Budapesti Műszaki és Gazdaságtudományi Egyetem

Kvantum-áramkör szimulátor tervezése és alkalmazásai

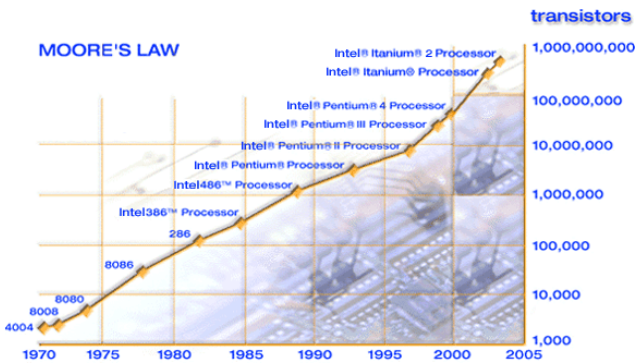
Szakdolgozat, 2005.

Készítette: Pereszlényi Attila, peresz@mcl.hu
Konzulens: Dr. Imre Sándor, Híradástechnikai Tanszék, imre@hit.bme.hu

Tartalomjegyzék

1 Bevezetés

A számítástechnika rohamos fejlődésével a mai típusú számítógépeket tervező mérnökök néhány éven belül eddig nem tapasztalt új akadályokba fognak ütközni. Az integrált áramkörök gyártásában hosszú idő óta az a tendencia, hogy az egységnyi felületen elhelyezett tranzisztorok száma néhány évente megduplázódik (Gordon Moore, 1965 [19]). Ha ez folytatódik, akkor a tranzisztor mérete nemsokára elérheti az atomi mérettartományt. Egy ilyen szerkezet viselkedése egyre kevésbé fog hasonlítani egy tranzisztor viselkedéséhez, mivel a klasszikus fizika törvényei ezen a mérettartományon használhatatlanokká válnak, helyettük a kvantummechanika törvényei lépnek életbe. Néhány éve a kutatások fő célja az volt, hogy minél kisebb szerkezetekre erőltessék rá a tranzisztor jellegű működést. Ma már nem ez a helyzet. Rolf Landauer 1961-ben megmutatta, hogy információ törlése szükségképpen hőfelszabadulással jár [7], sőt minden irreverzibilis kapu (például ÉS kapu) működése közben egy adott minimális hőmennyiség mindenképpen felszabadul. Ez reverzibilis kapuk használatával kiküszöbölhető, sőt törlés sem kell, ha a gép az eredmény kiszámolása után a reverzibilis kapuk visszafordításával visszatér a kezdeti állapotba. Habár a mai számítógépek még nagyságrendekkel több hőt juttatnak a környezetbe, mit az elvi minimum, ez a későbbiekben komoly akadálya lehet a számítási kapacitás növelésének.



1-1. ábra: Moore törvénye. A tranzisztorok számának növekedése exponenciális. [19]

A kvantummechanikai rendszerek nehézkes szimulációja 1982-ben arra az ötletre vezette Richard Feynmant, hogy kvantumrendszert kvantumrendszerrel lenne célszerű szimulálni, és ezzel elkerülhető lenne az exponenciális lassúság. Ez volt az első ötlet, hogy a kvantummechanika elvei alapján építsünk számítógépet. (Paul Benioff szintén hasonló következtetésre jutott nagyjából ugyanekkor.) A kvantummechanikai kapuk ráadásul reverzibilisek. Később sikerült olyan problémákat találni, amelyek kvantum

megoldása exponenciálisan gyorsabb, mint a klasszikus. (Ilyen például a Deutsch-Jozsa algoritmus [8].) Sajnos ezeknek a problémáknak nem volt sok gyakorlati hasznuk. 1994-ben azonban Peter Shor bemutatott egy olyan kvantum algoritmust, amellyel összetett számokat polinom időben lehet prímtényezőkre bontani [9]. Ha ezt sikerül fizikailag megvalósítani, akkor a manapság széles körben elterjedt titkosító algoritmusok használhatatlanok lesznek (pl. RSA, El-Gamal). Egy korábbi eredménynek, a kvantum kulcsszétosztásnak köszönhetően azonban létezik biztonságos megoldás a rejtjelezés problémájára (Charles Bennett és Gilles Brassard, 1984 [10]). 1996-ban újabb jelentős eredmény született. Lov K. Grover bemutatott egy olyan kvantum algoritmust, amellyel egy N elemű rendezetlen adatbázisban való keresés $O\left(\sqrt{N}\right)$ adatbázis hozzáféréssel megoldható [12].

A kvantum algoritmusok fizikai megvalósításának egyik nehézsége, hogy a kvantumállapotok tökéletlen elszigeteléséből adódóan hibák lépnek fel és ezek a számítás végére olyannyira akkumulálódhatnak, hogy a végére már a zaj elnyomja az adatot. 1995-ben Shor bebizonyította, hogy a kvantumállapotok hibavédő kódolása lehetséges [13]. Arra is lehetőség van, hogy a nem tökéletes kapuk hatásait korrigáljuk [14]. A kvantum információelmélet is megszületett, a klasszikus információelmélettel analóg eredmények sorát sikerült bebizonyítani.

A kvantum algoritmusokat leggyakrabban kvantum-áramkörökkel adják meg. Munkám célja egy program kifejlesztése, amely ilyen áramkörök tervezésére és szimulációjára alkalmas. A szimulátor használható algoritmusok működésének vizsgálatára, helyességének ellenőrzésére. A programban lehetőség van kvantum csatornák használatára, így kommunikációs protokollok is tesztelhetők, illetve segítségükkel lehetőség nyílik a kvantumszámítógépek valósághűbb modellezésére azzal, hogy az építőelemekben véletlen hibázásokat idézünk elő. A program ismertetése mellett dolgozatom célja, hogy bemutassam a program lehetséges felhasználási területeit ismert algoritmusokon keresztül (pl. teleportálás, „Super-dense Coding”, Deutsch-Jozsa algoritmus [8], kvantum hibavédő kódolás [13], stb.). Ezek után a program egy konkrét, gyakorlatban is használható alkalmazását is bemutatom. Kvantum kulcsszétosztó protokollokat vizsgáltam, azt feltételezve, hogy a kommunikáció zajos kvantum csatornán zajlik. Olyan kérdésekre kerestem a választ,

mint hogy mennyire biztonságos egy adott csatornával egy adott protokoll, illetve egy adott csatornához milyen protokollt célszerű választani.

A dolgozat felépítése a következő. A 2. fejezetben a tudományterület matematikai alapjait mutatom be, olyan részletességgel, ami a program használatához és a dolgozatban lévő algoritmusok megértéséhez szükséges. A 3. fejezetben a szimulátor programomat mutatom be. Először egy használati útmutatót adok, majd a program felépítését és a szimulációt írom le, végül összehasonlítom a programot más hasonló célú programmal. A 4. fejezet célja, hogy a program széleskörű alkalmazhatóságát demonstrálja. Különböző algoritmusokat szimulálok, és példákat adok, hogy a program nyújtotta lehetőségekkel az algoritmusok milyen tulajdonságaira lehet rávilágítani. Az 5. fejezetben írom le a fentebb említett kvantum kulcsszétosztás vizsgálatát, és a gyakorlati felhasználás szempontjából is érdekes eredményeit. Végül a 6. fejezet zárja a dolgozatot az összefoglalással.

2 Kvantum informatika

A matematikai formalizmus bevezetését az axiómák definiálásával kezdem. A négy axiómát a „kvantummechanika posztulátumainak” hívjuk, bemutatásuk nagyrészt „Mark Oskin: Quantum Computing - Lecture Notes” [2] szerinti.

2.1 A kvantummechanika posztulátumai

Az elmélet a következő négy posztulátumra épül:

1. A kvantum állapot
2. Kvantumrendszer időbeli fejlődése
3. Mérés
4. Összetett rendszer

2.1.1 A kvantum állapot

Minden zárt kvantumrendszerhez hozzárendelhetünk egy belső szorzattal rendelkező $\mathcal{H}$ komplex vektorteret, amit Hilbert-térnek hívunk. A rendszer állapotát egy $|\varphi\rangle \in \mathcal{H}$ egység hosszú vektor írja le.

A Hilbert-tér esetünkben mindig véges dimenziós, azaz $\mathcal{H} = \mathbb{C}^n$. A tér oszlopvektorait a „ket”, sorvektorait a „bra” Dirac formalizmussal jelöljük. Például egy oszlopvektor lehet: $|\varphi\rangle$, $\langle\varphi| = |\varphi\rangle^\dagger$ pedig egy sorvektor. A † az adjungálás operátora:

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}^\dagger = \begin{bmatrix} \overline{a}_{11} & \cdots & \overline{a}_{m1} \\ \vdots & \ddots & \vdots \\ \overline{a}_{1n} & \cdots & \overline{a}_{mn} \end{bmatrix} \tag{2.1}$$

Ahol $\overline{a}_{ij}$ az a_{ij} komplex szám konjugáltját jelöli. A belső vagy skalár szorzatot a következőképpen számítjuk:

$$\langle\varphi||\psi\rangle = \sum_{i=1}^{n} \overline{\varphi}_i \psi_i \tag{2.2}$$

Egy vektor hossza a következő:

$$\left\| |\varphi\rangle \right\| = \sqrt{\langle\varphi||\varphi\rangle} \tag{2.3}$$

A legegyszerűbb eset a két állapotú kvantum rendszer, amit kvantum bitnek vagy qubitnek nevezünk. Egy elektron spinje, vagy egy foton polarizációja jó példa erre. Ennek a rendszernek a $\mathbb{C}^2$ Hilbert-tér felel meg, a rendszer állapota, azaz a qubit értéke:

$$|\varphi\rangle = a|0\rangle + b|1\rangle = \begin{bmatrix} a \\ b \end{bmatrix}, \quad a,b \in \mathbb{C},\ |a|^2 + |b|^2 = 1 \tag{2.4}$$

Itt $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ és $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ a tér egymása merőleges egység hosszú bázisvektorai (ortonormált bázisa). Az a és b számokat valószínűségi amplitúdóknak nevezzük, ennek jelentése később kiderül.

2.1.2 Kvantumrendszer időbeli fejlődése

Zárt kvantumrendszer időbeli fejlődését unitér transzformációval írhatjuk le. Azaz, ha a rendszer egy kezdeti időpillanatban a $|\varphi_0\rangle$ állapotban volt, akkor egy későbbi időpillanatban a

$$|\varphi_1\rangle = \mathbf{U}|\varphi_0\rangle \tag{2.5}$$

állapotban lesz. $\mathbf{U}$ *csak a kezdeti és a későbbi időpillanattól függ.*

Egy négyzetes $\mathbf{U}$ mátrix unitér, ha teljesül rá az alábbi összefüggés:

$$\mathbf{U}^\dagger\mathbf{U} = \mathbf{I} \tag{2.6}$$

Unitér transzformációt gyakran kapunak is neveznek. Például a Hadamard kapu:

$$\mathbf{U} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \mathbf{U}^\dagger, \quad \mathbf{U}^\dagger\mathbf{U} = \mathbf{I} \tag{2.7}$$

$$\mathbf{U}|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \tag{2.8}$$

A Hadamard kapu a $|0\rangle$ bázisállapotot a két bázisállapot szuperpozíciójába alakította át.

2.1.3 Mérés

Legyen $\mathcal{X}$ a mérés lehetséges eredményeinek a halmaza. Egy mérés az $\mathcal{M} = \{\mathbf{M}_x\}$, $x \in \mathcal{X}$, $\mathbf{M}_x \in \mathcal{H}$ mérési operátorok (Kraus operátorok) halmazával adható meg. A mérési operátoroknak teljesíteniük kell az úgynevezett teljességi feltételt:

$$\sum_x \mathbf{M}_x^\dagger \mathbf{M}_x = \mathbf{I} \tag{2.9}$$

Ha a megmérendő rendszer állapota $|\varphi\rangle$, akkor annak a valószínűsége, hogy a mérés az x eredményt adja:

$$p_x = \langle \varphi | \mathbf{M}_x^\dagger \mathbf{M}_x | \varphi \rangle \tag{2.10}$$

A mérés után a rendszer állapota a következő lesz:

$$\frac{\mathbf{M}_x |\varphi\rangle}{\sqrt{p_x}} \tag{2.11}$$

Nagyon fontos, hogy a kvantummechanikában a mérés egy véletlenszerű folyamat. Ugyanazt az állapotot megmérve különböző eredményeket kaphatunk. A mérés eredménye csak valószínűségi alapon jósolható, a konkrét esemény teljesen véletlenszerűen következik be. Másik fontos tulajdonság a mérés hatása a rendszer állapotára, ami a mérés után megváltozik.

Nézzünk egy példát! A mérés legyen a következő:

$$\mathcal{M} = \{\mathbf{M}_0, \mathbf{M}_1\} \tag{2.12}$$

$$\mathbf{M}_0 = |0\rangle\langle 0| = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \ \mathbf{M}_1 = |1\rangle\langle 1| = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.13}$$

A $|\psi\rangle\langle\psi|$ mátrix a $|\psi\rangle$ vektor által kifeszített egydimenziós altérre történő vetítés mátrixa. Nézzük meg, mi történik, ha egy qubitet megmérünk! Mi a valószínűsége annak, hogy a mérés eredménye 0 lesz?

$$|\varphi\rangle = a|0\rangle + b|1\rangle = \begin{bmatrix} a \\ b \end{bmatrix} \tag{2.14}$$

$$p_0 = \langle\varphi|\mathbf{M}_0^\dagger\mathbf{M}_0|\varphi\rangle = |a|^2 \tag{2.15}$$

Hasonlóan:

$$p_1 = \langle\varphi|\mathbf{M}_1^\dagger\mathbf{M}_1|\varphi\rangle = |b|^2 \tag{2.16}$$

Ebből látszik a valószínűségi amplitúdó elnevezés jogszerűsége. A mérés után a rendszer állapota a következő lesz:

$$\frac{\mathbf{M}_0|\varphi\rangle}{|a|} = |0\rangle \tag{2.17}$$

A mérés során egy állapotvektor levetítődik valamelyik mérési operátor által meghatározott altérre, majd egy hosszúra normálódik. A rendszer állapota tehát összeomlik, cserébe azonban információt kapunk a rendszerről. Látható, hogy a mérés újbóli végrehajtása már mindig ugyanazt az eredményt adja, és tovább már a rendszer állapota sem változik.

2.1.4 Összetett rendszer

Legyen $\mathcal{H}_1$, $\mathcal{H}_2$ két kvantumrendszerhez rendelt Hilbert-tér. Ekkor a két rendszerből álló összetett rendszerhez a $\mathcal{H}_1 \otimes \mathcal{H}_2$ Hilbert-tér rendelhető.

$\otimes$ a tenzor szorzás operátora. Nézzünk egy példát:

$$|\varphi_1\rangle = a|0\rangle + b|1\rangle = \begin{bmatrix} a \\ b \end{bmatrix}, \quad |\varphi_1\rangle \in \mathcal{H}_1 = \mathbb{C}^2 \tag{2.18}$$

$$|\varphi_2\rangle = c|0\rangle + d|1\rangle = \begin{bmatrix} c \\ d \end{bmatrix}, \quad |\varphi_2\rangle \in \mathcal{H}_2 = \mathbb{C}^2 \tag{2.19}$$

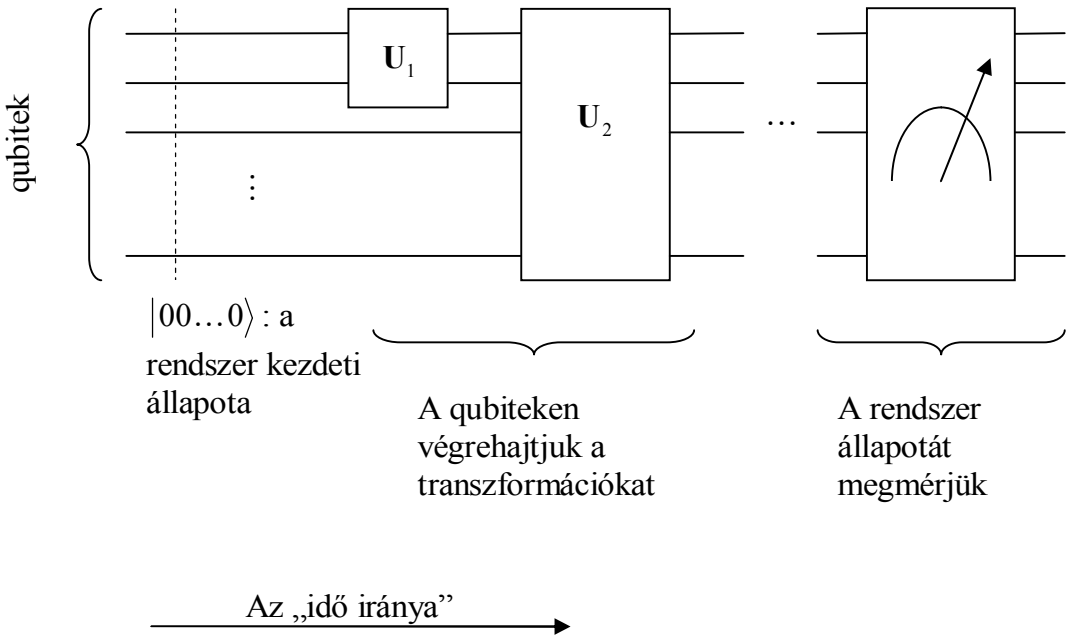
Az összetett rendszer állapota:

$$|\varphi_1\rangle \otimes |\varphi_2\rangle = |\varphi_1\varphi_2\rangle = \begin{bmatrix} ac \\ ad \\ bc \\ bd \end{bmatrix} = ac|00\rangle + ad|01\rangle + bc|10\rangle + bd|11\rangle, \qquad (2.20)$$

$$|\varphi_1\varphi_2\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2 = \mathbb{C}^4 \qquad (2.21)$$

2.2 Kvantum-áramkör

Most, hogy az alapfogalmakat bevezettem, bemutatom a kvantum algoritmusokat, és leggyakoribb megadási módjukat, a kvantum-áramköröket. Az algoritmusok általános felépítése a 2-1. ábrán látható. A rendszer valamilyen kezdeti állapotban van, ez általában $|00\ldots0\rangle$. A rendszer qubitjein különböző unitér transzformációkat hajtunk végre, majd a rendszer állapotát megmérjük. A mérés eredménye az algoritmus kimenete. A kvantum algoritmus „futási ideje” arányos az áramkörben lévő elemi kapuk számával. Az elemi kapukat nem definiálom pontosan, a következő fejezetben bemutatásra kerülők tekinthetőek azoknak.



2-1. ábra: Egy kvantum algoritmus vázlatos áramköri rajza

2.3 Nevezetes kvantum kapuk és mérések

Az alábbiakban áttekintjük a leggyakrabban használt kvantum kapukat illetve méréseket.

Hadamard kapu:

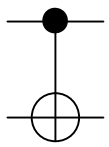
$$\mathbf{H} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.22}$$

Pauli I, X, Y, Z kapu:

$$\mathbf{I} = \sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \mathbf{X} = \sigma_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ \mathbf{Y} = \sigma_2 = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix},\ \mathbf{Z} = \sigma_3 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{2.23}$$

A CNot kapu és áramköri jele:

$$\mathbf{CNot} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.24}$$



A Swap kapu:

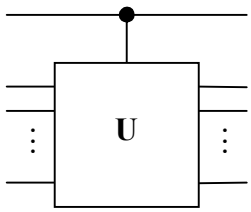
$$\mathbf{Swap} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.25}$$

Ez megcseréli a két bemeneti qubit állapotát $\left(|\varphi\rangle|\psi\rangle \to |\psi\rangle|\varphi\rangle\right)$.

Vezérelt kapuk:

A vezérelt kapu szintén egy unitér transzformáció, amely a következőképpen működik. Ha a vezérelt **U** kapu felső qubitje $|0\rangle$, akkor az alsó qubitek nem változnak, ha $|1\rangle$, akkor az alsó qubiteken az **U** transzformáció hajtódik végre. A vezérelt **U** kapu mátrixa és áramköri jele: (**U** n qubites.)

$$\mathbf{U}_{vezérelt} = \begin{bmatrix} \mathbf{I}^{\otimes n} & \mathbf{0} \\ \mathbf{0} & \mathbf{U} \end{bmatrix} \qquad (2.26)$$



($^{\otimes n}$ az önmagával vett n-szeres tenzor szorzást jelenti.)

Ortogonális mérés a bázisállapotokban: (A rendszer n qubites.)

$$\mathcal{M}_+^n = \bigcup_{i=0}^{2^n-1} \left\{ |e_i\rangle\langle e_i| \right\}, \quad |e_i\rangle \in \mathbb{C}^{2^n} \qquad (2.27)$$

Itt $|e_i\rangle$ a tér i-edik egységvektora, azaz:

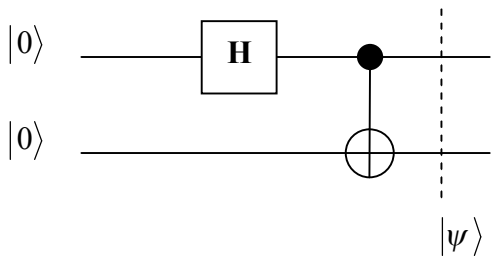
$$|e_i\rangle = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} 0. \\ \\ i-1. \\ i. \\ i+1. \\ \\ 2^n-1. \end{matrix} \qquad (2.28)$$

2.4 Összefonódás

Tekintsük a 2-2. ábrán látható kvantum áramkört. A rendszer állapota a transzformációk végén:

$$|\psi\rangle = \mathbf{CNot} \cdot (\mathbf{H} \otimes \mathbf{I})(|0\rangle \otimes |0\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \qquad (2.29)$$

Ezt a konstrukciót EPR párnak hívjuk Albert Einstein, Boris Podolsky és Nathan Rosen után. Könnyen észrevehetünk néhány furcsa tulajdonságot. Mérjük meg az első vagy a második qubitet az $\mathcal{M}_+^1$ méréssel! Kiszámítható, hogy a mérés eredménye



2-2. ábra: EPR pár létrehozása

mindkét esetben ½ valószínűséggel 0 illetve 1, a rendszer pedig a mérés után a $|00\rangle$ vagy az $|11\rangle$ állapotban lesz. Ezután, ha a másik qubitet mérjük meg, az már 1 valószínűséggel ugyanazt az eredményt adja, mint az első mérés! Ez akkor is igaz, ha a két qubit egymástól nagyon messze van. Ezt a furcsa tulajdonságot összefonódásnak hívjuk, és majd a későbbiekben, mint erőforrás használjuk. Látható, hogy az összefonódott állapot nem állítható elő két qubit kompozíciójaként (nem szeparálható)!

$$\neg\exists\,|\psi_1\rangle,|\psi_2\rangle\in\mathbb{C}^2:\quad |\psi_1\rangle\otimes|\psi_2\rangle=|\psi\rangle \tag{2.30}$$

2.5 Sűrűségmátrix

Eddig zárt rendszerek jellemzéséről volt szó. Lehetséges azonban, hogy a kvantumrendszerünk össze van fonódva egy másik kvantumrendszerrel, mint ahogy a fenti EPR pár esetében a két qubit. Mint arról már volt szó, az EPR pár egy qubitjének állapota nem írható le állapotvektorral. Az ilyen állapotokat kevert (mixed) állapotoknak hívjuk, egyébként az állapot tiszta (pure). A tiszta állapotokat leírhatjuk állapotvektor segítségével, ha azonban kevert állapotokat is szeretnénk leírni, akkor egy új fogalomra van szükségünk. Ezt sűrűségmátrixnak nevezik, és általában ρ-val jelölik. Tiszta állapot esetén:

$$\rho=|\varphi\rangle\langle\varphi|. \tag{2.31}$$

Tegyük fel, hogy egy két komponensből álló összetett rendszerünk van, amelynek komponenseihez a $\mathcal{H}_A$ illetve a $\mathcal{H}_B$ Hilbert teret rendeljük. Ha az összetett rendszer állapota tiszta, akkor leírható egy $|\varphi\rangle_{AB}\in\mathcal{H}_A\otimes\mathcal{H}_B$ állapotvektorral:

$$|\varphi\rangle_{AB}=\sum_{i,\mu}a_{i\mu}|i\rangle_A\otimes|\mu\rangle_B. \tag{2.32}$$

Itt $\{|i\rangle_A\}$ ortonormált bázisa $\mathcal{H}_A$-nak, $\{|\mu\rangle_B\}$ pedig $\mathcal{H}_B$-nek. Az *A* rendszer állapotát leíró ρ_A sűrűségmátrixot a következőképpen kaphatjuk az *AB* rendszer állapotából:

$$\rho_A=\mathrm{tr}_B\left(|\varphi\rangle_{AB\ AB}\langle\varphi|\right)\equiv\sum_{i,j,\mu}a_{i\mu}\overline{a}_{j\mu}|i\rangle_{A\ A}\langle j|. \tag{2.33}$$

A (2.31) és (2.33) egyenletből látható, hogy egy ρ sűrűségmátrixra teljesülnek az alábbiak:

1. ρ önadjungált, azaz $\rho^\dagger = \rho$.
2. ρ pozitív szemidefinit, azaz $\forall|\varphi\rangle: \langle\varphi|\rho|\varphi\rangle \geq 0$.
3. $\mathrm{tr}(\rho) = 1$, azaz a főátlóban lévő elemek összege (a mátrix nyoma) 1.

Ezek a feltételek elégségesek is ahhoz, hogy ρ sűrűségmátrix legyen, mert minden ilyen ρ felírható úgy, mint

$$\rho = \sum_a p_a |\varphi_a\rangle\langle\varphi_a|, \tag{2.34}$$

ahol $0 < p_a \leq 1$ és $\sum_a p_a = 1$. ρ tehát a konvex kombinációja a $\rho_a = |\varphi_a\rangle\langle\varphi_a|$ tiszta állapotoknak. Sőt, ρ minden szempontból úgy viselkedik, mintha egy kvantumrendszer állapotáról azt tudnánk, hogy p_a valószínűséggel a ρ_a állapotban van. Így tehát egy tetszőleges ρ fizikai megvalósításának egy lehetséges módját is megkaptuk.

Szükséges még a posztulátumok kiterjesztése sűrűségmátrixokra. Az időbeli fejlődés unitér transzformációjának hatását könnyen kaphatjuk (2.5)-ből és (2.34)-ből:

$$\rho(t) = \mathbf{U}\rho(0)\mathbf{U}^\dagger. \tag{2.35}$$

Hasonlóan (2.10), (2.11) és (2.34)-ből következik a mérés kiterjesztése. Az x eredmény valószínűsége

$$p_x = \mathrm{tr}\left(\mathbf{M}_x^\dagger \mathbf{M}_x \rho\right). \tag{2.36}$$

A mérés utáni rendszerállapot pedig

$$\rho_x' = \frac{\mathbf{M}_x \rho \mathbf{M}_x^\dagger}{p_x}. \tag{2.37}$$

Összetett rendszert sűrűségmátrixok esetében is tenzor szorzással képzünk:

$$\rho_1, \rho_2 \to \rho_1 \otimes \rho_2 . \tag{2.38}$$

2.6 Bloch gömb

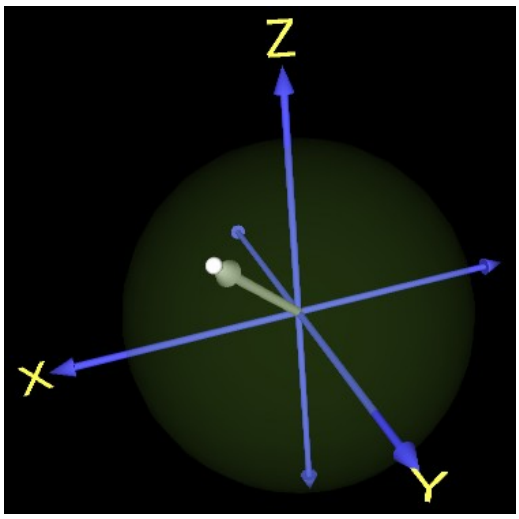
Egy tetszőleges 2×2-es önadjungált mátrix jellemezhető egy 4 dimenziós $\vec{P}'$ valós vektorral, a következőképpen:

$$\rho\left(\vec{P}'\right) = \frac{1}{2}\vec{P}'\cdot\vec{\sigma} = \frac{1}{2}\begin{bmatrix} P_0 + P_3 & P_1 - iP_2 \\ P_1 + iP_2 & P_0 - P_3 \end{bmatrix}, \tag{2.39}$$
$$\vec{P}' = \begin{bmatrix} P_0 & P_1 & P_2 & P_3 \end{bmatrix} \in \mathbb{R}^4, \quad \vec{\sigma} = \begin{bmatrix} \sigma_0 & \sigma_1 & \sigma_2 & \sigma_3 \end{bmatrix}.$$

A mátrix nyoma 1 akkor és csak akkor ha $P_0 = 1$, és belátható, hogy ha ez igaz, akkor a pozitív szemidefinitség szükséges és elégséges feltétele

$$\left\|\vec{P}\right\| \le 1, \tag{2.40}$$

ahol $\vec{P} = \begin{bmatrix} P_1 & P_2 & P_3 \end{bmatrix}$. Egy 1 qubites rendszer állapota tehát leírható egy valós vektorral, amely a 3 dimenziós egységgömb része. Ezt Bloch gömbnek nevezzük (2-3. ábra). A Bloch gömb felületén vannak a tiszta, belsejében a kevert állapotok.



2-3. ábra: Egy tiszta állapot ábrázolása a Bloch gömbön, a QCircuit programmal rajzolva

Bebizonyítható, hogy a Bloch gömbön bármely tengely körüli forgatás unitér transzformáció, illetve bármely unitér transzformációnak megfeleltethetünk a Bloch gömbön valamilyen forgatást. Egy tetszőleges $\mathbf{U}$ tehát felírható úgy, mint

$$\mathbf{U} = e^{i\theta}\mathbf{R}_x(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_z(\gamma), \tag{2.41}$$

ahol $\mathbf{R}_x(\alpha)$, $\mathbf{R}_y(\beta)$ és $\mathbf{R}_z(\gamma)$ az x, y és z tengely körüli α, β és γ szöggel történő forgatások mátrixai, $e^{i\theta}$ pedig a globális fázis, aminek nincs fizikai jelentősége. A forgatások mátrixai:

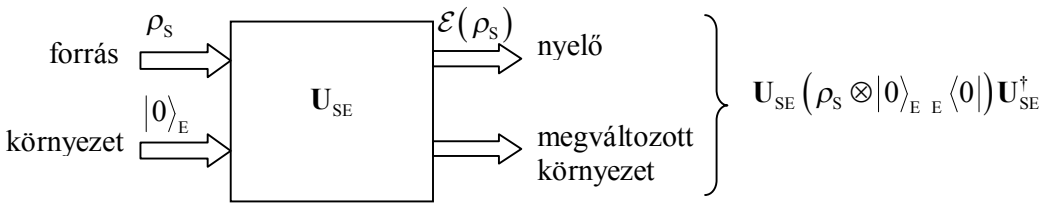
$$\mathbf{R}_x(\alpha) = \begin{bmatrix} \cos\frac{\alpha}{2} & -i\sin\frac{\alpha}{2} \\ -i\sin\frac{\alpha}{2} & \cos\frac{\alpha}{2} \end{bmatrix}, \tag{2.42}$$

$$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\frac{\beta}{2} & -\sin\frac{\beta}{2} \\ \sin\frac{\beta}{2} & \cos\frac{\beta}{2} \end{bmatrix}, \tag{2.43}$$

$$\mathbf{R}_z(\gamma) = \begin{bmatrix} e^{-i\frac{\gamma}{2}} & 0 \\ 0 & e^{i\frac{\gamma}{2}} \end{bmatrix}. \tag{2.44}$$

2.7 Kvantum csatorna

Tegyük fel, hogy egy ρ_S kvantumállapotot küldünk át egy kvantum csatornán. Ilyenkor ρ_S nem tekinthető zárt rendszernek, mert kölcsönhatásba lép a „környezettel”, melynek állapotát jelöljük $|0\rangle_E$-vel. Egy $\mathbf{U}_{SE}$ unitér transzformáció hat az átküldendő állapotból és a környezetből létrejövő összetett rendszeren, aminek hatására a kettő összefonódik, és a ρ_S-ben kódolt kvantum információ áttevődik a ρ_S és a környezet korrelációjába, és így hozzáférhetetlenné válik. Ez okozza a kvantum csatorna zajos viselkedését. Az 2-4. ábrán látható a folyamat illusztrációja.



2-4. ábra: A kvantum csatorna modellje

Ha mellőzzük a környezetre való hivatkozást, akkor a csatorna egy $\mathcal{E}$ lineáris leképezésnek tekinthető, ami sűrűségmátrixokat sűrűségmátrixokba visz. Egy tetszőleges ilyen $\mathcal{E}$ – mint egy kvantumos mérés – jellemezhető az $\{\mathbf{M}_x\}$ Kraus operátorok halmazával, amelyre a teljességi feltételnek (2.9) fenn kell állnia. A csatorna transzformációját a következőképpen számítjuk:

$$\mathcal{E}(\rho_S) = \sum_x \mathbf{M}_x \rho_S \mathbf{M}_x^\dagger . \tag{2.45}$$

Az $\mathbf{M}_x$ mátrixok elemei $\mathbf{U}_{SE}$-ből az alábbi összefüggéssel kaphatók:

$${}_S\langle k|\mathbf{M}_x|l\rangle_S = \left({}_S\langle k| \otimes {}_E\langle x|\right)\mathbf{U}_{SE}\left(|l\rangle_S \otimes |0\rangle_E\right). \tag{2.46}$$

Itt $\{|i\rangle_E\}$ a környezet Hilbert terének ortonormált bázisa, $|k\rangle_S$ és $|l\rangle_S$ pedig a k-dik és l-dik egységvektora annak a bázisnak, amelyben $\mathbf{M}_x$-et felírjuk. Összehasonlítva (2.37)-et (2.45)-tel látható, hogy $\mathcal{E}(\rho_S)$ a konvex kombinációja a ρ_x' állapotoknak, súlyozva a p_x valószínűségekkel. Egy kvantum csatornát tehát tekinthetünk egy olyan mérésnek, amikor nincs információnk a mérés kimeneteléről. A csatorna kimenete a lehetséges ρ_x' állapotok valószínűségi eloszlása.

A következőkben három egy qubites kvantum csatornát definiálok. A „depolarizing” csatornát úgy írhatjuk le, hogy $1-p$ valószínűséggel a qubit nem változik, p valószínűséggel hiba történik, ekkor a csatorna kimenete a „teljesen véletlen” $0.5\mathbf{I}$ állapot lesz. A csatorna Kraus reprezentációja:

$$\mathbf{M}_0 = \sqrt{1-\frac{3}{4}p}\,\mathbf{I},\ \mathbf{M}_1 = \sqrt{\frac{p}{4}}\mathbf{X},\ \mathbf{M}_2 = \sqrt{\frac{p}{4}}\mathbf{Y},\ \mathbf{M}_3 = \sqrt{\frac{p}{4}}\mathbf{Z}. \tag{2.47}$$

A „phase-damping” csatornát az alábbi Kraus operátorok definiálják:

$$\mathbf{M}_0 = \sqrt{1-p}\,\mathbf{I},\ \mathbf{M}_1 = \sqrt{p}\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix},\ \mathbf{M}_2 = \sqrt{p}\begin{bmatrix}0 & 0\\ 0 & 1\end{bmatrix}. \tag{2.48}$$

A csatornán átküldött állapot sűrűségmátrixának főátlójában lévő elemek nem változnak, míg a nem átlós elemek lecsökkennek. A csatorna előnyben részesíti a

$\{|0\rangle_S, |1\rangle_S\}$ bázist, ebben a bázisban bithiba nem fordul elő. A „phase-damping” (fázis csillapítás) név világossá válik, ha megnézzük hogy a $a|0\rangle_S + b|1\rangle_S$ állapot – ami a $|0\rangle_S$ és $|1\rangle_S$ koherens szuperpozíciója – a $p = 1$ paraméterű csatornán átérve elbomlik a $|a|^2 |0\rangle_S \langle 0| + |b|^2 |1\rangle_S \langle 1|$ inkoherens szuperpozícióba.

Az „amplitude-damping” csatorna egy két állapotú kvantumrendszer gerjesztett állapotának spontán bomlásának modellje. A csatorna Kraus operátorai:

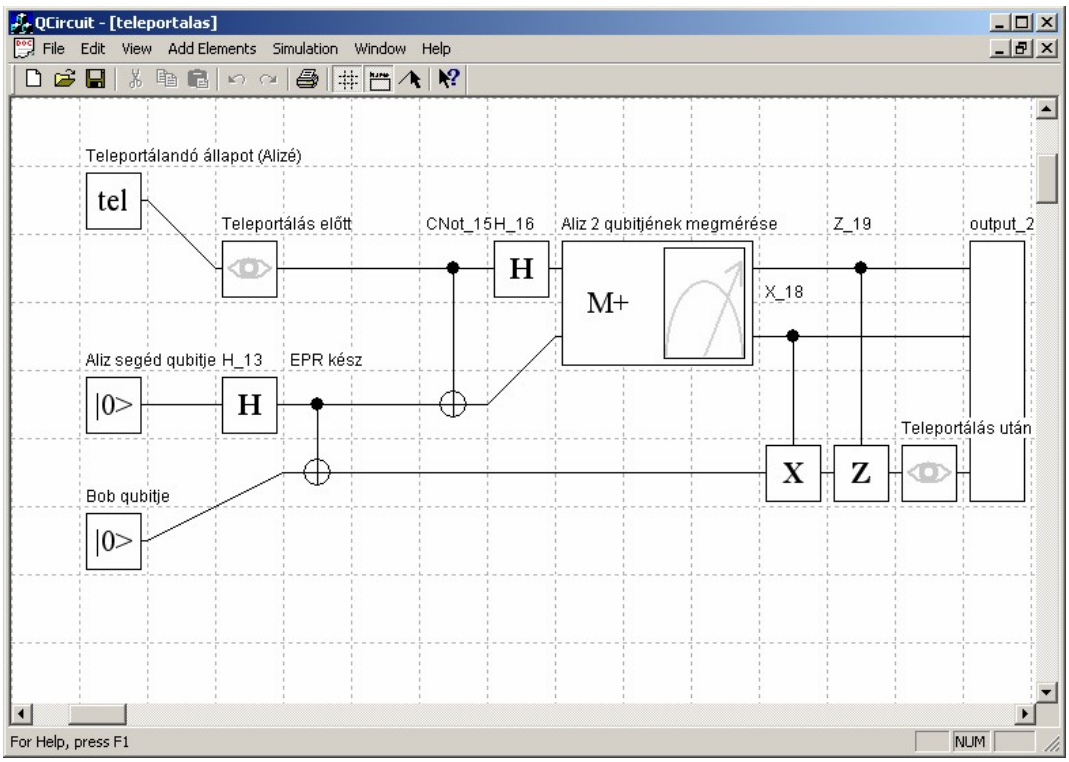
$$\mathbf{M}_0 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{bmatrix}, \quad \mathbf{M}_1 = \begin{bmatrix} 0 & \sqrt{p} \\ 0 & 0 \end{bmatrix}. \tag{2.49}$$

Látható, hogy a $|0\rangle_S$ alapállapot változatlan marad, míg a gerjesztett $|1\rangle_S$ állapot elbomlik az alapállapotra p valószínűséggel.

A matematikai formalizmus bevezetése után a következő fejezetben a szimulációs programom bemutatásával folytatom. Hogy az eddig felépített eszköztár mire használható a gyakorlatban, azt a 4. és 5. fejezetben mutatom be.

3 A QCircuit program

A célom egy olyan kvantum-áramkör szimulátor program kifejlesztése volt, amellyel a bonyolult áramkörök is könnyedén megszerkeszthetőek, és a kezelőfelülete nagyon egyszerű. Ezért a programot Windows operációs rendszerhez készítettem. A programban lehetőség van bármely – a kvantummechanikai törvényeknek eleget tevő – áramkör megépítésére, és az egyszerűbb felhasználás érdekében több előre definiált objektumot is tartalmaz, pl. kapuk, mérések, csatornák, stb. A bonyolultabb áramkörök könnyebb megszerkeszthetősége érdekében lehetőség van a hierarchikus építkezésre azzal, hogy egyes áramkör részeket külön fileban építünk meg, majd ezeket egyetlen építőelemként használjuk akár többször is, és bármilyen mélységben. További követelmény, hogy a szimulált rendszer állapotait könnyedén megfigyelhessük. Lehetőség van a szimuláció egymás után sokszori lefuttatására, ami azért hasznos, mert az áramkörben elhelyezett mérések következtében a rendszer fejlődése többféleképpen alakulhat. A többszöri futtatással lehetőség van a rendszer viselkedését statisztikusan vizsgálni.



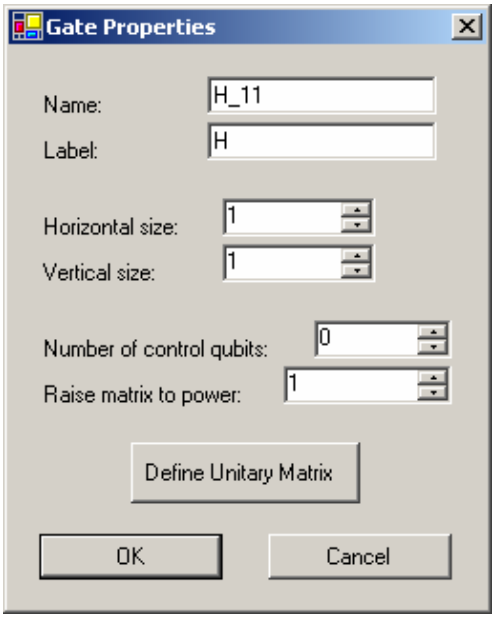
3-1. ábra: Pillanatkép a QCircuit programból

3.1 Használati útmutató

A QCircuit web oldaláról [21] a program legfrissebb verziója letölthető, a futtatáshoz Windows 98/NT/XP mellett .NET Framework [20] szükséges. A program nyelve angol. Kezelése, mint általában a Windows alapú programoké nagyon egyszerű. A File menü elemei (New, Open, Close, Save, stb.) a szokásosak, ezek működését nem írom le. Az Edit, View és a Window menüből is csak azokat a menüpontokat írom le, amelyek működése nem triviális.

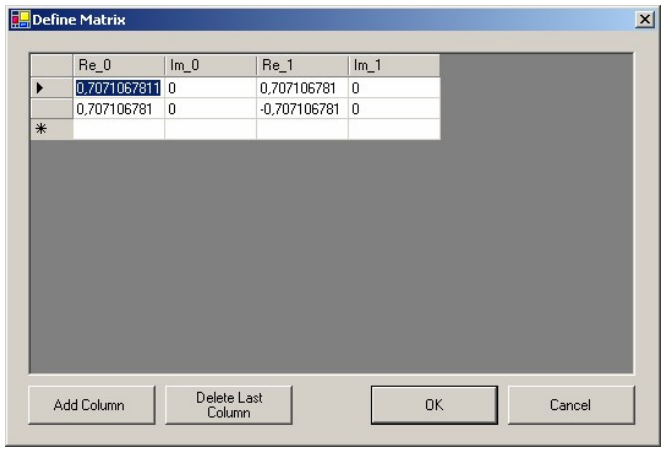
3.1.1 Áramkör építése

Kvantum kaput (3-9. (a) ábra) hozzáadni az Add Elements / Gates menüvel lehet. Van néhány előre definiált, gyakran használt kapu, úgy mint Hadamard (2.22), Pauli X, Y, Z (2.23), CNot (2.24), $\mathbf{R}_x(\alpha)$ (2.42), $\mathbf{R}_y(\beta)$ (2.43) és $\mathbf{R}_z(\gamma)$ (2.44), és ami a legfontosabb, lehetőség van tetszőleges kaput megadni (Add Elements / Gates / Custom). A kapu tulajdonságai párbeszéd-panelen (3-2. ábra) megadhatjuk a kapu nevét, a dobozra ráírt címkét, a doboz méretét, a kapu vezérelő qubitjeinek számát, definiálhatjuk az unitér transzformáció mátrixát, és lehetőség van a mátrix adott hatványra emelésére is. A kapu mátrixát, és általában a programban minden mátrixot és vektort a Define Matrix párbeszédpanelen (3-3. ábra) nagyon egyszerűen – egy táblázat kitöltésével – adhatunk meg. Láthatjuk, hogy egy elemnek vannak kimeneti és/vagy bemeneti interfészei, amiket majd össze kell kötni. Ha egy kaput elhelyeztünk, kattintsunk rá a jobb gombbal. Ekkor egy gyorsmenü jelenik meg, amellyel lehetőség van a kaput törölni (Delete), létrehozni egy ugyanilyen kaput (Clone), és a kapu tulajdonságait beállítani (Gate Properties). Bármely más elemre kattintva szintén előugrik egy ehhez hasonló gyorsmenü, ahol az elemnek megfelelő műveletek hajthatók végre, illetve a tulajdonság párbeszédpanel az elemre duplán kattintva is elérhető. Ezt a későbbiekben már nem részletezem.

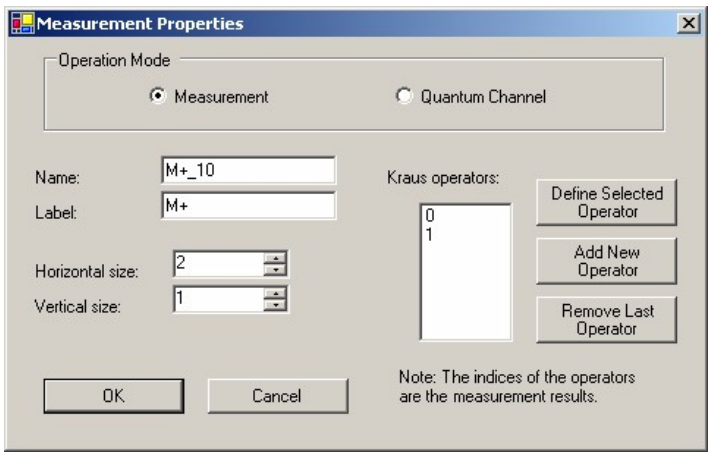


3-2. ábra: Kapu tulajdonságai párbeszédpanel

Mérést (3-9. (b) ábra) megadni az Add Elements / Measurements menüvel lehet. Az Orthogonal-lal egy $\mathcal{M}_+^n$ mérést definiálhatunk, miután megadtuk *n* értékét. Miután lehelyeztük, a jobb gombbal előugró gyorsmenüből a Mérés tulajdonságai párbeszédpanelen (3-4. ábra) megnézhetjük a mérés operátorait, hogy pontosan lássuk a mérés működését. A Custom menüpont is ezt a párbeszédpanelt jeleníti meg. A programban a mérés eredményei csak természetes számok lehetnek, amelyek 0-tól egyesével következnek ($x = 0,1,\ldots,|\mathcal{X}|-1$).



3-3. ábra: Define Matrix párbeszédpanel

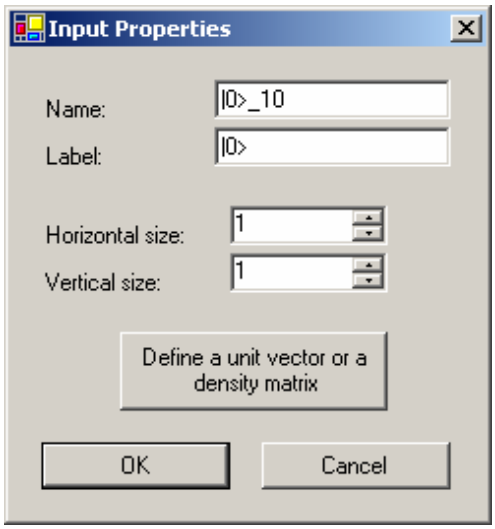


3-4. ábra: Mérés tulajdonságai párbeszédpanel

Minden eredményhez definiálható az $\mathbf{M}_x$ mérési operátor (Define Selected Operator gomb). Újat az Add New Operator gombbal lehet felvenni, az utolsót törölni a Remove Last Operator-ral lehet. A mérés dobozának jobb oldalán lévő téglalapba a szimuláció során – ha többször futtatjuk a szimulációt, akkor az utolsó – mérés eredménye kerül.

A programban inputnak nevezzük azokat az elemeket, melyekkel a qubitek kezdeti állapotát lehet beállítani (3-9. (c) ábra). Ugyanúgy kell felvenni, mint például a kapukat, és itt is vannak előre definiáltak, úgy mint $|0\rangle$, $|1\rangle$ és a (29)-ben definiált EPR pár. A tulajdonság panelen (3-5. ábra) nem egy unitér mátrixot, hanem egy vagy 2^n dimenziós egy hosszú oszlopvektort, vagy egy $2^n \times 2^n$ dimenziós sűrűségmátrixot kell megadni. Az inputnak természetesen nincs bemeneti interfésze. Az áramkör végén (jobb oldalán) lévő elemek kimeneti interfészeit be kell kötni egyetlen output elem (3-

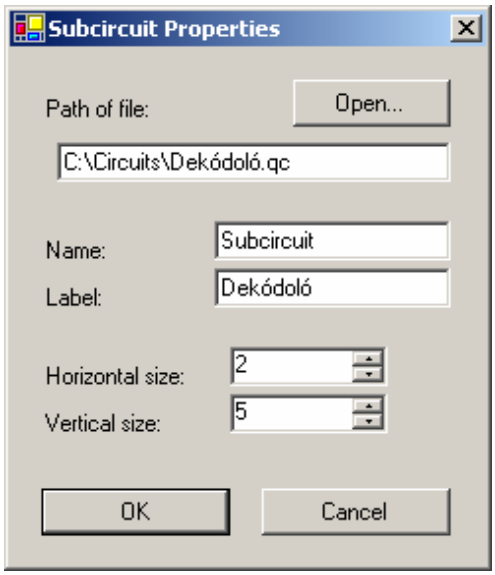
9. (h) ábra) bemeneti interfészeibe. Ez azért kell, mert az áramkörben nem lehet bekötetlen interfész a szimulációkor. Outputot felvenni az Output menüponttal lehet, és csak egy Output lehet az áramkörben.



3-5. ábra: Input tulajdonságai párbeszédpanel

A programban a „depolarizing”, „phase-damping” és „amplitude-damping” előre beépített csatornamodelleken túl tetszőleges csatornát is megadhatunk (3-9. (d) ábra), Kraus operátoros alakban. Ezt ugyanazzal a párbeszédpanellel tehetjük meg, mint amelyiken a mérés tulajdonságait állítjuk be (3-4. ábra), a különbség csak annyi, hogy az Operation Mode-nál a Quantum Channel van bekapcsolva. Csatorna esetében az operátorok sorrendjének (indexeinek) nincs jelentősége. A programban bármikor lehetőség van a mérések és a csatornák között átkapcsolni, ilyenkor a Kraus operátorok változatlanok maradnak, az elem azonban a (2.11) vagy (2.37)-ben definiált transzformációt végzi mérés esetén vagy a (2.45)-ben definiáltat csatorna esetén.

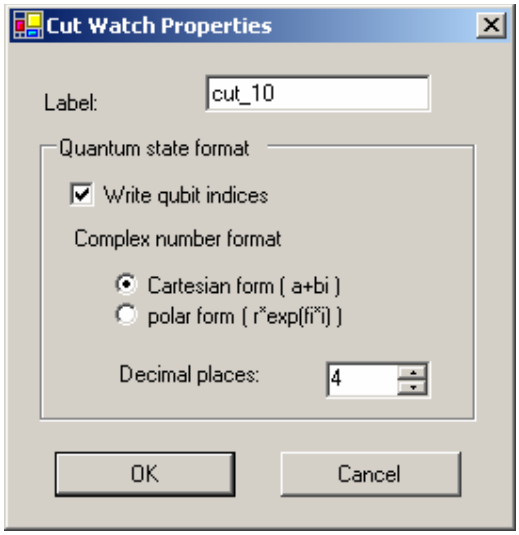
A Subcircuit menüponttal lehetőség van alhálózat (3-9. (g) ábra) megadására. A párbeszédpanel (3-6. ábra) Open gombjával megadhatunk tetszőleges QCircuit filet, amit aztán egyetlen építőelemként használhatunk. A szimuláció során majd ezeket a program kibontja, azaz a fileban lévő elemeket beszerkeszti az áramkörbe, kivéve az input és az output elemeket, mivel azok a subcircuit bemeneti illetve kimeneti interfészei lesznek. A subcircuit kimeneti interfészeinek sorrendje meg fog egyezni az egyetlen output bemeneti interfészeinek sorrendjével. A bemeneti interfészek sorrendje az inputok sorrendjével fog



3-6. ábra: Subcircuit tulajdonságai párbeszédpanel

megegyezni, amit az olvasással megegyező sorrendben kaphatunk (balról jobbra, fentről le). Ha a subcircuitre duplán kattintunk, akkor megnyithatjuk szerkesztésre, de ne felejtsük el elmenteni, ha szimulálni akarjuk azt az áramkört, amelyiknek része.

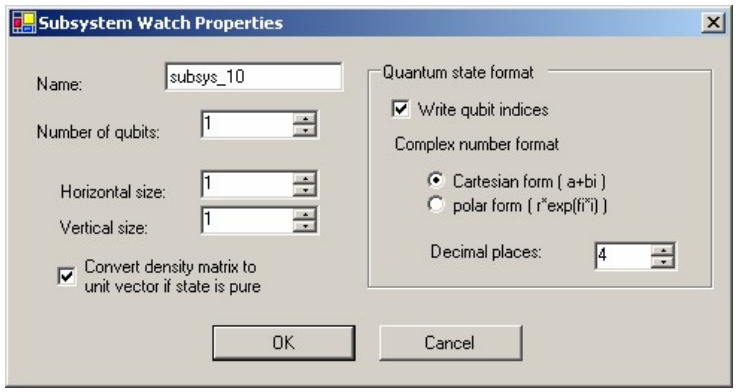
A Watch-ok olyan elemek, amelyekkel qubitek állapotát lehet figyelni a szimuláció során, és ennek megfelelően nem befolyásolják az áramkör működését. Három fajta watch-ot használhatunk. A Cut Watch (3-9. (i) ábra) különbözik az eddigi elemektől. Nem doboz kinézetű, és nem is kell bekötni az áramkörbe. Ez a watch egy vágás az áramkör gráfjában. Annyi élet kell metszenie, mint ahány qubites a rendszer. A szimuláció során a teljes rendszer állapotát lehet monitorozni vele, azoknál az éleknél, amiket elmetsz. A szimuláció eredményeként létrejövő fileba ki lesz írva a Cut Watch helyén előfordult rendszerállapotok, azok előfordulásainak száma, és az előfordulások számának aránya a szimulációs futások számához viszonyítva. Ez a watch alkalmas arra, hogy az áramkört építése közben szimulálva ellenőrizhessük, hiszen az áramkörbe nem kell beszerkeszteni, drag-and-drop elven egyszerűen áthelyezhető, és ha az áramkört Subcircuit-ként használjuk, akkor nem írja ki az állapotot az eredmény fileba. A Cut Watch létrehozásakor megjelenő, és a gyorsmenün keresztül is elérhető párbeszédpanelen (3-7. ábra) megadhatjuk a Cut Watch nevét, azt hogy a bázisállapotok címkéi ki legyenek-e írva (Write qubit indices), továbbá a komplex számok kiírásának formátumát.



3-7. ábra: Cut Watch tulajdonságai párbeszédpanel

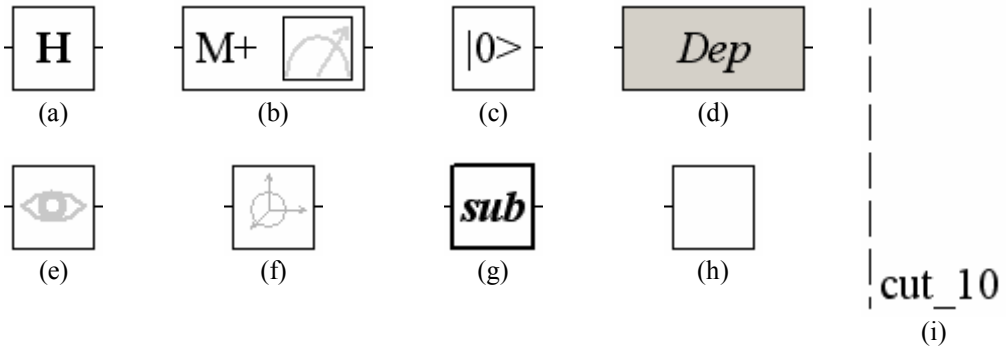
A Subsystem Watch (3-9. (e) ábra) a többi elemhez hasonlóan doboz kinézetű, szintén a rendszerállapot figyelésére szolgál, és a kimeneti fileba is hasonló eredményt produkál, mint a Cut Watch. Nagy különbség azonban, hogy teljes rendszer állapotának figyelése helyett itt lehetőség van tetszőleges számú és tetszőlegesen választott qubitek állapotának figyelésére. A Subsystem Watch tulajdonságai párbeszédpanelen (3-8. ábra) a Cut Watch-nál látottakon túl megadhatjuk, hogy hány

qubit állapotát akarjuk figyelni, illetve azt, hogy ha tiszta állapot fordul elő a watchnál, akkor azt egységvektorként vagy sűrűségmátrixként írja-e ki. (Kikapcsolva minden állapotot sűrűségmátrix formájában ír ki.)



3-8. ábra: Subsystem Watch tulajdonságai párbeszédpanel

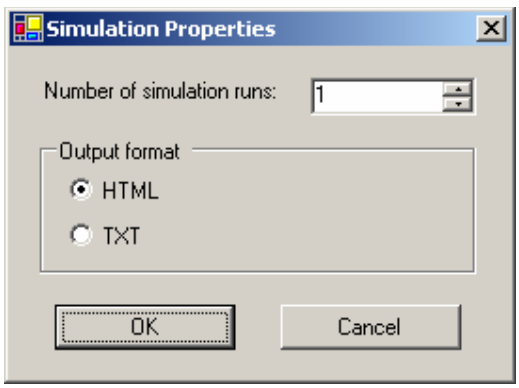
A Bloch Sphere watch (3-9. (f) ábra) esetében csak nevet lehet megadni. Ez egy 1×1-es, 1 qubites doboz, amivel a qubit állapotát ábrázolhatjuk a Bloch gömbön 3 dimenzióban (2-3. ábra).



3-9. ábra: Példák a programban használható áramköri elemekre. (a) Gate, (b) Measurement, (c) Input, (d) Channel, (e) Subsystem Watch, (f) Bloch Sphere, (g) Subcircuit, (h) Output, (i) Cut Watch

Az egérrel alaphelyzetben az elemeket mozgathatjuk drag-and-drop elven. Ahhoz, hogy az elemeket összeköthessük, kapcsoljunk át összekötés üzemmódra. Ezt az Edit menü Connect pontjával, vagy az eszköztár gombjával lehet. Kössük össze a kimeneti interfészeket a bemeneti interfészekkel. Ha az áramkör nem fér ki a képernyőre, akkor állíthatunk a nagyítás mértékén (View / Zoom). A Zoom to Fit menüponttal akkorára nagyíthatunk, hogy az áramkör éppen kiférjen a képernyőre. A rácsvonalak kirajzolását és az elemek neveinek kiírását ki- illetve bekapcsolhatjuk a View menü megfelelő menüpontjaival, vagy az eszköztár és gombjával. Alaphelyzetre visszatérni az ESC billentyűvel lehet.

3.1.2 Szimuláció



3-10. ábra: Szimuláció tulajdonságai párbeszédpanel

Ha az áramkört megépítettük, akkor leellenőrizhetjük, hogy helyes-e. Ezt a Simulation menü Circuit Check menüpontjával tehetjük meg. Ez olyan feltételeket ellenőriz, mint például: minden elem be van-e kötve, minden él előre mutat-e, az elemek a kvantummechanika törvényeinek megfelelnek-e, stb. Ha valami szerkesztési hiba van, akkor a program figyelmeztet, és a hibás elemet vagy elemeket egy piros E betűvel jelöli meg. Ha az áramkör helyes, akkor a Run Simulation menüponttal előhívhatjuk a szimuláció tulajdonságai párbeszédpanelt (3-10. ábra). (A szimuláció futtatása előtt a program mindig leellenőrzi, hogy az áramkör helyes-e.) Megadhatjuk, hogy hányszor fusson le egymás után a szimuláció, illetve az eredmény file formátumát. Ez utóbbi lehet HTML vagy TXT. A TXT esetében a szimuláció végeztével az eredményt egy szöveges fileba menti, és megnyitja a Jegyzettömbbel (Notepad). Ha HTML-t választottunk, az eredmény HTML filet az Internet Explorer-rel nyitja meg. Az eredmény fileban rögzítve lesz néhány általános adat, úgy mint a szimulálandó áramkör neve és elérési útvonala, a szimuláció futásainak száma, a szimuláció kezdetének és végének időpontja. Az eredmény fileba ezek után a mérések és a különféle watch-ok írják ki a szimuláció során összegyűjtött adataikat. A Bloch Sphere a TXT fileba csak a Bloch gömbi vektor koordinátáit írja ki, míg HTML esetében egy link is létrejön, amire kattintva megnézhetjük a vektort 3D-ben. A 3 dimenziós modellt a program X3D [22] formátumban generálja, amihez egy X3D player szükséges. A fejlesztés során az ingyenesen használható Flux Player-t [23] használtam.

3.2 A program felépítése

A program szerkezetének ismertetését két lépcsőben teszem meg. Először ismertetem azt a könyvtárat, ami a kvantummechanikai számításokat végzi (QCore), majd leírom, hogy a teljes program hogyan épül fel, és hogyan használja ezt a könyvtárat. A szimuláció algoritmusának leírására a következő fejezetet szentelem.

3.2.1 A QCore könyvtár

Ez a C++ könyvtár tartalmazza azokat az osztályokat, amik a kvantummechanikai számításokhoz kellenek. A program fejlesztésének kezdetén még Microsoft Visual C++ 6.0-t használtam, ennek a könyvtárnak nagyjából a felét ebben írtam, majd áttértem Microsoft Visual Studio .NET 2003-ra.

Először is szükségem volt egy „komplex számok” osztályra, ehhez a Standard C++ Library [24] `<complex>` template class-ját használtam, amit `double`-lel paramétereztem. Az internetről letöltöttem egy Matrix C++ template class-t [27], amiben az olyan szokásos mátrixműveletek vannak implementálva, mint például összeadás, szorzás, inverz, determináns, stb. Ezt a templatet paramétereztem a komplex osztállyal, és örököltettem belőle egy `HMatrix` osztályt, ami a Hilbert-teret definiálja. Néhány művelettel ki kellett egészíteni, úgy mint adjungálás, tenzor szorzás, unitérség vizsgálata, sűrűségmátrix három feltételének vizsgálata, Bloch gömb koordinátáinak kiszámítása, gyors hatványozás, stb. Definiáltam egy jó néhány gyakran használt unitér transzformációt, és állapotvektort. Ezekkel tulajdonképpen a harmadik kivételével az összes posztulátum által megkívánt számítást el lehet végezni. A mérés véletlen működésű, ezért kellett egy függvény, ami jó $[0,1)$ intervallumbeli valós véletlen számot ad. Ezután definiáltam a `Measurement` osztályt. Ez tartalmaz egy `HMatrix`-szal paraméterezett `<vector>` templatet, amiben a mérési operátorokat tárolom. A mérési operátorokat természetesen ellenőrizni is lehet. A mérés osztállyal lehetőség van az adott eredményhez tartozó valószínűség kiszámítására, illetve egy kvantumállapot megmérésére. Ez úgy működik, hogy sorsolunk egy $r \in [0,1)$ véletlen számot, a mérés eredménye az a k szám lesz, amelyre teljesül:

$$\sum_{i=0}^{k-1} p_i \le r < \sum_{i=0}^{k} p_i \,. \tag{3.1}$$

Itt p_i az i-edik eredményhez tartozó valószínűség. A megmért kvantumállapot természetesen megváltozhat. A `Measurement` osztály valósítja meg a kvantum csatornát is, erre a `ChannelTransform` nevű tagfüggvénye szolgál.

3.2.2 A QCircuit felépítése

A programot kezdetben Visual C++ 6.0-ban írtam, majd áttértem Microsoft Visual Studio .NET 2003-ra. A program az MFC Document/View [25] architektúrára épül. Ez megadja a felhasználói felület keretét, mint például a File, Edit, Wiev, Window, Help menüt. A program specifikus menüpontokat természetesen nekem kellett funkcionalitással felruházni. Multiple-Document Interface-t (MDI) használtam, ami azt jelenti, hogy a programban több dokumentum egyidejű megnyitására és szerkesztésére van lehetőség.

A program adatszerkezete a következőképpen néz ki. Az építőelemek a `CCirElement` absztrakt osztályból öröklődnek, ami a `CObject`-ből öröklődik, és az általános funkciókat definiálja, úgy mint a ki- és bemenő interfészek, ezek kirajzolása, piros E betű kirajzolása, ha hiba van, stb. Az osztály attribútumai közül néhány: elem egyedi azonosítója, neve, címkéje, helyzete, mérete, stb. Az áramköri elem be- és kimeneti interfészeit két tömbben tárolom. A tömb egy eleme megmondja, hogy az él másik vége melyik elem hányadik interfészéhez kapcsolódik. Ebből az osztályból örököltetem a `CGate`-et, ami a kvantum kapu osztálya. Ennek egy `HMatrix` attribútuma a kapuhoz tartozó unitér mátrix. A `CMeas` a mérés osztály, ami egy `Measurement` attribútummal hajtja végre a mérést. Hasonlóan van implementálva az input, output és subcircuit osztályok (`CInput`, `COutput`, `CSubcircuit`). A csatornát a `CMeas` osztály valósítja meg, hogy éppen mérés vagy csatorna funkcióban működik, azt egy `bool` változó tárolja. A Subsystem Watch-nak megfelelő `CSubsystemWatch`, és a Bloch Sphere-nek megfelelő `CBlochSphere` osztály megvalósítása az előbbiekhez hasonló, azonban Cut Watch megvalósítása más, mivel ez nem a szokásos építőelem. A `CWatch` osztály közvetlenül a `CObject`-ből öröklődik, és nyilvántartja, hogy hol metszi az áramkört. Az áramkör elemeit egy `CTypedPtrList<CObList, CCirElement*>` listában, a watchokat pedig egy `CTypedPtrList<CObList, CWatch*>` listában tárolja a document osztály.

A felhasználói interfészt a view osztály valósítja meg, ez kezeli az egér, a billentyűzet, a menük és az eszköztár gombjai által generált eseményeket. A adatok bevitelét legtöbbször párbeszédpaneleken tehetjük meg. Ehhez kezdetben a `CDialog` MFC osztályból örököltettem le a megfelelő osztályt. Az MFC elemkészlete később kevésnek és nehézkesen használhatónak tűnt, ezért a továbbiakban a

párbeszédpanelekhez a .NET Framework Class Library [26] `Form` osztályát használtam.

3.3 A szimuláció algoritmusának leírása

Ebben az alfejezetben a Simulation menü Run Simulation pontjának működését írom le. Miután az adatokat megadtuk a párbeszédpanelen, lefut ugyanaz az áramkör ellenőrző függvény, mint a Circuit Check menüpontnál. Ez a következő feltételeket ellenőrzi:

- Az áramkör nem lehet üres.
- Elemek nem lehetnek egymáson. Ez inkább csak szerkesztési konvenció, de nagyon hasznos.
- Nem lehetnek bekötetlen interfészek.
- Nem futhat él visszafelé. Ez garantálja azt is, hogy az áramkör gráfja körmentes (Directed Acyclic Graph, DAG) legyen, amit majd később a szimuláció során kihasználok.
- Csak egy output lehet. Ez a feltétel nem jelent korlátozást az áramkörépítésben, és talán most nem is tűnik ésszerűnek, a szimuláció során azonban ki fogom használni.
- Minden elemnek van egy `Check` metódusa, aminek igaz értékkel kell visszatérnie. Ez az elemre specifikus ellenőrzést hajtja végre, például kapu esetén megnézi, hogy a mátrix unitér-e, mérésnél ellenőrzi a teljességi feltételt, stb. Subcircuit esetében ez nagyon fontos, mert a `Check` metódusa rekurzívan hívja ezt az áramkör ellenőrzést a subcircuitban megadott filera. Ezzel bármilyen mélyen lévő hibára fény derül.
- Cut Watchok sem lehetnek egymáson.
- Minden Cut Watchnak annyi élet kell metszenie, mint az egyetlen output bemeneti interfészeinek száma. Ez azt jelenti, hogy az összes qubitnek megfelelő élet metszeni kell.

Ha az áramkör megfelelt az ellenőrzésnek, akkor az előfeldolgozása kezdődik. Az előfeldolgozás során átalakítom úgy, hogy a szimulációt el lehessen rajta végezni. Ehhez először is lemásolom, hogy az eredeti ne változzon. Az első lépés a Cut Watchok beszerkesztése az áramkörbe, a programban ugyanis eddig teljesen függetlenül tárolódtak. Ehhez meg kell határozni, hogy milyen éleket metsz a watch, és az is fontos, hogy milyen sorrendben, hiszen a rendszer állapotvektora ettől is függ. Az éleket el kell vágni, és be kell szúrni egy új áramköri elemet. Ezért a `CCirElement`-ből leörököltettem egy `CWatchVirt` osztályt, és ezt szúrom be úgy, hogy az interfészek sorrendje megfeleljen az elmetszett élek sorrendjének. A következő lépés a Subcircuitek kibontása. Meg kell nyitni a filet, és az elemeket hozzá kell venni az áramkörhöz, úgy hogy az interfészeket megfelelően össze kell kapcsolni. Mindezt rekurzívan kell tenni, hogy a bármilyen mélyen lévő subcircuitek is kibontásra kerüljenek. Az áramköri elemeket a létrehozásuk sorrendjében tárolja a lista. Ez a sorrend nem feltétlenül egyezik azzal, ahogy az elemeket sorra kell venni, és alkalmazni a nekik megfelelő transzformációt a rendszerállapoton. Ha egy A elem kimeneti interfésze össze van kötve egy B elem bemeneti interfészével, akkor az A elem transzformációját előbb kell használni, mint a B-t. Hogy egy ennek a feltételnek eleget tevő rendezést kapjak, az output elemből az éleken visszafelé haladva mélységi kereséssel (Depth First Search, DFS) bejárom a gráfot, és a visszatérési számoknak megfelelően sorrendezem a csúcsokat (elemeket). Itt használom ki, hogy csak egy output van, és mivel tudom, hogy a gráf DAG, ezért ez a számozás a gráf egy mélységi rendezését adja, ez a sorrend pedig a feltételnek megfelel. A következő lépésben meg kell határozni a rendszer kezdeti állapotát. Nehézséget okoz, hogy az outputoktól eltérően az inputokból több is lehet, és ezek elszórva is lehetnek. Ezért balról jobbra, fentről le, azaz az olvasás irányában végignézem az áramkört, és ebben a sorrendben összegyűjtöm az input elemeket. Ez azt jelenti, hogy az elemet törlöm a listából, a neki megfelelő kvantumállapotot pedig tenzor szorzom az eddigi állapottal. Így megvan a rendszer kezdeti állapota, és tudom azt is, hogy melyik elem transzformációját kell alkalmazni, csak azt nem tudom, hogy melyek azok a qubitek, amelyeken a transzformáció hat. Ezért tárolni kell egy listában, hogy a rendszer éppen aktuális állapotának milyen él sorrend felel meg.

Egy adott szimulációs futást mindig az előfeldolgozásban kapott kezdeti állapotból indítom. Sorra veszem az elemeket, és minden egyes elemnél a következőket teszem.

Minden elemnek van egy `Transform` metódusa, ami alkalmazza a neki megfelelő transzformációt a rendszerállapot – az elemtől függő számú – felső qubitjein. A nyilvántartott élsorrend azonban nem feltétlenül ilyen, ezért azt úgy kell alakítani, hogy tényleg a következő elemnek megfelelő élek legyenek felül a jó sorrendben. Természetesen az új élsorrendhez új rendszerállapot tartozik. Ezt közvetlenül nem tudom megtenni, két egymás melletti élet azonban könnyen felcserélhetek egy Swap kapuval. Ezért egy olyan rendezési algoritmust kellett választanom, ami csak egymás melletti éleket cserél fel. Az egyszerűsége miatt a buborék rendezést választottam. Természetesen csak olyan éllel hajtok végre cserét, ami a következő elembe befut. Minden egyes cseréhez a rendszerállapoton végrehajtom a megfelelő Swap kaput. Az így rendezett rendszerállapotot kapja meg a `Transform` metódus.

Az élek rendezése az előfeldolgozás utolsó, de legfontosabb és leghosszabb ideig tartó lépése. A szimulációs futások során már nem kell az élsorrendet nyilvántartani, csak végig kell menni annyiszor az áramkörön, amennyit megadtunk a szimulációs futások számának. Ha egy elemnél az élek rendezésre szorulnak, akkor a `Transform` metódusa előtt, – az előfeldolgozás során kiszámolt unitér kapuval – el kell végezni a rendszerállapoton az élrendező transzformációt.

A programban lehetőség van állapotvektorokat és sűrűségmátrixokat is használni, akár egyidejűleg is. Az állapotvektorral történő szimuláció jóval gyorsabb, mint sűrűségmátrixszal, például kapu esetén a mátrixot vektorral szorozni gyorsabb, mint mátrixot mátrixszal. A szimuláció során ezért az alapelv az, hogy amíg lehet, állapotvektorokkal számol, majd ha már nem, akkor átvált sűrűségmátrixra. Például, ha inputnak csak állapotvektort adtunk meg, és az áramkörben van valahol egy kvantum csatorna, akkor a csatornáig állapotvektorral számol a program, utána sűrűségmátrixszal. Ha azonban valamelyik inputnak sűrűségmátrixot adtunk meg, akkor végig sűrűségmátrixszal számol.

Érdemes néhány szót szólni az egyes elemek `Transform` metódusairól. A `CGate` metódusa megszorozza az unitér mátrixát a rendszerállapottal (2.5)-nek megfelelően, vagy a (2.35)-ben definiált transzformációt végzi. A `CMeas Transform` metódusa a (2.11)-ben vagy (2.37)- ben definiált módon megméri, de még mást is csinál. Minden mérés feljegyzi, hogy melyik eredmény hányszor jött ki. A szimuláció végeztével a kimeneti fileba ebből egy statisztikát készítek. Ehhez hasonlóan a három watch

feljegyzi, hogy milyen rendszerállapotok hányszor fordultak elő nála, és ezt szintén beleírom a kimeneti fileba. (A Bloch Sphere ezen túl még legenerálja a 3D-s modelleket.) Az utolsó futás mérési eredményeit átmásolom az eredeti áramkörbe, így a mérések dobozán az látható lesz.

Egy n qubites rendszer állapotvektorának dimenziója 2^n, ezért a szimulációs algoritmus futási ideje és tárigénye a qubitek számában exponenciális. Úgy gondolják, hogy tetszőleges kvantumrendszert nem is lehet polinom időben szimulálni, bár ez még nincs bebizonyítva.

3.4 Összehasonlítás más programokkal

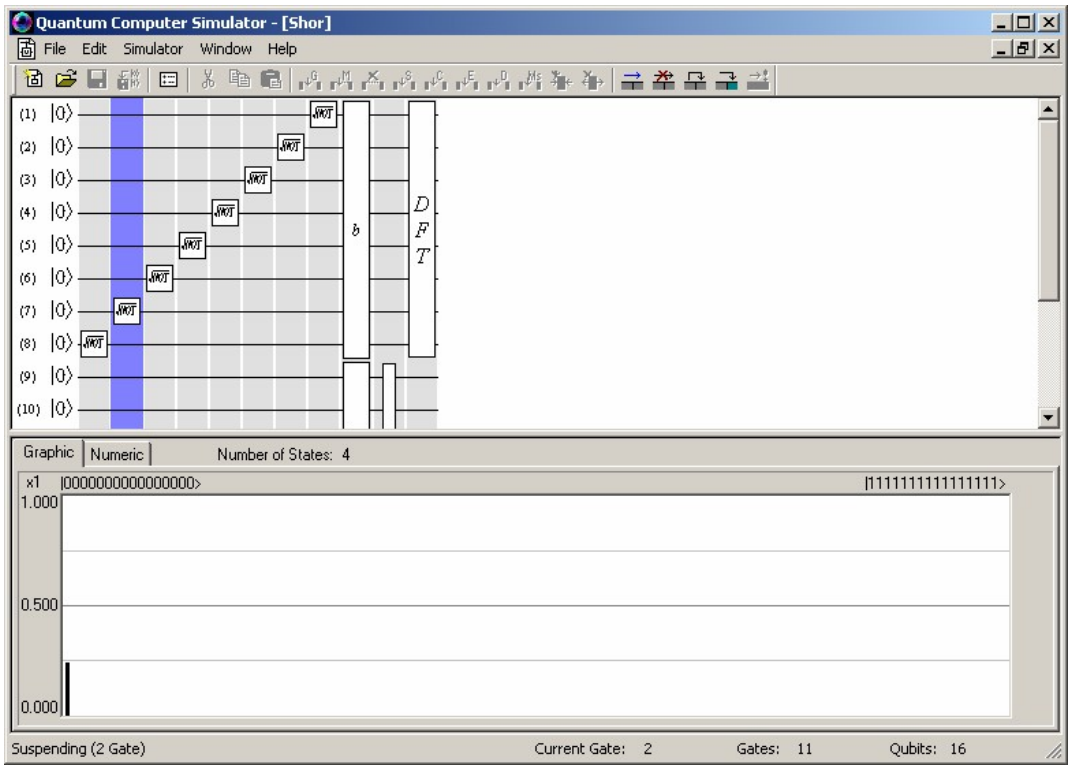
Az interneten sok kvantum számítógép szimulátor létezik. Vannak kvantum programnyelvek, amelyeket parancssoros program értelmez és futtat, vannak olyanok, amelyek fizikai implementációkat szimulálnak, vannak különböző programnyelveken írt könyvtárak, és számos kvantum áramkör szimulátor is létezik. Ebben a fejezetben ezekből válogatok, és hasonlítom össze a QCircuit programmal. Ennek célja, hogy bemutassa a QCircuit előnyeit, azon plusz funkciókat, amelyek motiváltak egy új áramkör szimulátor kifejlesztésére.

3.4.1 SENKO Quantum Computer Simulator

A Quantum Computer Simulator (QCS) a japán SENKO Corporation által készített kvantum áramkör szimulátor [28]. A program nem ingyenes, azonban egy demo verzió szabadon letölthető. A program kezelése a QCircuithez hasonlóan egyszerű. A kezelőfelület (3-11. ábra) két részre van osztva, felül láthatjuk a szerkesztett áramkört, alul a szimuláció eredményét. Az áramkör fölötti eszköztárral a program által nyújtott minden művelet elvégezhető.

Hátrányok

A program legnagyobb hátránya a QCircuittel szemben – azon túl hogy nem ingyenes – az, hogy nem tud sűrűségmátrixokkal számolni, és így kvantum csatornákat sem lehet használni. Az áramkörben tetszőleges kapukat is el lehet helyezni, de a mátrix megadása nehézkesebb, mint a QCircuit esetében. Itt ki kell választani a mátrix elemének indexeit, és úgy lehet megadni egy komplex számot, szemben a QCircuitben



3-11. ábra: Pillanatkép a SENKO Quantum Computer Simulator programból

alkalmazott táblázatos megoldással, ahol egyben lehet látni az egész mátrixot. (Ezen úgy próbáltak segíteni, hogy a QCS képes a Mathematica programból mátrixot importálni.) Másik nagy hátrány, hogy itt kezdeti állapotnak csak $|0\rangle$-t vagy $|1\rangle$-et lehet megadni, illetve mérésnek csak $\mathcal{M}_+^n$-t. Az áramkör szerkesztése is nehézkesebb, mert itt a vízszintes vonalakra kell pakolni az elemeket, és általunk megadott helyeken lehet a vezetékeket keresztbe kötni. Itt is lehetséges a Subcircuit-hez hasonlóan több elemet egy építőelemként kezelni az úgynevezett „Composite Gate” segítségével, véleményem szerint azonban a Subcircuit kényelmesebben használható, mert például lehetőség ad arra, hogy különböző fileokban lévő áramkörökbe ugyanazt a Subcircuitet illesszük be. További hátrány a Bloch gömbös megjelenítés, illetve a rész rendszer állapot megjelenítésének hiánya. A QCircuit további előnye a szimulációs eredmény külön fileba történő megjelenítése, ami lehetőséget ad arra, hogy elmentsük, és később a QCircuit nélkül is elemezhessük.

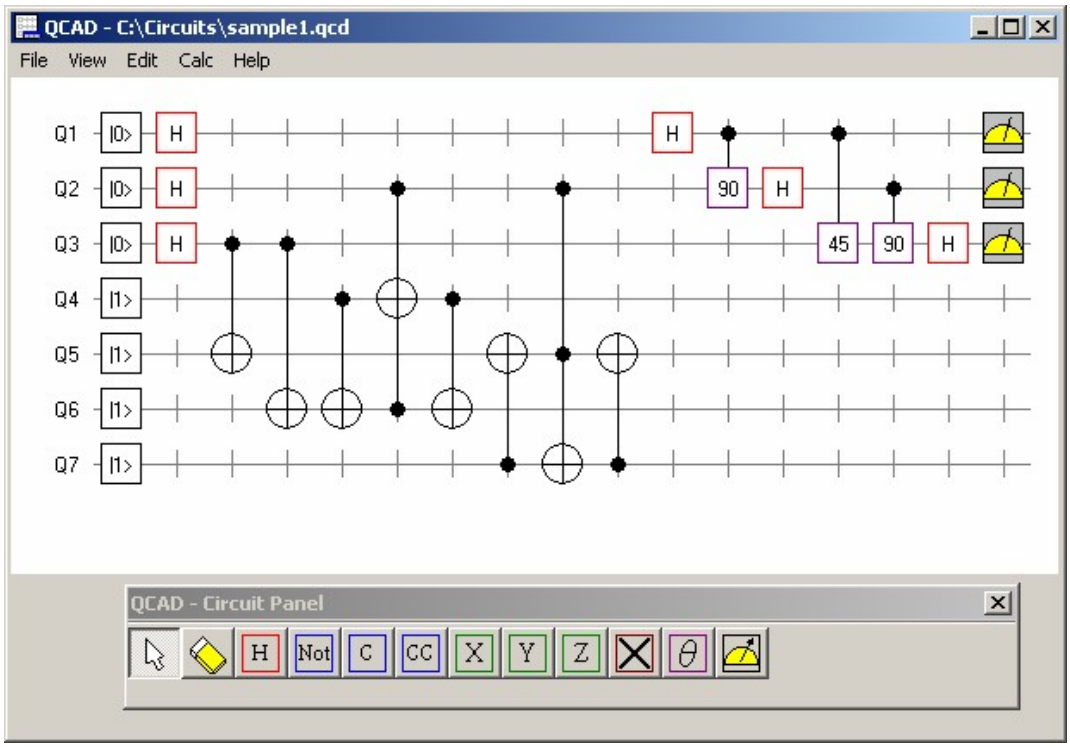
Előnyök

A legfőbb előnye a QCS-nek a QCircuittel szemben a gyorsabb és több qubitet kezelni tudó szimuláció. Bár kérdéses, hogy ha sűrűségmátrixok használatát is lehetővé tenné,

akkor is ilyen gyors lenne-e. Szintén előnyként lehet értékelni, hogy a szimulációt lehet kapunként léptetni, és minden lépés után a teljes rendszerállapotot figyelni. A rendszerállapotot a szokásos „numerikus” megjelenítés mellett lehetséges „grafikusan” is megnézni, ahol is az állapotvektor valószínűségi amplitúdóinak abszolút érték négyzeteit ábrázolja a program oszlopdiagram szerűen.

3.4.2 QCAD

A QCAD-ot [29] a tokiói egyetemen fejlesztették ki. A program ingyenes, azonban a QCS-nél és a QCircuitnél is jóval kevesebb funkciót tartalmaz. A program futása során két ablakot látunk (3-12. ábra), az egyikben az áramkör van, a másikban egy eszköztár, amiből kiválaszthatjuk, hogy milyen kaput akarunk lehelyezni. Sajnos csak az itt felsorolt kapukat használhatjuk, tetszőlegeset nem adhatunk meg. Lehetőség van mérést is elhelyezni az áramkörben, azonban a szimuláció során a program ezt figyelmen kívül hagyja. Szimuláció során csak a végső rendszerállapotot nézhetjük meg, ehhez a program színes oszlopdiagramot is rajzol, aminek hasznossága számomra kérdéses. Mindezeket összevetve megállapíthatom, hogy a QCAD nem konkurenciája sem a QCircuitnek, sem a QCS-nek.



3-12. ábra: Pillanatkép a QCAD programból

3.4.3 Quantum Qudit Simulator

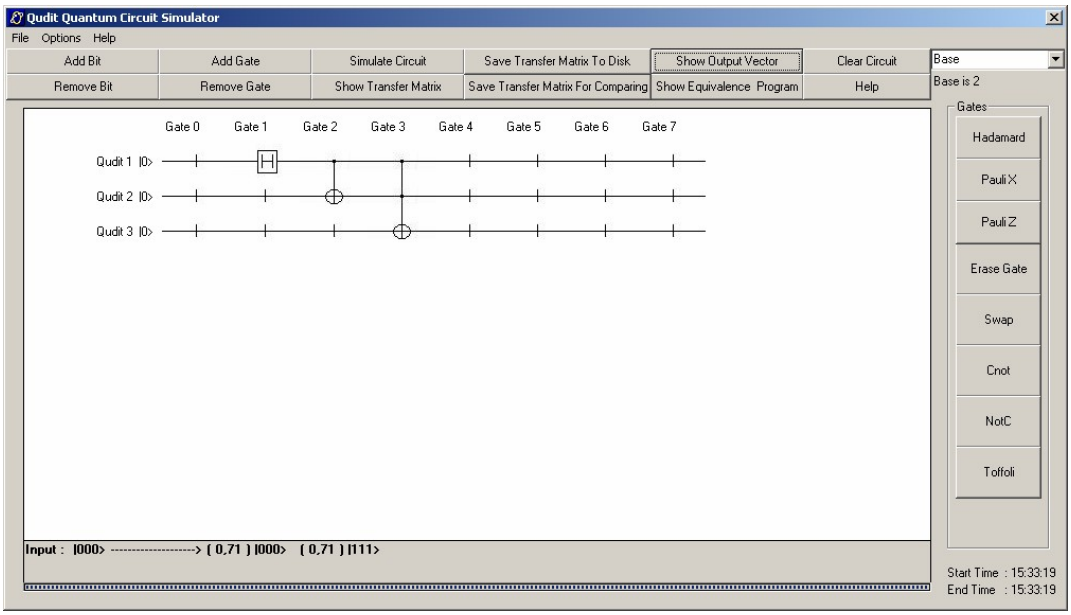
A Quantum Qudit Simulator-t [30] egy ír egyetemista fejlesztette ki szakdolgozata keretében. A program a forráskóddal együtt ingyenesen letölthető.

Hátrányok

A program kezelőfelülete (3-13. ábra) a QCAD-hoz hasonló. Tetszőleges kaput megadni sajnos itt sem lehet, és az áramkörbe kapukon kívül más elemet nem lehet elhelyezni. Ez a program is csak a rendszer végső állapotát képes megmutatni, ráadásul a szimuláció lassú, és a program is elég megbízhatatlanul működik. További hátrány, hogy az áramkört nem lehet elmenteni.

Előnyök

A programban lehetőség van a két bázisállapotú qubitek helyett több bázisállapotú „quditek” használatára. Azaz egy vezetéknek megfelelő Hilbert tér nem csak $\mathbb{C}^2$, hanem $\mathbb{C}^i$, $i = 2,3,\ldots,10$ is lehet. A program képes még kiszámolni az áramkörnek megfelelő unitér mátrixot – mivel csak kapukat lehet elhelyezni.



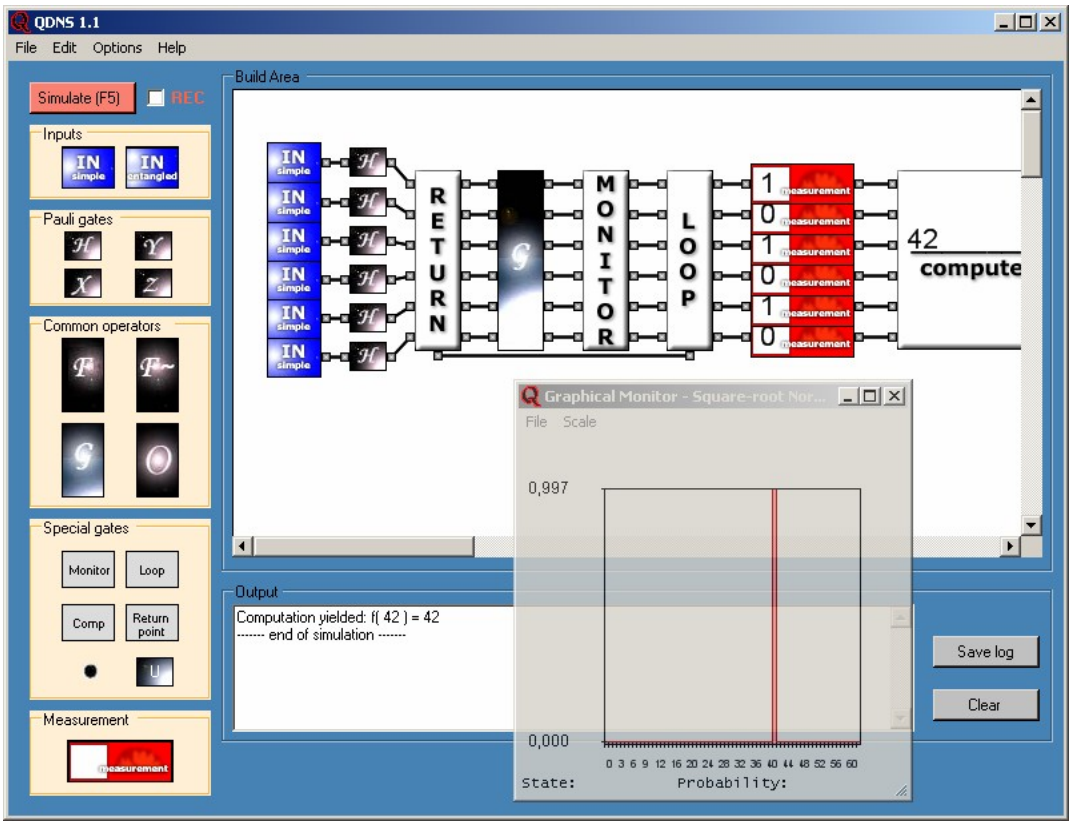
3-13. ábra: Pillanatkép a Quantum Qudit Simulator programból

3.4.4 Quantum Designer and Network Simulator

A Quantum Designer and Network Simulator (QDNS) programot (3-14. ábra) Imre Sándor, Abronits Péter és Darabos Dániel fejlesztették ki a BME Híradástechnikai Tanszékén [31].

Hátrányok

Ez a program sem képes sűrűségmátrixokkal számolni, és így kvantum csatornát sem lehet használni. A beépített kapukon túl csak maximum 3 qubites tetszőleges kaput lehet megadni. Hasonlóan, inputból is csak egy qubites tetszőlegeset lehet megadni, illetve EPR párt is definiálhatunk. Sajnos a program a kvantummechanikai törvények által meghatározott feltételek teljesülését nem mindig ellenőrzi, így megadhatunk például nem invertálható kaput, vagy tetszőlegesen hosszú állapotvektort. Mérésből itt is csak $\mathcal{M}_+^n$-t lehet megadni.



3-14. ábra: Pillanatkép a Quantum Designer and Network Simulator programból

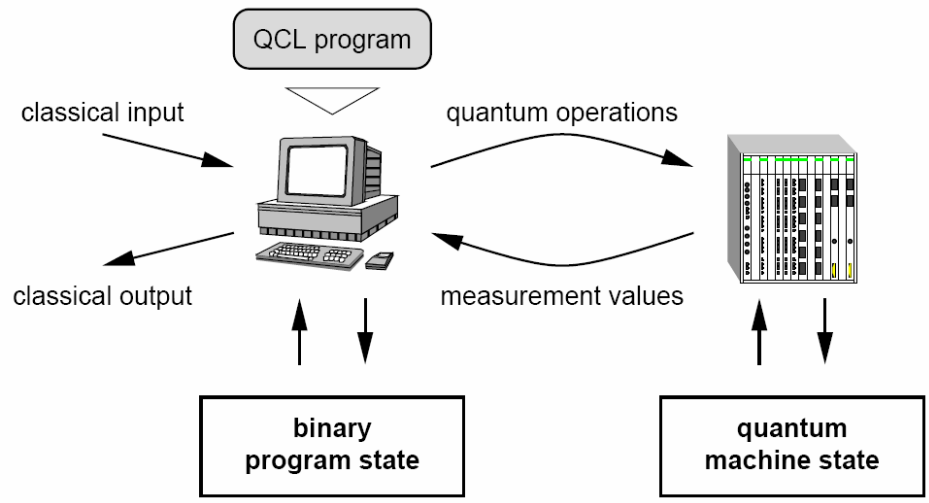
Előnyök

Ezzel a programmal is lehetséges a rendszerállapot oszlopdiagramos megjelenítése, amiből több féle is van. A Grover algoritmusához használandó „Oracle” kaput definiálhatjuk adatbázissal vagy C# kóddal, bár a kapu unitér mátrixát nem nézhetjük meg. Lehetőség van az áramkörben ciklus létrehozására is a „Loop” és „Return” elemekkel. Az áramkört lehetséges képként is kimenteni, a szimuláció eredményéről pedig Microsoft Word formátumú jelentést is készíthetünk.

3.4.5 QCL

A QCL [32] nem áramkör szimulátor, hanem egy „kvantum programnyelv”, azonban hasznos összehasonlítani a kvantum algoritmusok áramkör illetve programnyelv megadási módját. A QCL-t Bernhard Ömer fejleszti. A kvantum áramkör és kvantum programnyelv viszonya analóg a logikai áramkör és „klasszikus” programnyelv viszonyához. Míg az előző a fizikai törvényei adta kapukból állítja össze az algoritmust, az utóbbi a jövőbeli kvantum számítógépek vezérlő nyelve lehet.

A QCL a 3-15. ábrán látható felépítésű kvantum számítógép architektúrát feltételezi [33]. A QCL-ben megírt program nem a kvantum számítógépen fut, hanem egy klasszikus számítógépen, ami egy kvantum számítógépet vezérel. Ezt a felépítést úgy is leírhatjuk, mint egy klasszikus számítógép kiegészítve egy kvantum orákulummal. A felhasználó szemszögéből a rendszer úgy viselkedik, mint egy klasszikus



3-15. ábra: A QCL által feltételezett kvantum számítógép architektúra [33]

számítógép – klasszikus bemenete és klasszikus kimenete van. Ameddig ilyen kvantumszámítógépet nem tudnak készíteni, a QCL-ben megírt programunkat futtathatjuk az internetről letölthető parancssoros interpreterrel, amely képes a kvantum számítógép működésének szimulálására is.

Az eddigiekből következik, hogy a QCL-ben megvannak a „klasszikus" programnyelvekből ismert adattípusok (int, real, boolean, string, stb.) és vezérlési szerkezetek (függvényhívás, ciklus, feltételes elágazás, stb.). A QCL továbbá ki van egészítve a kvantum számítógépet vezérlő parancsokkal, amelyekkel unitér kapukat hajthatunk végre, mérést végezhetünk, törölhetjük a kvantum regiszterek állapotát, stb. Egy QCL-ben megírt kvantum algoritmus általános felépítése a következő:

```
{
     reset;
     myoperator(q);
     measure q,m;
} until ok(m);
```

A kvantum regisztert töröljük, elvégezzük a különböző unitér operátorokat, majd a regiszter állapotát megmérjük. Ha a mérés a rossz eredményt adja akkor az eljárást előről kezdjük. (Természetesen feltételezzük, hogy a kvantum algoritmus kimenete klasszikus algoritmussal hatékonyan ellenőrizhető.)

Hátrányok

A QCL legfőbb hátránya abból fakad, hogy programnyelv, tehát a szintaktikáját meg kell tanulni mielőtt használjuk, míg mondjuk a QCircuitnél a szakirodalomban elterjedt elnevezések és jelölések alapján egyszerűen meg lehet építeni egy áramkört. Ráadásul a QCL esetében bizonyos dolgok elnevezése teljesen egyedi, például a Hadamard transzformációt a „Mix" operátorral adhatjuk meg. Habár a QCL-ben van lehetőség kvantum regiszter állapotát lekérdezni, programnyelv lévén Bloch gömbös vagy más grafikus megjelenítés nem lehetséges. A QCL azt feltételezi, hogy a kvantum regiszterek tökéletesen el vannak szigetelve a környezettől, azaz az állapotuk tiszta. Ezért a QCL-ben sem lehet sűrűségmátrixokat vagy kvantum csatornákat használni.

Előnyök

A QCL legfőbb előnyei – programnyelv révén – a különböző vezérlési szerkezetek és adattípusok használatának lehetősége. Egy algoritmust valószínűleg kompaktabb és jobban strukturált formában adhatunk meg, mint egy áramkör esetében. További előny lehet, hogy a QCL-ben megírt program a jövőbeli kvantum számítógépeken talán módosítás nélkül is futtatható lesz. (Feltéve, hogy az architektúrája megegyezik a fentebb leírttal.)

3.4.6 Összegzés

Az eddigieket összefoglalva a QCircuit főbb előnyei a többi szimulátorral szemben, hogy állapotvektorokat és sűrűségmátrixokat is kezelni tud, az építőelemek minden típusából (kapu, input, mérés, csatorna, stb.) van lehetőség tetszőlegeset definiálni, illetve a kvantum csatornák használatának lehetősége.

A következő két fejezetben példákon megmutatom, hogy a csatornák milyen hasznosak lehetnek, segítségükkel milyen új dolgok szimulációjára nyílik lehetőség.

4 Kvantum algoritmusok szimulációja

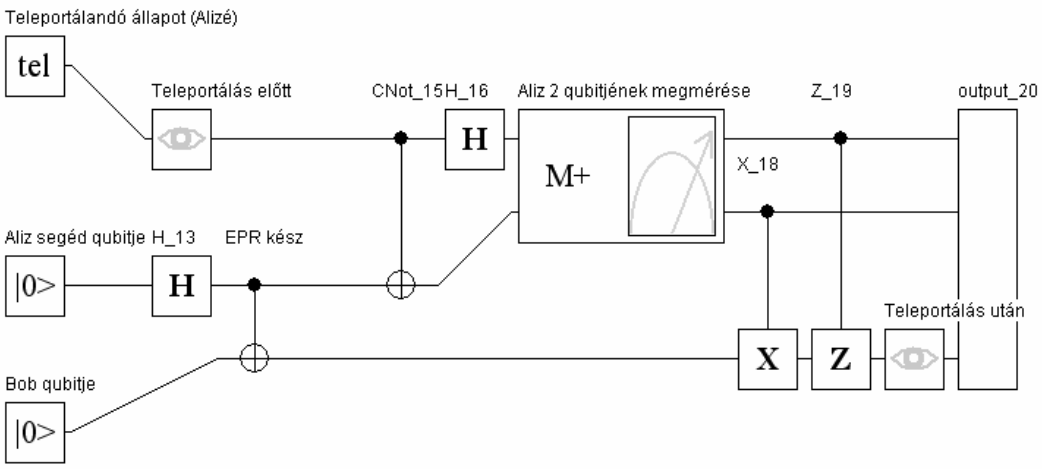
Ebben a fejezetben ismert kvantum algoritmusok szimulációját mutatom be. Ezzel egyrészt megmutatom, hogy a QCircuit program milyen sokrétűen használható, másrészt helyenként példákat adok, hogy a program nyújtotta lehetőségekkel hogyan lehet az algoritmusokat továbbfejleszteni vagy nem triviális tulajdonságait megérteni.

4.1 Teleportálás

A posztulátumokból bebizonyítható, hogy tetszőleges kvantumállapotot nem lehet lemásolni („No Cloning Theorem”, [3] 162. old.), azaz nincs olyan transzformáció, ami a következőt teszi:

$$\forall|\varphi\rangle:\ |\varphi\rangle|0\rangle\rightarrow|\varphi\rangle|\varphi\rangle \tag{4.1}$$

Azt azonban semmi nem tiltja, hogy Aliznál lévő qubitek állapotát úgy másoljuk át Bob qubitjeire, hogy a másolás után az Aliznál lévő qubitek állapota megváltozzon. (A másolás helyett tulajdonképpen áthelyezést végezzünk.) Ezt hívjuk teleportálásnak. ([2] 11. old.)



4-1. ábra: Egy qubit teleportálásának áramköri rajza. A „tel” állapotnak bármit megadhatunk.

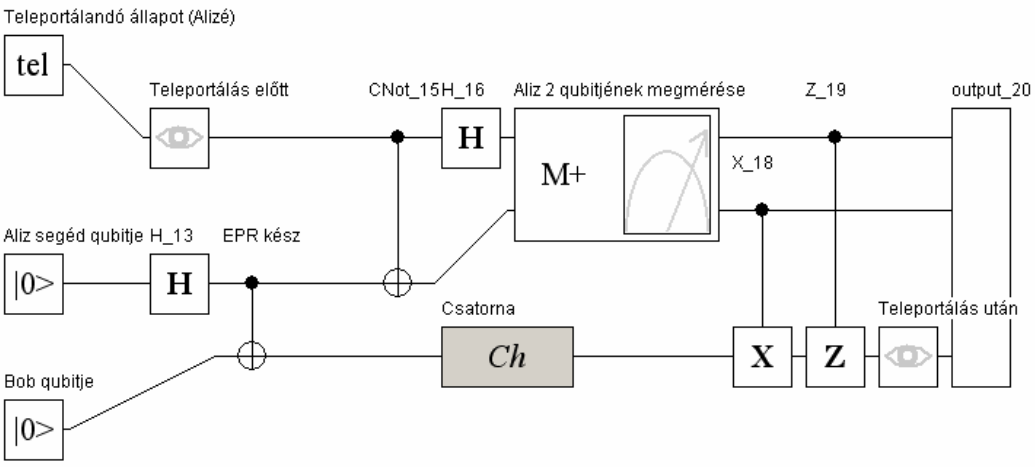
A 4-1. ábrán látható az egy qubites teleportáló áramköre. Az „EPR kész” kapu után előáll a (2.29)-ben említett EPR állapot. Ennek egyik qubitjét Aliz, másikat Bob kapja meg. Aliz hozzákapcsolja a teleportálandó qubitjét – ami bármilyen tiszta vagy kevert

állapotban lehet – az EPR párban lévő qubitjéhez, majd a két qubitjét megméri az $\mathcal{M}_+^2$ méréssel. A mérés eredményétől függően Bob elvégzi a szükséges transzformációkat (X_18 és Z_19), hogy előálljon nála a teleportálandó állapot. A mérés eredményéből tudni lehet, hogy Aliz két qubitje milyen állapotba került, így Aliznak elég csak a mérés eredményét tudatnia Bobbal, ami két klasszikus bit kommunikációja. Egy qubit „végtelen sok" lehetséges állapota tehát átvihető két bit kommunikációval! Ehhez persze az kell, hogy előzetesen szétosszunk egy EPR párt, amit erőforrásként használunk. A szimulációt sokszor lefuttatva láthatjuk, hogy a mérés kimenetelétől függetlenül a „Teleportálás előtt" állapot mindig meg fog egyezni a „Teleportálás után" állapottal. Ez természetesen analitikusan is bebizonyítható, illetve fizikailag is sikerült már megvalósítani.

4.1.1 Teleportálás csatornával

A valóságban Aliz és Bob egymástól valamilyen fizikai távolságban van. Ezért az EPR pár szétosztásakor qubitet (qubiteket) viszünk át egyik helyről a másikra. Mivel a kvantum információ nagyon törékeny, ha valósághű szimulációt akarunk, azt kell feltételezni, hogy az EPR párt zajos kvantum csatornán visszük át. (A két klasszikus bit hibamentes átvitele nem gond.) Ekkor azt várjuk, hogy a teleportálás utáni állapot nem fog teljesen megegyezni a teleportálandó állapottal.

A 4-2. ábrán látható áramkörben azt feltételeztem, hogy Aliz hozza létre az EPR párt, és az egyik qubitjét küldi át Bobnak egy kvantum csatornán. A szimuláció



4-2. ábra: Teleportálás kvantum csatornával

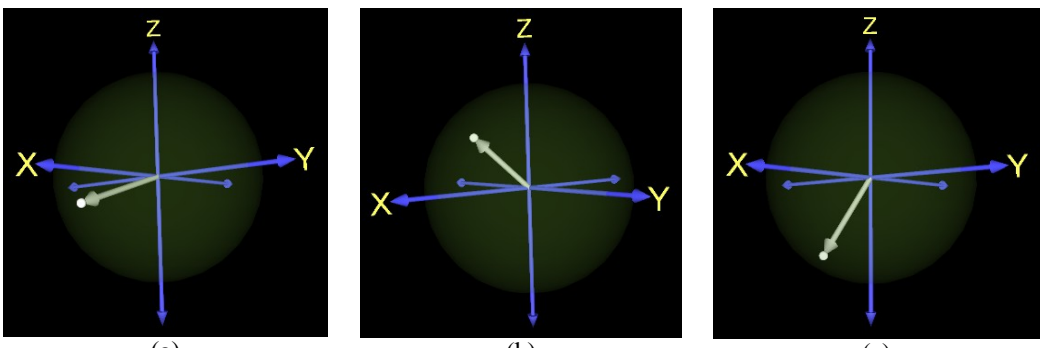
eredményének fontos következményei lehetnek. Ha ugyanis azt tapasztalnánk, hogy a teleportálás utáni állapot „jobban hasonlít” a teleportálandó állapotra, mintha a teleportálandó állapotot a kvantum csatornán küldtük volna át, akkor így a csökkenthetnénk a csatorna zaját, anélkül, hogy a qubitekben redundanciát használnánk. A „depolarizing” és a „phase-damping” csatorna esetében biztosan nem ez a helyzet, mert a csatornán átküldött és a teleportált állapot megegyezik. Az „amplitude-damping” csatornánál azonban érdekes helyzet alakul ki. Az esetek felében a két állapot megegyezik, a másik felében (ha a mérés során az X_18 vezérlő qubitjére $|1\rangle$ jön ki) azonban a teleportált állapot egész más lesz, mint a csatornán átküldött. Például a $0.6|0\rangle + 0.8|1\rangle$ állapotot átküldve a $p = 0.6$ paraméterű „amplitude-damping” csatornán a következő állapotot kapjuk:

$$\rho_{csat} \approx \begin{bmatrix} 0.7440 & 0.3036 \\ 0.3036 & 0.2560 \end{bmatrix}. \tag{4.2}$$

Ellenben ha ugyanezt az állapotot teleportáljuk ugyanezzel a csatornával, akkor az esetek felében a következőt kapjuk:

$$\rho_{tel} \approx \begin{bmatrix} 0.1440 & 0.3036 \\ 0.3036 & 0.8560 \end{bmatrix}. \tag{4.3}$$

A 4-3. ábrán látható az eredeti és ez a két állapot a Bloch gömbön ábrázolva.

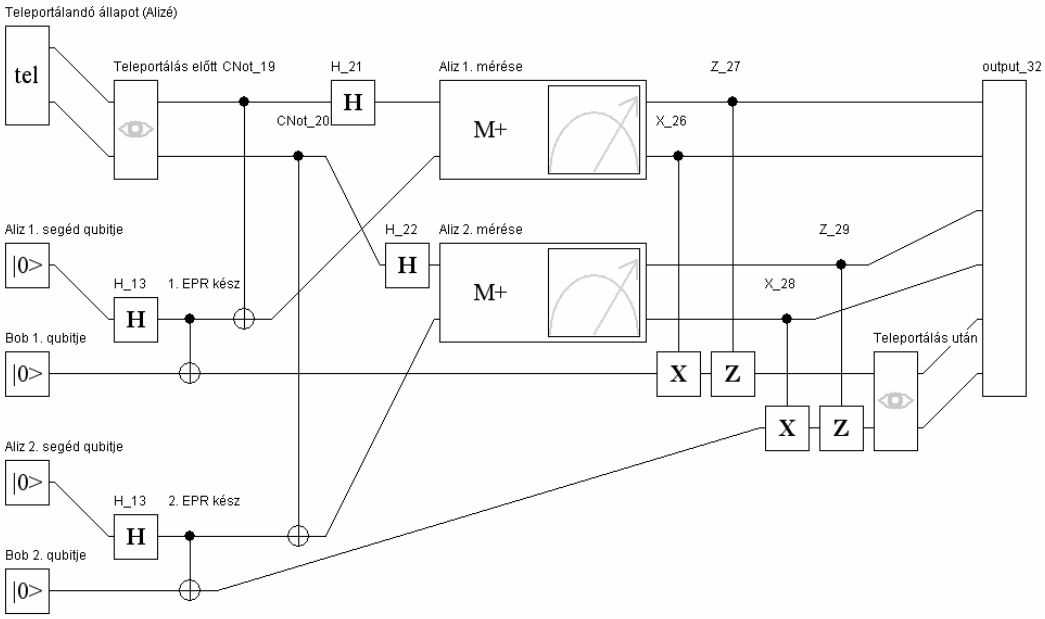


(a) (b) (c)

4-3. ábra: (a) $0.6|0\rangle + 0.8|1\rangle$, (b) ρ_{csat}, (c) ρ_{tel} ábrázolása a Bloch gömbön

4.1.2 Két qubit teleportálása

Természetesen felvetődik a kérdés, hogyan lehetne több qubites állapotot teleportálni. Az első gondolat, hogy teleportáljuk a qubiteket külön-külön, szerencsére működni fog. Ez nem triviális, hiszen a qubitek összefonódva is lehetnek, de szerencsére be lehet bizonyítani. A bizonyítás helyett szimuláljuk a 4-4. ábrán látható áramkört. A mérésektől függően 16 féleképpen alakulhat a rendszer végső állapota, de az alsó két qubit állapota („Teleportálás után") mindig a teleportálandó állapot lesz. A szimuláció ideje jelentősen megnő, mivel a szimulációs algoritmus futásideje a qubitek számában exponenciális.



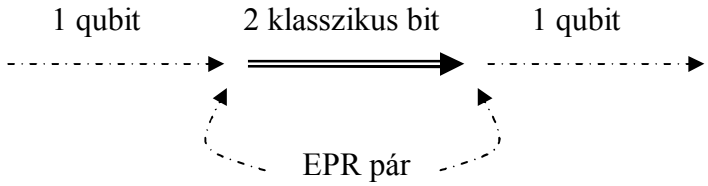
4-4. ábra: Két qubites állapot teleportálása

4.2 „Super-dense Coding"

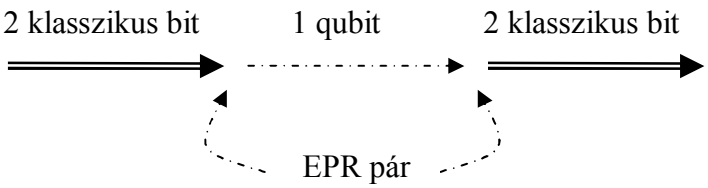
A „Super-dense Coding" ([2] 15. old.) a teleportálás fordítottja (4-5. ábra). Azaz egy előzetesen szétosztott EPR pár segítségével lehetőség van két bitnyi információ átvitelére egyetlen qubit átküldésével. A 4-6. ábrán látható az eljárás áramköre.

Az „EPR kész" kapu után létrejövő EPR pár felső qubitjét Aliz, az alsót Bob kapja meg. Aliz a bitjeitől függően elvégzi a megfelelő X és Z transzformációt a qubitjén, ezután azt átküldi Bobnak, aki a transzformációk után megméri a két qubitet. A mérés eredménye minden esetben meg fog egyezni az Aliz által átküldeni kívánt két bitnyi

Teleportálás:

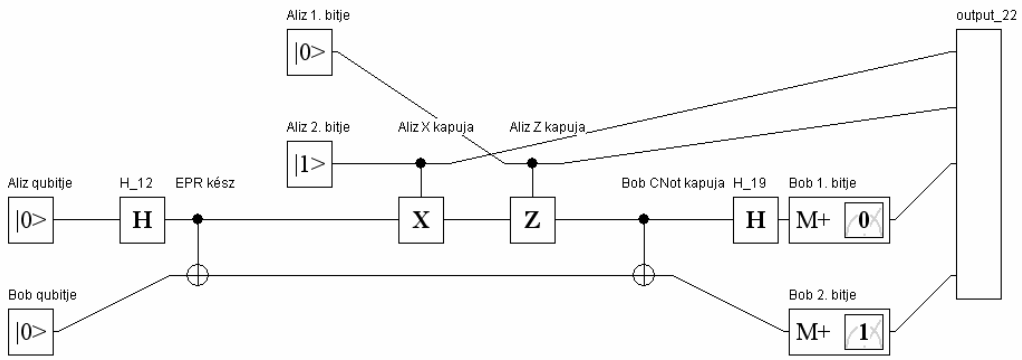


„Super-dense Coding”:



4-5. ábra: A teleportálás és a „Super-dense Coding” egymás fordítottja

információval. Ezt ellenőrizhetjük, ha Aliz 1. és 2. bitjét kicseréljük a $|0\rangle$ vagy $|1\rangle$ bázisállapototok valamelyikére. (Itt is valósághűbb azt feltételezni, hogy az EPR párt, és Aliz qubitjét kvantum csatornán visszük át. Ekkor a klasszikus bitekben valamilyen valószínűséggel hiba lép fel, ami javítható klasszikus hibavédő kódolással.)



4-6. ábra: „Super-dense Coding” áramköri rajza

4.3 Deutsch-Jozsa algoritmus

Vegyük a következő problémát: Az $f(x):\{0,1\}^n \to \{0,1\}$ függvényről a következőket tudjuk. Vagy konstans, azaz minden argumentumra az értéke konstans, vagy balansz,

azaz az argumentumok felére az értéke 1, a másik felére 0. A feladat eldönteni, hogy melyik igaz.

$$\forall x : f(x) = c, \quad c \in \{0,1\} \text{ vagy} \tag{4.4}$$

$$\left|\{x : f(x) = 0\}\right| = \left|\{x : f(x) = 1\}\right| \tag{4.5}$$

Ha klasszikusan akarjuk megoldani, akkor legalább a lehetséges argumentumok felénél egyel többre ki kell értékelni a függvényt, hogy biztosak legyünk a válaszban. Ez $2^{n-1}+1$ kiértékelést jelent, azaz a legjobb determinisztikus klasszikus algoritmus is exponenciális futási idejű n-ben.

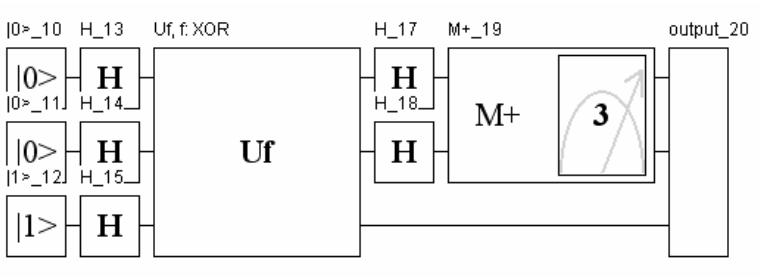
A Deutsch-Jozsa algoritmus [8] ezt egyetlen függvénykiértékeléssel oldja meg, ami összességében polinom idejű algoritmust ad! Ez a következőképpen működik (4-7. ábra). A függvényt kiszámoló kapu egy bázisállapotot a következő bázisállapotba visz:

$$\mathbf{U}_f : |\mathbf{x}\rangle|y\rangle \to |\mathbf{x}\rangle|y \oplus f(\mathbf{x})\rangle, \quad \mathbf{x} \in \{0,1\}^n, y \in \{0,1\} \tag{4.6}$$

Az így kapott $\mathbf{U}_f$ kapu minden f-re unitér. Az algoritmus a $|0\rangle^{\otimes n} \otimes |1\rangle$ állapotból indul. ($^{\otimes n}$ az önmagával vett n-szeres tenzor szorzást jelenti.) Erre alkalmazzuk a következő transzformációt: $\left(\mathbf{H}^{\otimes n} \otimes \mathbf{I}\right)\mathbf{U}_f\left(\mathbf{H}^{\otimes n} \otimes \mathbf{H}\right)$. A felső n qubitet az $\mathcal{M}_+^n$ méréssel megmérve egy valószínűséggel nullát kapunk, ha a függvény konstans. Ha a függvény balansz, a 0 eredmény valószínűsége 0. A mérés eredményéből tehát egyértelműen megválaszolható a kérdés.

Például vegyük az $f(x_1, x_2) = x_1 \oplus x_2$ balansz függvényt. A függvény kapuja ekkor:

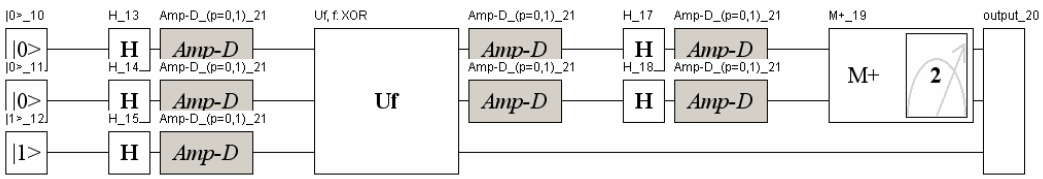
$$\mathbf{U}_f = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.7}$$



4-7. ábra: A Deutsch-Jozsa algoritmus mérése az XOR balansz függvény esetén soha nem ad nulla eredményt.

A mérés ekkor mindig a 3 eredményt adja, a függvény tehát tényleg balansz. Ha az $\mathbf{U}_f$ kaput lecseréljük a három qubites identitás mátrixra ($\mathbf{I}^{\otimes 3}$), akkor az az azonosan nulla konstans függvény ($f(x_1, x_2) \equiv 0$) mátrixa lesz. A mérés ebben az esetben mindig nullát ad.

A következő példában tegyük fel, hogy az algoritmust olyan kvantumszámítógéppel valósítjuk meg, ahol a qubiteket atomok alap- és gerjesztett állapota tárol. Egy atom alapállapota jelentse a $|0\rangle$, gerjesztett állapota pedig az $|1\rangle$ bázisállapotot. A gerjesztett állapot nem stabil, adott egységnyi idő alatt p valószínűséggel elbomlik az alapállapotra. Mint arról már volt szó, ezt a fajta bomlást az „amplitude-damping” csatornával lehet modellezni. Ha feltesszük, hogy a Hadamard és az $\mathbf{U}_f$ kapu végrehajtása egységnyi időt vesz igénybe, akkor a fizikai rendszerünket úgy szimulálhatjuk, hogy a kapuk után felvesszünk p paraméterű „amplitude-damping” csatornákat (4-8. ábra). A 4-8. ábrán $p = 0.1$. Ekkor a mérés az esetek 4.4%-ában 0 eredményt ad, azaz a várakozásoknak megfelelően néha rossz eredményt kapunk.



4-8. ábra: A Deutsch-Jozsa algoritmus áramköre, kiegészítve a fizikai implementáció során fellépő hibákat modellező kvantum csatornákkal.

4.4 Csatorna BER mérés

A következőkben arra mutatok példát, hogyan lehet egy kvantum csatorna klasszikus bithiba arányát (Bit Error Rate, BER) megmérni (4-9. ábra). A mérés során ½-½ valószínűséggel sorsolok 0-t vagy 1-et, és ezt a következőképp kódolom:

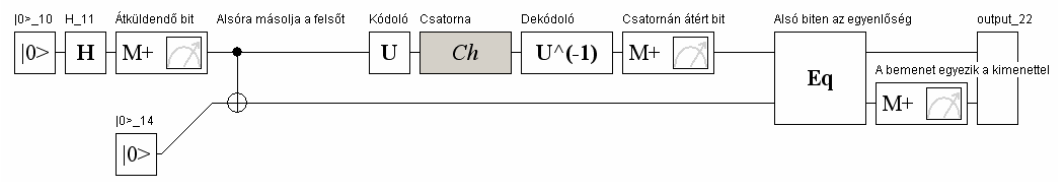
$$\begin{aligned} 0 &\to |e_0\rangle, \\ 1 &\to |e_1\rangle. \end{aligned} \tag{4.8}$$

$|e_0\rangle$, $|e_1\rangle$ a kvantum csatornán átküldött qubit két lehetséges állapota. Ahhoz, hogy ez a két állapot méréssel egy valószínűséggel megkülönböztethető legyen szükséges, hogy egymásra merőlegesek legyenek, ezért ezt a továbbiakban fel fogom tenni. Egy ilyen kódolást egy $\mathbf{U}$ unitér transzformációval lehet végrehajtani. A csatornán átért qubitet az $\mathbf{U}$ inverzével (az ábrán **U^(-1)**) dekódolom, majd az $\mathcal{M}_+^1$ méréssel megmérem, és az átküldendő bitet és a mérés eredményét összehasonlítom az **Eq** kapuval. Az **Eq** kapu alsó bitje 1, ha a bemenő két bit megegyezik:

$$\mathbf{Eq} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.9}$$

A szimulációt sokszor futtatva megkapjuk a „nem egyezések” arányát („A bemenet egyezik a kimenettel” mérés 0 eredményének előfordulási arányát), azaz a bithiba arányt.

Ha például p paraméterű „depolarizing” csatornát vizsgálunk, azt tapasztaljuk, hogy a bithiba arány mindig $\frac{p}{2}$ lesz, függetlenül $\mathbf{U}$-tól. Ez megegyezik azzal, amit elvárunk,



4-9. ábra: Csatorna bithiba arányának mérése

hiszen a csatorna hatása az, hogy a Bloch gömbön ábrázolt vektor megrövidül, az $\mathbf{U}$ transzformációé pedig a Bloch gömbön valamilyen forgatás. Ha tehát elforgatjuk a vektort, megrövidítjük, majd visszaforgatjuk, akkor azt kapjuk, mintha csak megrövidítettük volna. Általánosságban azonban nem mindegy, milyen bázisban küldjük a biteket. Mint arról már volt szó, a „phase-damping" csatorna esetében a $\{|0\rangle, |1\rangle\}$ bázisban (ha $\mathbf{U} = \mathbf{I}$) bithiba nem fordul elő, p értékétől függetlenül. Ha azonban a $\left\{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right\}$ bázisban küldjük a biteket – azaz ha $\mathbf{U} = \mathbf{H}$ –, akkor azt kapjuk, hogy a bithiba arány $\frac{p}{2}$ lesz.

4.5 Kvantum hibavédő kódolás

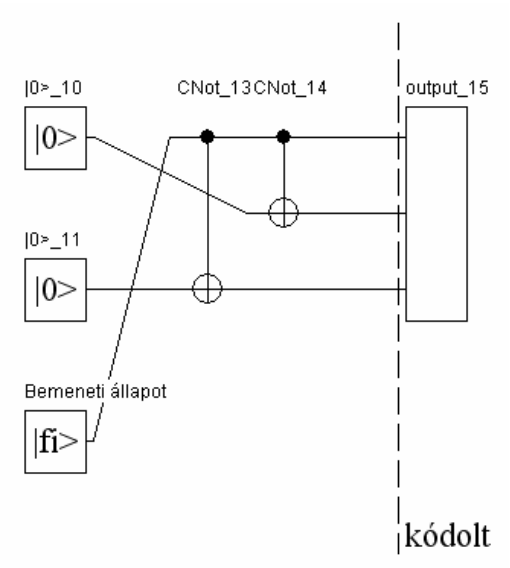
Peter Shor 1995-ben bebizonyította, hogy a kvantumállapotok hibavédő kódolása lehetséges [13]. Ez azért nagyon fontos, mert a gyakorlatban a kvantumállapotok nagyon törékenyek, a környezettel való összefonódás eredményeképpen a bennük tárolt információ hamar elveszik. Az itt bemutatott kódkonstrukciók elvi hátteréről Mark Oskin ([2] 47. old.) és John Preskill ([3] 26. old.) munkáiban olvashatunk.

4.5.1 3 qubites bit flip kód

Ez a kód a klasszikus ismétléses kód analógiájára épül. A klasszikus esetben, ha a csatornán 0-át vagy 1-et akarunk átküldeni, akkor 000-át ill. 111-et küldünk át. Ha egy bitnyi „bit flip" hiba van (azaz 0-ból 1, 1-ből 0 lesz), akkor az többségi döntéses elven javítható. Ezzel a módszerrel egy adott csatorna bithiba aránya tetszőlegesen kicsivé tehető, azonban a kódszó jelentősen megnő. Természetesen nem ez a leghatékonyabb módszer a hibavédelemre, viszont nagyon egyszerű.

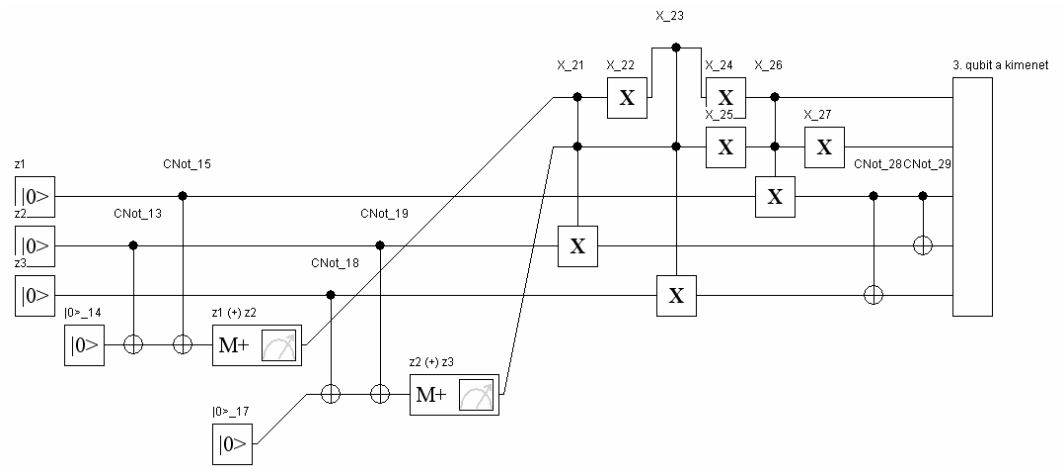
A kvantum esetben egy qubit állapotát 3 qubit állapotában tároljuk, azaz a kódoló (4-10. ábra) a következő transzformációt végzi:

$$|\varphi\rangle = a|0\rangle + b|1\rangle \rightarrow a|000\rangle + b|111\rangle \tag{4.10}$$



4-10. ábra: A 3 qubites kódoló áramköre

A dekódolás a 4-11. ábrán látható módon zajlik. A csatornán átért 3 qubitből (z_1, z_2, z_3) 2 segéd qubiten előállítom a $z_1 \oplus z_2$ és a $z_2 \oplus z_3$ állapotot, és ezeket megmérem. Ezzel a „folytonos bit flip" hiba „diszkrét lesz". (A „folytonos bit flip" hiba egy olyan unitér transzformáció, ami a Bloch gömb x tengelye körüli forgatásnak felel meg.) A két mérési eredményből meg lehet állapítani, hogy melyik qubiten történt a hiba. (A kód csak egy hibát tud javítani.) Például ha $z_1 \oplus z_2$-t megmérve 1-et, $z_2 \oplus z_3$-at megmérve 0-t kapok, akkor az első qubiten volt a hiba. A hibás qubiten végrehajtok egy **X** transzformációt. Ezután a kódoló inverzével a 3 qubit állapotát egyetlen qubiten előállítom. Ez a kódkonstrukció minden 1 qubitnyi



4-11. ábra: A 3 qubites dekódoló áramköre

„bit flip”, azaz amplitúdóban bekövetkezett hibát képes javítani. De amíg ez elegendő a klasszikus esetben, nem elegendő a kvantum esetben, mivel egy qubitnek fázisa is van! A dekódoló kimenetén egy tetszőleges 1 qubites hiba bekövetkezése és javítása után az alábbi két állapot lehetséges:

$$a|0\rangle + b|1\rangle \text{ vagy } a|0\rangle - b|1\rangle \tag{4.11}$$

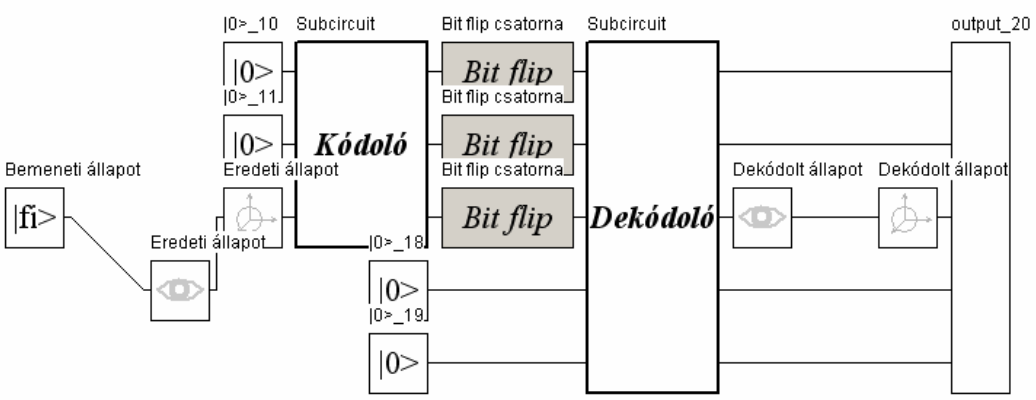
Ez a kód tehát egy 1 qubites hibát egy fázishiba erejéig tud javítani. Ezt majd a 9 qubites kód segítségével lehet kiküszöbölni.

Most azonban egy konkrét csatornán tesztelem ezt a kódolást (4-12. ábra). A kódoló és a dekódoló áramkörét subcircuitként töltöttem be. Mind a három qubitet ugyanazon a „bit flip” csatornán küldöm át. A „bit flip” csatorna az $\mathbf{X}$ transzformációt hajtja végre p valószínűséggel, $1-p$ valószínűséggel a qubit nem változik. A csatorna Kraus reprezentációja:

$$\mathbf{M}_0 = \sqrt{1-p}\mathbf{I},\ \mathbf{M}_1 = \sqrt{p}\mathbf{X}. \tag{4.12}$$

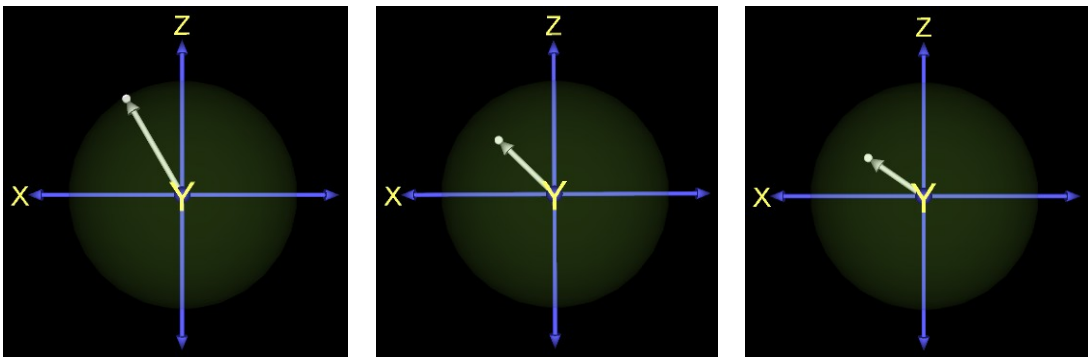
Ebben a példában $p = 0.3$. A dekódoló után a középső qubiten fog megjelenni a dekódolt állapot, a felső két qubit ($z_1 \oplus z_2$ és $z_2 \oplus z_3$) pedig jelzi, hogy volt-e javítás, és ha igen, akkor melyik qubiten.

A szimulációhoz először kapcsoljuk át a dekódolóban a két mérést kvantum csatornába. A dekódolás során ugyanis nem érdekel minket, hogy konkrétan melyik biten volt javítás, csak a dekódolás kimenete. Az átkapcsolás után a szimuláció a



4-12. ábra: A 3 qubites bit flip kód tesztelése „bit flip” csatornával

lehetséges rendszerállapotok „valószínűségi eloszlásával” számol. Végeredményként a „Dekódolt állapot” Subsystem watch egyetlen sűrűségmátrixot ad, amit összehasonlíthatunk a bemeneti állapottal. Például legyen a bemeneti állapot a $0.966|0\rangle + 0.259|1\rangle$. (Azért ez, mert a Bloch gömbön a vektora az x-z síkban van, és így könnyebb ábrázolni.) A szimulációt lefuttatva megkapjuk az eredeti (4-13. (a) ábra) és a dekódolt állapot (4-13. (b). ábra) Bloch gömbön történő ábrázolását is. E mellé célszerű megrajzoltatni azt az állapotot, ami akkor jön létre, ha $0.966|0\rangle + 0.259|1\rangle$-t egyszerűen átküldjük a „bit flip” csatornán (4-13. (c). ábra). Jól látható, hogy a dekódolt állapot „jobban hasonlít” az eredeti állapotra, mintha csak simán a csatornán küldtük volna át, azaz a csatorna zaját tényleg sikerült javítani. A jobban hasonlít alatt azt értem, hogy közelebb van a Bloch gömbön. Az is látható, hogy a dekódolt állapot a másik kettőt összekötő szakaszon van.



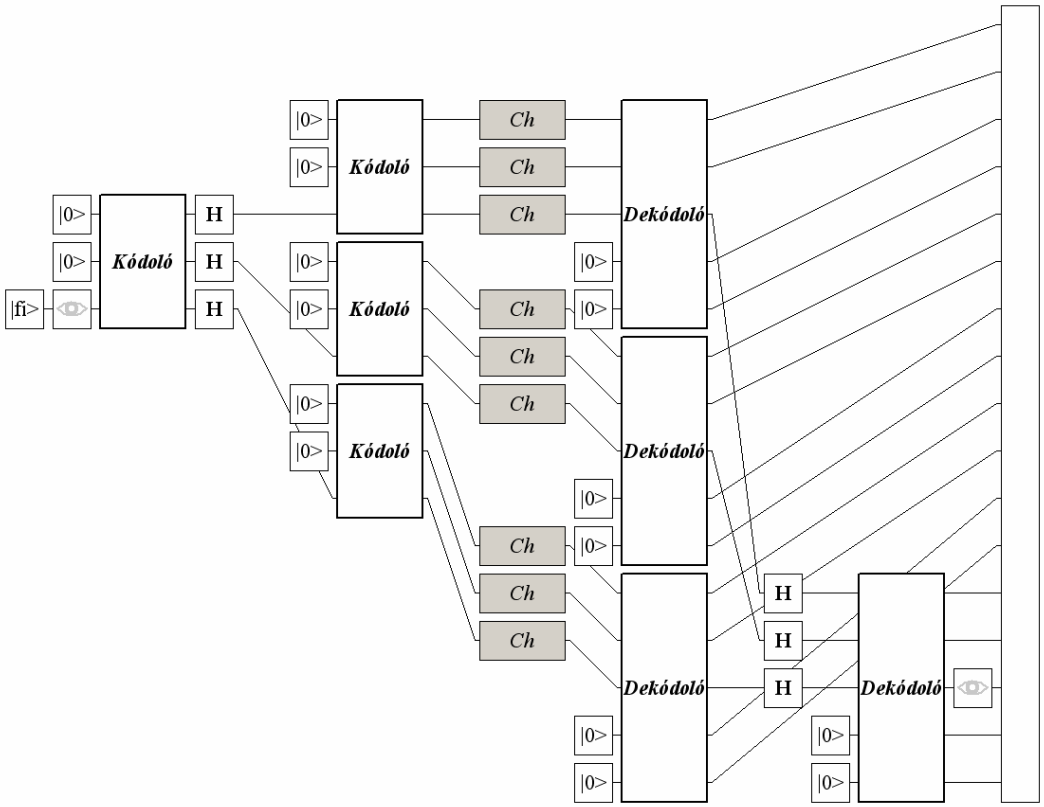
(a) (b) (c)

4-13. ábra: (a) Az eredeti állapot ($0.966|0\rangle + 0.259|1\rangle$), (b) a dekódolt állapot, (c) a csatornán átért állapot ábrázolása a Bloch gömbön

4.5.2 9 qubites kód

Úgy kell kiegészíteni a 3 qubites bit flip kódot, hogy az egy amplitúdóban bekövetkezett hiba mellett egy fázishibát is képes legyen javítani. Ehhez kihasználjuk azt a tényt, hogy $\mathbf{HZH} = \mathbf{X}$, azaz a fázishiba Hadamard kapukkal amplitúdó hibává alakítható és fordítva. Ha a 3 qubites kódoló kimenetére és a dekódoló bemenetére Hadamard kapukat teszünk, akkor a kód egy fázishiba kijavítására képes. A 9 qubites kód hierarchikusan épül fel 3 darab 3 qubites kódból, amiket egy olyan 3 qubites kód fog össze, amely ki van egészítve Hadamard kapukkal (4-14. ábra). A 3 db 3 qubites kód kijavítja az amplitúdó hibát, és ha fázis hiba is volt, akkor a Hadamard kapuknak

köszönhetően átalakul amplitúdó hibává, amit szintén kijavítunk. Kihasználva azt a tételt, mely szerint minden unitér transzformáció előáll a σ_i-k ($i = 0,1,2,3$) lineáris kombinációjaként, adódik, hogy ez a kód bármely 1 qubites hibát képes javítani. (Ugyanis σ_1 a „bit flip” hiba, σ_3 a „phase flip” hiba, $\sigma_2 = (-i)\sigma_3\sigma_1$ pedig mindkettő.) Habár a kód közel sem optimális, nagyon egyszerű. Sajnos az egész rendszer 17 qubites, ami már meghaladta a számítógépem teljesítményét.



4-14. ábra: A 9 qubites kód

5 Kvantum kulcsszétosztás vizsgálata szimulációval

Ebben a fejezetben a QCircuit egy konkrét, gyakorlatban is használható alkalmazását mutatom be. Mint arról már volt szó, Peter Shor 1994-es eredményének köszönhetően a jövőbeli kvantumszámítógépek képesek lesznek az RSA feltörésére. Az RSA-t és a többi nyilvános kulcsú kriptográfiai algoritmust ma főleg kriptográfiai kulcsszétosztásra használják. A kulcsszétosztás célja egy véletlen bitsorozat létrehozására a két kommunikáló fél (Aliz és Bob) között úgy, hogy ahhoz illetéktelen személy (Éva) ne férhessen hozzá. A kulcs ezek után használható biztonságos kommunikációhoz például a „One-Time-Pad” (vagy más néven „Vernam rejtjelező”) segítségével. Lehetséges tehát, hogy a jövőben az RSA-t leváltó új módszer után kell nézni. A kvantum kulcsszétosztás (quantum key distribution, QKD) egy ígéretes jelölt, mivel biztonsága bizonyított. A biztonság azon alapul, hogy egy kém, miközben információt próbál szerezni a kulcsról, elkerülhetetlenül megnöveli a kulcs bithiba arányát (Bit Error Rate, BER), így jelenléte leleplezhető. A BER könnyedén meghatározható néhány véletlenül választott bit egyeztetésével. A kulcs bithiba arányára azonban jelentős befolyással van a kvantum csatorna, amin a kommunikáció zajlik. Kísérletek igazolták, hogy a kvantum kulcsszétosztás megvalósítható üvegszálon [15] és levegőben [16] is, sőt működő példányok már kereskedelemben is kaphatók [17], [18].

Érdemes tehát feltenni a kérdést, hogy mennyire biztonságos egy adott csatornával egy adott protokoll, illetve egy adott csatornához milyen protokollt célszerű választani. E fejezet célja ennek megválaszolása.

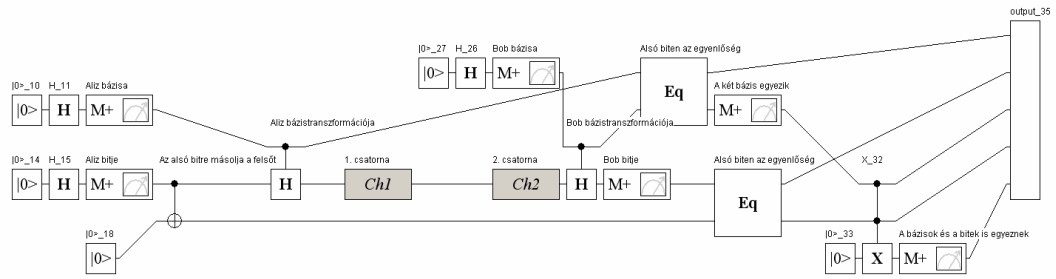
5.1 BB84 QKD protokoll

A BB84 az első kvantumfizikai elven működő kriptográfiai protokoll. Az ötlet Charles Bennettől és Gilles Brassardtól származik, melyet 1984-ben publikáltak [10].

5.1.1 Működési elv

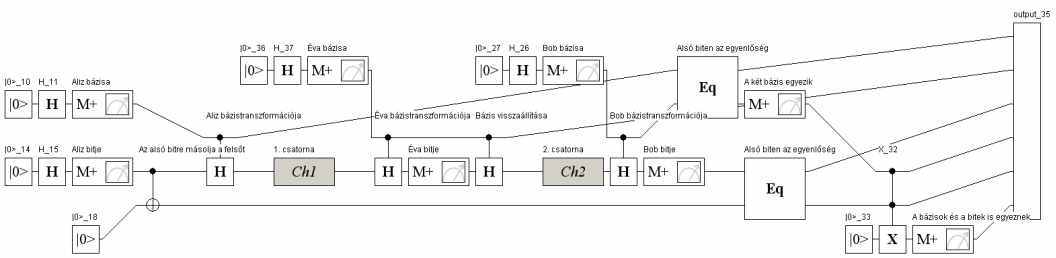
A protokoll működése röviden a következő. Aliz egyenletes eloszlással sorsol két véletlen bitsorozatot, az egyik a „bitek”, a másik a „bázisok”. Ha a sorsolt bitje 0,

akkor a $|0\rangle$ állapotot, ha 1, akkor az $|1\rangle$ állapotot veszi. Ha a bithez sorsolt bázisa 1, akkor végrehajt a qubiten egy Hadamard transzformációt. Az így kapott qubitet átküldi Bobnak a kvantum csatornán, aki a sorsolt bázisától függően végrehajtja a saját Hadamard transzformációját, és a kapott bitet megméri az $\mathcal{M}_+^1$ méréssel. Miután így átvitték a biteket, egy publikus csatornán egyeztetik, hogy melyik biteknél milyen bázisokat használtak. Azokat a biteket, amelyeknél különböző bázisokat használtak eldobják. Ha a csatorna zajmentes és nem hallgatta le senki, akkor az így kapott bitsorozatnak (a „nyers" kulcsnak) mindkét félnél azonosnak kell lennie. Azaz Aliz és Bob kulcsa közötti bithiba arány 0. Az 5-1. ábrán látható a protokoll áramköre. Az **Eq** kapu a (4.9)-es egyenletben megadott.



5-1. ábra: A BB84 protokoll lehallgató nélkül

Ha a csatornát Éva lehallgatta, akkor mivel nem ismeri Aliz bázisait, a legjobb amit tehet, hogy Bobhoz hasonlóan véletlenszerűen választott bázisokkal megméri a qubitet, és amit mér, azt továbbküldi. Az esetek felében azonban Éva más bázist választ, mint Aliz és Bob. Ekkor megváltoztatja a qubit állapotát, ami azt eredményezi, hogy Bob ½ valószínűséggel hibásan méri. Ekkor a bitek ¼ része hibás lesz, azaz a kulcsok közötti bithiba arány 0.25-re növekszik. A két BER közötti különbség tehát $0.25-0=0.25$. Ésszerű ezt a mennyiséget tekinteni a biztonság



5-2. ábra: A BB84 protokoll lehallgatóval

mértékeként, jelöljük ezt r_{sec}-kel ($0 \le r_{\text{sec}} \le 0.25$). Ha $r_{\text{sec}} = 0$, akkor a protokoll egyáltalán nem biztonságos. Aliz és Bob a bithiba arányt könnyedén meg tudja határozni néhány véletlenül választott bit egyeztetésével. A maradék bitek alkotják a kulcsot. Az 5-2. ábra mutatja a protokoll áramkörét lehallgató jelenlétében.

Az áramkörökben *Ch1* az Aliz és Éva, *Ch2* pedig az Éva és Bob közötti kvantum csatorna. Az érdekes eset az ha *Ch1* és/vagy *Ch2* nem zajmentes. Ilyenkor azt tapasztaljuk, hogy – a lehallgató nélküli és a lehallgatós esetben is – a bithiba arány nőni fog, a kettő különbsége azonban csökkeni. Azaz ilyenkor a protokoll kevésbé biztonságos, mint a zajmentes esetben ($r_{\text{sec}} < 0.25$).

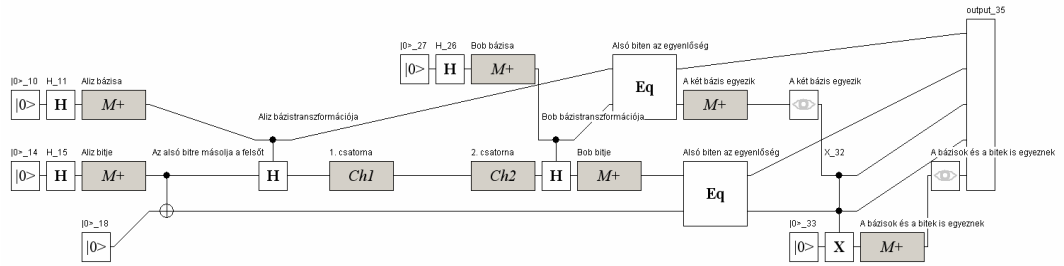
5.1.2 Szimuláció

Egy konkrét eset biztonságának számszerű kiértékeléséhez először meg kell határozni a csatorna matematikai modelljét. Ez leellenőrizhető, ha meghatározzuk szimulációval bizonyos tulajdonságait, mint például a 4.4 fejezetben. A csatornát beillesztve a két áramkörbe lefuttatható a szimuláció, és r_{sec} értéke meghatározható. A korrekt szimulációhoz a csatornát két szakaszban kell megadni – egy lehetséges kém előtti és utáni szakaszban. Talán megalapozatlannak tűnik, hogy a kémet egy konkrét helyre kell tenni, de valós helyzetekben gondolkodva a csatorna zaja olyan okokból adódik, mint a küldő és a fogadó berendezés tökéletlen fizikai megvalósítása, az átviteli közeg hatása, stb. Az utóbbit talán nehéz két részre osztani, de a küldő berendezés tökéletlenségét egyértelműen *Ch1*-gyel, míg a fogadóét *Ch2*-vel kell modellezni.

Miután a szimulációt sokszor lefuttattuk, r_{sec} értéke meghatározható a mérések eredményeiből a következőképpen. Az szimuláció eredményeit mutató fileból „A két bázis egyezik” mérés 1-es eredményének előfordulási arányát kiolvashatjuk. Jelöljük ezt r_{bases}-zel a lehallgató nélküli esetben, és $r_{\text{Eve,bases}}$-zel a lehallgatós esetben. Hasonlóan jelöljük „A bázisok és a bitek is egyeznek” mérés 1-es eredményének előfordulási arányát r_{bits}-szel és $r_{\text{Eve,bits}}$-szel. A protokoll biztonsága a következőképp kapható:

$$r_{\text{sec}} = \frac{r_{\text{bits}}}{r_{\text{bases}}} - \frac{r_{\text{Eve,bits}}}{r_{\text{Eve,bases}}}. \tag{5.1}$$

Lehetőség van a protokoll biztonságát analitikusan is kiszámolni, ha kihasználjuk a csatorna és mérés közötti analógiát. (Lásd (2.46) utáni bekezdés.) Ha tehát minden mérést átkapcsolunk kvantum csatornába, akkor a szimuláció az összes lehetséges rendszerállapot valószínűségi eloszlásával számol. Hogy eredményt kapjunk, fel kell venni két „Subsystem watch" áramköri elemet „A két bázis egyezik" és „A bázisok és a bitek is egyeznek" nevű – mostmár – csatornák után, mint ahogyan az 5-3. ábrán látható. A „Subsystem watch" elemek a megfelelő csatornák után kapták a nevüket.



5-3. ábra: A BB84 protokoll analízishez átalakított áramköre

Miután a szimulációt egyszer lefuttattuk, két sűrűségmátrixot kapunk eredményül. Jelöljük a „A két bázis egyezik"-et ρ_{bases}-zel illetve $\rho_{\text{Eve,bases}}$-zel; a „A bázisok és a bitek is egyeznek"-et ρ_{bits}-szel illetve $\rho_{\text{Eve,bits}}$-szel, hasonlóan, mint fentebb. r_{sec} pontos értéke:

$$r_{\text{sec}} = \frac{\langle 1|\rho_{\text{bits}}|1\rangle}{\langle 1|\rho_{\text{bases}}|1\rangle} - \frac{\langle 1|\rho_{\text{Eve,bits}}|1\rangle}{\langle 1|\rho_{\text{Eve,bases}}|1\rangle}. \tag{5.2}$$

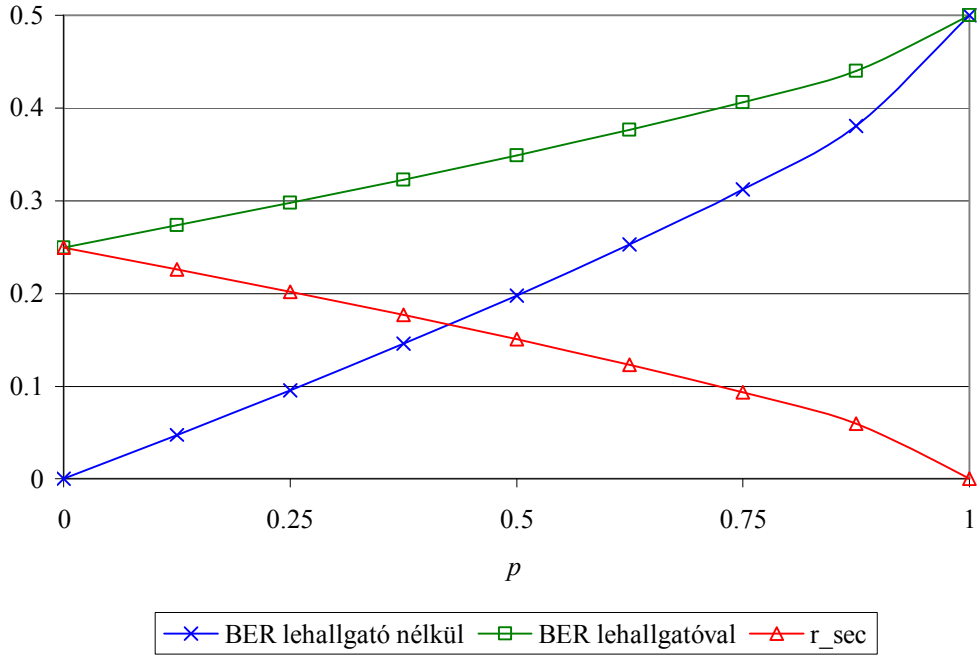
($\langle 1|\rho|1\rangle$ a ρ 2. sorának 2. eleme.) Észrevehetjük, hogy $\langle 1|\rho_{\text{bases}}|1\rangle$ és $\langle 1|\rho_{\text{Eve,bases}}|1\rangle$ mindig 0.5-tel egyenlő, mivel Aliz és Bob a bázisokat egymástól függetlenül sorsolja. Tehát (5.2) leegyszerűsíthető:

$$r_{\text{sec}} = 2\langle 1|\rho_{\text{bits}}|1\rangle - 2\langle 1|\rho_{\text{Eve,bits}}|1\rangle. \tag{5.3}$$

A következőkben minden eredményt ezzel az analitikus módszerrel kaptam, mert ez pontos és kevesebb időt igényel.

A következő példában *Ch1* zajmentes, míg *Ch2* a (45)-ben definiált „amplitude-damping" csatorna. Az 5-4. ábra mutatja, hogyan változnak a bithiba arányok és r_{sec},

ha a p csatornaparaméter a zajmentes 0-ról a teljesen zajos 1-re nő. Látható, hogy r_{sec} csökken, ahogy a csatorna egyre zajosabb lesz.



5-4. ábra: A diagramon látható hogyan változnak a BB84 protokoll bithiba arányai és r_{sec} az „amplitude-damping” csatorna p csatornaparaméterének függvényében.

5.2 B92 QKD protokoll

A B92 protokollt Charles Bennett publikálta 1992-ben [11]. A protokoll működése nagyon hasonlít a BB84-éhez.

5.2.1 Működési elv

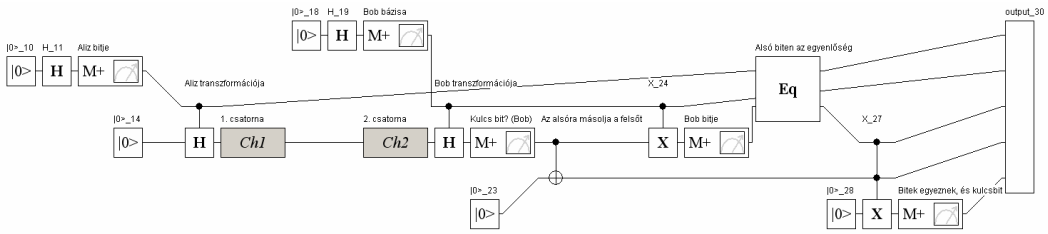
Aliz egyenletes eloszlással sorsol egy véletlen bitsorozatot, amit a következőképpen kódol:

$$\begin{aligned} 0 &\rightarrow |0\rangle, \\ 1 &\rightarrow \mathbf{H}|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right). \end{aligned} \tag{5.4}$$

Aliz elküldi ezeket a qubiteket Bobnak a kvantum csatornán, aki minden bithez sorsol egy bázist (0 vagy 1). Ha Bob az 1-es bázist sorsolta, akkor végrehajt egy Hadamard transzformációt a qubiten, ami megcseréli a qubit két lehetséges állapotát:

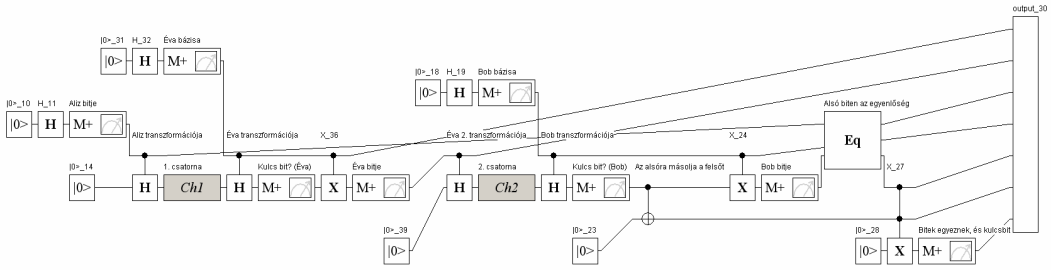
$$|0\rangle \leftrightarrow \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right). \tag{5.5}$$

Bob ezután megméri a qubitet $\mathcal{M}_{+}^{1}$-gyel. Ha 0-t kap, az nem mond semmit, mert ez előállhatott $|0\rangle$-ból és $\frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)$-ből is. Az 1-es mérési eredmény azonban csak $\frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)$-ből jöhet létre. Ezek a bitek alkotják a „nyers” kulcsot, ugyanis ekkor Bob biztos lehet benne, hogy Aliz 1-et küldött, ha a bázisa 0 volt, illetve 0-t, ha a bázisa 1 volt. Tehát ha Bob invertálja ezeket a biteket azokban az esetekben, amikor a Bázisa 1 volt, akkor megkapja a kulcsot. Az 5-5. ábra mutatja a protokoll áramkörét. Miután a kommunikáció lezajlott, Bob egy publikus csatornán tudatja Alizzal a kulcs bitjeinek pozícióját. Ez a „nyers” kulcs Aliznál ás Bobnál azonosnak kell lenni, ha nem volt lehallgató, és a csatorna zajmenetes.



5-5. ábra: A B92 protokoll lehallgató nélkül

Éva a csatornán hallgatózva ugyanazt az eljárást végzi, mint Bob (5-6. ábra). Ha Aliz $|0\rangle$-t küld, Éva ½ valószínűséggel 1 bázist választ, utána ½ valószínűséggel 0-nak méri, ami az invertálás és újraküldés után $\frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)$ lesz. Hasonlóan Éva $\frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)$-et $|0\rangle$-ra változtatja ¼ valószínűséggel. Következésképpen Éva 0.25-ös bithiba arányt okoz.



5-6. ábra: A B92 protokoll lehallgatóval

r_{sec} ismét definiálható úgy, mint a lehallgató nélküli és a lehallgatós eset bithiba arányainak különbsége. $0 \le r_{\text{sec}} \le 0.25$ ismét igaz.

5.2.2 Szimuláció

A B92 szimulációja nagyon hasonlít a BB84-ére. r_{sec} értéke itt is a mérési eredményekből kapható. A BB84-hez hasonlóan jelöljük a „Kulcs bit? (Bob)” mérés 1-es eredményének előfordulási arányát r_{rawkey}-vel és $r_{\text{Eve,rawkey}}$-vel, a „Bitek egyeznek, és kulcsbit” mérés 1-es eredményének előfordulási arányát r_{bits}-szel és $r_{\text{Eve,bits}}$-szel, megkülönböztetve a lehallgató nélküli és a lehallgatós esetet. A biztonság mértéke (5.1)-hez hasonlóan kapható:

$$r_{\text{sec}} = \frac{r_{\text{bits}}}{r_{\text{rawkey}}} - \frac{r_{\text{Eve,bits}}}{r_{\text{Eve,rawkey}}}. \tag{5.6}$$

Az analitikus vizsgálat itt is lehetséges. A mérések csatornába történő átkapcsolása és a két „Subsystem watch” elem felvétele után lefuttatott szimuláció a ρ_{rawkey}, $\rho_{\text{Eve,rawkey}}$, ρ_{bits}, $\rho_{\text{Eve,bits}}$ sűrűségmátrixokat generálja. (5.2)-höz hasonlóan r_{sec} pontos értéke:

$$r_{\text{sec}} = \frac{\langle 1|\rho_{\text{bits}}|1\rangle}{\langle 1|\rho_{\text{rawkey}}|1\rangle} - \frac{\langle 1|\rho_{\text{Eve,bits}}|1\rangle}{\langle 1|\rho_{\text{Eve,rawkey}}|1\rangle}. \tag{5.7}$$

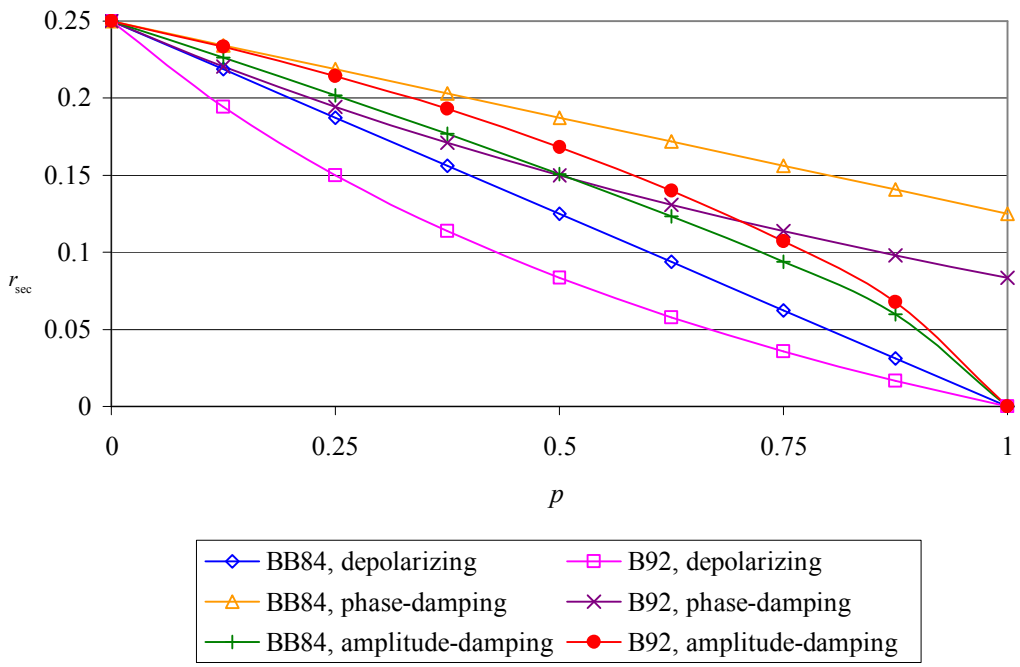
Most (5.3)-hoz hasonló egyszerűsítés nem lehetséges, mivel $\langle 1|\rho_{\text{rawkey}}|1\rangle$ vagy $\langle 1|\rho_{\text{Eve,rawkey}}|1\rangle$ – a csatorna vagy egy esetleges más lehallgató stratégia miatt – nem feltétlenül 0.25.

5.3 A BB84 és a B92 összehasonlítása

A két protokoll biztonságát vizsgáltam a „depolarizing” (2.47), „phase-damping” (2.48) és „amplitude-damping” (2.49) csatornákkal. Minden esetben *Ch1* zajmentes volt, *Ch2* pedig egy a három csatorna közül, azaz azt feltételeztem, hogy a lehallgató veszteség nélkül hozzáfér az Aliz által küldött qubitekhez, de továbbküldeni már csak a kvantum csatornán tudja. Minden esetben a p csatornaparamétert a zajmentes 0-ról a teljesen zajos 1-re növeltem 0.125-ös lépésekben. Az 5-7. ábra mutatja az analízis eredményét.

A „depolarizing” csatorna esetében a BB84 biztonsága lineárisan csökken 0.25-ről 0-ra. A B92 kevésbé biztonságos, ha $0 < p < 1$. A két protokoll r_{sec} értéke közötti különbség nagyjából $p = 0.4$-nél van, legnagyobb értéke kb. 0.043. Ha a csatorna kevéssé ($p \approx 0$) vagy nagyon zajos ($p \approx 0.25$), akkor a két protokoll biztonsága nagyjából azonos.

A „phase-damping” csatorna esetében r_{sec} egyik protokollnál se csökken 0-ra még a legzajosabb esetben sem. Egy lehallgató tehát még akkor is lefülelhető, ha $p = 1$. Ez



5-7. ábra: A diagram a BB84 és a B92 protokoll – három csatornával történő – analízisének eredménye. r_{sec} értékét mutatja a p csatornaparaméter függvényében.

azért van, mert némely biteket a csatorna által előnyben részesített bázisában küldünk. A „phase-damping” csatorna esetében is a BB84 a biztonságosabb protokoll, a legnagyobb különbség r_{sec}-ben $p = 0.9$-nél van, értéke 0.043.

Az „amplitude-damping” csatorna esetében B92 a biztonságosabb, ha $0 < p < 1$. A két protokoll biztonsága közötti különbség nem nagy, a maximum 0.017, $p = 0.5$-nél.

5.3.1 Konklúzió

Az eredményeket összefoglalva kijelenthetjük, hogy nincs legjobb protokoll. Az hogy melyiket célszerű használni attól függ, milyen csatornánk van. Úgy tűnik viszont, hogy ha ismerjük a csatorna matematikai modelljét, akkor ki tudjuk választani a csatornához a jobb protokollt, a p csatornaparaméter konkrét értékétől függetlenül. (Eltekintve a $p = 0$ – és némely csatornánál a $p = 1$ – irreleváns esetektől.)

6 Összefoglalás

A dolgozatomban a számítástechnika fejlődésének egy új irányával, a kvantum informatikával foglalkoztam. Bemutattam a tudományterület matematikai alapjait, a kvantummechanika posztulátumait, majd erre építve bevezettem a kvantum algoritmus és a kvantum-áramkör fogalmát. Leírtam egy tipikusan kvantummechanikai jelenséget, az összefonódást, amelyet később az algoritmusokban erőforrásként használtam. Bevezettem a sűrűségmátrixokat és erre építve a Bloch gömöt és a későbbiekben sokat használt kvantum csatornát.

Kvantum-áramkörök szerkesztésére és szimulációjára terveztem a QCircuit programot. A dolgozatban ismertettem a program kezelését és működését, különös tekintettel a szimuláció algoritmusára, illetve más hasonló célú programmal is összehasonlítottam.

A program széleskörű alkalmazhatóságának demonstrálása céljából dolgozatomban bemutattam néhány alkalmazási területet ismert algoritmusokon keresztül. A teleportálás és a „Super-dense Coding” az összefonódást használja ki, hogy kvantum információt klasszikus információvá alakítson, és fordítva. A Deutsch-Jozsa algoritmus olyan kvantum megoldása egy adott problémának, amely exponenciálisan gyorsabb minden determinisztikus klasszikus algoritmusnál. A bemutatott hibavédő kódolások segítségével lehetőség van a törékeny kvantum információ megvédésére.

A kvantum kulcsszétosztó protokollok az első olyan kvantum informatikai alkalmazások, amelyek fizikai megvalósításai már kereskedelemben is kaphatóak. A szimulációs programmal lehetőség nyílik különböző kvantum csatorna modellekkel vizsgálni a protokollok működését, és a kapott bithiba arányok segítségével a protokollok biztonságát meghatározni.

7 Irodalomjegyzék

[1] Colin P. Williams, Scott H. Clearwater: Explorations in Quantum Computing. New York: Springer, 1998.

[2] Mark Oskin: Quantum Computing - Lecture Notes. http://www.cs.washington.edu/homes/oskin/quantum-notes.pdf

[3] John Preskill: Lecture Notes for Physics 229: Quantum Information and Computation. http://www.theory.caltech.edu/people/preskill/ph229/#lecture

[4] S. Imre, F. Balázs, Quantum Computing and Communications – An Engineering Approach, Published by John Wiley and Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England, 2004, ISBN 0-470-86902-X.

[5] Charles H. Bennett, David P. DiVincenzo, and John A. Smolin: Capacities of Quantum Erasure Channels. http://arxiv.org/abs/quant-ph/9701015

[6] E. Knill, R. Laflamme, A. Ashikhmin, H. Barnum, L. Viola andW. H. Zurek: Introduction to Quantum Error Correction. http://arxiv.org/abs/quant-ph/0207170

[7] R. Landauer, "Irreversibility and heat generation in the computing process," IBM Journal of Research and Development 5:183-191 (1961). http://www.aeiveos.com/~bradbury/Authors/Computing/Landauer-R/IaHGitCP.html

[8] D. Deutsch and R. Jozsa, Rapid Solution of Problems by Quantum Computation, Proc. Roy. Soc. London Ser. A 439 (1992) 553.

[9] Peter W. Shor: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. http://arxiv.org/abs/quant-ph/9508027

[10] C. H. Bennett and G. Brassard, “Quantum cryptography: Public-key distribution and coin tossing,” in Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, Bangalore, India, 1984, (IEEE Press, 1984), pp. 175–179.

[11] C. H. Bennett, “Quantum Cryptography Using Any Two Non-Orthogonal States,” Phys. Rev. Lett. 68, 3121, 1992.

[12] Lov K. Grover: A fast quantum mechanical algorithm for database search. http://arxiv.org/abs/quant-ph/9605043

[13] A. R. Calderbank and Peter W. Shor: Good quantum error-correcting codes exist. http://arxiv.org/abs/quant-ph/9512032

[14] Peter W. Shor: Fault-Tolerant Quantum Computation. http://arxiv.org/abs/quant-ph/9605011

[15] D. Stucki, N. Gisin, O. Guinnard, G. Ribordy, H. Zbinden, “Quantum key distribution over 67 km with a plug&play system,” New Journal of Physics 4, 2002, 41.1–41.8. http://www.idquantique.com/files/njp-2002.pdf

[16] R. J. Hughes, J. E. Nordholt, D. Derkacs, C. G. Peterson, “Practical free-space quantum key distribution over 10 km in daylight and at night.” http://arxiv.org/abs/quant-ph/0206092

[17] id Quantique quantum cryptography system. http://www.idquantique.com/files/spec-qkd.pdf

[18] MagiQ QPN Security Gateway. http://www.magiqtech.com/

[19] Moore's Law, Intel Research.
http://www.intel.com/research/silicon/mooreslaw.htm
[20] Microsoft .NET Framework.
http://www.microsoft.com/net/
[21] QCircuit program weboldala.
http://www.hszk.bme.hu/~pa310/quantum/qc/
[22] X3D Documentation
http://www.web3d.org/x3d/
[23] Flux Player
http://www.mediamachines.com/download.html
[24] Standard C++ Library.
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vcstdlib/html/vclrfcplusspluslibraryoverview.asp
[25] Microsoft Foundation Class (MFC) Library, Document/View Architecture.
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vccore/html/_core_document.2f.view_architecture_topics.asp
[26] .NET Framework Class Library.
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconTheNETFrameworkClassLibrary.asp
[27] Matrix TCL Lite v1.13
Copyright (c) 1997-2002 Techsoft Pvt. Ltd.
Matrix.h: Matrix C++ template class include file
http://www.techsoftpl.com/matrix/
[28] SENKO Quantum Computer Simulator weboldala.
http://www.senko-corp.co.jp/qcs/
[29] QCAD weboldala.
http://acolyte.t.u-tokyo.ac.jp/~kaityo/qcad/
[30] Quantum Qudit Simulator weboldala.
http://www.compsoc.nuigalway.ie/~damo642/QuantumSimulator/QuantumSimulator/QuantumQuditSimulator.htm
[31] Quantum Designer and Network Simulator (QDNS) weboldala.
http://www.hit.bme.hu/people/imre/pages/QDNS/
[32] QCL weboldala.
http://tph.tuwien.ac.at/~oemer/qcl.html
[33] Bernhard Ömer: Quantum Programming in QCL
http://tph.tuwien.ac.at/~oemer/doc/quprog.pdf

Köszönettel tartozom konzulensemnek, Dr. Imre Sándornak, hogy hasznos szakmai tanácsaival és támogató munkájával segítette a tanulmányaimat és a dolgozatom elkészítését. Hálás vagyok továbbá a szüleimnek a támogatásukért és a fáradozásaikért, amellyel sokat segítettek a dolgozat stílusának kialakításában és a hibák felderítésében.