Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Hálózati Rendszerek és Szolgáltatások Tanszék

Kertész Gergő Csaba

KVANTUM NEURÁLIS HÁLÓZAT TESZTELÉSE SZIMULÁTORON

KONZULENS

Dr. Bacsárdi László

BUDAPEST, 2019

Tartalomjegyzék

Összefoglaló

A kvantuminformatika és a gépi tanulás két nagyon népszerű terület a mai informatikában. Mindkettő informatikusok százait foglalkoztatja, és teljes potenciáljuk egyelőre nem is látható. A gépi tanulás többek között az intelligens rendszerek, BigData és IoT területeken is rendkívül fontos téma, viszont az implementációk nagyon számítás igényesek, némely tanulási algoritmus akár napokat vagy heteket is igénybe vehet, ezért jó néhány – pl. az online learning architektúra – még meg sem valósítható. Az előbbieket figyelembe véve nem is csoda, hogy próbálkozások történtek a két terület összefűzésére. A kvantuminformatika által nyújtott számítási sebesség és különleges tulajdonságok jelentős előrelépést nyújthatnak a gépi tanulás területén. Mindezek ellenére még nem sok kvantum tanuló algoritmus implementáció létezik, mivel mindkét területnek megvannak a maga kihívásai.

A kvantuminformatika fejlődésének egyik nagy akadálya a hardverkomponensek ára, különösen a kvantumprocesszoré. Jelenleg csak néhány cég gyárt kvantumszámítógépet, ezek közül a legismertebbek a D-Wave termékei, de az áraik átlag felhasználónak nem megfizethetőek. Erre a problémára egy megoldást jelentenek a kvantumáramkör szimulátorok, ezekből van néhány elérhető az interneten, de nagyrészt már nem támogatottak, vagy fizetősek, esetleg nehezen használhatóak.

A TDK munkám során egy saját, open-source szimulátort mutatok be, amin kvantum neurális hálózat architektúrákat vizsgálok. A szimulátor képességei között grafikus adatmegjelenítés is van, ami a legtöbb mostani szimulátorból hiányzik. Az elméleti része a TDK dolgozatomnak pedig a szimulátoron futtatott neurális hálózatok teljesítményének analizálásáról fog szólni.

Abstract

Quantum informatics and machine learning is two of the most popular fields of today's information science. Hundreds of engineers and information scientists research both of these fields, and the potential benefits are yet to be seen. Machine learning is a large and important part of BigData and IoT systems, however, most of the current implementations require huge amounts of computational power which creates a bottleneck on the improvements of these fields. Some machine learning algorithms can even take days or weeks to complete and this makes some of the architectures – like on-line learning – impractical. Considering the points above, it's not surprising that there have been attempts to combine the advantages of these fields. The computational power and other unique features of quantum informatics can greatly improve the potential of machine learning algorithms, and make the aforementioned bottleneck disappear. In spite of the opportunities that the combination of quantum informatics and machine learning can provide, there haven't been many implementation of quantum learning algorithms, mainly because of the unique challenges that quantum informatics faces.

One of the biggest obstacles in the advancement of quantum informatics is the price of the hardware components, a large part of this price is the procedures to create (and cool) a quantum processor. Up to date, there are few companies that deal in quantum computers, the most known one of them is D-Wave, but these computers are not accessible for many, due to their prices. A half solution to this problem, considering the advancement and possibilities of machine learning, is the existence of quantum circuit simulators. Simulators are inexpensive, and even though they can't simulate the computational power on a non-quantum computer, using the results of experiments, we can approximate how efficient an algorithm would be on a real quantum computer. There are already existing simulators available on the internet, but most of them lacks something that is essential to an educational environment, some are no longer supported, some have too high prices or aren't intuitive to use.

In my work, I will present my own, open-source simulator, on which I will use to examine the efficiency and feasibility of some quantum neural network architectures. Among the capabilities of my simulator is data imaging, which is, in my opinion, highly lacking in most available simulators.

1 Bevezető

A tudományban – így az informatikai területen is - sok olyan kísérlet van, mely a gyakorlatban vagy egyáltalán nem, vagy csak nagyon körülményesen és drágán valósítható meg, ilyen helyzetekben legtöbbször szimulátorokhoz fordulunk. Tipikusan ilyen terület a kvantuminformatika gyakorlata is. Jelen TDK dolgozat témája egy ilyen szimulátor megalkotása, és azon egy neurális hálózat futtatása. Ebben a fejezetben röviden bemutatom a dolgozathoz szorosan kapcsolódó területek mai állását, valamint a dolgozat felépítését.

1.1 Kvantuminformatika mai állása

A kvantuminformatika gyakorlata és elmélete még közel sem teljes. Rendszeresen jelennek meg új cikkek különböző platformokon, és a kvantumszámítógépek területén is folyamatos fejlődés mutatkozik. A gyakorlatban a fejlődési irány arra mutat, hogy a D-Wave hamarosan elveszti a kvantum számítógépek feletti monopóliumát, ahogy egyre több cég tör be a piacra. Idén (2019) jelentette be az IBM első kereskedelembe bocsátott architektúráját, az IBM Q System One-t [1]. Lassan pedig ipari verseny kezd kialakulni az egy chip-re ültetett kvantumbitek számában: a Google és az Intel tavaly (2018) erősítette meg, hogy dolgoznak egy 72-qbites (Google – Bristlecone[2]) és egy 49-qbites szupravezető (Intel – Tangle Lake[3]) chip-en.

Elméleti alapokon az egyik legizgalmasabb kérdés jelenleg a kvantum hibajavítás (Quantum Error Correction – QEC). A hibajavítás legalább olyan fontos a kvantuminformatikában, mint a klasszikus informatikában, a csatornák zajossága mellett a kvantum információra veszélyes a dekoherencia is. A terület annyira tárgyalt, hogy nemzetközi konferenciát is tartanak róla, sőt 2014-ben felvetődött az elmélet egy laboratóriumban, hogy a tér-idő maga is egy kvantum hibajavító kód [4].

Magyarország is beszállt a kvantuminformatika kutatásba több fronton is. Folyamatban van a HunQuTech[5] program, mely Magyarország felzárkóztatását célozza meg a kvantumtechnológia területén. A HunQuTech programban a BME is részt vesz, például a „*Kvantumbitek előállítása, megosztása és kvantuminformációs hálózatok fejlesztése* projektben [6]. Az egyetemen ezenkívül önálló kvantuminformatikai kutatások is folynak a Hálózati Rendszerek és Szolgáltatások Tanszék Mobil Kommunikáció és

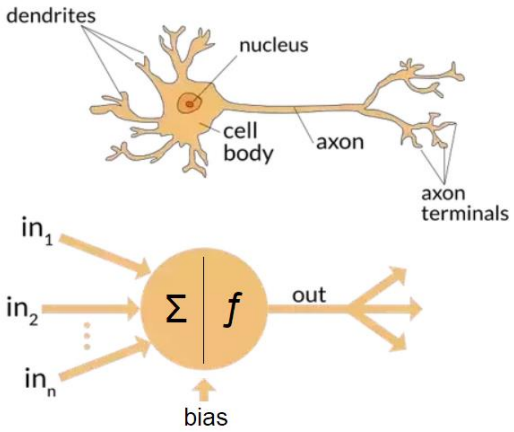
Kvantumtechnológiák Laboratóriumban, mely rendelkezik fizikai kvantumeszközökkel is.

1.2 Gépi tanulás mai állása

A gépi tanulás egy másik jelenleg népszerű és rohamosan fejlődő területe az informatikának, a terület nevét Arthur Samuel adta 1959-ben [7]. Mivel a terület jelentős részének gyakorlatában nem előfeltétel a legújabb, vagy még nem is létező hardver, a gépi tanulás gyakorlata elterjedtebb és gyorsabban tud fejlődni, legalábbis a területek jelenlegi állapotában. Az igen alacsony hardver követelmények miatt, ami az elméletek gyakorlati kipróbálásához szükséges a gépi tanulás elméletével is nagyon sokan foglalkoznak. Az alkalmazási terület igen széles, néhány érdekesebb: BigData feldolgozás, IoT, intelligens város és képfeldolgozás.

1.2.1 Neurális hálózatok

A gépi tanulás egyik legnépszerűbb típusa a neurális hálózatok. A neurális hálózatok az informatika többi részétől igen különböző tulajdonsága az, hogy a legtöbb architektúra a biológiából merít ihletet. Az egyik első neurális architektúra a Rosenblatt Perceptron [8], mely az idegsejtet modellezi. Az *1. ábra* megfigyelhető az analógia a két struktúra között, az idegsejt dendritjeit modellezzük egy bemeneti vektorral, amelyet összegzünk a sejttestben és egy úgynevezett aktiváló függvényen keresztül kivezetjük a kimenetre amely a neuron axon-ját modellezi.



1. ábra Biológiai neuron és az általa inspirált architektúra

A terület nem mindig élvezte a rivaldafényt, volt időszak, amikor csak nagyon kevesen foglalkoztak neurális architektúrák kidolgozásával, miután Marvin Minsky és

Seymour Papert a terület lehetőségeit túlságosan korlátozottnak könyvelték el publikációjukban [9]. Az ez utáni lappangó érdeklődés rövidesen újra felújult, mikor a multilayer perceptronok és az error feedback tanulási algoritmusok a tévesen felállított korlátokat megdöntötték.

1.2.2 Evolúciós algoritmusok

Az evolúciós algoritmusok a gépi tanulás egy másik egyedi területe. Az algoritmusok alapvetően az evolúciót modellezik (innen is ered az elnevezés), ahol az egy adott célra legmegfelelőbb egyedek tulajdonságai maradnak fenn a jövő populációjában. Az algoritmusok az evolúció más tulajdonságait is átveszik a lehető legjobb eredmény garantálása érdekében, mint például a mutációt.

A terület 1950-ben ütötte fel a fejét, és 1960-ra három különböző interpretáció is elterjedt. Az egyik fő alkalmazási terület az optimalizáció. Optimalizáción belül különösen gyakran alkalmaznak evolúciós algoritmusokat nagy komplexitású gráfelméleti problémák megoldására, az egyik legközismertebb ezek közül is az utazó ügynök probléma.

1.2.3 Korlátok

Az eddig elmondottak alapján talán még nem egyből világos, hogy a gépi tanulás területének miért is van szüksége bármilyen más területre. A gépi tanulás területének egyik legnagyobb korlátjára eddig azért nem került fény, mert főleg az elméleti részét mutattam be, ha a gyakorlatra ki is tértem, az csak az elmélet verifikációjára vonatkozott, tehát nem tértem ki a legfontosabb részre: a mindennapi alkalmazásra. Mivel a gépi tanulást a gyakorlatban legtöbbször olyan feladatokra alkalmazzuk, ahol nagyon nagy adathalmazt szeretnénk hatékonyan feldolgozni, vagy egy komplex feladatot szeretnénk valós időben (realtime) megoldani. A most említett feladatok megoldásának az árát valamikor meg kell fizetni, a gépi tanulás ezt mindössze előrehozza a tanulási fázisba, így az élesben való problémamegoldást már gyorsan végzi. Egy komplexebb feladatra előállított gépi tanuló architektúra tanulási fázisa viszont órákig, napokig akár hetekig is eltarthat a jelenleg rendelkezésre álló hardvereken.

1.3 A dolgozat felépítése

A **Bevezető** egy rövid áttekintést adtam a dolgozat témájának két fő területéről, azon belül is a dolgozat szempontjából fontos kérdésekre és részterületekre kitérve.

A téma validálása című fejezetben a dolgozat főbb témáit szeretném indokolni, bemutatom, hogy miért fontos és előnyös a kvantuminformatikát és a gépi tanulást együtt is vizsgálni, valamint a Quantum Toy Box (QTB) bemutatott szimulátor szükségességét.

A **harmadik** fejezet a megvalósított szimulátoromat mutatja be, elmélettel indokolva az egyes tervezési döntéseket, valamint az elkészült program működésének helyességét fogom bemutatni néhány elméleti alapokon fekvő kvantumáramkör próbafuttatásával.

A **negyedik** fejezetben megvizsgálok néhány most is elérhető kvantumáramkör szimulátort és összehasonlítom őket a sajátommal.

A **Kvantum** gépi tanulásról általában fejezetben bemutatom a kvantum gépi tanulás, mint a két terület összefogása, fejlődését és mai állapotát, majd megvizsgálom és bemutatom a megvalósításra kiválasztott architektúrákat.

A **Egy** szimpla kvantum perceptron fejezetben a választott architektúra futásának eredményeit szeretném bemutatni, és megmutatni a kvantum neurális hálók előnyét a klasszikus párjaikkal szemben.

Végezetül összefoglalnám a dolgozatot és röviden tárgyalnám a szimulátorom és a projekt jövőjét, lehetőségeit.

2 A téma validálása

Ebben a fejezetben a dolgozat témájának fontosságát indoklom: miért is érdemes a kvantum gépi tanulással, valamint a kvantum áramkörök szimulálásával foglalkozni.

2.1 Kvantum gépi tanulás előnyei

A bevezetőben bemutattam a kvantuminformatika és a gépi tanulás területeinek néhány előnyét és korlátját, ezekből esetleg már érezhető, hogy hol és mennyire tudja kiegészíteni a gépi tanulás területét a kvantuminformatika. A gépi tanulás egyik legnagyobb problémája a mindennapi gyakorlatban a számításigény. Vegyük például a neurális hálózatokat. A legtöbb neurális hálózat architektúra nagy mennyiségű egyszerű feladatokat megoldó egységet foglal magába, melyek a legoptimálisabb tanuló algoritmusokkal is sok ezerszer fognak műveletet végezni, míg a súlyaik a kívánt állapotba kerülnek. Ezeknek a feldolgozó egységek részhalmazai általában párhuzamosíthatóan működnek, ennek a párhuzamosításnak a megvalósítása klasszikusan nagy erőforrásokat igényel, míg ha a számolásban az összefonódás, szuperpozíció tulajdonságait kihasználjuk, a klasszikus futásidőt drasztikusan csökkenthetjük. A futásidő csökkenésére példa Grover algoritmusa, mely használható rendezetlen adatbázisban való keresésre mely a klasszikus $O(N)$ futásidőt elméleti síkon $O(\sqrt{N})$-re csökkentette, ahol N a lekérdezések száma.

Egy másik igen fontos előnye a kvantuminformatikának, hogy tenzor műveleteket nagyon hatékonyan lehet megoldani, 2009-ben jelent meg egy publikáció, mely bemutatott egy algoritmust, ami logaritmikus futásidővel tud megoldani lineáris egyenletrendszereket[10]. A tenzor műveletek sok gépi tanuló algoritmus alapja, sok architektúra felírható egy tenzor rendszerként, és a tenzorokkal lehet számolni az architektúrák eredményeit, valamint az algoritmus is a tenzorok módosításával tanítható. A kvantum gépi tanulás területének részletesebb taglalására még visszatérünk egy későbbi fejezetben.

2.2 Kvantuminformatika és szimulátorok

A kvantuminformatika gyakorlata, ahogy a bevezetőben is említve volt, igen korlátozott egyelőre, melynek fő oka a hardveres megvalósítás ára. Kezdeményezések

vannak egy kereskedelembe bocsátható kvantumszámítógépre, de a mai napig megvalósított gépeket egyelőre csak nagy cégek vásárolták, mert csak nekik vannak meg a finanszírozási képességeik hozzá. Egy másik nagy hátránya a jelenlegi architektúráknak a mérete, melynek legnagyobb faktora a processzort megfelelő hőmérsékleten tartó hűtőrendszer. A klasszikus számítógépek története alapján következtethetünk arra, hogy a kvantumszámítógépek mérete is folyamatosan csökkenni fog, de addig is a legtöbb gyakorlati problémára egy kompromisszum a működés szimulálása. Szimulátorokat nagy sikerrel alkalmaznak ma is drága eszközök vizsgálatára vagy tervezésre, például egyetemi környezetben digitális áramkörök vizsgálatára.

Egy kvantumáramkör szimulátor klasszikus gépen is tudja emulálni egy kvantum áramkör működésének logikáját, habár igen erőforrás igényesen, valamint egy véletlenszám-generátor minőségétől függően. Egy helyesen implementált kvantumáramkör szimulátoron az elméletben megalkotott algoritmusokat és áramköröket olcsón lehet tesztelni és analizálni.

2.2.1 Futásidő problémája

A legnagyobb hátránya a kvantumáramkör klasszikus szimulálásának, hogy a futásidőt nem tudjuk reprodukálni – ha tudnánk, már nem lenne biztonságos az RSA -, mivel egy kvantumregisztert $f*2^n$ klasszikus biten tudunk reprezentálni, ahol n a regiszterben lévő kvantumbitek száma és f egy komplex szám ábrázolásához szükséges bitek száma, és utána ugyanennyi klasszikus bittel kellene műveleteket végezni, ami bár bizonyos szinten párhuzamosítható, elég nagy n-re esélytelen. Tehát, habár még az összefonódást is tudjuk szimulálni a matematikai leírásának köszönhetően, algoritmusok pontos futásidőjének analízisére egy szimulátor nem alkalmas, bár következtetések alapjául szolgálhat.

3 Quantum Toy Box (QTB)

A szimulátoromat Quantum Toy Box-nak(QTB) neveztem el, mivel a célja az, hogy könnyen lehessen vele kísérletezni, tanulni, esetleg oktatni. A teljesen elkészült szimulátor open-source és github-on elérhető lesz. Jelen dolgozat írásakor a program kvantum áramkörök szimulálására teljesen alkalmas, de néhány funkciót még hozzá fogok adni, mielőtt publikusan is elérhetővé tenném github-on.

3.1 Áttekintés

A szimulátort C++ nyelven írtam a lehető legnagyobb hatékonyság érdekében, mivel a kvantuminformatika szimulálása közben nagyméretű mátrixokkal számolunk, ami a minél optimálisabb memóriafelhasználást kívánja. Fejlesztőkörnyezetnek a Microsoft Visual Studio 2019-et használtam, mivel ezen eszköz kezelésében van a legtöbb tapasztalatom.

A program – ahogy azt a **2. ábra** is szemlélteti – három funkcionális rétegből áll: matematikai alapfunkciók, kvantuminformatikai szabályok és konstrukciók, áramkör építési eszközök. A matematikai alapfunkció réteg tartalmazza a komplex szám megvalósítást, valamint a vektor és mátrix műveleteket, melyek fontosak a kvantuminformatika szempontjából. A kvantuminformatikai réteg a szimulátor fő logikai része, mely a kvantuminformatika konstrukcióit tartalmazza, korlát és szabályellenőrzéssel. Az áramkör építő réteg tartalmazza az áramkörépítő funkciókat, valamint a megjelenítést megvalósító kódot.

2. ábra A szimulátor struktúrájának vázlata: A három fő funkcionálisan elkülöníthető rész, valamint azokon belül a főbb funkciók.

3.2 Felépítés

Ebben az alfejezetben bemutatom a QTB felépítését, a tervezői döntéseimet indokolva és alátámasztva kvantuminformatikai elmélettel, így a szimulátor leírása közben röviden ismertetni fogom a kvantuminformatika alapjait is.

3.2.1 Komplex számok és lineáris algebra

A kvantuminformatika egyik sajátos tulajdonsága, hogy a kvantumbitek reprezentálására komplex számokat is használunk. Implementáció során az egyik első tervezői döntés az volt, hogy már megírt könyvtárat használok, avagy „újra feltalálom a kereket". Végül az előbbi mellett döntöttem, a már korábban is említett hatékonyság érdekében. A kvantuminformatikai alkalmazás szempontjából nem volt szükségem a komplex számok összes tulajdonságára, ezért egy teljes komplex könyvtárat felhasználni felesleges lett volna, valamint a jövőben a szimulátort képessé teszem emberileg értelmezhetőbb matematikai kifejezésekkel való számolásra is ($1 + \sqrt{2}$-vel számol 2.4142 helyett). Végül tehát írtam egy saját, kvantuminformatikai szempontból tervezett komplex szám könyvtárat.

A lineáris algebrával kapcsolatban hasonló kérdés merült fel, és a válasz is ugyanaz lett, megéri többlet időt áldozni egy saját lineáris algebra könyvtár megírására, mely a célra van optimalizálva, mint egy általános könyvtárat használni, mely a felhasználás szempontjából fölösleges extra funkciókkal rendelkezik. A kvantuminformatika szempontjából a legfontosabb eszköz a lineáris algebrából a mátrix műveletek, és a programban a reprezentáció is ezt mutatja. A könyvtár legnagyobb része a különböző mátrixműveletek különféle módon optimalizált kódja.

3.2.2 Kvantumregiszterek

A kvantumbit és kvantumregiszter fogalmát a kvantuminformatika 1. és 4. posztulátuma definiálja:

1. ***Posztulátum (Állapottér):*** *Zárt fizikai rendszer állapota leírható egy v komplex együtthatójú és egység hosszú állapotvektorral, mely egy V Hilbert*[1] *tér része.*

[1] Komplex lineáris vektor tér mely rendelkezik belső szorzattal.

Tehát egy kvantumbit leírható egy kétdimenziós vektorral:

$$\boldsymbol{v} = [a, b]^T = a\mathbf{0} + b\mathbf{1}$$

Ahol $\mathbf{0} = [1,0]^T$ és $\mathbf{1} = [0,1]^T$a V Hilbert tér ortonormális bázisvektorai és $a, b \in \mathbb{C}$. Az egységnyi hossz betartása érdekében, kikötjük még, hogy $|a|^2 + |b|^2 = 1$. Az állapotvektor koordinátáit gyakran hívják valószínűségi amplitúdóknak, mivel méréskor annak a valószínűsége, hogy a rendszert valamelyik klasszikus állapotra mérjük, megegyezik az adott együttható négyzetével. Kvantumbitek esetén gyakran használjuk a bra-ket jelölést, például „ket0" = |0> és „bra0" = <0| , a ket és bra jelölés között a kapcsolat: |φ> = $(<\varphi|)^{\dagger}$, ahol a † jel a vektor esetén a komplex konjugált transzponáltat jelöli.

4. Posztulátum (Kompozit rendszer): *Egy W kompozit fizikai rendszer állapottere kiszámolható az alkotó rendszerek tenzor szorzatával:* $W = V \otimes Y$. *Valamint, ha V állapottér* $\boldsymbol{v}$ *és az Y állapottér* $\boldsymbol{y}$ *állapotban van, akkor a kompozit rendszer állapota* $\boldsymbol{w}$ *kiszámolható:* $\boldsymbol{w} = \boldsymbol{v} \otimes \boldsymbol{y}$

A negyedik posztulátum által leírt kompozit rendszer állapot vektorát nevezzük kvantum regiszternek. A fentiek alapján egy kvantumbit reprezentálása egészen magától értetődő, egy kettő magas komplex számoszlop, mely koordinátáinak négyzetösszege mindenképpen egyenlő egyel. Ez a lineáris algebra könyvtár alkalmazásával egyszerűen megoldható, viszont egy kvantumbit viselkedése nem sokban különbözik egy kvantumregiszterétől a szimulátorom szempontjából, így egy kvantumbit egy 1 bites kvantumregiszterként értelmezhető. Kvantumbitek kompozit rendszere pedig kiszámítható a bitek Kroenecker szorzatával.

3.2.2.1 Tiszta és kevert állapotok, sűrűség mátrix

Az első posztulátum egy úgynevezett **tiszta állapotot** ír le, melyet felfoghatunk ortonormális bázisvektorok kvantum lineáris kombinációjaként melyek valószínűségi amplitúdókkal vannak súlyozva. Egy **kevert állapot** egy kvantumrendszer lehetséges kvantumállapotait írja le az adott állapot valószínűségével súlyozva, tehát a kevert állapot tiszta állapotok klasszikus lineáris kombinációja valószínűségekkel súlyozva. Szemléletes példának vegyük mondjuk a $|\varphi> = \frac{1}{\sqrt{2}}|0> + \frac{1}{\sqrt{2}}|1>$ és $|\phi> = 1\,|0>$ tiszta állapotokat. Ekkor kvantumrendszer, mely a fenti két állapot valamelyikében van, például leírható a következőképpen: $|\vartheta> = p_1|\varphi> + p_2|\phi>$, ahol p_1 és p_2 klasszikus

valószínűségek, tehát $p_1 + p_2 = 1$ és $0 \leq p_1, p_2 \leq 1$. A definíciókból következik, hogy egy $|\varphi>$ tiszta állapot értelmezhető úgy, mint egy kevert állapot, melynek az egyetlen lehetséges állapota a $|\varphi>$ 1 valószínűséggel.

A leírtakból látszik, hogy a kevert állapot leírható egy *<tiszta állapot, valószínűség>* párokból álló vektorral, valamint az is, hogy a tiszta állapotok helyett elég kevert állapotokat ábrázolni.

A tiszta és kevert állapotok ismerete a sűrűség mátrix fogalmának bevezetése miatt fontos. Egy kvantumrendszer sűrűség mátrixa a rendszert tömören reprezentálja, és a sűrűségmátrixon keresztül minden információt kinyerhetünk a rendszerről. A sűrűség mátrix egy pozitív szemidefinit mátrix, melyet a következőképpen számolhatunk ki:

$$\mathbf{p} = \sum_i p_i |\psi_i >< \psi_i|$$

Ahol **p** a sűrűség mátrix $|\varphi_i>$ az i-edik rendszer állapot p_i valószínűséggel, a braket jelöléssel pedig a $|><|$ jelölés a Kroenecker szorzatot jelöl a komplex konjugált transzponáltal . Mivel már tudjuk, hogy egy tiszta állapot kifejezhető kevert állapottal is, így a sűrűség mátrix képlete is leegyszerűsödik, $|\psi>$ tiszta állapotra:

$$\boldsymbol{p} = |\psi >< \psi|$$

A programban a sűrűségmátrix kiszámolása tehát igen egyszerű feladat, egyszerűen a regiszterben tárolt állapotnak vesszük a valószínűségével súlyozott Kroenecker szorzat összegeket.

Felmerülhet olyan eset, ahol két kvantum rendszer sűrűség mátrixa megegyezik, ekkor a két állapot megkülönböztethetetlen. A sűrűség mátrixon végzett műveletek között kiemelt fontosságú a nyom operátor. Ha egy sűrűség mátrixnak vesszük a nyomát, az mindig egy lesz: $\mathrm{Tr}(\mathbf{p}) = 1$. Amennyiben viszont a sűrűség mátrix négyzetének a nyomát vesszük, az csak tiszta állapotok esetén lesz egy: $\mathrm{Tr}(\mathbf{p}^2) = 1$, míg kevert állapotok esetén kisebb lesz, mint egy: $\mathrm{Tr}(\mathbf{p}^2) < 1$.

A sűrűség mátrix bevezetése után a felső két posztulátum összefoglalható:

1. ***Posztulátum:*** Mivel egy állapot sűrűség mátrixa ugyanannyi információt tartalmaz mint az állapotvektor, bármilyen kvantum rendszer, mely p_i valószínűséggel van $\mathbf{p}_i$ állapotban, teljes sűrűség mátrixa*:* $\boldsymbol{p} = \sum_i^k p_i \boldsymbol{p_i}$.

4. ***Posztulátum:*** Ha vannak független alrendszereink, $|\psi_i>$-vel reprezentálva, akkor az összetett rendszer leírható $\otimes_i \boldsymbol{p_i}$, ugyanígy számolható ki a $\{\mathbf{p_i}\}$ által reprezentált alrendszerek kompozit rendszere

3.2.2.2 Összefonódás és redukált sűrűség mátrix

Az előző alfejezetben definiált fogalmak egyik haszna a szimulátor szempontjából a kvantum összefonódás reprezentálása és ellenőrzése. A kvantum összefonódás (avagy Einstein szavaival élve: „Spooky action a distance") a kvantuminformatika egyik legkülönösebb és leghasznosabb tulajdonsága. Képzeljünk el két pénzérmét, amelyek között valamilyen módon létrehozunk egy kapcsolatot úgy, hogy az egyiket a hátunk mögé tesszük a kezünkben, a másikat feldobjuk és megnézzük az eredményt. Ha a feldobott pénz fej, akkor a kezünkben tartott pénz is fejjel felfele van, ha pedig írás, akkor a kezünkben tartott pénz is írást mutat. Ez a jelenség pedig a kapcsolat létesítése után megmarad addig, amíg meg nem nézzük, hogy mit mutat a feldobott pénzérme, akármilyen messzire is visszük egymástól a két pénzérmét. Első ránézésre úgy tűnik, hogy elcsaltuk a relativitás alaptörvényét, vagyis, hogy információ nem utazhat gyorsabban a fénynél, ez a kvantum informatika többi törvénye miatt azonban nem igaz.

Bár a fizikai szimulálása az összefonódásnak egy hosszasabb munka lenne, a matematikai viselkedésének reprezentálása már nem olyan nehéz. Ha belegondolunk egy kompozit rendszerbe, az alapból szimpla kvantumbitek tenzor szorzatával jön létre, ebből következnie kellene, hogy egy kompozit rendszer dekompozíciója után vissza kaphatjuk az eredeti szimpla rendszereket. Ez az esetek többségében igaz, viszont kvantumállapotok, melyeknek nem létezik dekompozíciója, vegyük például a következő állapotot:

$$\frac{1}{\sqrt{2}}|00> + \frac{1}{\sqrt{2}}|11>$$

Ez a kvantumállapot előállítható kvantumkapukkal, viszont nincs két olyan 1 kvantumbites tiszta rendszer, melyből ez előállítható. A fentebb leírt érmés gondolatkísérlet belátható az eddig bemutatott információkból, ha a rendszer egyik kvantumbitjét megmérem, az 0.5 valószínűséggel 0 lesz, 0.5 valószínűséggel pedig 1. Ezután a második kvantumbit a kompozit rendszer összeállításának módjának következményeként, kénytelen 0-ra váltani, ha 0 eredményt kaptunk az első biten, és 1 eredményt adni, ha 1-et mértünk az első biten.

A dekompozíció általános vizsgálata nagyon erőforrás igényes, itt jön be az egyik fontos felhasználása a sűrűségi mátrixoknak. Annak ellenére, hogy egy összefonódott állapotot lehetetlen felbontani tiszta állapotok tenzorszorzatára, ha a részrendszerek állapotától nem várjuk el, hogy tiszták legyenek, a feladat már közel sem olyan megoldhatatlan: egy kompozit rendszer sűrűségmátrixából ki lehet számolni a tagrendszereket sűrűség mátrixait, úgy, hogy a nem kívánt részrendszereket kiszűrjük, így kapjuk meg a redukált sűrűség mátrixot. A sűrűség mátrix egyszerre reprezentálhat tiszta és kevert állapotokat is. Mivel egy összefonódott állapot nem bontható fel tiszta állapotokra, csak kevert állapotokra, a sűrűségmátrix nyomaira vonatkozó megkötésekből ki lehet deríteni, hogy egy állapot összefont-e vagy sem: ha van egy kvantumrendszer |ψ> akkor, ha $Tr\left(p_{|\psi>}^2\right) = 1$ akkor a rendszer állapota tiszta. Ezután megnézzük, hogy az összes redukált sűrűség mátrixra szintén teljesül a fenti feltétel, akkor tudjuk, hogy a rendszer tiszta állapotban és nem összefonódott, hiszen egy tiszta állapotról van szó, ami felbontható tiszta állapotokra, így nem lehet összefonódott. Ha az eredeti rendszerre teljesül a kritérium de egy részrendszerére nem, akkor pedig egy tiszta állapotú rendszerről van szó, mely nem bontható fel tiszta állapotokra, tehát összefonódott.

Ahogy az kiderült, a redukált sűrűségmátrixból meg lehet határozni egy rendszer összefonódottságát, ezután már csak a redukált sűrűségmátrix meghatározása maradt meg feladatnak, ezt a következő összefüggés alapján lehet meghatározni:

$$Tr_B(\boldsymbol{p}) = \sum_{|b> \in B} (I_A \otimes < b|)\boldsymbol{p}(I_A \otimes |b>)$$

ahol <b| és |b> a bra és ket vektorai a B rendszer ortonormális bázisvektorai, míg I_A az identitás operátor az A rendszer elemeinek.

3.2.3 Kvantumkapuk

Egy kvantumkapu fogalmát a kvantuminformatika második posztulátuma alapján vezethetjük be:

2. ***Posztulátum (evolúció):*** *Bármely fizikai rendszer időbeli evolúciója karakterizálható unitér transzformációkkal, melyek csak az evolúció kezdési és befejezési idejétől függ.*

Az evolúció eredeti formája a Schrödinger egyenlet, viszont a lineáris algebrai reprezentáció az elterjedt a kvantuminformatika szempontjából. A reprezentációja egy U

unitér operátornak egy **U** mátrix, melynek az U_{ij} eleme a feltételes valószínűségi amplitúdó mely összekapcsolja a bemeneti **j** bázis vektort az **i** bázis vektorral.

A szimulátorom szempontjából a kvantumkapuk tehát mátrixok, melynek az unitér tulajdonságát minden új kapunál ellenőrizni kell. A kapuk mátrixainak unitérségét a definícióval ellenőrzöm, mely a következő: $U^{\dagger}U= UU^{\dagger} =I$, ahol I az identitás operátor. Miután egy kvantumkapu mátrixreprezentációját megalkottuk, az operátor alkalmazása a kvantum regiszteren egy egyszerű mátrix szorzás kérdése.

Bebizonyították, hogy 3 elemi kvantumkapu segítségével az összes kvantum operáció emulálható[11], tehát a szimulátoromba is elég volt beégetni ezeket, az egyszerűbb felhasználás érdekében azonban további operátorokat is elérhetővé tettem, melyeknek száma a jövőben is bővülni fog.

A jelenleg elérhető operátorok a következők:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

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

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

$$P(\alpha) = \begin{pmatrix} 1 & 0 \\ 0 & e^{j\alpha} \end{pmatrix}$$

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

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

Ezek közül a három elemi kapu, amelyből minden más kapu megépíthető, a következő: a P(45°), Hadamard (H) és CNOT-kapuk. A CNOT-kapu az összefonódás, a Hadamard kapu a szuperpozíció és interferencia míg a P(45°) kapu a komplex interferencia megvalósításáért felelős. Az X,Y,Z és P kapukat Pauli kapuknak is nevezik.

3.2.3.1 Dekompozíció

A kvantumregisztereken való műveletvégzés megvalósításánál a legfontosabb tervezői döntés az áramkörbe csatolás miatt az volt, hogy a kvantumbiteken külön, vagy csak együtt, regisztereken lehet műveletet végezni. Az összefonódás jelensége miatt a külön-külön műveletvégzés komplikáltabb és erőforrás igényesebb út lett volna, hiszen akkor kevert állapotokon is kellett volna műveleteket végezni, annak ellenére, hogy a kevert állapot a kvantuminformatikában ritkán használt fogalom. Tehát az operátoroknak az egész kvantumregiszteren kell műveletet végezni akkor is, ha egyébként csak egy kvantumbiten szeretnénk őket alkalmazni. Ennek a problémának létezik megoldása, a kvantum operátorok mátrixreprezentációjából előállíthatóak azon operátorok, mely a kívánt biteken a kívánt műveletet elvégzik.

Az egybites kapuk esetében egyszerű dolgunk van. Legyen |ψ> = a|0> + b|1> kvantumbitünk, ezen az X operátort végrehajtva az eredmény:

$$X|\psi> = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} * \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} b \\ a \end{pmatrix}$$

Tehát az eredmény a |0> és |1> bázisvektorok valószínűségi amplitúdójának megcserélődése. Legyen |φ> = c|0> + d|1> egy másik kvantumbit, és nézzük meg a kompozit rendszereket:

$$|\psi> |\varphi> = ac|00> + ad|01> + bc|10> + bd\,|11>$$

$$(X|\psi>)|\varphi> = bc|00> + bd|01> + ac|1> + ad|11>$$

A feladatunk tehát egy olyan U operátor keresése, melyre a következő egyenlet teljesül:

$$U(|\psi> |\varphi>) = (X|\psi>)|\varphi> = (X|\psi>)(I|\varphi>)$$

Ahol I az identitás operátor. Mivel két kvantumbit tenzor szorzata egy négydimenziós vektorként írható fel, egyértelmű, hogy a keresett U operátor is egy 4x4-es mátrix. Ahogy a fenti egyenletből is látszódik, az U operátor az első kvantumbiten X míg a második kvantumbiten az I operátort fogja megvalósítani, célszerű tehát ezekből az operátorokból megpróbálni kombinálni U-t. Két 2x2-es mátrixon végzett műveletek közül az, ami egy 4x4-es mátrixot ad a Kroenecker szorzat, és mivel a kompozit rendszert is a részrendszerek Kroenecker szorzatával kaptuk, érdemes ezt megpróbálni:

$$U = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Gyors ellenőrzés után látszik, hogy az U operátor tényleg a kívánt operátor. Általánosan:

Felhasználva a Kroenecker szorzat és a mátrixszorzás közötti kapcsolatot:

Legyenek U_i, $0 \leq i \leq k$, 1 kvantumbiten végezhető operátorok és legyenek $|\varphi_i>$-k kvantumbitek tiszta állapotai. Ekkor:

$$U_1|\varphi_1 > \otimes U_2|\varphi_2 > \otimes \dots \otimes U_k|\varphi_k > =$$

$$(U_1 \otimes U_2 \otimes \dots \otimes U_k)(|\varphi_1 > \otimes |\varphi_2 > \otimes \dots \otimes |\varphi_k >)$$

Tehát a kívánt 1 bites operátorok Kroenecker szorzata megadja azt a 2^kx2^k-on operátort, amely a kvantumbitekből álló kvantumregiszteren ugyanazt a műveletet végzi, mintha az 1 bites operátorokat végrehajtottuk volna az egyes kvantumbiteken, majd aztán építettünk volna belőlük regisztert.

A CNOT kapu különbözik a többitől abban, hogy két kvantumbiten működik. Ennek ellenére egy bizonyos eseten kívül működik a CNOT kapu általánosítása a fenti módszerrel. A CNOT kapu általánosítása akkor működik egyszerű Kroenecker szorzatokkal, amennyiben az irányító és irányított bitekegymás mellett vannak. Ha nincsenek egymás mellett, akkor egy új általánosítási algoritmust kell bevezetni.

Kiindulási alapnak nézzük a következő esetet: három kvantumbites regiszterünk van, és az első kvantumbittel szeretnénk irányítani a harmadik bitet, ekkor:

$$CNOT_{1,3} = (CNOT_{1,2} \otimes I_{2x2})(I_{2x2} \otimes CNOT_{2,3})\ (CNOT_{1,2} \otimes I_{2x2})(I_{2x2} \otimes CNOT_{2,3})$$

A fenti állítást már bizonyították[12], ebből kiindulva levezetem az általános módszert. A levezetés megértéséhez először meg kell érteni a CNOT kapu működését:

$$CNOT_{1,2}(|\varphi > |\psi >) = |\varphi > |\psi \oplus \varphi >)$$

Ahol:

$$|\varphi > |\psi >= a|00 > +b|01 > +c|10 > +d|11 >$$

$$|\varphi > |\psi \oplus \varphi >= a|00 > +b|01 > +c|11 > +d|10 >$$

Tehát a CNOT kapu értelmezhető úgy, mint az irányított bit helyettesítése az irányító és irányított bit XOR-jával. Tehát a hárombites CNOT kapu szemléletesen felírva:

$$CNOT_{1,3}(|A> |B> |C>) = |A> |B> |A \oplus C>$$

4-bitre az általánosítás:

$$CNOT_{1,4} = (CNOT_{1,3} \otimes I_{2x2})(I_{4x4} \otimes CNOT_{3,4})\ (CNOT_{1,3} \otimes I_{2x2})(I_{4x4} \otimes CNOT_{2,3})$$

Ahol $CNOT_{13}$ már kiszámolható. A CNOT kapu szemléletes formáját használva a bizonyítás:

1. Eredeti állapot: |A>|B>|C>|D>
2. Első CNOT után: |A>|B>|$C \oplus A$>|D>
3. Második CNOT után: |A>|B>|$C \oplus A$>| $C \oplus A \oplus D$ >
4. Harmadik CNOT után (kihasználva, hogy bináris B-re B XOR B = 0 és 0 XOR B = B): |A>|B>|C>| $C \oplus A \oplus D$ >
5. Negyedik CNOT után: |A>|B>|C>| $A \oplus D$ >

Az algoritmus végére tényleg $CNOT_{1,4}$ et kapunk. Ezután felírhatjuk az általános képletet:

$$CNOT_{i,j} = (CNOT_{i,j-1} \otimes I_{2x2})(I_{2^{j-i-1}x2^{j-i-1}} \otimes CNOT_{j-1,j}) *$$

$$* (CNOT_{i,j} \otimes I_{2x2})(I_{2^{j-i-1}x2^{j-i-1}} \otimes CNOT_{j-1,j})$$

A bizonyítás a 4 bitessel analógan, rekurzívan megkapható.

A CNOT kapunál már csak egy kérdés maradt, mi van akkor, amikor megfordítjuk az irányított és irányító bit kapcsolatát? Ekkor elég kiszámolni a két szomszédos bites $CNOT_{2,1}$ kaput, és azt lehet általánosítani az előbb leírt módszerrel. A $CNOT_{2,1}$ kapu kiszámítása ezek után kézileg, vagy két oldalról Hadamard kapuval szorzással megvalósítható.

Tehát mivel tudjuk, hogy a P(45°), H, és CNOT kapukkal minden más kapu kifejezhető, és ezeket a kapukat bizonyítottan akármekkora regiszteren tudjuk használni, ezek után bármely kapu megépíthető a szimulátoromban.

3.2.4 Mérések

A kvantuminformatikának már csak egy fontos lépését nem írtam le a szimulátoromban, ez pedig a klasszikus és kvantum világ közötti híd: a mérések. A méréseket a kvantuminformatika harmadik posztulátuma fogalmazza meg:

3. ***Posztulátum (mérés):*** *Bármely kvantummérés leírható egy mérési operátor halmazzal $\{M_m\}$, ahol m a mérés lehetséges eredményeit jelöli. A valószínűsége annak, hogy m-et mérünk ha a rendszer v állapotban van, kiszámolható a következő egyenlettel:*

$$P(m|v) = \boldsymbol{v}^{\dagger} M_m^{\dagger} M_m \boldsymbol{v}$$

A rendszer a m mérése után pedig a következő állapotba megy:

$$\boldsymbol{v}' = \frac{M_m \boldsymbol{v}}{\sqrt{\boldsymbol{v}^{\dagger} M_m^{\dagger} M_m \boldsymbol{v}}}$$

Mivel a klasszikus valószínűségszámítás elmélete megköveteli hogy:

$$\sum_m P(m\,|\boldsymbol{v}) = \sum_m \boldsymbol{v}^{\dagger} M_m^{\dagger} M_m \boldsymbol{v} = 1$$

A mérési operátoroknak pedig teljesíteni kell a teljességi relációt:

$$\sum_m M_m^{\dagger} M_m = I$$

A mérési operátorok nem unitérek, de ez a szimulátor szempontjából nem akadályozza meg azt, hogy a kompozíciós általánosítás, amit a **Dekompozíció** alfejezetben bemutattam, a mérési operátorokra is kiterjeszthető legyen.

Az egyetlen újdonság, melyet a mérési operátorok bevezetnek, az adott kvantumbit klasszikus állapotba „csapódása”. Miután kiszámoltuk a valószínűségét az adott eredmény mérésének, a szimulátor a Mersenne-Twister pszeudo véletlenszám-generátorral generál egy számot, mely alapján a szimulátor eldönti, hogy mi lesz a mérés eredménye.

3.2.5 Kvantumáramkör

A kvantumáramkör az előző alfejezetekben bemutatott kvantuminformatikai módszereken kívül újdonságot nem tartalmaz, ez lényegében csak egy konstrukció, mely az operátorokat folytonosan elvégzi a megadott kvantumregiszteren.

Egy érdekes konstrukció még a mérések által irányított kapuk, sok kvantumáramkör, mint például a kvantum teleportáció használ kapukat, melyek egy mérés eredményétől függően végeznek, vagy nem végeznek egy operációt a kvantumregiszteren.

3.2.6 Kitérő az ábrázolásra

Az egyik legelemibb ábrázolás a kvantuminformatikában, amit mindenki ismer – aki egy kicsit is jártas a területen - a Bloch gömbön való ábrázolás. Ez az ábrázolás csak egy kvantumbitet tud egyszerre ábrázolni. Az ábrázolás elméleti része a kvantum bit legáltalánosabb formájából indul[13]:

$$|\varphi >= e^{j\gamma}[\cos(\frac{\alpha}{2})|0> + e^{j\beta}\sin(\frac{\alpha}{2})|1>]$$

A kvantuminformatikai alkalmazásban az $e^{j\gamma}$ globális fázis elhagyható. Így |0> együtthatója mindenképpen valós, míg |1> együtthatója lehet komplex is, így ez a konstrukció ábrázolható egy háromdimenziós ábrán vektorként, a vektor 1 hosszúságú, és a koordinátái a következő összefüggéssel határozhatók meg:

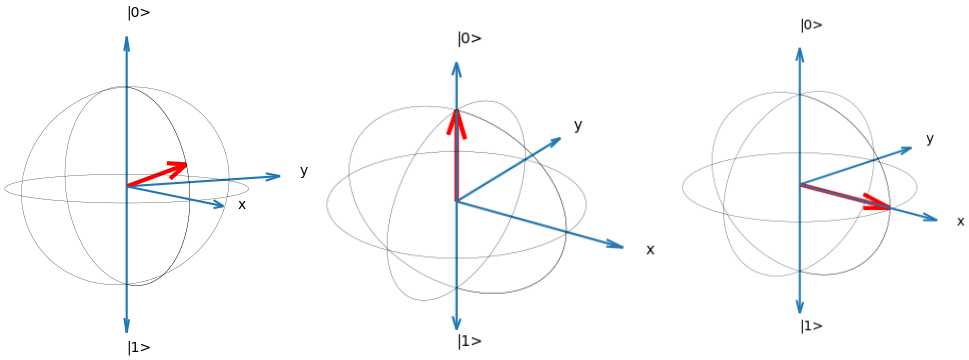
$$[x, y, z] = [\cos(\beta)\sin(\alpha), \sin(\beta)\sin(\alpha), \cos(\alpha))]$$

A szögek a következőképpen határozhatóak meg egy |φ> = a|0> + b|1>-re:

$$\alpha = 2 * \text{acos}(\,|\frac{a}{e^{\text{i}*\arg(\text{a})}}|)$$

$$\beta = \arg\left(\frac{b}{e^{\text{i}*\arg(a)} * \sin\left(\frac{\alpha}{2}\right)}\right)$$

A **3. ábra** a szimulátorommal készült néhány Bloch gömb látható:

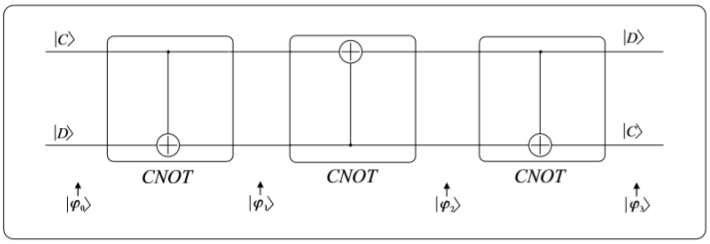


3. ábra Az ábrán sorban a $\frac{1}{\sqrt{3}}|0> + \sqrt{\frac{2}{3}}|1>, |0>$ és $\frac{1}{\sqrt{2}}|0> + \frac{1}{\sqrt{2}}|1>$, állapotok Bloch gömb ábrázolásai találhatóak.

3.3 Verifikáció

Ebben az alfejezetben szeretném bizonyítani, hogy a szimulátorom helyesen működik. A **Dekompozíció** fejezetben olvasható a bizonyítása annak, hogy a helyesen implementált kapukból helyesen általánosított operátorok helyesen működnek. A következő alfejezetekben a [13] irodalomból vett elméleti alapon bizonyított áramköröket implementálom, és vizsgálom az eredményeket, ezzel bizonyítva, hogy az egyes kapuk, mérések helyesen működnek, ezáltal bizonyítva, hogy a szimulátorommal összerakható áramkörök elméleti alapon is helyesek.

3.3.1 Swap Gate



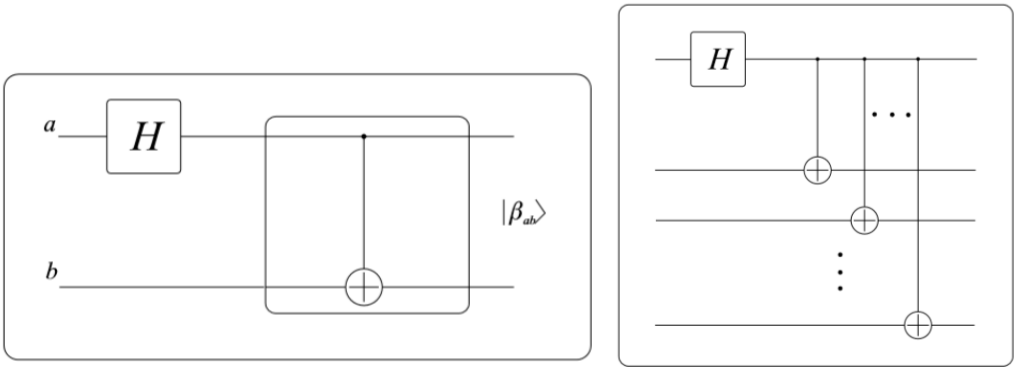
4. ábra Swap áramkör rajza

A swap áramkör két kvantumbit valószínűségi amplitúdóit cseréli meg. Az áramkör indulásakor a $|+> = \frac{1}{\sqrt{2}}|0> + \frac{1}{\sqrt{2}}|1>$ és a $|0>$ kvantumbiteket tesszük az áramkör regiszterébe, az áramkör kimenetén található kvantumregiszter az **5. ábra** látható. Ez az áramkör a CNOT és a CNOT kontrol invertált kapuk működését mutatja.

```
SWAP.......
Start:  ( 0.707107 |00>, 0.707107 |10>)[c]

Swap |+> with |0>:  ( 0.707107 |00>, 0.707107 |01>)[c]
```

5. ábra A swap áramkör kimenete

3.3.2 Bell állapotok generálása



6. ábra a Bell állapot generátor és az általános összefonó áramkörök áramköri rajza

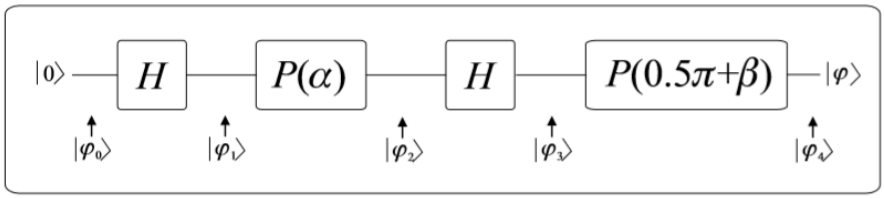
A CNOT kapuk helyes működését már láttuk, most bemutatom, hogy az általánosított Hadamard és az általánosított CNOT kapuk is működnek.

Az első áramkör a négy Bell állapotok generálja le a |0>|0>, |0>|1>, |1>|0> és |1>|1> kvantumregiszterek bemenetére, míg a második egy n bites összefonódott állapotot hoz létre, az áramkörök kimenetei a **7. ábra** figyelhetők meg.

```
Bell generator circuit.......
First wire: ( 1 |0>)[c]
Second wire: ( 1 |0>)[c]
Output: ( 0.707107 |00>, 0.707107 |11>)[c]

First wire: ( 1 |0>)[c]
Second wire: ( 1 |1>)[c]
Output: ( 0.707107 |01>, 0.707107 |10>)[c]

First wire: ( 1 |1>)[c]
Second wire: ( 1 |0>)[c]
Output: ( 0.707107 |00>, -0.707107 |11>)[c]

First wire: ( 1 |1>)[c]
Second wire: ( 1 |1>)[c]
Output: ( 0.707107 |01>, -0.707107 |10>)[c]

General entangler circuit.......
2 bit entangled: ( 0.707107 |00>, 0.707107 |11>)[c]

3 bit entangled: ( 0.707107 |000>, 0.707107 |111>)[c]

4 bit entangled: ( 0.707107 |0000>, 0.707107 |1111>)[c]
```

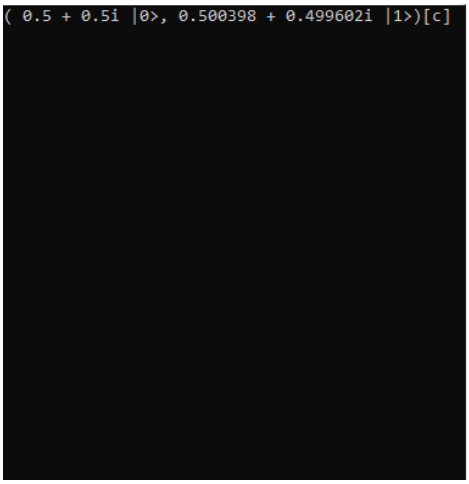
7. ábra A Bell állapot generátor és az általános összefonó áramkörök kimenete

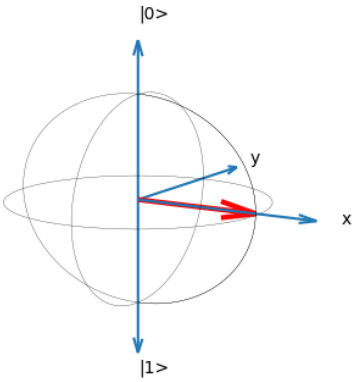
3.3.3 Tetszőleges kvantumállapot létrehozása



8. ábra Általános kvantum szuperpozíció előállítása

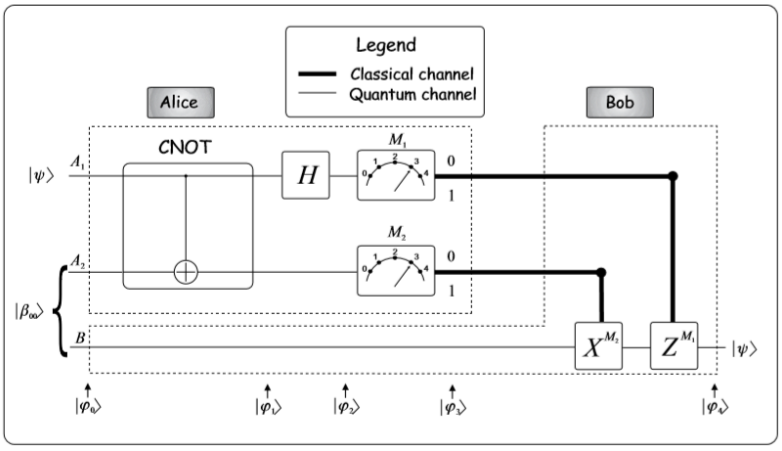
A 3 elemi kvantumkapuból kettőt már ellenőriztük, ezzel az áramkörrel a fázistoló kaput teszteljük. Az áramkör mindig a |0> állapotból indul ki, és a fázisoknak olyan szöget választottam, hogy a |+> állapot valamilyen globális fázissal (mely a kvantuminformatikai számítások során elhanyagolható) eltoltját kapjam meg.





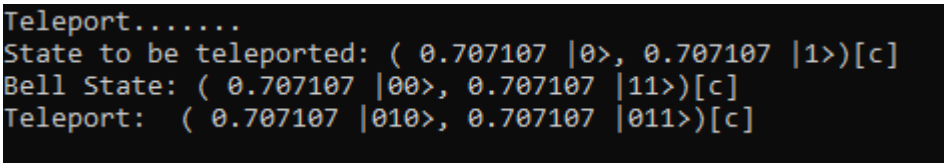
9. ábra A tetszőleges Kvantum állapotot előkészítő áramkör kimenete és az előkészített kvantumállapot Bloch gömbje.

3.3.4 Kvantum teleportáció



10. ábra a kvantum teleportáció áramköri rajza

Már bebizonyítottuk, hogy a 3 elemi kapu a szimulátorban helyesen működik, végezetül összerakjuk a kvantuminformatika egyik legkülönlegesebb áramkörét, ami egy kicsit komplexebb az eddigieknél, és van benne mérés és mérés által irányított kapu is. A szimulátorban a |+> állapotot teleportáljuk, melynek eredménye a **11. ábra** látható.



11. ábra A teleport áramkör eredménye a szimulátoron, az $\frac{1}{\sqrt{2}}|0> + \frac{1}{\sqrt{2}}|1>$ kvantumállapotot teleportáljuk át az 1-es vezetékről a 3-as vezetékre.

4 Ma is elérhető szimulátorok

Most, hogy bemutattam a saját készítésű szimulátoromat, röviden be szeretném mutatni a jelenleg elérhető szimulátorokat. Szinte az összes neves cég egyből nekiállt kvantum szimulációs rendszereket fejleszteni: Microsoft, Google, IBM. Már elérhető webes felületen is kvantumáramkör szerkesztő.

4.1 Microsoft Quantum Develpoment Kit (QDK)

A QDK egy open-source fejlesztői csomag, mely tartalmazza a Q# programozói nyelvet, melyet a Microsoft fejlesztett kvantumalkalmazások fejlesztésére, könyvtárakat melyek kvantum funkcionalitást tartalmaznak a Q# nyelven írva. Egy hoszt alkalmazást mely a Q# nyelven írt operációkat futtatja.

A QDK rugalmas csomag, fejleszthetünk benne Python nyelven, Jupyter notebookban, C# nyelven Visual Studio-t vagy VS Code-ot használva, vagy a dotnet command-line eszközt.

Microsoft termék lévén folyamatosan fejlesztés alatt áll a fejlesztői csomag, és nagyon széles funkcionalitást nyújt a felhasználóinak.

4.2 IBM Qiskit

A Qiskit egy open-source kvantum szoftverfejlesztő keretrendszer, melyet az IBM fejleszt és tart karban a felhasználókkal együtt. A Qiskit egy kizárólag Python nyelvbe beágyazható rendszer, mely Pythonban és kvantuminformatikában gyakorlott felhasználók számára könnyen használható. A Qiskit képes áramkör vizualizációra és az eredmény értékelés könnyítésére sokféle adatvizualizáció is elérhető.

A Qiskit egy egyelőre behozhatatlan előnye a többi rendszerhez képest az, hogy az IBM-nek saját kvantum számítógépe van, melyre a Qiskitben készült kvantumprogramokat egyből a Python API-ból lehet feltölteni és kipróbálni.

4.3 Google Cirq

A Google kvantumáramkör építője szintén egy Python API. A Cirq nem egy általános célú szimulátor, hanem a NISQ(Noisy Intermediate Scale Quantum) számítógépekre írandó algoritmusok gyakorlati hasznosságának tesztelésére lett

fejlesztve. Manapság a Google AI Quantum csapat a Cirq-t használja a Google Bristlecone-ra célzott áramkörök fejlesztésére.

4.4 Quirk

A Quirk egy weboldalba ágyazott kvantum szimulátor, mely alkalmas áramkörök gyors demonstrálására egy minimalistán vizualizált szerkesztőben.

12. ábra Quirk kvantum áramkör építő.

4.5 A Quantum Toy Box

Az általam készült szimulátor még kiforratlan és kezdetleges, de ennek ellenére is van néhány kitüntetett tulajdonsága:

A nagy cégek szimulátoraival ellentétben nincs bekötve egy másik nyelvbe, így szimplán elég a kvantuminformatikához érteni, és lehet benne áramköröket építeni.

C++-ban írodott, melyben még nincs nagyon elterjedten használt kvantuminformatikai könyvtár, és ezért a legtöbb kezdetleges szimulátorral szemben performancia előnye is van.

Egyszerűen használható, mivel nem tudósoknak szánt a lehető legoptimálisabb és leggyorsabb fejlesztésre, hanem a kvantuminformatika iránt érdeklődő emberek íródott kísérletezésre és tanulásra.

5 Kvantum gépi tanulásról általában

Ebben a fejezetben a kvantum gépi tanulás területét szeretném bemutatni általánosan, hogy a későbbiekben bemutatott eredményeket jobban lehessen értékelni.

Egy jó kvantumalgoritmus tervezése nehéz feladat, a nehézség fő okozója a területtel szemben támasztott elvárások: gyorsabb és kisebb komplexitású legyen, mint bármely ismert klasszikus algoritmus, melyet ugyanarra a célra találtak ki. A kvantuminformatika mostani haladási iránya, és a fentebb tárgyalt előnyök arra mutatnak, hogy a távkommunikáció mellett a terület legjobb gyakorlati alkalmazása a gépi tanulással együtt képzelhető el. A terület tárgyalásakor nem érdemes általános kvantumszámítógépre alapozni, mivel a gépitanulásnak specifikus igényei vannak.

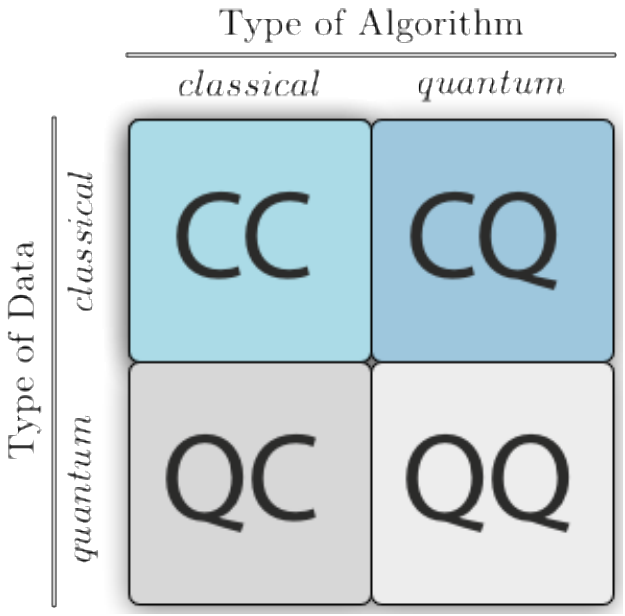
A kvantuminformatikát a mai állásában és közeljövőben a területen foglalkozó tudósok többsége nem önállóan, a klasszikus informatikát helyettesítve képzelik el. Fölösleges is lenne teljesen helyettesíteni velük, a klasszikus számítógépek évtizedek óta teljesítenek különféle feladatokat teljesen elfogadható módon. A mai napokban nagyon elterjedt a gyorsítók használata, melyek kiegészítik a hagyományos CPU-t. A gyorsítók a pontosságot mérsékelten feláldozva nagy sebességgel képesek párhuzamos műveletet végezni, de például a véletlen elérés bevezetése ezt az előnyt elvenné. A kvantum számítógépek modellje is hasonlóan képzelhető el két okból, először is a kvantum hardver irányító rendszere klasszikus számítógép lesz, másrészt az adatbevitel és a mérés kiolvasás is klasszikus hardveren fog alapulni[14].

Az előbbiek alapján a fejezetben tárgyakat olvasva ne lepődjünk meg tehát, hogy a legtöbb kvantumalgoritmus nem egy teljes architektúrát képvisel, hanem annak csak egy részét valósítja meg, gyorsítja fel a klasszikus párjához képest.

Mielőtt a részletekbe belemennénk érdemes felfedezni még egy pozitív hatást, melyet a kvantuminformatika gyakorolt a gépi tanulás területére: a kvantum-szerű algoritmusokat, melyek bár nem tartalmaznak olyan lépéseket, melyek igényelnek kvantum áramköröket, mégis a kvantum informatika által inspirált klasszikus lépéseket alkalmaznak.

5.1 Csoportosítás

A kvantum gépi tanulás még igen új terület a tudomány történetének szempontjából, és annyi féle szálon lehet elindulni mind a kvantuminformatika, mind a klasszikus gépi tanulás szempontjából a kvantum gépi tanulás felé, hogy a csoportosítás egy nehéz feladat. Az egyik legátfogóbb csoportosítási módszer az adat és az algoritmus típusán keresztüli csoportosítás, ennek miként je a **13. ábra** figyelhető meg.



13. ábra Kvantum gépi tanuló algoritmusok csoportosítása az adat és az algoritmus típusa szerint.

A klasszikus adatok klasszikus feldolgozása (CC) a hagyományos mesterséges tanulás területe, mely bár még mindig fejlődik, már számos olvasmány bemutatja részletesen, ezért erre bővebben nem térek ki.

A kvantum adatok klasszikus feldolgozása (QC) már közelebb áll a kvantuminformatikához, erre tipikus példa a kvantum állapot tomográfia, ahol egy kvantumállapotot tanulunk mérésekből, példa lehet erre még kvantum véletlenszámok vizsgálata klasszikus algoritmusokkal[15].

A kvantum adatok kvantumos feldolgozása (QQ) a színtiszta kvantumalgoritmusokat jelöli, ahol a tanuló rendszer, a környezet, az adatok és ezek interakciója is teljesen kvantumos.

A típus, mely utoljára maradt és a dolgozatom fókuszpontja a klasszikus adatok kvantum feldolgozása (CQ). Ezt a csoportot a hibrid rendszerek jellemzik, melyek részben klasszikusak részben kvantumosak.

5.2 Nehézségek

A sok előny ellenére, amelyet a gépi tanuló algoritmusok élvezhetnek a kvantuminformatika által, a két terület összefüggésének megvannak a nehézségei, melyek a kvantuminformatika inherens tulajdonságaiból születnek.

Az első a non-linearitások problémája, a kvantuminformatika linearitása, mivel a *Posztulátum (**evolúció):** Bármely fizikai rendszer időbeli evolúciója karakterizálható unitér transzformációkkal, melyek csak az evolúció kezdési és befejezési idejétől függ.* alapján, egy kvantumrendszer időbeli evolúcióját unitér operátorokkal írhatunk le, az uniter operátok pedig lineárisok azaz non-linearitásokat nem tudunk leírni velük. Ennek a problémának a megoldására különféle módszerek vannak. Annak ellenére, hogy az evolúciós operátorok unitérek, a ***Posztulátum (mérés):** Bármely kvantummérés leírható egy mérési operátor halmazzal {M_m}, ahol m a mérés lehetséges eredményeit jelöli. A valószínűsége annak, hogy m-et mérünk ha a rendszer v állapotban van, kiszámolható a következő egyenlettel:* alapján tudjuk, hogy a mérési operátorok nem, tehát mérésekkel tudunk közelíteni non-linearitásokat. Egy második módszer a közvetlen fizikai részecskékből épített neurális hálózatok építése (QQ megoldás), ekkor a Feynman féle út integrál formuláción keresztül a rendszerbe bevezethetünk non-linearitást. CQ megoldások esetét két módszer is ajánlja magát, az egyik egyszerűen az, hogy a non-linearitást nem próbáljuk meg kvantumosan megoldani, hanem az architektúrába klasszikus elemként beépítjük a non-linearitást. Egy bonyolultabb QC megoldás, amivel bár nem lépünk re a kvantuminformatika domain-jéről, még is meg tudunk közelíteni egy non-lineáris viselkedést, a RUS (Repeat-Until-Success) áramkör.

A második nehézségét, ami a kvantuminformatika egy általános hátránya, a *No Cloning Theorem* mondja ki: A kvantuminformatikában egy kvantumállapot tökéletes átmásolása előzetes információ nélkül lehetetlen, ezért amikor megmérünk egy kvantumbitet, akkor az információ amit addig tartalmazott elveszik. Ez a legtöbb kvantuminformatikai algoritmusban problémát okoz, melyet meg valahogy meg kell kerülni. Ez a probléma, társítva a kvantuminformatika másik legnagyobb hátrányával, a dekoherenciával, mely az információ idővel való elvesztését jelenti, igen kellemetlen

következményeket von maga után, például azt, hogy egy kvantum gépi algoritmus változóit mindig klasszikusan kell lementeni, majd visszatenni a rendszerbe, különben a tanulást elölről kell kezdeni, mert az információ elveszik.

5.3 Kvantum Neurális Hálózatok

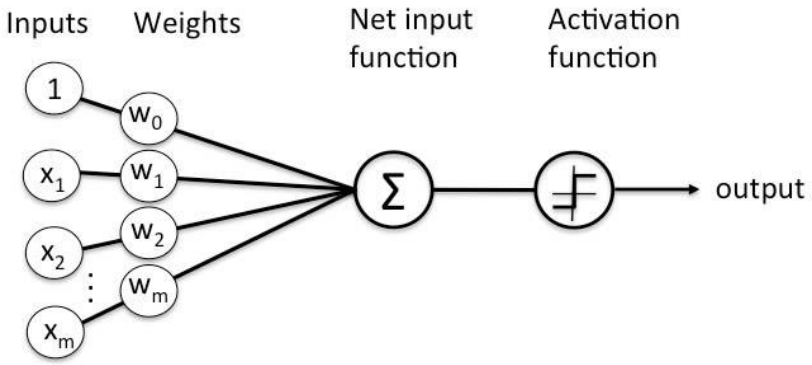
A gépi tanulás területén talán a legjobban felkapott szó talán a „valamilyen neurális hálózat". Ez jó okkal van így, a neurális hálózatok alkalmazási területe széles, és sok oldalról meg lehet közelíteni a témát. Ebből következően a kvantum gépi tanulás területén is a legelterjedtebbek a kvantum neurális hálózatok. A kvantum neurális hálózatok megvalósítása iránt sok irányban megindultak kísérletek, néhány főbb motívumot vizsgáljunk meg.

5.3.1 CQ Hibrid módszerek

Ahogy a csoportosításnál is említettem, a dolgozat témája és a vizsgálandó rendszerek a klasszikus adatokon végzett kvantum feldolgozás. A hibrid módszerek közül be szeretném mutatni a klasszikus neurális egyes részeinek lehetséges kvantum verzióját, és hogy ezeket milyen módszerekkel lehet integrálni egy közös rendszerbe. Általánosságban egy neurális hálózatot három részre lehet felbontani: Az alapvető felépítés, a non-linearitások és a tanítási algoritmus.

5.3.1.1 Általános felépítés

Az általános architektúrára már sok megvalósítás létezik klasszikusan, ezek mintájára kidolgoztak kvantum verziókat is. Vegyük példának a klasszikus perceptron architektúráját, mely látható a **14. ábra**. Ahogy az ábráról is leolvasható, egy perceptron egyszerűen egy súlyozott összegző, mely kimenetén egy aktivációs függvényt értékelünk ki. Az aktivációs függvény megvalósításával a non-linearitások pont alatt fogunk foglalkozni.



14. ábra Egy perceptron általános felépítése

Ha végig gondoljuk, akkor tehát az aktivációs függvény bemenetére a perceptron bemeneteinek a súlyokkal vett skaláris szorzatát kapjuk, melyet a neuron bemenetének hívjuk és jelöljük θ-val. Egy neuront tehát matematikailag a következőképpen írhatunk le:

$$output = f_{akt}(x_1 w_1 + x_2 w_2 + \cdots + x_n w_n) = f_{akt}(\theta)$$

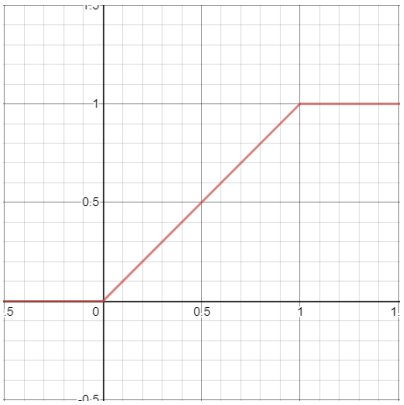
A skaláris szorzat viselkedését emulálni tudjuk egy ancilla kvantumbit segítségével. Használhatjuk az $x_1,x_2,\ldots,x_n$ bemenetekből alkotott kvantumregisztert arra, hogy irányítsunk egy $R_y(2w_i)$ kaput az ancilla biten, melyek az i-edik kvantumbitre kondícionálunk. Ennek az eredményeként $R_y(2\theta)$ elforgatást eszközöltünk az ancilla kvantum biten, lényegében az ancilla kvantumbitet a kívánt állapotba hozva. Ezután ha az ancilla biten valamilyen módon egy aktivációs függvényt kiértékelünk, akkor teljesen megvalósított perceptron architektúrát kapunk, hiszen az elfordításoknak hála az ancilla kvantumbiten a súlyok függvényében |0>-t vagy |1> fogunk nagyobb eséllyel mérni.

5.3.1.2 Aktivációs függvény

Az aktivációs függvénnyel a neuron súlyainak a befolyását erősítjük fel, egyfajta küszöbfüggvényt valósítunk meg vele, mely azt dönti el, hogy a súlyok adott összegére, a neuron mit válasszon. A legegyszerűbb aktivációs függvény a lineáris függvény, ezt még a kvantuminformatikával is nagyon egyszerű megvalósítani, ha belegondolunk, akkor egy kvantum bit projektív mérése[2] pontosan ezt fogja megvalósítani, ha a méréseket kellően sokszor futtatjuk le. Gondoljuk végig: ha egy kvantumbitet mérünk projektíven,

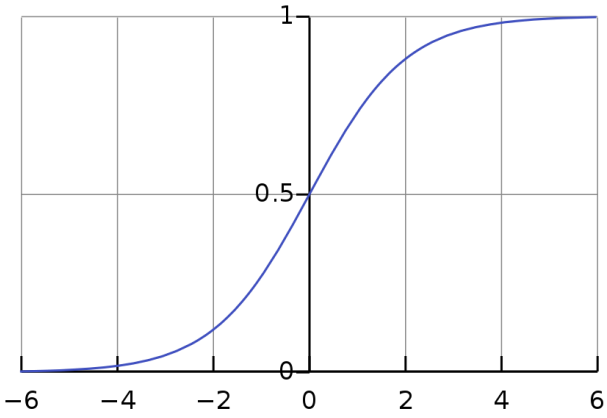
[2] A projektív mérések nem változtatják meg a mérés során a bázis állapotok valószínűségi amplitúdóit.

akkor |0> mérése $|a|^2$ valószínűséggel történik meg, |1> mérése pedig, az $|a|^2+|b|^2=1$ összefüggés miatt $1-|a|^2$ valószínűséggel, tehát ahogy |0> mérésének valószínűsége nő, úgy csökken |1> mérésének valószínűsége, ez látható a **15. ábra**.



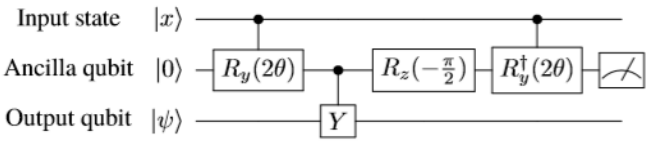
15. ábra Mérésekkel megvalósítható aktivációs függvény $|a|^2$ függvényében, az y=1 értékekre biztos, hogy |0>-t az y=0 értékekre pedig biztosan |1>-et mérünk.

A tipikus aktivációs függvényeknél azonban nem lineárisan szeretnénk az inputot feldolgozni, hanem olyan viselkedést szeretnénk megvalósítani, mely során az input annál kevésbé változik, minél közelebb van az egyik szélsőértékhez, ilyenre példa a szigmoid függvények családja melyből példát a **16. ábra** láthatunk.



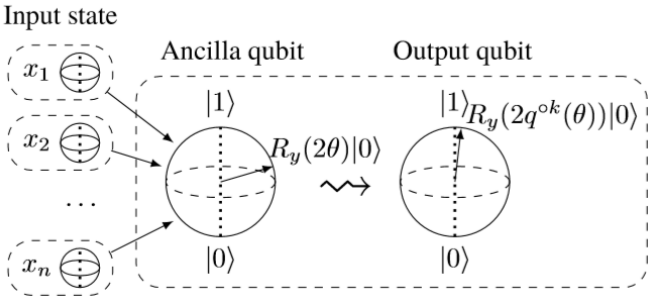
16. ábra Logisztikus szigmoid függvény: $f(x) = \frac{1}{1+e^{-x}}$

Az ábrán tisztán látszik, hogy a függvény közel sem lineáris, melyet egyszerű kvantum operátorral nem lehet megvalósítani. Kvantuminformatikusok ennek ellenére kidolgoztak egy áramkört, mellyel meg lehet közelíteni ezt a viselkedést. Ennek a neve a már említett RUS áramkör, az áramköri rajz látható a **17. ábra.**



17. ábra A RUS áramkör áramköri rajza.

Az áramkör az $R_y(2q(\theta))$ operációt valósítja meg, ahol $q(\theta) = \arctan(\tan^2\theta)$. Az áramkör viselkedése az ancilla kvantum bit mérésének eredményétől függ. Ha |0>-át mérünk, akkor az áramkör sikeresen megvalósított egy $2q^k(\theta)$ forgatást az output kvantum biten, azaz sikeresen megvalósítottuk egy kvantum áramkör működését, ha nem sikerült, akkor az output kvantumbiten $R_y(\pi/2)$-es forgatást végeztünk. Ebben az esetben $R_y(-\pi/2)$ forgatással javítunk, és ezt ismételjük addig, amíg nem |0>-t mérünk. Tehát végül megépítettünk egy működő kvantum perceptron architektúrát mely a **18. ábra** látható. A RUS áramkör általánosításáról bővebben lehet olvasni a [17]-es publikációban.



18. ábra A kvantum perceptron

5.3.1.3 Tanítás

Az egyetlen probléma, melyet még nem oldottunk meg, a perceptron tanítása. A klasszikus tanító algoritmusoknak különböző paramétereket adhatunk meg a minél jobb eredmények elérésének érdekében, egy nélkülözhetetlen paraméter, mely nélkül nem tudjuk helyesen tanítani az architektúránkat az a hiba, tehát, hogy az architektúra által adott tipp mennyiben különbözik az elvárt eredménytől. A hiba számítására különböző módszerek vannak. A tanulás megvalósításában viszont az előzőleg választott lépések függvényében felmerülhet probléma: amennyiben nem méréssel valósítottuk meg a non-linearitást kvantumosan, a non-linearitás eredménye kvantum információként van elrejtve a rendszerben, ezért az információt vagy sokszoros méréssel tudjuk kinyerni, mely a *No*

Cloning Theorem miatt az egész rendszer sokszoros újra futtatását jelenti[3], vagy pedig a hibát is kvantumosan kell számolni. Ezután viszont a hiba is kvantuminformáció lesz, mely megint a fenti dilemma elé helyez minket, tehát a hiba hatékony felhasználásához a tanulást is kvantumosan kell megoldani. A tanítás nagyban függ a választott architektúrától, és sokféle elképzelés kialakult, ami egy érdekes megfigyelés, hogy meglepően sok kvantum tanulási algoritmus alkalmazza Grover keresési algoritmusát, ami érthetővé válik, amint belegondolunk, hogy Grover algoritmusa lényegében az amplitúdó erősítésről szól, melyre gondolhatunk úgy, mint egy adott súly manipulációjára.

5.4 Implementációk

A kvantum gépi tanulás területe még az emberi élethosszal számolva is fiatal, az első megvalósított kvantum gépi tanulási algoritmus a Google és a D-Wave együttműködésével valósult meg a D-Wave adiabatikus kvantum számítógépén 2009-ben[16]. Ezután sok hasonló kísérlet követte az előbbit, ahogy egyre több vezető cég érdeklődését felkeltette a terület. 2013-ban a Google Research, NASA és a Universities Space Research Association elindította a Quantum Artificial Intelligence Lab-ot mely a D-Wave adiabatikus kvantum számítógép felhasználását vizsgálja. Különböző kísérletek és implementációk valósultak meg azóta is, de a terület gyakorlata továbbra sem tekinthető elterjedtnek.

[3] Hiszen ha nem tudjuk másolna a kvantumbitet- és mivel tudjuk, hogy a mérés elpusztítja a kvantumállapotot – ahhoz, hogy ugyanazt a kvantumbitet többször megmérhessük, a kvantumbitet újra és újra elő kell készíteni ugyanabba az állapotba.

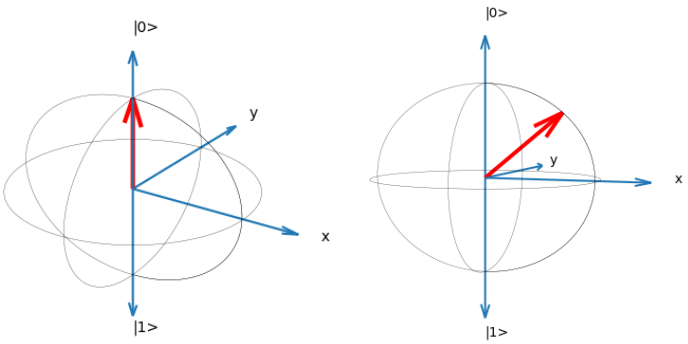
6 Egy szimpla kvantum perceptron

Miután az előző fejezetben megvizsgáltuk a kvantum gépitanulást, és azon belül is a kvantum neurális hálózatokat, a szimulátoron megpróbálunk megépíteni egyet. Először ki kell választani, hogy mit építünk meg, ahogy korábban is említettem a dolgozat fókusza a hibrid CQ architektúrák, így az implementált architektúra is ehhez hasonló lesz.

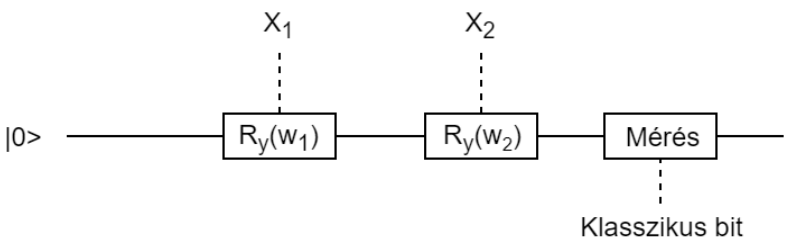
Az architektúra egy neuronból áll, melynek felépítése a **20. ábra** megfigyelhető. Az ábrán látható R_y kapu a kvantumbitet az Y tengely körül forgatja el, innen is kapta a nevét, a kaput a következő mátrix írja le:

$$R_y(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix}$$

A kapukat az áramkörben a klasszikus bemenetek irányítják, ha a bemenet alacsony(0) a kapu érintetlenül átengedi a kvantumállapotot, míg ha magas(1) akkor elvégzi rajta az Y tengely körüli forgatást θ radiánnal. A forgatás szöge a mi esetünkben a súlyok függvényében, $\theta_i = w_i * \frac{\pi}{2} + \frac{\pi}{2}$ szöggel történik. A forgatás szemlétesen megfigyelhető a **19. ábra.**



19. ábra Az y tengely körüli forgatás |0>-ból $\frac{\pi}{4}$-el



20. ábra A saját perceptron architektúrám a leírtak alapján inspirálva. Jelölés: Folytonos vonal kvantum vezetéket jelöl míg a szaggatott klasszikusat.

Nézzük meg az áramkör futását néhány bemenetre a **21. ábra.**

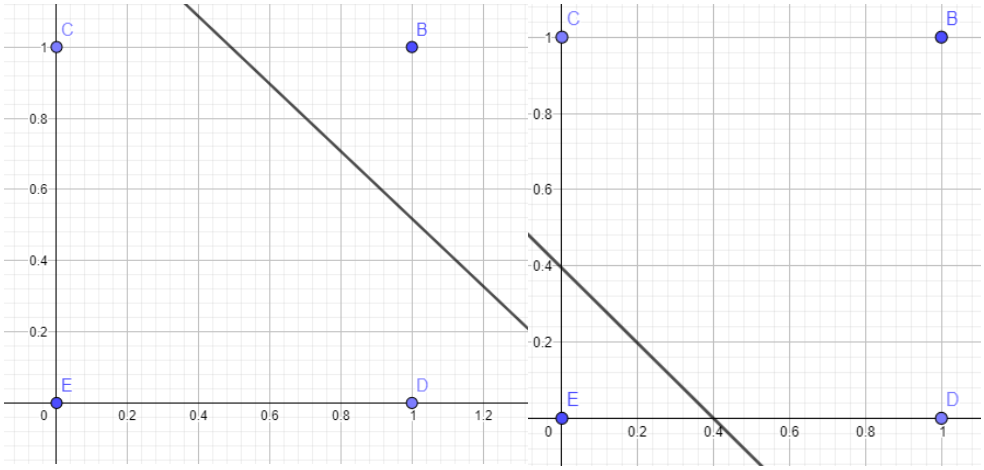
```
w1 = 1, w2 = 1
X1 = 0, X2 = 0
After 100 runs, we have 100 zeros and 0 ones.
X1 = 1, X2 = 0
After 100 runs, we have 0 zeros and 100 ones.
X1 = 0, X2 = 1
After 100 runs, we have 0 zeros and 100 ones.
X1 = 1, X2 = 1
After 100 runs, we have 100 zeros and 0 ones.



w1 = -0.3, w2 = -0.3
X1 = 0, X2 = 0
After 1000 runs, we have 1000 zeros and 0 ones.
X1 = 1, X2 = 0
After 1000 runs, we have 714 zeros and 286 ones.
X1 = 0, X2 = 1
After 1000 runs, we have 717 zeros and 283 ones.
X1 = 1, X2 = 1
After 1000 runs, we have 201 zeros and 799 ones.



w1 = 0.34, w2 = 0.34
X1 = 0, X2 = 0
After 1000 runs, we have 1000 zeros and 0 ones.
X1 = 1, X2 = 0
After 1000 runs, we have 240 zeros and 760 ones.
X1 = 0, X2 = 1
After 1000 runs, we have 239 zeros and 761 ones.
X1 = 1, X2 = 1
After 1000 runs, we have 305 zeros and 695 ones.
```

21. ábra A kvantum perceptron eredményei, w1 és w2 súlyok, X1 és X2 klasszikus bemenetek.

6.1 Quantum Supremacy

„quantum supremacy is the potential ability of devices to solve problems that classical computers practically cannot”

Mielőtt egy megdöbbentő felfedezést tennénk meg az eredményekben, egy kicsit vizsgáljuk meg a klasszikus perceptronokból épített neurális hálózatok képességeit. A klasszikus egyrétegű perceptron hálók a súlyaikkal egy választóvonalat határoznak meg a bemeneti vektorok által meghatározott síkon, ez azt jelenti, hogy bizonyos problémákat egy klasszikus perceptronnal lehetetlen megoldani, ez szemléletesen megfigyelhető a **22. ábra.** Egy ilyen probléma az XOR is, melynek két eredményhalmazát (igaz, hamis), nem lehet egy egyenessel kettéválasztani. Gondoljuk végig: az {[0,0][1,1]} és {[0,1][1.0]} pontokat kell kettéválasztani az $y=w_1*x + w_2*x$ hipersíkkal, mely lehetetlen.



22. ábra bal oldalon egy példa tanulható választóvonal az X AND Y probléma megoldására{Hamis: E,C,D ; Igaz: B}, a jobb oldalon pedig az X OR Y {Hamis: E; Igaz: C,D,B } probléma megoldására.

Figyeljük meg alaposan a **21. ábra** a kimeneteket a bemenetek függvényében:

$$KvantumNeuron(0,0) = 0$$

$$KvantumNeuron(0,1) = 1$$

$$KvantumNeuron(1,0) = 1$$

$$KvantumNeuron(1,1) = 0$$

A kvantumneuronunk a két súly megfelelő megválasztásával tökéletesen megoldotta a XOR problémát, és megtippelt súlyokkal nagy magabiztossággal megoldja az AND és OR problémákat is. Tehát egy darab kvantumbitből álló kvantum neuron két súly és két bemenet függvényében meg tudja oldani az XOR problémát, melyet klasszikus megfelelője képtelen megoldani. Az architektúra ezek után láncolható és elég méréssel tanítható is bonyolultabb problémák megoldására. A kvantum neurális hálózatok tehát potenciálisan képesek olyan problémákat megoldani, melyekre a klasszikus megfelelőjük képtelen.

7 Összefoglalás, kitekintés és köszönetnyilvánítás

A dolgozat során bemutattam a saját készítésű kvantum áramkör szimulátoromat, a Quantum Toy Box-ot, verifikáltam a működését, ismertettem a kvantum gépi tanulás területét általánosan, bevezetve az olvasót felületesen a kvantum neurális hálózatok területére is. Végül bemutattam egy saját (bár mások által inspirált) kvantum perceptront, mely képes megoldani a XOR problémát, melyre egy klasszikus perceptron képtelen lenne, így kimutatva a „Quantum Supremacy”-t a neurális hálózatok területén.

Mindkét területnek (a szimulátoroknak és a kvantum gépi tanulásnak) hatalmas potenciálja van a jövőben, az általam tovább vinni tervezett motívumok a szimulátoromban a fraktálalapú kvantumregiszter vizualizálás, a grafikus áramkörszerkesztés és a különböző kvantum assembly nyelvekre fordítása a tervezett áramköröknek. A kvantum perceptronom a jövőben csatlakozni fog társaihoz egy teljes kvantum neurális hálózatban, különböző tanítási módszereket kipróbálva.

Végül, de nem utolsó sorban köszönetet szeretnék mondani a konzulensemnek, Dr. Bacsárdi Lászlónak aki a Föld másik oldaláról is korrektúrázta a dolgozatomat, és bármikor segített, ha szakmai kérdésem támadt.

Irodalomjegyzék

[1] https://www.hpcwire.com/2019/01/10/ibm-quantum-update-q-system-one-launch-new-collaborators-and-qc-center-plans/ *(Utolsó látogatás, 2019.10.14)*

[2] https://ai.googleblog.com/2018/03/a-preview-of-bristlecone-googles-new.html *(Utolsó látogatás, 2019.10.14)*

[3] https://spectrum.ieee.org/tech-talk/computing/hardware/intels-49qubit-chip-aims-for-quantum-supremacy *(Utolsó látogatás, 2019.10.14)*

[4] https://www.quantamagazine.org/how-space-and-time-could-be-a-quantum-error-correcting-code-20190103/ *(Utolsó látogatás, 2019.10.14)*

[5] https://wigner.mta.hu/quantumtechnology/hu/fooldal/ (*Utolsó látogatás, 2019.10.16*)

[6] http://www.eit.bme.hu/calls_and_projects/2017-121-nkp-2017-00001/ (*Utolsó látogatás, 2019.10.16*)

[7] Samuel, Arthur (1959). *"Some Studies in Machine Learning Using the Game of Checkers".* IBM Journal of Research and Development.

[8] Rosenblatt, F. (1958). *"The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain".* Psychological Review

[9] Marvin Minsky, Seymout Papert (1969) *„Perceptrons: an introduction to computational geometry”*

[10] Aram W. Harrow, Avinatan Hassidim, Seth Lloyd*, „Quantum algorithm for linear systems of equations”*

[11] Nikolay Raychev (2015), *„Universal quantum operators”*

[12] Juan Carlos, Garcia-Escartin, Pedro Chamorro-Posada (2011), *„Equivalent Quantum Circuits”*

[13] Sándor Imre, Ferenc Balázs, Wiley kiadó, *„Quantum Computing and Communications: An Engineering Approach”*

[14] Peter Wittek, *„Quantum Machine Learning: What Quantum Computng Means to Data Mining”* 7.o.

[15] Kertész Gergő Csaba (2018), *„Kvantum alapú véletlenszámok tesztelése neurális hálók segítségével”*

[16] Google, D-Wave Systems Inc. (2019) *„NIPS 2009 Demonstration: Binary Classification using Hardware Implementation of Quantum Annealing”*

[17] Yudong Cao, Gian Giacomo Guerreschi, Alán Aspuru-Guzik (2017), *„Quantum Neuron: an elementary building block for machine learning on quantum computers"*

[18] Bob Ricks, Dan Ventura, *„Training a Quantum Neural Network"*

[19] Lov K. Grover (1996), *„A fast quantum mechanical algorithm for database search"*