začali s postupným definováním abstraktního rámce kvantové informatiky v podobě kvantového Turingova stroje, teorie o kvantové složitosti problémů a určitých námětů funkčního modelu prvního kvantového počítače (Kupča, 2001). Další krok pro kvantovou informatiku učinil Peter Willston Shor kolem roku 1994. Vypracoval kvantové algoritmy pro faktorizaci velkých čísel[1] anebo vyhledávání v nesetříděné databázi.

Už tehdy bylo jasné, že takovýto stroj bude mít mnoho nových možností, jak prozkoumat využití kvantového paralelismu nebo vlastností přirozeně náhodné jevy. Bohužel sestavení tohoto stroje bylo velmi obtížné. Manipulovat se samostatnými atomy není vůbec jednoduché a měření systému je též obtížné. Poprvé se to povedlo Jonathanu A. Jonesovi a Michele Mosca z Oxfordské univerzity při demonstrování používaní kvantového algoritmu k řešení Deutschova problému na funkční 2-qubitovém NMR kvantovém počítači.V dnešní době společnost IBM vyvinula kvantový procesor Osprey, který disponuje objemem 433 qubitů a dále chce společnost ve zlepšování pokračovat. Chtějí pokořit hranici 4000 qubitů do konce roku 2025 (IBM, 2022).

[1] Faktorizace velkých čísel znamená jejich rozklad na dvě prvočísla, která vzájemným součinem dají původní číslo

3 Kvantová informatika

Tato kapitola už se bude blíže zabývat kvantovou informatikou, jak fungují výpočty za pomoci těchto zvláštních kvantových pravidel, kolik společného má kvantová informatika s tou klasickou. Popisuje, jaké logické brány se používají či jaké algoritmy lépe fungují prostřednictvím kvantového počítač a výhody i nevýhody kvantové informatiky.

3.1 Kvantový stav

Kvantový stav je spojení všech relevantních fyzikálních veličin u kvantového systému (Laforest, 2015). Mezi tyto veličiny patří především pozice, spin u elektronu, polarizace a další. Tyto informace o kvantovém systému neboli kvantový stav lze zapsat pomocí vektoru v komplexním lineárním Hilbertově prostoru.

Tyto veličiny nabývají pouze hodnot z diskrétní množiny. Jenže tomu tak je jen kvůli měření daného systému. Ve skutečnosti mohou veličiny odpovídat nekonečně mnoho kvantovým stavům. Kvantový systém, který je izolovaný od okolí (není měřený) má spojitý vývoj. Tento vývoj popisuje Schrödingerova vlnová rovnice pro daný systém.

Neskládá se však vždy jen z jedné vlnové funkce. Kvantové systémy se mohou skládat i z více vlnových funkcí, z toho vyplývá, že kvantový stav je vyjádřen jako součet daných vlnových funkcí. Pro přehlednost každá funkce vlastní takzvanou amplitudu pravděpodobnosti. Ta určuje, z jaké části se podílí na celkovém stavu. Zajímavé je, že může nabývat i komplexních či negativních hodnot. Další význam má amplituda pravděpodobnosti pro výpočet pravděpodobnosti. Jedná se o pravděpodobnost naměření daného stavu. Tato pravděpodobnost odpovídá druhé odmocnině z absolutní hodnoty amplitudy pravděpodobnosti.

$$|\psi\rangle = \omega_0|\psi_0\rangle + \omega_1|\psi_1\rangle \qquad (1)$$

Tato rovnice ukazuje, jak je možné zapisovat kvantový stav o dvou vlnových funkcích. ψ je označení kvantového stavu. ψ_0 a ψ_1 jsou vlnové funkce, ze kterých je složený onen kvantový systém a $\omega_0,\ \omega_1 \in \mathbb{C}$ jsou jejich amplitudy pravděpodobnosti.

3.2 Hilbertův prostor

Jak už bylo uvedeno, kvantové stavy popisujeme jako vektory. Tyto vektory se nachází v takzvaném komplexním lineárním Hilbertově prostoru s konečným počtem dimenzí. Každý stav, který může kvantový systém nabýt při měření, má vlastní dimenzi (Kupča, 2001).

Protože chceme vyjádřit kvantový bit, potřebujeme dvoustavový/dvouhladinový systém. To je takový kvantový systém, který nabývá pouze dvou různých stavů. Zde je značná podobnost s klasickým bitem, buď nabude hodnoty 0 nebo 1. Dva stavy znamenají tedy dvourozměrný Hilbertův prostor. Takto je zapisován takový prostor:
$|\psi\rangle = \omega_0|\psi_0\rangle + \omega_1|\psi_1\rangle$, stejná rovnice jako v kapitole 3.1. Obvykle se stavy označují jako $|0\rangle$ a $|1\rangle$, stavy totiž odpovídají klasickým bitům. Rovnice po úpravě vypadá takto:

$$|\psi\rangle = \omega_0|0\rangle + \omega_1|1\rangle \qquad (2)$$

Pro prostory, kde je dimenzí více platí následující.

$$\sum_{i=0}^{n-1}|\omega_i|^2 \qquad (3)$$

Samozřejmě tato rovnice platí i pro systémy právě s dvěma stavy (Kupča, 2001).

3.3 Kvantový bit

Nejmenší jednotkou dat v klasické informatice je bit. Není překvapením, že kvantová informatika označení převzala a vznikl kvantový bit. Jindy také označovaný jako qubit.
O kvantových bitech toho bylo něco málo řečeno už v předchozích kapitolách. Jako vhodný dvoustavový kvantový systém se v praxi používá například foton[2] v polarizovaném stavu. Nulový stav je při zaznamenání horizontální polarizace a jedničkový stav při vertikální polarizaci. Při jiných polarizacích, jako je například diagonální nebo elipsoidní polarizace, se systém nachází v superpozici. Další možností jsou atomová jádra, kde rozhoduje atomový spin či ionty a jejich energetický systém (Vyvlečka, 2020).

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad (4)$$

α a β reprezentují hodnotu amplitudy pravděpodobnosti neboli velikost pravděpodobnosti výsledku, který dostaneme po změření. Protože se jedná o kvantový výpočet, mohou α a β nabývat komplexních hodnot. Jak už víme, pravděpodobnost z α a β získáme jejich umocněním. Tedy $|\alpha|^2$ se rovná pravděpodobnosti výsledku $|0\rangle$ při měření.
Z tohoto faktu vyplývá rovnice:

$$|\alpha|^2 + |\beta|^2 = 1 \qquad (5)$$

[2] Nejmenší částice světla.

Například takto:

$$|\varphi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \qquad (6)$$

vypadá zápis qubitu v superpozici, ve které je pravděpodobnost stavů $|0\rangle$ a $|1\rangle$ totožná a rovná se 50 %.

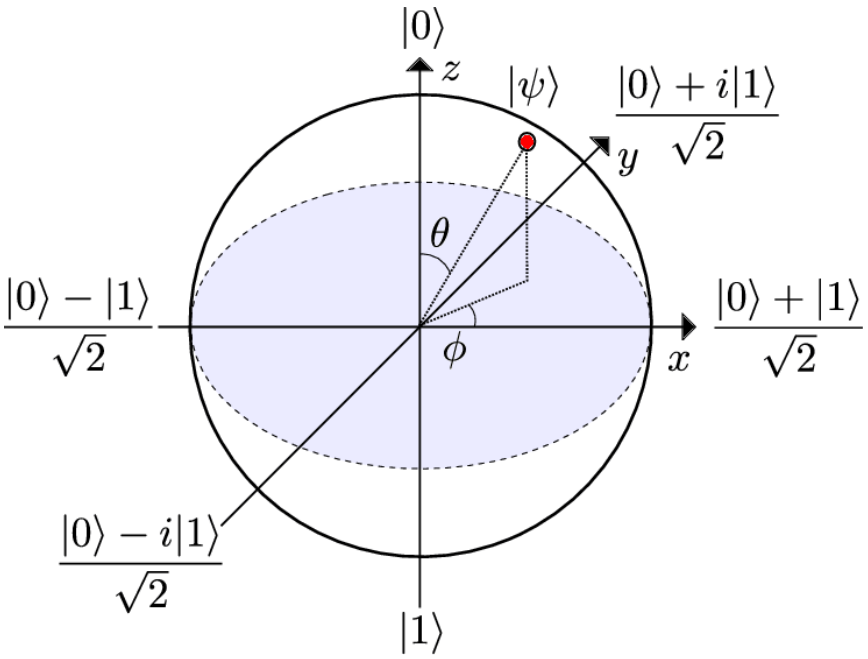
3.3.1 Schrödingerova kočka

Pro jednoduší a snadnější pochopení paradoxu kvantové superpozice vymyslel Erwin Rudolf Josef Alexander Schrödinger „myšlenkový experiment“. Velmi zjednodušeně se dá říct, že se jedná o živou kočku umístěnou do neprůhledné, uzavřené krabice či boxu. Spolu s kočkou je vložena nádoba s kyselinou, taktéž uzavřená. Nádoba je opatřená mechanismem, který ji rozbije s pravděpodobností 50 % a se stejnou pravděpodobností ne. Takový stav bychom vyjádřili funkcí $|\psi\rangle = \omega_0|živá\rangle + \omega_1|mrtvá\rangle$.
Myšlenka je taková:
Dokud krabici neotevřeme a nezjistíme, zda je kočka živá či mrtvá, v jakém stavu je tato kočka? A úplně ve stejném stavu se nachází kvantový bit před změřením. Jediné, co je o něm před měření známé, je poměr pravděpodobností.

3.3.2 Blochovo sféra

Jako jedno z vyjádření kvantového bitu se používá takzvaná Blochova sféra. Jakožto grafická reprezentace je snadnější a přehlednější pro pochopení kvantového bitu a jeho transformace pomocí logických kvantových bran. Jedná se o kouli a qubity se zobrazují na jejím povrchu. Pozice je ovlivněna amplitudami pravděpodobností stavů a fázový posun.

Obrázek 1: Blochovo sféra



Zdroj: (Anton Frisk Kockum, 2019)

Obrázek 1 jasně ukazuje, jak změna znamének u pravděpodobnostních amplitud mění polohu kvantového bit na povrchu, takzvaný fázový posun. Každý bod na povrchu této koule je qubit v superpozici, kromě dvou modrých bodů. Tyto body se nacházejí na severním a jižním pólu. K jednomu z nich se qubit přesune po změření. Kvantové bity můžeme psát takovýmto způsobem:

$$|\psi\rangle = cos(\tfrac{\varphi}{2})|0\rangle + e^{-i\theta} sin(\tfrac{\varphi}{2})|1\rangle \qquad (7)$$

Kde φ je úhel nabývající hodnot mezi 0 a π a jedná se o úhel, o který je odtočený vektor mířící k bodu, na kterém se nachází stav qubitu na blochově sféře, kolem vertikální osy koule od osy x. Tento úhel tedy ukazuje, jak moc jsou v pravděpodobnosti stavů začleněny komplexní hodnoty. Druhý úhel, θ, který svírá osa z a vektor mířící k bodu, na kterém se nachází stav qubitu na blochově sféře a nabývá hodnot 0 až 2π. Úhel θ tedy reprezentuje poměr pravděpodobností jednotlivých stavů kvantového bitu (Jae-weon Lee, 2002).

Tento způsob vyobrazení kvantového bitu je velmi přehledný, ale nelze jej využít pro zápis více qubitů najednou

3.3.3 Diracův zápis vektoru

Diracova notace je způsob zápisu vektoru, který je používaný v oboru kvantové mechaniky. Samozřejmě se používá i ve kvantové informatice. Tento zápis vektoru vytvořil a zavedl Paul A. M. Dirac. Za pomoci této notace můžeme kvantový stav vyjádřit takto: $|0\rangle = \begin{pmatrix}1\\0\end{pmatrix}$, $|1\rangle = \begin{pmatrix}0\\1\end{pmatrix}$.

Pro lepší pochopení je druhá rovnice, která systém zápisu ještě trochu osvětluje

$$|\psi\rangle = \omega_0|0\rangle + \omega_1|1\rangle = \begin{pmatrix}\omega_0\\\omega_1\end{pmatrix} \tag{8}$$

Tato notace se také nazývá Bra-ket. Symbol identifikující vektor zapisujeme jako |x⟩, do takzvaného ket. Duální vektor je zapisován takto $\langle x| = (\omega_0, \omega_1)$ a nazývá se bra.
Pomocí této notace lze pro dva vektory, které označíme například $\vec{x}, \vec{y}$. Budou odpovídat stavům $|x\rangle$ a $|y\rangle$, definovat skalární součin jako $(|x\rangle, |y\rangle)$, což se obvykle zkracuje na zápis $\langle x|y\rangle$ (Kupča, 2001). Zkratka tvořící závorku (anglicky bracket). Pro označení tenzorového součinu se používá $|x\rangle\langle y|$.

3.4 Tenzorový součin

Tenzorový součin je nutno znát, pokud pracujeme s více jak jedním kvantovým bitem. Představme si, že tyto bity jsou dva. Dva qubity také znamenají dva Hilbertovy komplexní prostory. Pokud chceme pracovat s oběma bity najednou je potřebný další prostor, ve kterém budou ty dva předchozí spojené. K výslednému prostoru se lze dostat přes tenzorový součin dvou původních prostorů. Tenzorový součin značíme jako |x⟩ ⊗ |y⟩. Mějme například dva kvantové stavy |x⟩ a |y⟩. Ty jsou zapisovány: $|x\rangle = \omega_{x0}|0\rangle + \omega_{x1}|1\rangle$ a $|y\rangle = \omega_{y0}|0\rangle + \omega_{y1}|1\rangle$. Převedeme je do Diracovi notace, $|x\rangle = \begin{pmatrix}\omega_{x0}\\\omega_{x1}\end{pmatrix}$ a $|y\rangle = \begin{pmatrix}\omega_{y0}\\\omega_{y1}\end{pmatrix}$.
Poté stavy spojíme tenzorovým součinem jako je napsáno na následující rovnici.

$$|x\rangle \otimes |y\rangle = \begin{pmatrix}\omega_{x0}|y\rangle\\\omega_{x1}|y\rangle\end{pmatrix} = \begin{pmatrix}\omega_{x0}\omega_{y0}\\\omega_{x0}\omega_{y1}\\\omega_{x1}\omega_{y0}\\\omega_{x1}\omega_{y1}\end{pmatrix} \tag{9}$$

Z tohoto vzorce je zcela zřejmé, jak tenzorový součin funguje.

3.4.1 Spojení více kvantových bitů

Pokud by kvantová informatika byla schopná pracovat pouze s jedním kvantovým bitem, bylo by možné ji využít tak maximálně na generátor náhodných celých čísel od nuly do jedné.

Jak je vysvětleno v kapitole 3.4, při propojení dvou kvantový stavů vzniká další nový stav v novém Hilbertově prostoru za pomocí tenzorového součinu.

Například při spojení dvou kvantových bitů v superpozici mohou nastat tyto stavy:

$$|00\rangle = \begin{pmatrix}1\\0\end{pmatrix} \times \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} \tag{10}$$

$$|01\rangle = \begin{pmatrix}0\\1\end{pmatrix} \times \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix} \tag{11}$$

$$|10\rangle = \begin{pmatrix}1\\0\end{pmatrix} \times \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \tag{12}$$

$$|11\rangle = \begin{pmatrix}0\\1\end{pmatrix} \times \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix} \tag{13}$$

Při spojení dvou kvantových bitů vzniká Hilbertův prostor o čtyřech dimenzích. Platí tedy, fakt: počet stavů a zároveň počet dimenzí prostoru je roven 2^n kde n je počet spojených qubitů.

Další věc je zapisování takového kvantového stavu. Příklad opět na dvou qubitovém systému. Při použití většího množství qubitů se zápis značně prodlužuje a dva budou pro pochopení stačit.

$$|x\rangle = \omega_0|00\rangle + \omega_1|01\rangle + \omega_2|10\rangle + \omega_3|11\rangle = \begin{pmatrix}\omega_0\\\omega_1\\\omega_2\\\omega_3\end{pmatrix} \tag{14}$$

3.5 Provázání

Kromě superpozice má kvantový systém ještě další zajímavé vlastnosti, kterých lze využít ve kvantové informatice. Jedná se o jev provázání dvou nebo více kvantových bitů. Kvantový stav každého z bitů je závislí na ostatních bitech. Takové bity mají korelovanou jednu z vlastností, může to být spin nebo polarizace.

„Provázané stavy jsou ty, které nelze simulovat klasickými korelace.“ (Lluis Masanes, 2007) V. Kupča ve své práci napsal: *„Zejména je s podivem, že lze bez přítomnosti výměnných částic ovlivňovat částici, která je třeba na opačné straně vesmíru.“* (Kupča, 2001).

Na tomto principu se vědci v dnešní době pokoušejí postavit takzvanou Teleportaci stavů. Ta by mohla transportovat informace mnohem větší rychlostí, než jsme doposud zvyklí. Takovéto provázané Kvantové bity mohou vzniknout například při vyslání ultra fialového fotonu do krystalu β−BaB^2O^4. Tento foton se může po průchodu krystalem přemění na dva fotony o nižší energii. Jeden je polarizovaný vertikálně a ten druhý horizontálně, ale jsou vzájemně provázané.

3.6 Interference

Pod pojmem interference si zajisté všichni vybaví výsledek působení dvou vlnění na sebe navzájem. Vlny na sebe působí buď to konstruktivně, pokud se hodnoty vlny nacházejí na stejné straně od uzlu (nad nebo pod ním), jsou tedy zesilovány. Pokud je tomu naopak vlny na sebe působí destruktivně a tlumí se. Mohou se tlumit do takové míry, že úplně zaniknou.

Díky jevu vlny a částice je interference i součástí kvantové informatiky. Skládání amplitud pravděpodobnosti několika možností je možné využít v některých algoritmech.

Amplitudy se mohou vzájemně tlumit, zesilovat nebo dokonce vyrušit.

Interference se využívá v kvantovém počítání při získávání výsledku z kvantového registru. Pokud známe „hledanou vlnu“ můžeme s ní působit na registr amplitud pravděpodobností stavů. To má za výsledek zvýraznění příznivých stavů, tedy správných řešení (konstruktivní interference) a potlačení či utlumení těch nepříznivých (destruktivní interference).

3.7 Brány

Jednou z věcí, kterou mají klasické počítače společné s těmi kvantovými, jsou logické brány. Jednotky klasické informatiky, tedy bity, jsou vedeny po vodičích a na tyto vodiče jsou umisťovány logické brány či hradla. Dalo by se říct, že kvantové počítače to mají podobné, jen s tím rozdílem, že jsou aplikovány na qubity. Jak je tomu zvykem ve kvantové informace, nic není úplně totožné. Jednou z těchto věcí je reverzibilita kvantových bran, kterou většina těch klasických nesplňuje. Reverzibilita znamená, že z výstupu brány lze získat vstup/vstupy.

Pokud aplikujeme kvantovou bránu na qubit, výsledek této akce získáme tenzorovým součinem matice kvantového bitu a matice dané brány. Z toho vyplývá, že každá brána má svou matici. Nyní následuje pár příkladů takových bran.

3.7.1 Jednoqubitové brány

Pro snadnější pochopení začneme od těch nejjednodušších bran. Těmi jsou brány, které pracují jen v rámci jednoho kvantového bitu. Nejlépe pochopitelné budou ty, které jsou podobné těm klasickým logickým branám jako je například NOT, OR a AND.

3.7.1.1 NOT

Nejzákladnější, zároveň nejpodobnější té klasické, kvantovou branou je NOT. U klasického obvodu brána not také působí pouze na jeden bit/qubit. 1 na vstupu mění na 0 a obráceně. Pokud je kvantový bit ve stavu $|\varphi\rangle = 1|0\rangle + 0|1\rangle$, kdy při měření vždy vyjde stav $|0\rangle$, po aplikace brány je jasné, že výsledek musí být opačný, tedy $|1\rangle$. Tomuto výsledku odpovídá stav $|\varphi\rangle = 0|0\rangle + 1|1\rangle$. Brána tedy mění $\binom{1}{0}$ na $\binom{0}{1}$. Zde můžete vidět, jak vypadá maticový zápis hradla a na rovnici pod jak funguje.

Obrázek 2: Schéma kvantové brány NOT

Zdroj: (IBM, nedatováno)

$$NOT\ |\varphi\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \binom{\alpha}{\beta} = \binom{\beta}{\alpha} = (|0\rangle\langle 1| + |1\rangle\langle 0|)\,(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle \quad (15)$$

Obrázek 2 znázorňuje značku brány Not ve vývojovém prostředí společnosti IBM. Brána je také známá jako překlápění qubitu nebo Pauliho X brána (IBM Corporation, 2016).

3.7.1.2 Pauliho brány

Mezi tyto brány patří již zmiňovaný NOT nebo také X brána. Mezi Pauliho brány dále řadíme bránu Y a bránu Z. Není náhodou, že brány mají stejné označení jako osy v Blochově sféře 3.3.2.

Brána Y tedy bude mít co dočinění s osou y. Jedná se o otočení vektoru kolem osy y o 180̊. Ve kvantové informatice se používají radiány, tudíž je to posun o π.

$$Y = \begin{bmatrix} 1 & -i \\ i & 0 \end{bmatrix}$$

Z předchozího odstavce jasně vyplývá, jakým způsobem bude fungovat brána Z. Jedná se tedy o rotaci kolem osy Z na Blochove sféře o π kolem osy. Maticový zápis vypadá následovně:

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Tato brána nemění |0⟩ na |1⟩, ale mění fázový posun qubitu. To znamená, že mění tento stav |φ⟩ = α|0⟩ + β|1⟩ na tento |φ⟩ = α|0⟩ - β|1⟩.

3.7.1.3 Hadamardova brána

Jedna z bran, které jsou nejvíce používány ve kvantové informatice, dostává kvantový bit do stavu superpozice. Tedy aplikací této brány se nastavuje stav |0⟩ na $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ a stav |1⟩ na $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ (Phillip R. Kaye, 2007). Velmi zjednodušeně to znamená, že brána provede rotaci o π/2 kolem osy y a poté druhou rotaci kolem osy x a to o π. Zde je znázorněný maticový zápis brány:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Za krátkou zmínku dále stojí brány, které mění pouze fázový posun. Jedná se o brány otáčející kolem osy z jako brána Z. Brány tedy mění fázový posun, ale nijak nezasahují do změny amplitudy pravděpodobností jednotlivých stavů. Jediný rozdíl mezi těmito branami je velikost otočení kolem dané osy. Již zmiňovaná brána Z se otáčí kolem osy z o π, tedy $\theta = \pi$. Obecný zápis takových bran:

$$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix}$$

Mezi tyto brány se řádí také brána S, u které platí $\theta = \pi/2$ a další kvantová brána tohoto typu je T, jejíž rotace je pouze o $\pi/4$, z toho vyplývá $\theta = \pi/4$.

3.7.1.4 Operátor U

Jedná se o univerzální bránu, pomocí které jsme schopni s kvantovým bytem udělat v podstatě cokoliv. Tato brána vychází z evolučního operátoru, jak píše ve své práci V. Kupča. Tento operátor může popsat jakýkoliv dynamický vývoj kvantového systému v čase.

Jeho zápis vypadá takto:

$$U(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

Pokud tedy chceme kvantový systém se stavem $|\varphi\rangle = 1|0\rangle + 0|1\rangle$ přivést do superpozice, použijeme θ s hodnotou $\pi/4$. Výpočet vypadá následovně:

$$U\left(\frac{\pi}{4}\right)|0\rangle = U\left(\frac{\pi}{4}\right)\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{pmatrix} = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \qquad (16)$$

Zdroj: (Kupča, 2001)

3.7.1.5 U3

Některé brány otáčejí vektor kolem osy x, jiné y anebo osy z. Ideální by bylo spojení brány, která dokáže spojit všechny rotace dohromady. V prostředí, ve kterém se pracuje na platformě společnosti IBM, jedna taková brána je. Lze ji najít pod označením U stejně jako název operátoru z předchozí kapitoly 3.7.1.4.

V této bráně nastavujeme čtyři různé parametry. První tři jsou theta (Θ), phi (φ) a lambda (λ), poslední určuje, na který qubit bude brána aplikována. Všechny parametry jsou nastavovány v radiánech. Následující maticový zápis osvětluje fungování této brány.

$$U(\theta, \phi, \lambda) = \begin{bmatrix} \cos(\frac{\theta}{2}) & -e^{i\lambda}\sin(\frac{\theta}{2}) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) & e^{i(\phi+\lambda)}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}$$

Jednoduše shrnuto, první parametr (theta) mění amplitudu pravděpodobnosti a tím i pravděpodobnost změření jednotlivých stavů. Druhý z parametrů, tedy phi, ovlivňuje fázový posun.

Pokud máme kvantový systém ve stavu, kdy při měření vždy změříme $|0\rangle$ a chceme ho dostat do superpozice, stačí aplikovat tuto bránu s parametry U $(\pi/2, \varphi, 0)$. Na místo φ lze dosadit jakákoliv hodnota posunu fáze, protože ten výsledek nemění.

Za parametry lze dosazovat jakékoliv číselné hodnoty i ty záporné. Používat ale hodnoty menší než -2π nebo větší, než 2π nemá smysl. Již posun o 2π neprovede žádnou změnu na kvantovém stavu.

3.7.2 Dvouqubitové brány

Jako je tomu i u klasických počítačů i ty kvantové používají logické brány aplikované na dva kvantové bity. Jak je známo všechny z klasických hradel pracujících se dvěma bity, jako AND, OR, NAND, XOR a další, mají pouze a jen jeden bit na výstupu. Proto neplatí

reverzibilita, a jsou tedy ve kvantových obvodech nepoužitelné. Není totiž možné zjistit vstupy, pokud známe jen výstup a typ použitého hradla.
Z toho vyplývá, že kvantové brány mají výstupy dva. Jejich chování a fungování je popsáno v následujících odstavcích.

3.7.2.1 CNOT

Z těch jednodušších bran, pracujících na dvou kvantových bitech, je CNOT neboli kontrolovaná negace z anglického controlled-NOT. Funkce této brány odpovídá velmi přesně jejímu názvu. Brána pracuje na dvou kvantových bitech, ty označujeme jako kontrolní a cílový. Jedná se o bránu NOT, která pracuje pouze tehdy, kdy na druhém, kontrolním qubitu je stav |1⟩. Pravdivostní Tabulka 1 znázorňuje fungování této brány.

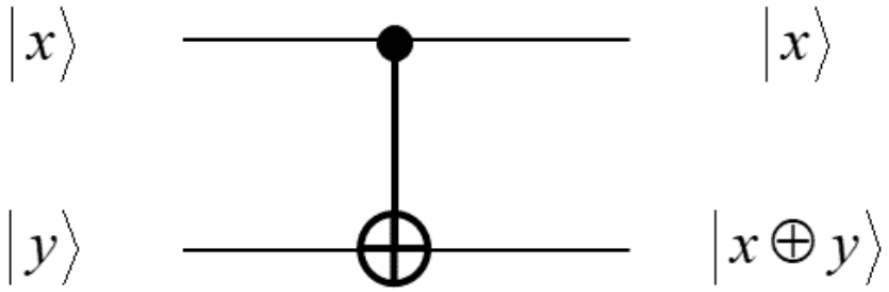
Tabulka 1: Pravdivostní tabulka Kvantové brány CNOT

INPUT		OUTPUT	
Y (NOT)	X (kontrolní)	Y	X
0	0	0	0
1	0	1	0
0	1	1	1
1	1	0	1

Zdroj: Vlastní zpracování

Na Obrázek 3 je vyobrazené schéma zapojení brán CNOT. Kvantový bit Y je negovaný pouze tehdy, kdy se na qubitů X nachází stav |1⟩.

Obrázek 3: Schéma zapojení kvantové brány CNOT



Zdroj: (S., 2019)

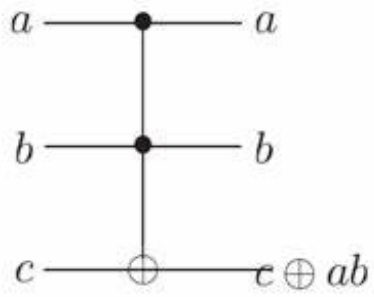
Jak je zobrazeno na předešlém schématu, při aktivaci brány, stavem |1⟩, nevzniká pouze negovaný vstupní stav qubitu |y⟩, ale spojení dvou qubitů, jak již bylo uvedeno za pomocí tenzorového součinu. Pokud je kontrolní qubit v superpozici vzniká provázaný stav těchto dvou kvantových bitů (IBM Corporation, 2016).

Jako každá brána i tato má vlastní maticový zápis, který vypadá tímto způsobem:

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Za zmínku ještě stojí velmi podobně fungující brána. Její název je Toffoliho brána či hradlo, jedná se v podstatě o CCNOT bránu. Funguje na stejném principu jako standardní brána CNOT jen pro negování jednoho z kvantových bitů musí být stav |1⟩ na dvou kontrolních qubitech.

Obrázek 4: Schéma zapojení kvantové Toffiliho brány



Zdroj: (Gajevic, nedatováno)

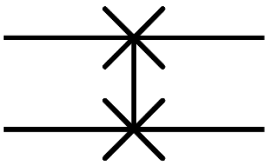
3.7.2.2 Swap

Následující kvantová logická brána je další ze základních bran. Jedná se o bránu aplikovanou na dva qubity. Její název po překladu z anglického jazyka (vyměnit či prohodit) napovídá jakou operaci brána s kvantovými bity provede. Brána tedy prohodí kvantové stavy dvou qubitů. Následuje maticový popis brány SWAP:

$$SWAP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Obrázek 5 znázorňuje schéma SWAP brány.

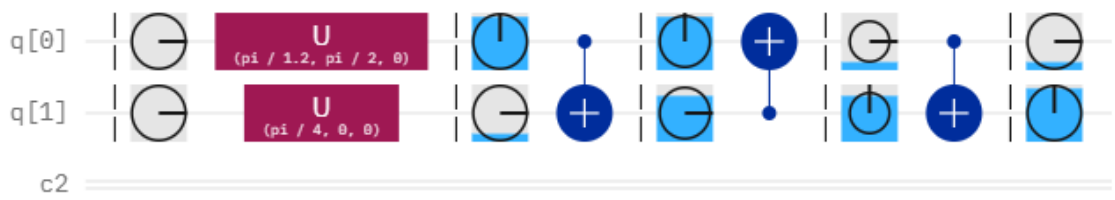
Obrázek 5: Grafická zančka brány SWAP



Zdroj: (Wikipedia contributors, 2011)

Zajímavostí je, že při správném zapojení tří bran CNOT dosáhneme stejné funkcionality jako při použití brány SWAP. Takový obvod lze sestavit na platformě IBM.

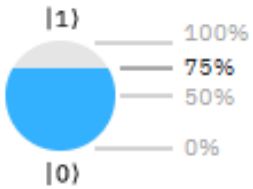
Obrázek 6: Kvantový obvod znázorňující CNOT bránu



Zdroj: (IBM, nedatováno)

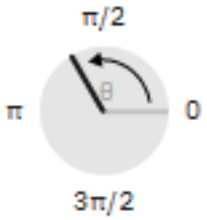
Pomocí bran U je na začátku nastavený stav kvantových bitů. Prvek, který se opakuje v obvodu, na Obrázek 6, častěji a následuje za branou U se nazývá „Phase disk info“. Znázorňuje aktuální stavy na daném místě v obvodu. Na Obrázek 7 níže je možné vidět, jak hladina modré oblasti znázorňuje pravděpodobnost, se kterou bude změřen stav |1⟩. Druhý ukazatel (Obrázek 8) vypadající jako hodinová ručička ukazuje, jak velký fázový posun kvantový bit v daný okamžik má. Posledním z ukazatelů je na obrázku 9 a poukazuje na provázanost qubitů. Čím větší provázání, tím blíže je zvýrazněný kroužek ke středu disku.

Obrázek 7: Ukazatel pravděpodobnosti stavu |1⟩



Zdroj: (IBM, nedatováno)

Obrázek 8: Ukazatel fázového posunu



Zdroj: (IBM, nedatováno)

Obrázek 9: Ukazatel propletení kvantových bitů

Zdroj: (IBM, nedatováno)

A následující brán je už již zmiňovaný CNOT. Po užití každé jednotlivé brány je díky „Phase disk info“ vidět, jak se transformují kvantové bity až nabydou podoby druhého z nich.

3.8 Kvantové algoritmy

Zprvu vypadaly kvantové počítače jako zajímavá vědecká kuriozita a nikdo netušil, zda budou k něčemu užitečné a v jakých oborech by mohly najít své uplatnění. První jiskřička naděje přišla s prvním použitelným algoritmem. Autorem tohoto algoritmu je Peter Shor, jedná se o Shorův faktorizační algoritmus (Kupča, 2001). Následuje výběr ze zajímavých kvantových algoritmů.

3.8.1 Shorův faktorizační algoritmus

Již je nám známo že Shorův algoritmus se zabývá faktorizaci celých velkých čísel. Jedná se o jeden z nejznámějších a nejpoužívanějších kvantových algoritmů.
Existuje tedy číslo N, které není prvočíslem a hledány jsou dvě prvočísla jejichž vynásobením získáme ono číslo N. Platí tedy $X \times Y = N$ a zároveň 1<X; Y<N.
První, co je nutné vědět je fakt, že problém faktorizace čísel je velmi složitá operace k řešení, ale tento problém se dá převést na úkon hledání periody určité periodické funkce (Kupča, 2001). Je nutné tedy vytvořit onu periodickou funkci.

$$f_{y,N}(a) = y^a mod\ N \qquad (17)$$

Použití operace modulo takovou funkci poskytuje. y je náhodně vybrané číslo které není soudělné s číslem N. Zároveň pro něj platí 1 <y < N. Nyní lze dosadit hodnoty do rovnice a zjistit periodu funkce. Délka periody je označována jako r.

Platí následující vztah:

$$y^r \equiv 1 \, mod \, N. \qquad (18)$$

Ten lze upravit do rovnice:

$$\left(y^{r/2} - 1\right)\left(y^{r/2} + 1\right) \equiv 0 \, mod \, N \qquad (19)$$

Jak píše V. Kupča tento vztah platí pouze pro sudé periody, pokud vyjde lichá je třeba vybrat jiné náhodné y. Za pomoci této rovnice se problematika převádí na hledaní nejmenšího společného dělitele. Hledaná čísla jsou společnými děliteli N a $\left(y^{r/2} - 1\right)$ a N $\left(y^{r/2} + 1\right)$. Pro tento úkon existuje vhodný efektivní algoritmus i pro klasický počítač. Vše zní poměrně jednoduše. Nejnáročnější částí tohoto postupu je hledání periody dané funkce. S tím ale značně ulehčuje práci aplikací Shorova řešení se zapojením kvantových výpočtů.

Využívá kvantový paralelismus pro výpočet funkčních hodnot funkce, které hledáme periodu (IBM Corporation, 2016). Po tomto výpočtu na scénu přichází zapojení kvantové Fourierovi transformace. Její hlavní úkolem je na registr vypočítaných hodnot zapůsobit kvantovou interferenci, která je buď konstruktivní pro stavy poblíž hodnoty periody nebo destruktivní pro hodnoty vzdálené. Pokud je perioda nalezena stačí zapojit Euklidův algoritmus a najít již zmiňované největší společné dělitele.

Při použití takzvaného algoritmu hrubé síly, který prochází jednotlivá čísla postupně, je zapotřebí čas $\sqrt{N}$. Zatím co na stejnou úlohu použití Shorova algoritmu vyžaduje pouze čas o délce $\left((\log N)^2(\log\log N)\right)$ (IBM Corporation, 2016). Podle (Patrick J. Coles, 2018) není možné faktorizovat číslo větší než 1000 bitů pomocí klasický algoritmů na klasickém počítači.

3.8.2 Groverův algoritmus

Další z problémů, na který existuje lepší řešení na kvantovém počítači, než na tom klasickém je prohledávaní nesetříděného seznamu. Myšlenka je taková: existuje seznam, kde N je počet položek. Úkolem je naleznout v co nejkratším čase prvek ze seznamu, který je předem známý. Klasickému počítači by tento úkon trval v průměru $\frac{N}{2}$. Bral by prvek po prvku a srovnával s hledaným. Časový úsek by záležel pouze na tom, ve které části seznamu se prvek nachází (začátek/konec). V nejhorším případě je třeba porovnat všech N hodnot (IBM Corporation, 2016).

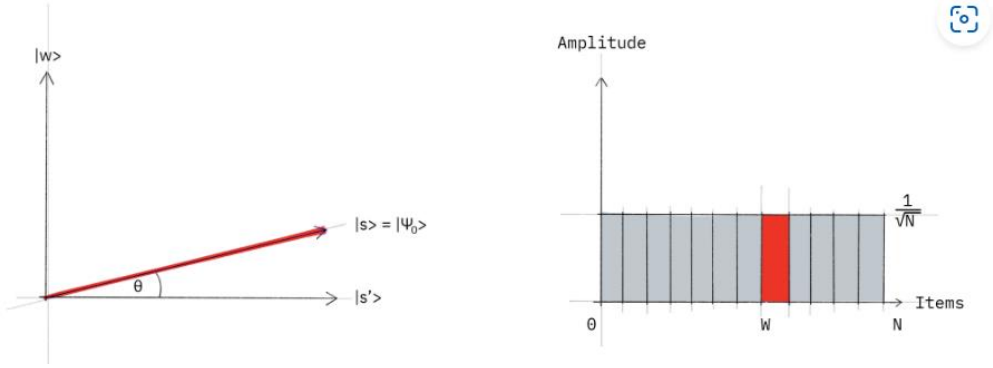
Podstatou tohoto algoritmu je aplikování funkce na všechny hodnoty databáze. Funkce poté označí pozici hledaného prvku. Funkce je definována takto:

$$f(x) = \begin{cases} 1 & if\ x = x^* \\ 0 & if\ x \neq x^* \end{cases} \qquad (20)$$

x označuje jednotlivé hodnoty v prohledávaném seznamu a x^* označuje hodnotu hledanou (Patrick J. Coles, 2018).

Zprvu jsou hodnoty ze seznamu uvedeny do superpozice, tedy změření každé z nich je stejně pravděpodobné. Stav registru lze vidět na Obrázek 10. Pravděpodobnost změření a získání správné hodnoty se rovná $\frac{1}{\sqrt{N}}$.

Obrázek 10: Hodnoty seznamu v superpozici



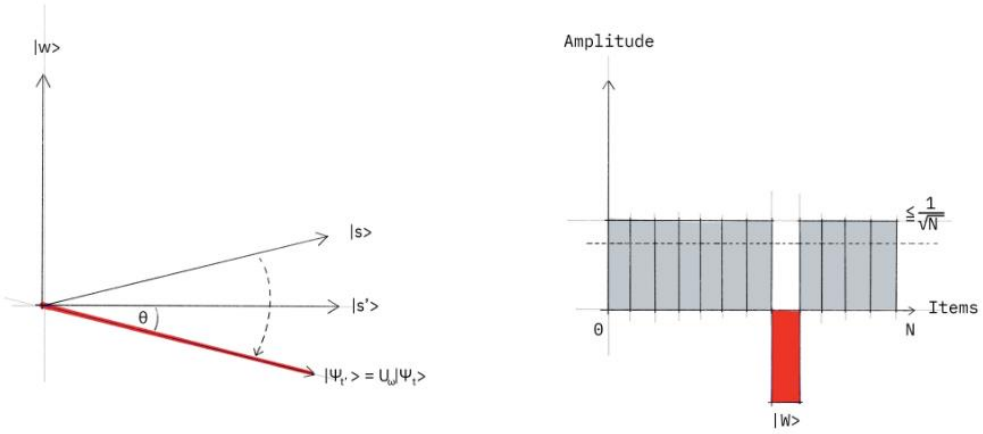
Zdroj: (IBM Quantum, 2021)

Následuje použití operátoru, který se nazývá Orákulum. Následující matice je maticové vyjádření takového orákula.

$$U_w = \begin{bmatrix} (-1)^{f(0)} & 0 & \dots & 0 \\ 0 & (-1)^{f(1)} & \dots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \dots & (-1)^{f(2^n-1)} \end{bmatrix}$$

Je zřejmé, že diagonála matice bude obsahovat samé hodnoty jedna kromě těch kvantových bitů, na kterých se bude nacházet hledaná hodnota. Na takových místech se bude nacházet -1. Tudíž po aplikaci hledaná hodnota bude mít otočenou fázi.

Obrázek 11: Hodnoty seznamu po aplikaci orákula



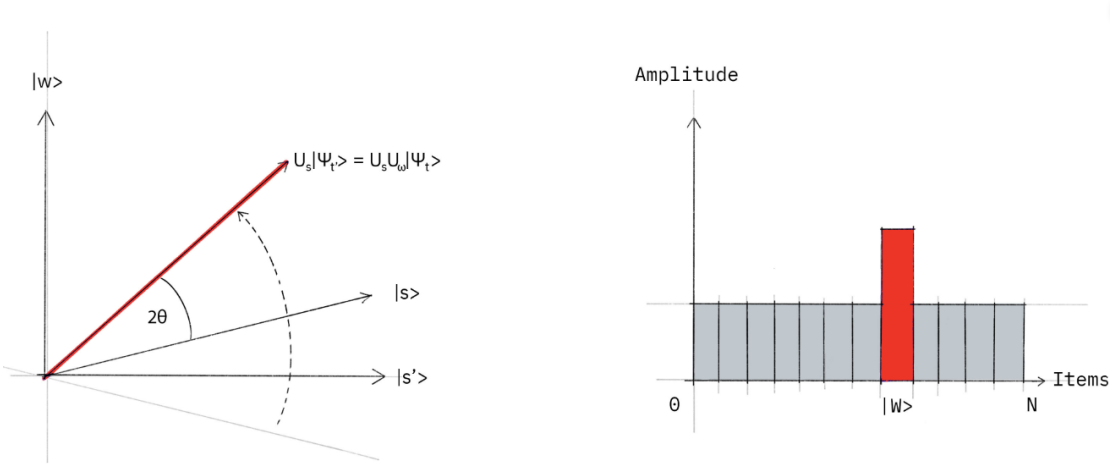
Zdroj: (IBM Quantum, 2021)

Obrázek 11 popisuje, jakým způsobem funguje orákulum. Na Obrázek 10 je původní stav všech hodnot v superpozici se stejnou amplitudou. Na pravé straně je sloupcový graf amplitud stavů. Na Obrázek 11 už je hledaný prvek s posunutou fází. Posun fáze ale nemění pravděpodobnost změření hodnoty. Tudíž tato transformace není dostatečnou pro změření správné hodnoty. Jenže průměrná amplituda pravděpodobnosti bude nižší, protože hledaná hodnota je záporná. Tento pokles je znázorněn na pravém straně Obrázek 11 přerušovanou čarou.

Dalším krokem pro naměření správného výsledku je aplikace dalšího operátoru. Ten může být zapsán jako $(2|\psi\rangle\langle\psi| - I)$ (Phillip R. Kaye, 2007). Operátor otočí amplitudu zpět do kladných hodnot. Po této transformaci je amplituda hledané hodnoty zhruba na dvou až tří násobku původní amplitudy. To lze pozorovat na následujícím Obrázek 12.

To je celá práce tohoto algoritmu. Pouze je zopakován několikrát dokola dokud není amplituda na požadované hladině pro jisté změření správné hodnoty. Počet opakování cyklu je přibližně $\sqrt{N}$. Poté bude jisté, že naměřená hodnota je hodnotou hledanou.

Obrázek 12: Hodnoty seznamu po dokončení prvního cyklu algoritmu



Zdroj: (IBM Quantum, 2021)

3.9 Měření

Již bylo uvedeno, jakým způsobem kvantové stavy a kvantové bity fungují, jak s nimi pracovat nebo ovlivňovat, jak je využívat. Nyní bude blíže přiblíženo, jak dostat výsledné hodnoty. Jedná se o měření.

Podle převládající teorie, která se nazývaná Kodaňská interpretace, kvantový systém zůstává v superpozici tak dlouho, dokud není ovlivněn vnějším světem nebo není pozorován vnějším světem (měřen). Pokud k tomu dojde, superpozice se zhroutí do jednoho z možných výsledných stavů onoho pozorovaného systému. Pokud pracujeme a měříme pouze jeden bit skončí v jednom nebo druhém z možných určitých stavů, tedy 0 či 1.

Z čistě teoretického hlediska se měření dá popsat pomocí Bornova pravidla. Dalo by se říct, že máme kvantový stav x a ortonormální bázi, ve které se nacházejí všechny změřitelné stavy. Poté je definováno že pravděpodobnost naměření určitého stavu, například stavu x_i je vyjádřena vzorcem

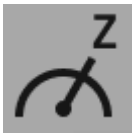
$$P(x_i) = |\langle x_i|x\rangle|^2. \qquad (21)$$

Po provedení tohoto úkonu kvantový stav kolabuje, jak je zmíněno v kapitole 1, do jednoho ze stavů kvantového stavu (Laforest, 2015).

Kvantový počítač je schopný masivního paralelismu, avšak jakékoliv měření způsobuje změnu existujícího celkového kvantového stavu, což tuto výhodu značně omezuje.

Jak již bylo řečeno, kvantové bity mohou být realizovány různými způsoby. Ty na cloudu od společnosti IBM jsou realizovány za pomoci Josephsonova přechodu[3] (Pathak, 2018). Architektura, kterou IBM použila, je známá pod názvem Transmon qubity. Transmon je typem supravodivého nábojového qubitu. Název vznikl ze zkratky anglických slov transmission line shunted plasma oscillation qubit**.** Existují tři základní typy návrhů kvantového bitu pomocí Josephsonova přechodu. Tyto návrhy jsou známé jako nabíjecí qubit, flux qubit a fázový qubit. V nábojových qubitech lze měřit náboj na supravodivém ostrově, např. pomocí jednoelektronového tranzistoru. Vzhledem k tomu, že stavy nábojových kvantových bitů jsou ve vlastní bázi náboje, takové měření poskytuje přímo informaci o stavu daného qubitu (Anton Frisk Kockum, 2019).

Obrázek 13: Grafické označení měření na platformě IBM



Zdroj: (IBM, nedatováno)

Na platformě IBM je měření označeno ikonou, která je vyobrazena na Obrázek 13. Na rozdíl od ostatních bran, které zde jsou aplikovatelné, blok měření není reverzibilní operací.

3.10 Dekoherence

Slovo koherence znamená soudržnost, tudíž dekoherence musí být v zásadě jejím opakem. Vzhledem k tomu, že kvantové počítače jsou realizovány za pomocí nejmenších části na světe jsou to velmi jemná a citlivá zařízení. Tyto subatomární částice mohou být ovlivněny jakýmkoliv druhem vibrací, protože není možné systém dokonale neproniknutelně odizolovat od okolního světa. Kvůli tomu jsou velmi náchylné k chybám. Interakce systému s vnějším světem může vést ke ztrátě soudržnosti a může vést k "dekoherenci" (Zhao, 2022).

Dekoherence má totožný efekt, jako provedení měření, čímž zničíme superpozici a všechny interference během výpočtu, jak je uvedeno v kapitole 3.9. Doba, ve které systém vydrží stabilní a neovlivněný, je závislá na velikosti systému a teplotě okolního prostředí. Jedná se o čase okolo 10^{-20} sekundy.

[3] Jedná se o dva supravodiče s velmi tenkou izolační vrstvou. Mezi nimiž vzniká proud. Jedná se o specifický případ tunelového jevu 1.

Pro kvantové stavy, o nichž nejsou známy kompletní informace, nelze je tudíž popsat superpozicí či vlastním stavem, existuje takzvaný smíšený stav. Takový zápis lze také využít, pokud je systém rušen. Smíšené stavy jsou zapisovány do hustých matic. Například (Kupča, 2001) uvádí ve své práci hustá matice standardního kvantového stavu $|\varphi\rangle = \omega_0|0\rangle + \omega_1|1\rangle$ bude vypadat tímto způsobem:

$$\rho = |\varphi\rangle\langle\varphi| = \begin{pmatrix} |\omega_0|^2 & \omega_0\omega_1^* \\ \omega_0^*\omega_1 & |\omega_1|^2 \end{pmatrix}. \qquad (22)$$

Na začátku bylo uvedeno, že stabilita systému je časově omezená, proto je třeba do matice zahrnout časovou proměnou. Tato proměnná se nazývá dekoherenční čas a značí se τ. Hustá matice potom vypadá následovně: $\rho_t = \begin{pmatrix} |\omega_0|^2 & e^{-\frac{t}{\tau}}\omega_0\omega_1^* \\ e^{-\frac{t}{\tau}}\omega_0^*\omega_1 & |\omega_1|^2 \end{pmatrix}$. Dekoherenci bohužel nejde úplně zabránit, ale existují sofistikované samoopravné mechanismy (Kupča, 2001).

Rušení lze brát jako určitý operátor aplikovaný na kvantový bit. Jakékoliv rušení je možné popsat za pomoci těchto čtyř matic.

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \qquad I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

σ_x způsobuje negaci kvantového stavu, matice je totožná s maticí brány NOT. σ_z mění fázový posun qubitu a σ_y funguje stejně jako předchozí dvě matice dohromady, tedy inverze bitů a k tomu změna fázového posunu. Poslední maticí je matice identity a ta je používána pro vytváření operátorů chyb pro více kvantových bitů. Pro lepší pochopení následuje příklad σ_z na dvou qubitech. Takový operátor vznikne direktivním součinem σ_z a matice identity ($I \otimes \sigma_z$). To znamená:

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}. \qquad (23)$$

První způsob je založený na redundanci, tedy na vícero kvantových počítačích provádíme stejný výpočet. Nazývá se oprava symetrizací. Každý počítač, z R počtu počítačů, má trochu jiný stav, než je ten ideální, díky ovlivňování prostředím. Všechny stavy spojíme, jak je zobrazeno v následující rovnici

$$|\psi\rangle = |\varphi_1\rangle \otimes \ldots \otimes |\varphi_R\rangle \qquad (24)$$

Pokud jsou jednotlivé stavy shodné, nacházejí se pouze na malé části Hilbertova prostoru 3.2. Tato část se označuje jako symetrická. V této části nezáleží na pořadí prvků

tenzorového součinu 3.4. *„Pokud provedeme měření, které je projekcí aktuálního stavu kopií do tohoto ideálního podprostoru, pak bezchybné stavy zůstanou zachovány, kdežto chybové se odstraní.“* (Kupča, 2001). Tato metoda není ovšem ideální na velké chyby, například v podobě prohození bitu, spíše na drobné odchylky od ideálního bez chybového stavu.

Větší chyby lze opravit pomocí strategie zvané kvantové opravné kódy. Jeden logický qubit je zakódovaný pomocí více fyzických kvantových bitů vzájemně závislých. První kvantový opravný kód navržený Peterem Shorem k tomu využíval 9 qubitu a novější metodě navržené Raymondem Laflammem stačí pouze 5.

Postup je jednoduchý. Zprvu jsou stavy zakódovány. Následně vzniknou chyby ve stavech zakódovaných qubitů, načež jsou stavy dekódovány. Určení, jaká chyba nastala a následná transformace bitu do stavu neovlivněného chybou (Kupča, 2001).

3.11 Kvantový paralelismus

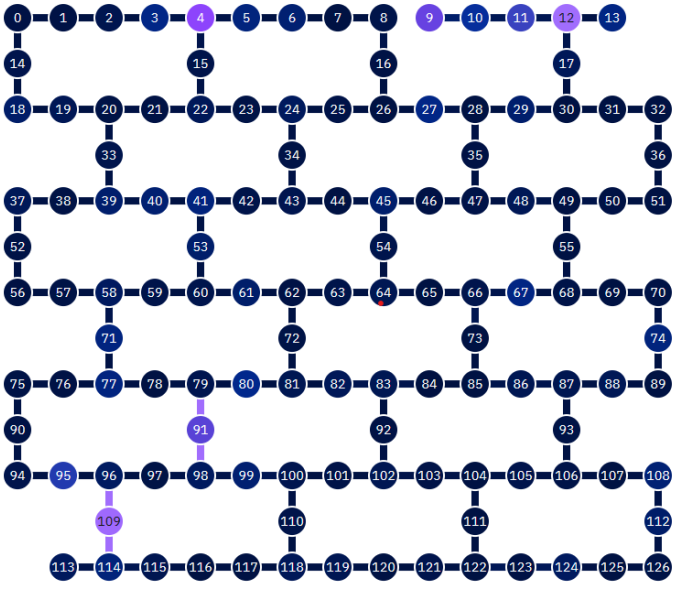
Klasické bity jsou schopné reprezentovat stejné množství informací jako ty kvantové. Například 3 klasické bity jsou schopné obsáhnout čísla z množiny {0,1,2,3,4,5,6,7}. Kvantové bity jsou na tom zcela stejně. Podstatný rozdíl mezi klasický a kvantovým bitem dělá superpozice 1. Kvantové bity mohou totiž nést všech 8 hodnot najednou, což klasický počítač nedokáže. To se dá skvěle využít a může to urychlit mnoho výpočtů. Tato vlastnost je pojmenovaná jako kvantový paralelismus. Díky tomuto jevu kvantové algoritmy poskytují takovou efektivitu tím, že vypočítávají 2^n stavů pouze n počtem qubitů a implementují kvantové výpočty paralelně, nikoliv sériově jako je tomu u počítače klasického (Patrick J. Coles, 2018). Tedy není nutné do funkce dosahovat osm proměnných v osmi krocích, ale dosadí se v kroku jednom. Paralelismus je příčinou exponenciálního zrychlení některých algoritmů. Například Shorův faktorizační algoritmus. Velikost paralelismu je závislá pouze na množství kvantových bitů.

4 IBM QX

Jedná se o projekt společnosti IBM, která byla založena 16. června 1911. Tato technologická společnost se zabývá vším, co se počítačového hardwaru i softwaru týká. Od umělé inteligence, databáze, servery přes sítě, počítačovou bezpečnost až po kvantové počítače. Dále se zabývá výzkumem v těchto oblastech. Díky tomu se může pyšnit ziskem 5 Nobelových cen, 6 Turingových cen, 10 národních medailí v technologické oblasti a 5 národních medailí za vědu (Přispěvatelé Wikipedie, 2023). Samozřejmě ne přímo ale prostřednictvím vědců pracujících pod jejich záštitou. Co se kvantové informatiky týká, zabývají se vývojem kvantových počítačů, kryptografii na kvantové bázi a konečně IBM QX. Písmena QX reprezentují anglická slova Quantum eXperience. IBM QX je označení online platformy, na které si libovolný uživatel může vyzkoušet práci s kvantovým počítačem. Tedy kvantové cloudové výpočetní služby. V dnešní době sdílí společnost IBM 24 kvantových počítačů na cloud. Zde jsou k dispozici zařízení od 5 qubitů až po 127 qubitů.

Na platformě jsou vypsané všechny kvantové počítače a jejich parametry. Například ibm_washington jeden z největších. Pracuje se 127 kvantovými bit, typ jeho procesoru je Eagle r1. Dále zde je možné najít počet výpočtů čekajících ve frontě. Obrázek 14 reprezentuje propojení qubitů daného procesoru a rozdílné odstíny naznačují chybovost jednotlivých qubitů (čím světlejší, tím vyšší chybovost).

Obrázek 14: Mapa kvantového procesoru typu Eagle r1

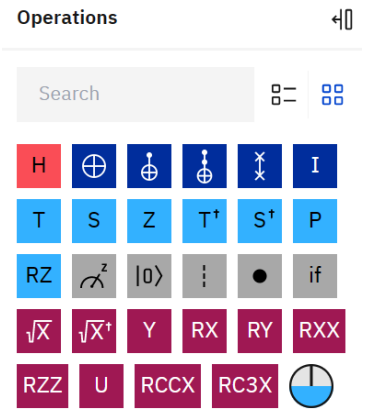


Zdroj: (IBM Quantum, 2021)

Pokud uživatel nechce vyčkávat ve frontě na kvantový výpočet je možnost využít simulátoru kvantového počítače (Pathak, 2018). Simulátory jsou omezené na 10000 s (2,75 hodiny).

Podstatnější funkcionalit platformy je Quantum Composer, ve kterém lze graficky konstruovat kvantové obvody. Platforma je velmi přehledná a intuitivní, jak lze vidět na následujících obrázcích (obrázky 15-18).

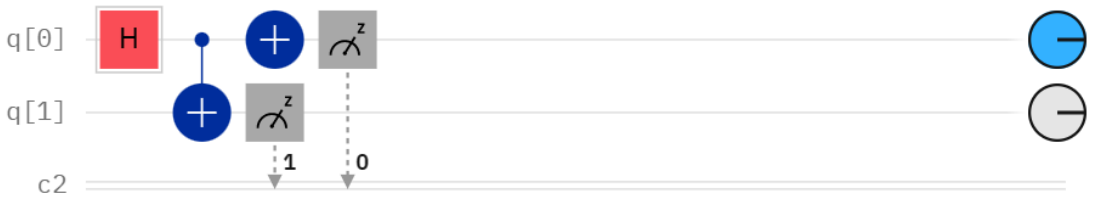
Obrázek 15: Nabídka kvantových bran a operátorů



Zdroj: (IBM, nedatováno)

Obrázek 15 znázorňuje nabídku operátorů obvodu, které lze přetažením aplikovat na jednotlivé qubity obvodu. Obsahuje operátory, které již byly zmíněny, jako Hadamardova brána, NOT, CNOT, Swap, ale také jiné prvky, než jsou kvantové logické brány. Například Phase disk info nebo měření.

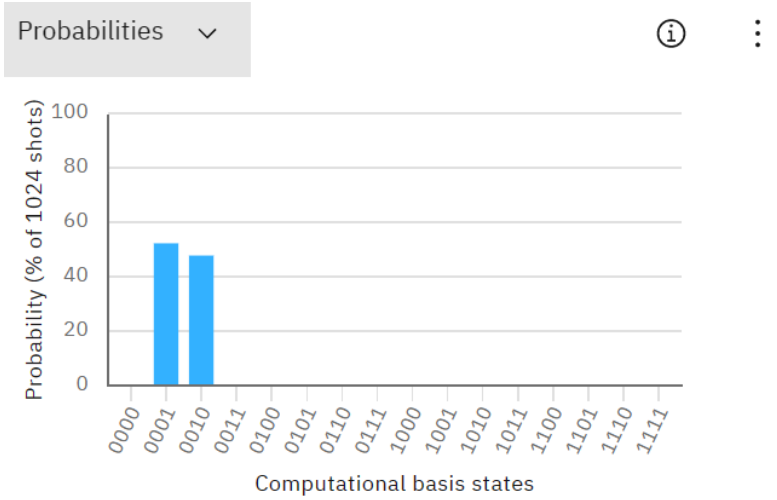
Obrázek 16: Sestavený kvantový obvod



Zdroj: Vlastní zpracování prostřednictvím IBM Composer

Obrázek 16 zobrazuje vzhled kvantového obvodu. Lze zde vidět jednotlivé qubity a brány na nich aplikované.

Obrázek 17: Grafické zobrazení pravděpodobností jednotlivých výsledků



Zdroj: (IBM, nedatováno)

Obrázek 17 ukazuje jedno z nejdůležitějších částí, a to zobrazování pravděpodobností naměření jednotlivých výsledků. Další možností je změnit amplitudu pravděpodobnost. S možností amplitudy je zobrazen maticový zápis výchozího stavu.

Obrázek 18: Zápis kvantového obvodu v OpenQASM 2.0

OpenQASM 2.0

Open in Quantum Lab

```
1   OPENQASM 2.0;
2   include "qelib1.inc";
3
4   qreg q[2];
5   creg c[2];
6   h q[0];
7   cx q[0], q[1];
8   x q[0];
9   measure q[1] -> c[1];
10  measure q[0] -> c[0];
```

Zdroj: Vlastní zpracování prostřednictvím IBM Composer

Poslední důležitou částí je blok převádějící graficky zobrazený obvod do kódu. Samozřejmě funguje i obráceně, pokud je zapisován kód, lze vidět jeho grafickou podobu. Kód je zapsán v programovacím jazyce OpenQASM 2.0 (Open Quantum Assembly Language). Další možností, jak jdou vytvářet kvantové obvody je programování pomocí Qiskit. Jak je uvedeno na jejich webu jedná se open-source software development kit pro práci s

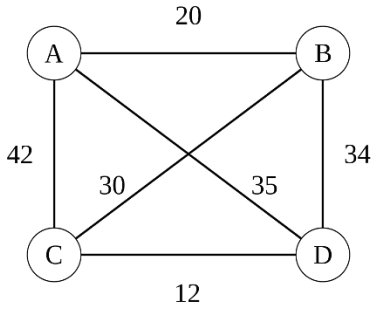
kvantovými počítači na úrovni impulzů, obvodů a aplikačních modulů. Poté mohou být kompilovány do OpenQASM 2.0 a spuštěny na cloudových kvantové systémy IBM.
Dále platforma nabízí velké množství dokumentace nebo výukových programů.

5 Problém k řešení

Pro vyhotovení aplikace bylo zprvu nezbytné určit si, na jaké téma bude tato aplikace. Zadaní jasně určuje, že aplikace musí být založena na bázi platformy IBM QX. Tedy bude využívat sílu kvantových výpočtů. Je nutné zvolit problém, který vyžaduje velkou výpočetní sílu a mohlo by být lepší řešit ho za pomoci kvantových počítačů, nikoliv jen na počítačích klasických.

Jeden z takových problémů se jmenuje Problém obchodního cestujícího. Někdy se také označuje jako Problém čínského listonoše. Zadaní je velmi jednoduché, obchodní cestující vyráží na pracovní cestu z výchozí destinace. Má seznam měst/míst, které musí navštívit. Po tom všem se vrací zpět do výchozího bodu. Úkolem tedy je naleznout pro obchodního cestujícího co nejkratší cestu která spojuje všechny města. Dalo by se říci, že se jedná o logistický problém.

Obrázek 19: Graf problému obchodního cestujícího



Zdroj: (Přispěvatelé Wikipedie, 2022)

Obrázek 19 zobrazuje problém graficky. Body označené písmeny reprezentují jednotlivá města a úsečky mezi nimi cesty, každá i se svou délkou.

Existuje několik algoritmů pro řešení tohoto problému. První z nich je algoritmus postupných permutací. Vytváří všechny možné cesty a z nich vybere tu nejkratší. Vždy najde tu nejkratší cestu. To není u všech algoritmů jisté. Nevýhodou tohoto algoritmu je, že není možné ho provozovat pro více něž patnáct měst, protože už bychom museli porovnat 16! cest. Dalším algoritmem je takzvaný hladový algoritmus. Hledá vždy nejbližší město a takto utváří postupně cestu. V porovnání s předchozím algoritmem je mnohem rychlejší. Ne vždy však najde tu úplně nejkratší cestu. Jeden z mnoha dalších algoritmů pro výpočet tohoto problému se nazývá algoritmus polární trasy s dvojitou optimalizací. Prvním krokem je hledání pomyslného těžiště všech bodů, které je nutné propojit. Následuje výpočet polárních souřadnic. Poté se body spojí do jedné cesty v pořadí podle jejich velikosti úhlu.

Dalším krokem je použití optimalizace. Vertexová optimalizace vyčlení úsek trasy, v němž zhustíme uzly spojnicemi. Poté hledá na takovém zahuštěném úseku cestu s kratší délkou. Pokud nalezneme kratší cestu, než je dosavadní, nahradíme jej za původní a tím dostane i kratší celkovou trasu (Svoboda, 2020). Algoritmus je schopen pracovat s mnohem větším počtem bodů než předchozí algoritmy (cca 200 měst/bodů). Je velmi rychlý, co se výpočtu týká. Ne vždy však nachází nejkratší trasu.

Existují i mnohé další algoritmy na řešení tohoto problému. Mým úkolem bylo zvolit, který z těchto algoritmů využiji ve své práci pro výpočet problému obchodního cestujícího. Mé rozhodnutí mělo dvě hlavní části. Zaprvé, zvolím algoritmus, jehož největším problémem je časová náročnost a pomocí výpočtů na kvantovém počítači se ho budu snažit urychlit. Zadruhé, dle mého názoru pro použití v reálném životě je velmi důležité mít vždy k dispozici tu nejkratší trasu. Tudíž jsem zvolil první z algoritmů, a to algoritmus postupných permutací.

6 Návrh aplikace

Již je zřejmé, jaký problém bude aplikace řešit. Dalším krokem je výběr Jazyka, ve které bude program napsán. Protože aplikace má být na bázi IBM QX je použit Qiskitu jako programovacího jazyku jasnou volbou. Existují ještě další jako Q Sharp (Q#) nebo QCL (quantum computing language), ale nebyl důvod pro jinou volbu. Je založen na programovacím jazyce Python. V Pythonu byla poté napsána celá aplikace.
Bylo vybráno již téměř vše potřebné. Prvním krokem je vytvoření algoritmu pro vyhledávání nejkratší cesty obchodního cestujícího.

6.1 Algoritmus

Algoritmus postupných permutací se skládá ze tří hlavních částí. První z částí je vytvoření všech permutací měst nutných navštívit. Druhou částí je převedení jednotlivých permutací na vzdálenost, která je nutná urazit při cestě daným pořadím. A třetí poslední fáze, tou je hledání té nejkratší cesty.

6.1.1 První část

Tuto část jsem se rozhodl vyřešit za pomoci klasického počítače. Vstup do této části je seznam měst. Ten je v mé aplikaci realizovaný listem, který uchovává indexy měst, která musejí být navštívena během cesty. Názvy u měst jsou nahrazeny indexy, pro usnadnění práce a přístupu k nim. Poté jsem pomocí cyklu for nalezl všechny permutace, které připadají v úvahu. Následuje ukázka kódu.

Ukázka kódu 1: Hledání všech permutací

```
for x in cisla:
	if len(cisla)==1:
		seznam.append(str(x))
		break
	else:
		for y in cisla:
			if len(cisla)==2:
				if x == y:
					continue
				else:
					seznam.append(str(x)+str(y))
					break
			for z in cisla:
				if len(cisla)==3:
					if x == y or x == z or y == z:
						continue
					else:
						seznam.ap-
pend(str(x)+str(y)+str(z))
					break
```

Výstupem tohoto kódu je list, označovaný v kódu jako seznam všech permutací, pokud na vstupu přijdou tři města. Toto je jen ukázka. Aplikace je schopna spočítat permutace i pro více měst. Na vstupním seznamu se nenalézá výchozí a zároveň konečný bod. U toho je jasné, že jeho pořadí na cestě budě první a poslední. Tento kód tedy hledá permutace tří měst, ale navštívena budou města čtyři. Například na vstup přijde list s dvěma hodnotami [2, 3, 4]. Výstupem aplikace tedy bude list s hodnotami [234, 243, 324, 342, 423, 432].

6.1.2 Druhá část

Ve druhé části je převáděny permutace na vzdálenost. Pro tuto část je zásadní Tabulka 2: Vzdálenosti mezi městy. Pro přehlednost jsem si tabulku vytvořil prostřednictvím aplikace Excel. Vybral jsem sedm měst rovnoměrně rozložených po celé České republice. Každému z nich jsem přidělil jeho index.

Tabulka 2: Vzdálenosti mezi městy

Index		0	1	2	3	4	5	6
	Města	České Budějovice	Praha	Brno	Liberec	Ostrava	Hradec Králové	Plzeň
0	České Budějovice	null	148	213	250	380	253	134
1	Praha	148	null	208	110	373	117	92
2	Brno	213	208	null	309	170	171	298
3	Liberec	250	110	309	null	437	101	203
4	Ostrava	380	373	170	437	null	242	466
5	Hradec Králové	253	117	171	101	242	null	208
6	Plzeň	134	92	298	203	466	208	null

Zdroj: Vlastní zpracování

Takováto tabulka je velmi přehledná, ale v kódu je nepoužitelná. Je nutné převézt ji tak, aby byla čitelná pro program. Ideálním formátem pro uloženích těchto dat je dvourozměrné pole.

Ukázka kódu 1: Dvourozměrné pole vzdáleností

```
vzdalenosti = [
                    [0, 148, 213, 250, 380, 253, 134],
                    [148, 0, 208, 110, 373, 117, 92],
                    [213,208, 0, 309, 170, 171, 298],
                    [250,  110, 309, 0, 437, 101, 203],
                    [380,  373, 170, 437, 0, 242, 466],
                    [253,  117, 171, 101, 242, 0, 208],
                    [134,  92,   298, 203, 466, 208, 0]
        ]
```

Ještě předtím, než dojde k samotnému počítání délky dané trasy, je třeba přidat výchozí/konečný bod cesty. Budu pokračovat v příkladu z kapitoly 6.1.1. Řekněme, že

výchozí bod bude město s indexem 0. Je nutné tento index přidat na začátek a konec každé cesty. Po této drobné úpravě bude list vypadat takto [02340, 02430, 03240, 03420, 04230, 04320]. Poté přichází na řadu práce s : Dvourozměrné pole . Aby byl program schopný vypočítat celkovou vzdálenost, stačí brát postupně dvojice čísel z jednotlivých položek listu. Pro ukázku dvojice první varianty cesty jsou [02, 23, 34, 40]. Tyto dvojice se postupně dosazují jako indexy do dvojrozměrného pole vzdálenosti. To okamžitě nabývá hodnoty vzdálenosti dvou měst, které jsou reprezentovány čísly. Takovýmto způsobem jsou vypočítány délky všech tras z listu permutací. Výstupem této části je list číselných hodnot.

U této části není co zrychlovat, tudíž nemá smysl snažit se zapojit kvantový počítač a výpočty jsou provedeny prostřednictvím klasického počítače.

6.1.3 Třetí část

Konečnou částí algoritmu postupných permutací je nalezení nejkratší trasy, kterou tento algoritmus nalezne vždy. Vstupem do této fáze je list vzdáleností tras. Hledání v seznamu je časově náročný úkon, tudíž se pokusím ho zjednodušit pomocí zapojení kvantového počítače a jeho výpočtů. Jak byl v kapitole 3.8.2 popsáno již existuje jeden velmi známý a funkční algoritmus na prohledávání seznamů. Jenže tento algoritmus hledá v seznamu prvek, který je předem známý. Nebylo by možné aplikovat funkci na stavy v superpozici, když nám není známá hodnota x*. To znamená, že pro tuto situaci není vhodný.

Klasické počítače by tento problém řešily tím způsobem, že by vzaly dvě vzdálenosti, porovnaly je a tu menší z nich vyřadily ze seznamu možných kandidátu na nejkratší trasu. To znamená, co krok to vyřazení jedné cesty.

Pokusím se tedy vymyslet, jakým způsobem by šli porovnávat hodnoty pomocí kvantového bitu. Pokud chceme naměřit některou hodnotu, je nutné zvýšit její amplitudu pravděpodobnosti a tím i pravděpodobnost jejího zachycení. Z kapitoly 3.7 o kvantových branách je možné zvolit vhodnou bránu pro tento úkon. V úvahu připadají U3 a Brána RY, což je v podstatě modifikovaná Pauliho brána. Lze na ní nastavovat velikost rotace. Má volba padla na bránu U3 kvůli její univerzálnosti.

Dalším problémem je převedení vzdáleností do kvantového bitu. Myšlenka je tedy taková, že délku cesty bude reprezentovat otočení qubitu na blochově sféře směrem k jednomu z pólů. Budeme tedy měnit úhel θ. Při porovnání dvou hodnot, první hodnotou se bude qubit otáčet blíže k severnímu pólu a druhou hodnotou obráceně tedy k pólu jižnímu.

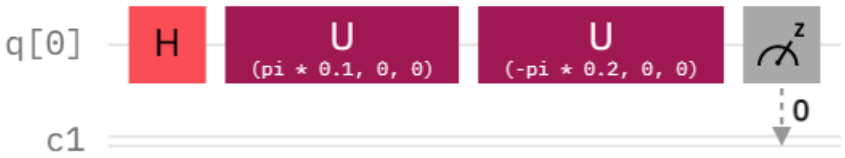
První věc, která je třeba udělat, je nastavit kvantový bit do výchozí pozice pro posuny. Touto pozicí je kvantový stav: $|\varphi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. Protože po aplikaci rotací bude nejlépe poznat rozdíl mezi jejich velikostmi.

Řekněme, že máme dvě hodnoty vzdáleností a označíme je X a Y. První hodnota vzdálenosti X bude otáčet qubit nahoru, a bude tedy aplikovat kladné hodnoty úhlu θ. Druhá hodnota Y tedy musí působit obráceným směrem. Úhel θ bude tedy aplikován v záporné hodnotě.

Z toho vyplývá, že pokud bude $X = Y$ rotace se vzájemně vyruší a při měření dostaneme obě hodnoty se stejnou pravděpodobností. Druhou z možností je stav, kdy $X > Y$, tehdy posun úhlu θ ve výsledku bude v kladných hodnotách. Tudíž se zvýší pravděpodobnost naměření stavu $|0\rangle$. Poslední možností je stav, kdy hodnoty vzdáleností budou $X < Y$, a to bude mít za výsledek zvýšení pravděpodobnosti naměření stavu $|1\rangle$. Pokud by stav $|1\rangle$ byl změřen víckrát znamená to, že hodnota Y je vetší, tudíž bude ze seznamu délek odstraněna.

Takové zapojení by vypadalo jako Obrázek 20. Za hodnoty jsem dosadil $X = 0{,}1$ a $Y = 0{,}2$.

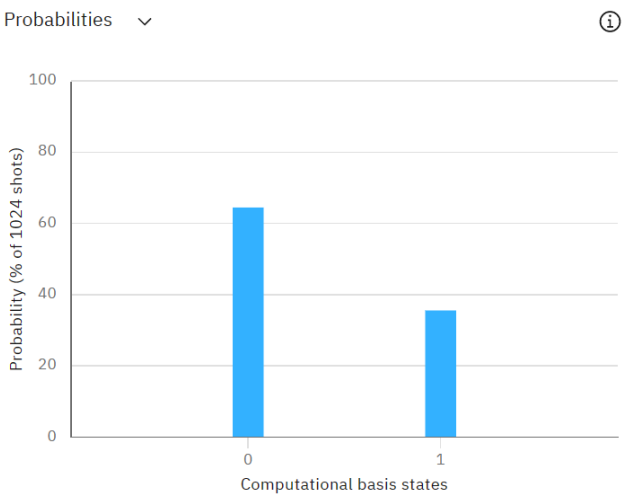
Obrázek 20: Porovnání dvou hodnot pomocí brány U3



Zdroj: Vlastní zpracování prostřednictvím IBM Composer

Výsledky takového zapojení vypadají následovně (Obrázek 21: Výsledky porovnávání). Výsledky ukazují že změření stavu $|0\rangle$ je 64,45 % a stavu $|1\rangle$ zbylých 34,55 %. To znamená, že z více jak jedné třetiny dostaneme špatný výsledek. Řešením tohoto problému je provézt celou operaci vícekrát.

Obrázek 21: Výsledky porovnávání



Zdroj: Vlastní zpracování prostřednictvím IBM Composer

Jak lze vidět na Obrázek 21 vedle osy y je napsáno pravděpodobnost v procentech při 1024 střelách. To značí, kolikrát byl obvod změřen. Z předchozích kapitol je známo, že po měření se změní kvantový stav. Zde jsou stavy pouze dva a systém zaznamenává, kolikrát je změřen který stav. V tomto případě by mělo být jasno ohledně správného výsledku už od tří střel.

Pokud se využije na porovnaní pouze jeden kvantový bit bude vyhledávání nejmenší hodnoty ze seznamu trvat stejně dlouho jako při zapojení pouze počítače klasického. Při každém kroku by kvantový počítač vyřadil pouze jednu hodnotu.

V tom je tedy to zrychlení. Ale co kdyby toto zapojení bylo aplikováno ne na jeden, ale na více qubitů.

Všechny kvantové bit počítače pracují paralelně a nečekají vzájemně na sebe. Pokud se toto zapojení aplikuje na dva kvantové bity je možné během jednoho kroku porovnávat čtyři číselné hodnoty najednou. Tedy v jednom kroku vyřadit dvě hodnoty z listu. Nyní si představme, že IBM vlastní kvantový počítač disponující 433 kvantovými bity, jak je zmíněno v kapitole 2. To by znamenalo vyřazení 433 hodnot během jednoho kroku.

Ve své aplikaci jsem použil pouze 3 kvantové bity pro porovnání hodnot. Rozšířit výpočet a další qubity není problém. Jediné, co je složitější při zapojení většího množství kvantových bitů je počet střel/měření systému. Při malých rozdílech ve vzdálenostech jsou malé rozdíly i v pravděpodobnostech. A potom je těžší získat správné výsledky. Tomu pomůže zvýšeno počtu střel. Má aplikace využívá 2000 na 3 qubity.

Ukázka kódu 2: Vytvoření kvantového obvodu

```
if len(pole)<6:
	qr = QuantumRegister(1, 'q')
	cr = ClassicalRegister(1, 'c')
else:
	qr = QuantumRegister(3, 'q')
	cr = ClassicalRegister(3, 'c')
obvod = QuantumCircuit(qr, cr)
```

Ukázka kódu 2 znázorňuje vytváření kvantového registru se třemi kvantovými bity a se třemi klasickými. Klasické bity slouží k měření obvodu. Pokud není splněna podmínka, porovnávaných hodnot je méně než šest, vytvořený obvod se bude skládat pouze z jednoho a jednoho bitu.

Ukázka kódu 3: Aplikace bran a měření na kvantový obvod

```
if len(pole)<6:
	obvod.h(qr[0])
	obvod.u((-pole[0])*pi, 0, 0, qr[0])
	obvod.u((pole[1])*pi, 0, 0, qr[0])
	obvod.measure(qr[0], cr[0])
else:
	obvod.h(qr[2])
	obvod.h(qr[1])
	obvod.h(qr[0])
	obvod.u((-pole[0])*pi, 0, 0, qr[2])
	obvod.u((-pole[2])*pi, 0, 0, qr[1])
	obvod.u((-pole[4]])*pi, 0, 0, qr[0])
	obvod.u((pole[1])*pi, 0, 0, qr[2])
	obvod.u((pole[3])*pi, 0, 0, qr[1])
	obvod.u((pole[5])*pi, 0, 0, qr[0])
	obvod.measure(qr[2], cr[2])
	obvod.measure(qr[1], cr[1])
	obvod.measure(qr[0], cr[0])
```

Ukázka kódu 3 v programu aplikuje brány, a nakonec i měření na kvantový obvod vytvořený v Ukázka kódu 2. Podmínka plní stejnou funkci jako v předchozím kódu. Tomuto kroku ještě předchází transformace délek tras. Parametr brány U3, který je použit pro porovnávání hodnot je uveden v radiánech, tudíž maximální hodnota, o kterou můžeme posouvat je π/2. Pokud bychom otočili o víc, pravděpodobnost by zase začala klesat a fázový posun by nabyl hodnoty π. Tudíž rozmezí, na kterém lze zaznamenávat porovnávané vzdálenosti je 0 až π/2. V dekadických hodnotách se jedná o hodnoty od 0 do 1,57. Protože se jedná o Českou republiku stačí vzdálenost v kilometrech vydělit 1000, protože nejdelší vzdálenost mezi dvěma městy je přibližně 630 km. Limit této metody je ovšem 1570 km.

Ukázka kódu 4: Získání výsledků z obvodu

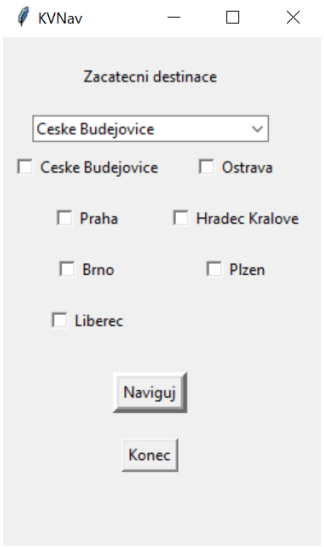
```
simulator = Aer.get_backend('qasm_simulator')
job = execute(obvod, simulator, shots=2000)
pocty = job.result().get_counts()
return pocty
```

Ukázka kódu 4 vytváří simulátor, na kterém bude měřený kvantový obvod. Dále se v této části nastavuje počet střel do kvantového obvodu (zde jsou nastaveny 2000). Výstupem tohoto kódu je list výsledných stavů a u každého z nich, kolikrát byl naměřen. Poté jsou vyhozeny hodnoty, které byli naměřené jako větší. Tento proces se opakuje, dokud nezbude poslední hodnota a tou je nejkratší cesta obchodního cestujícího.

6.2 Vizualizace

Pro vytvoření grafického uživatelského rozhraní aplikace jsem využil tkinter. Je velmi jednoduchý a multiplatformní. Zajistí požadovaný vzhled podle operačního systému. Vzhled úvodního okna ukazuje Obrázek 22.

Obrázek 22: Úvodní okno aplikace



Zdroj: Vlastní zpracování

V úvodním okně je možné si vybrat výchozí město požadované cesty a jaká města je třeba navštívit během cesty. Potom už jednoduše stačí kliknout na tlačítko naviguj.

Výsledkem algoritmu je z předchozí kapitoly 6.1 je řetězec indexů měst srovnaných za sebou tak aby vytvořili nejkratší cestu. Výsledné okno aplikace zobrazí pořadí měst. Již vypsaných pod jejich jmény tak, jak jsou známé, nikoliv jako indexy. Obrázek 23 ukazuje mapu, na které je zobrazována výsledná cesta. V tomto případě se jedná o cestu s výchozím bodem v Praze a města nutná navštívit jsou Brno, Hradec Králové, Liberec, a Plzeň.

Poslední informace zobrazená na konečné stránce je údaj o celkové délce trasy v kilometrech.

Obrázek 23: Výsledkové okno

Zdroj: Vlastní zpracování

7 Vyhodnocení

Tato bakalářská práce měla za cíl seznámení s problematikou kvantového počítání i počítače. Dalším z cílů bylo představení platformy IBM QX a vytvoření jednoduché aplikace na její bázi. Kapitola 3 se věnovala osvětlení kvantové mechaniky a informatiky, jejich principů a fungování. Následovala kapitola 4, která se věnovala pouze již zmiňované platformě společnosti IBM. A na konec Zvolení problému k řešení a tvorba aplikace. A samozřejmě návrh algoritmu pro řešení daného problému, který byl poté použit v aplikaci.

Práce ukazuje fakt, že Kvantová informatika je velmi silný nástroj pro řešení mnohých problémů. Jen je třeba naleznout problémy, pro které budou její specifické vlastnosti přínosem. Rozhodně nelze každý problém řešit lépe pomocí kvantového počítače, mnoho problémů je snadnější řešit prostřednictvím počítače klasického.

Aplikaci se sestavit povedlo a je zcela funkční. Algoritmus řešící problém obsahuje kvantovou složku. Aplikace plní požadavky, pro které byla stvořena. Jen se domnívám, že aplikace není zcela použitelná v reálné praxi. Ze začátku nebyl zvolen vhodný algoritmus pro řešení problému obchodního cestujícího, který byl následně vylepšen kvantovou složkou. Zrychlení se dá lépe ocenit při větším množství měst, která je nutná navštívit. Bohužel je algoritmus postupných permutací limitovaný na počet 15 měst. Další nevýhodou, pro použití v praxi, je jasně dané menu předvolených lokalit. Po přepracování by mohla aplikace naleznout uplatnění v logistice.

8 Citovaná literatura

Aaserud, F. (20.. Únor 2023). *Niels Bohr*. Načteno z Encyklopedie Britannica: https://www.britannica.com/biography/Niels-Bohr

Anton Frisk Kockum, F. N. (2019). Quantum Bits with Josephson Junctions. *Materials Science*.

Gajevic, D. (nedatováno). *Quantum computers gates circuits and programming quantum gates 2*. Načteno z Slidetodoc: https://slidetodoc.com/quantum-computers-gates-circuits-and-programming-quantum-gates-2/

IBM. (9.. Listopad 2022). *IBM představuje kvantový procesor 400 qubit-plus a novou generaci IBM Quantum System Two*. Načteno z IBM: https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two

IBM. (nedatováno). *composer*. Načteno z IBMQuantum: https://quantum-computing.ibm.com/composer/files/new

IBM Corporation. (2016). *composer*. Načteno z quantum-computing.ibm: https://quantum-computing.ibm.com/composer/docs/iqx

IBM Quantum. (2021). *Grover's algorithm*. Načteno z IBM Quantum Lab: https://quantum-computing.ibm.com/lab/docs/iqx/guide/grovers-algorithm

IBM Quantum. (2021). *Resources*. Načteno z IBM Quantum: https://quantum-computing.ibm.com/services/resources?system=ibm_washington

Jae-weon Lee, C. H. (2002, Září 16). Qubit geometry and conformal mapping. Tedžon, Korea.

Kockum, A. F. (Prosinec 2014). *Bloch sphere representation of the state of qubit*. Načteno z ResearchGate: https://www.researchgate.net/figure/The-Bloch-sphere-representation-of-a-qubit-The-basis-states-are-located-at-the-north_fig2_284259345

Kulhánek, P. (2016). *TF2: Kvantová teorie.* Praha: AGA.

Kupča, V. (23.. leden 2001). *Teorie a perspektiva kvantových počítačů*. Načteno z Teorie a perspektiva kvantových počítačů: https://www2.karlin.mff.cuni.cz/

Laforest, M. (2015). *The mathematics of quantum mechanics.*

Lluis Masanes, Y.-C. L. (29. Březen 2007). *All bipartite entangled states display some hidden nonlocality.* Načteno z rxiv: https://arxiv.org/abs/quant-ph/0703268v1

Pathak, A. (16. květen 2018). Experimental quantum mechanics in the class room: Testing basic ideas of quantum mechanics and quantum computing using IBM quantum computer. Noida, Uttarpradéš, Indie.

Patrick J. Coles, S. E. (10.. Duben 2018). Quantum Algorithm Implementations for Beginners. Los Alamos, New Mexico, USA.

Phillip R. Kaye, R. L. (2007). An Introduction to Quantum. New York, Spojené státy americké: Oxford University Press Inc.

Přispěvatelé Wikipedie. (8. Září 2022). *Problém obchodního cestujícího*. Načteno z Wikipedie: Otevřená encyklopedie: https://cs.wikipedia.org/w/index.php?title=Probl%C3%A9m_obchodn%C3%ADho_cestuj%C3%ADc%C3%ADho&oldid=21657536

S., M. (9. Listopad 2019). *input and output qubit notation in quantum gates*. Načteno z stackexchange: https://quantumcomputing.stackexchange.com/questions/8444/input-and-output-qubit-notation-in-quantum-gates

Svoboda, A. (Únor 2020). Trasa obchodního cestujícího v GIS (algoritmy řešení). *ArcRevue*, stránky 42-45.

Tyc, T. (2006). Základy kvantové mechaniky. Brno, Jihomoravský kraj, Česká republika.

Vyvlečka, M. (23. 2 2020). *Kvantový počítač, když jedničky a nuly přestávají stačit.* Načteno z Science & Technology Club: https://www.youtube.com/watch?v=arSYHeXXJJ0

Wikipedia contributors. (11. Březen 2011). *Quantum logic gate*. Načteno z Wikipedia: https://en.wikipedia.org/wiki/Quantum_logic_gate

Zhao, Y. (2022). Quantum Computing and Communications. Londýn, Spojené království: IntechOpen.

Seznam obrázků

Seznam ukázek kódu

Seznam tabulek

Summary and keywords

The development of quantum computers is moving forward constantly. In recent years, people have been looking to utilize the enormous power of quantum computing. The main problem is the difference between creating programs or applications on quantum computers and also on standard computers. The hardest part of creating an application is the algorithmization of the problem which we want to solve. It is already clear that quantum computing isn't only the future of computer science, but also it can solve unknown facts from other fields such as physics and chemistry.

There are now only a few quantum computers in the world and they are inaccessible to the public, therefore, IBM launched a project called IBM Quantum Experience (IBM QX) in May 2016. The main idea of the project is the sharing of the computing power of a quantum computer. Users can try their own program for free using several Qubits (the basic unit of quantum information).

The thesis identifies the possibilities of using the IBM QX platform and describes the advantages and disadvantages of quantum computers.

Keywords: quantum computing, quantum algorithm, IBM QX, quantum bit

Přílohy:

Kód aplikace