

Issue X: мова програмування для КВАНТОВИХ КОМП'ЮТЕРІВ

Максим Сохацький ¹

¹ Національний технічний університет України
«Київський Політехнічний Інститут» ім. Ігора Сікорського
29 жовтня 2018

Анотація

Ця робота є спробою огляду існуючих мов програмування для квантових обчислень, їх теоретичних основ та особливостей. Як приклад розглядається дискретне перетворення Фур'є, яке в якості вправи імплементується на двох мовах програмування: практичній, імперативній QCL від Бернхарда Омера[16] та теоретичній формальній, функціональній мові програмування, лямбда-численні з лінійними функціями від Андре ван Тондера[14]. Як висновок пропонуємо нарис своєї мови PLQ для квантових обчислень з залежними, залежними-лінійними та квантовими типами.

Ключові слова: Теорія типів, квантові комп'ютери

Зміст

1	Попередні відомості	2
1.1	Лінійна алгебра	2
2	Інтерпретація квантової механіки	3
2.1	Пам'ять квантового комп'ютера	4
2.2	Оператори обчислювального ядра	5
3	Огляд існуючих мов	7
3.1	Імперативні мови програмування	8
3.2	Функціональні мови програмування	9
4	Висновки	11
4.1	Мова PLQ	11

1 Попередні відомості

1.1 Лінійна алгебра

Нотація Дірака це компактний формалізм лінійної алгебри який будемо застосовувати для визначень квантової механіки¹.

Таблиця 1: Нотація Дірака

Нотація	Визначення
$ \psi\rangle$	загальний кет-вектор, наприклад $(c_0, \dots, c_n)^T$
$\langle\psi $	дуальний бра-вектор, наприклад (c_0^*, \dots, c_n^*)
$ n\rangle$	n-й базис вектор стандартного базису $N = (0\rangle, \dots, n\rangle)$
$ \tilde{n}\rangle$	n-й базис вектор альтернативного базису $\tilde{N} = (\tilde{0}\rangle, \dots, \tilde{n}\rangle)$
$\langle\phi \psi\rangle$	скалярний добуток
$ i, j\rangle$	тензорний добуток базисних векторів $ i\rangle$ та $ j\rangle$
$ \phi\rangle \otimes \psi\rangle$	тензорний добуток

Визначення 1. (Векторний простір). Множина V називається векторним простором над скалярним полем F , тоді і тільки тоді, коли визначені операції $+$: $V \times V \rightarrow V$ (сума векторів) та \cdot : $F \times V \rightarrow V$ (добуток скаляра та вектора) з наступними властивостями: i) $(V, +)$ утворюють комутативну групу; ii) $\lambda|\psi\rangle = |\psi\rangle\lambda$; iii) $\lambda(\mu|\psi\rangle) = (\lambda\mu)|\psi\rangle$; iv) $(\lambda + \mu)|\psi\rangle = \lambda|\psi\rangle + \mu|\psi\rangle$; v) $\lambda(|\psi\rangle + |\varphi\rangle) = \lambda|\psi\rangle + \lambda|\varphi\rangle$. Далі будемо розглядати скалярне поле комплексних чисел $F = C$.

Визначення 2. (Скалярний добуток). Функція $\langle\cdot|\cdot\rangle : V \times V \rightarrow C$ називається скалярним добутком, тоді і тільки тоді, коли: i) $\langle\psi|(\lambda\varphi + \mu\chi)\rangle = \lambda\langle\psi|\varphi\rangle + \mu\langle\psi|\chi\rangle$; ii) $\langle\psi|\varphi\rangle = \langle\varphi|\psi\rangle^*$; iii) $0 < \langle\psi|\psi\rangle \in \mathbb{R}$. Скалярний добуток визначає норму $\| |\psi\rangle \| = \sqrt{\langle\psi|\psi\rangle} = \| \psi \|$.

Визначення 3. (Повний векторний простір). Нехай V векторний простір з нормою $\| \cdot \|$ та $|\psi_n\rangle \in V$ послідовність векторів. i) $|\psi\rangle$ є послідовністю Коші тт. $\forall \epsilon > 0 \exists N > 0 : \forall n, m > N, \| |\psi_n\rangle - |\psi_m\rangle \| < \epsilon$. ii) $|\psi\rangle$ сходиться тт. $\forall \epsilon > 0 \exists N > 0 : \forall n > N, \| |\psi_n\rangle - |\psi\rangle \| < \epsilon$. Простір V повний тт. кожна послідовність Коші сходиться.

Визначення 4. (Гільбертів простір). Повний векторний простір H зі скалярний добутком $\langle\cdot|\cdot\rangle$ та відповідною нормою $\| \psi \| = \sqrt{\langle\psi|\psi\rangle}$ називається Гільбертовим.

Визначення 5. (Лінійний оператор). Нехай V – векторний простір, а A – функція $A : V \rightarrow V$. Тоді A називається лінійним оператором тт.

$$A(\lambda|\psi\rangle + \mu|\varphi\rangle) = \lambda A|\psi\rangle + \mu A|\varphi\rangle$$

В C^n лінійний оператор є матрицею $m \times n$ з елементами $a_{i,j} = \langle i|A|j\rangle$, де $A = \sum_{i,j} a_{i,j} |i\rangle\langle j|$. За визначенням лінійності оператора A можна записати

¹І.О. Вакарчук. Квантова Механіка. 2012

через лінійну суму векторів базису B :

$$A : |n\rangle \rightarrow \sum_k a_{kn} |k\rangle, \text{ де } |k\rangle \in B.$$

Визначення 6. (Тензорний добуток гільбертових просторів). Нехай H_1 та H_2 — Гільбертові простори з базисами B_1 та B_2 . Тоді тензорний добуток

$$H = H_1 \otimes H_2 = \{ \sum_{|i\rangle \in B_1} \sum_{|j\rangle \in B_2} c_{ij} |i, j\rangle \mid c_{ij} \in \mathbb{C} \}.$$

також Гільбертів простір з базисом $B = B_1 \times B_2$ та скалярним добутком:

$$\langle i, j | i', j' \rangle = \langle i | i' \rangle \langle j | j' \rangle = \delta_{ii'} \delta_{jj'}, \text{ де } |i\rangle, |i'\rangle \in B_1, |j\rangle, |j'\rangle \in B_2.$$

Визначення 7. (Тензорний добуток лінійних операторів). Нехай A та B лінійні оператори на Гільбертових просторах H_1 та H_2 , тоді тензорний добуток

$$A \otimes B = \sum_{i,j} \sum_{i',j'} |i, j\rangle \langle i | A | i'\rangle \langle j | B | j'\rangle \langle i', j'|$$

лінійний оператор на гільбертовому просторі $H_1 \otimes H_2$.

Визначення 8. (Комутатор та антикомутатор). Нехай A та B лінійні оператори на гільбертовому просторі H . Оператор $[A, B] = AB - BA$ називається комутатором, а $A, B = AB + BA$ називається антикомутатором.

2 Інтерпретація квантової механіки

В залежності від того як саме моделюються та конструюються гільбертові простори та гамільтоніани, виникають різні теорії, від нерелятивістської квантової електродинаміки до квантової хронодинаміки яка вводить поняття кварків та глюонів.

Теорія квантових обчислень — це ще одна теорія поверх абстрактного квантового формалізму та є інтерпретацією квантової механіки. Однак це не фізична теорія в тому сенсі, що вона не описує природний процес, а є ближчою до схемотехніки, з квабітами та квантовими вентилями, без визначення як саме моделюється квантова система, вона може бути або фізичним об'єктом або симулятором.

Точно так як для апаратного забезпечення будуються мови програмування та вищі мови програмування, так само для квантових обчислень, квантових станів та квантових логічних елементів (вентилів), існують свої мови програмування. У наступній секції дамо огляд існуючих мов та підходів до їх побудови, а тут дамо основні принципи та компоненти архітектури квантових обчислень, аби пояснити основні мовні елементи, та їх перетворення.

2.1 Пам'ять квантового комп'ютера

Визначення 9. (Квантовий біт). Квантовий біт або квабіт визначається як квантова система, стан якої може бути повністю виражений як суперпозиція (лінійна комбінація) двох ортонормованих власних базових станів позначених $|0\rangle$ та $|1\rangle$. Загальний стан $|\psi\rangle$ квабіта тоді визначається як $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, $|\alpha|^2 + |\beta|^2 = 1$. Значення квабіта описується спостереженням $N = |1\rangle\langle 1|$. $\langle N \rangle$ дає вірогідність знайти систему в стані $|1\rangle$, якщо над квабітом були проведені виміри. Простір станів квабіта є гільбертовим простором $H = \mathbb{C}^2$. Ортонормована система $|0\rangle, |1\rangle$ називається обчислювальним базисом.

Визначення 10. (Сфера Блоха). Загальний стан квабіта може бути виражений в полярних координатах θ та ϕ :

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle.$$

Одиничний вектор стану $|\psi\rangle$ називається вектором Блоха \tilde{r}_ψ , та має наступну властивість $\tilde{r}_\phi = -\tilde{r}_\xi \leftrightarrow \langle\phi|\xi\rangle = 0$. Оберти навколо осей X, Y та Z вектора

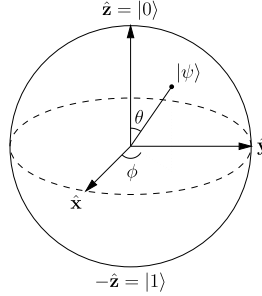


Рис. 1: Сфера Блоха як представлення квабіта $|\psi\rangle$

Блоха тоді визначаються як оператори:

$$X(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}, Y(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}, Z(\theta) = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$$

Визначення 11. (Квантова система).

Визначення 12. (Еволюція системи). Темпоральна еволюція стану системи описується рівнянням Шрьодінгера:

$$i\hbar\frac{\delta}{\delta t}|\psi\rangle = H|\psi\rangle.$$

де $\hbar \equiv 1.05457 \cdot 10^{-43}$ експериментальна константа Планка, а H — фіксований самоспряжений оператор на гільбертовому просторі, знаний як Гамільтоніан квантової системи. В квантовій фізиці нормують відносно \hbar тоді рівняння можна записати у безвимірній формі $i|\psi\rangle = H|\psi\rangle$. Гамільтоніан H повністю

визначає квантову систему. Другий спосіб визначення, через унітарний оператор $U = e^{-iH}$. Темпоральна еволюція замкненої квантової системи зі стану $|\psi\rangle$ та часу t_1 в стан $|\psi'\rangle$ та часу t_2 може бути описана унітарним оператором $U = U(t_2 - t_1)$, таким, що $|\psi'\rangle = U|\psi\rangle$.

Визначення 13. (Вимірювання). Проективне вимірювання визначається як самоспряжений оператор M який називається спостереженням зі спектральною композицією $M = \sum_m m P_m$, де P_m проекція на власний простір власного значення m . Власні значення m оператора M відповідаються усім можливим результатам вимірювання. Вимірювання $|\psi\rangle$ дасть результат m з вірогідністю $p(m) = \langle \psi | P_m | \psi \rangle$, таким чином через скорочення $|\psi\rangle$ отримаємо новий стан системи $|\psi'\rangle = \frac{1}{\sqrt{p(m)}} P_m |\psi\rangle$. Для стану кванбіта, самоспряжений оператор N знаний як стандартне вимірювання.

$$N = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = 0 \cdot |0\rangle\langle 0| + 1 \cdot |1\rangle\langle 1|.$$

Біль загально для простору стану $H = C^n$ стандартне вимірювання визначається як $N = \sum_i i |i\rangle\langle i|$.

Визначення 14. (Виважене середнє). Виважене середнє $\langle M \rangle$ усіх можливих результатів вимірювання M називється очікуваним значенням та визначається як

$$\langle M \rangle = \sum_n p(m) m = \sum_m \langle \psi | m P_m | \psi \rangle = \langle \psi | M | \psi \rangle.$$

2.2 Оператори обчислювального ядра

Визначення 15. (Оператори Паулі). Оператори Паулі пов'язані з довільним обертотом вектора Блоха формулою:

$$R_{\vec{n}}(\theta) = e^{-\frac{i}{2}\theta\vec{n}\cdot(x,y,z)} = \cos\frac{\theta}{2}I - i\sin\frac{\theta}{2}(n_xX + n_yY + n_zZ)$$

де X, Y, Z — оператори:

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

Найпростіший випадок унітарного перетворення це оператор який діє на одиничний кванбіт. Загальна форма 2-вимірної комплексної унітарної матриці $U \in SU(2)$ має вигляд:

$$U = e^{i\varphi} \begin{pmatrix} e^{\frac{i}{2}(-\alpha-\beta)} \cos\frac{\theta}{2} & -e^{\frac{i}{2}(-\alpha+\beta)} \sin\frac{\theta}{2} \\ e^{\frac{i}{2}(\alpha-\beta)} \cos\frac{\theta}{2} & e^{\frac{i}{2}(\alpha+\beta)} \sin\frac{\theta}{2} \end{pmatrix}$$

Нехай $U \in SU(2)$ має базисні вектори $|u\rangle$ та $|v\rangle$ та власні значення u та v . Тоді U можна виразити:

$$U = u|u\rangle\langle u| + v|v\rangle\langle v| = e^{i\varphi} R_{\vec{n}}(\delta),$$

де δ — фазова різниця між u та v — пов'язана як $v = e^{i\delta}u$.

Визначення 16. (Оператор Адамара).

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

Визначення 17. (Фазовий та $\pi/8$ оператори).

$$S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, R_3 = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$$

Визначення 18. (Контрольований вентиль). Нехай U унітарний q -квабітний вентиль, контрольований U -вентиль з n -контрольними бітами тоді визначається як

$$C^m[U] = \begin{bmatrix} I & \dots & 0 & 0 \\ \vdots & \ddots & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & U \end{bmatrix}$$

в просторі станів $B^{\otimes n+m}$.

Визначення 19. (Контрольований НЕ вентиль).

$$CNot : |x, y\rangle \rightarrow |x \otimes y, y\rangle$$

$$CNot = C[X] = \begin{bmatrix} I & 0 \\ 0 & X \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Визначення 20. (Вентиль перестановки).

$$Swap : |x, y\rangle \rightarrow |y, x\rangle$$

Визначення 21. (Контрольований фазовий вентиль).

$$CPhase : |x, y\rangle \rightarrow i^{xy}|x, y\rangle$$

Визначення 22. (Вентиль Тофолі).

$$CCNot : |x, y, z\rangle \rightarrow i^{xy}|x, y\rangle$$

3 Огляд існуючих мов

З огляду на новизну предмету розроблено не так багато мов, усі що є можна розділити на імперативні (по духу своєї імплементації) та функціональні (або побудовані на базі певного виду лямбда числення). Ми наведемо приклади програм для обох підходів. У якості порівняльної характеристики візьмемо алгоритм дискретного перетворення Фур'є.

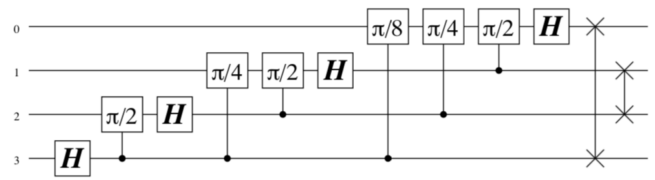


Рис. 2: Графічне представлення квантової схеми вентилів дискретного перетворення Фур'є для 4-квабітного регістру

3.1 Імперативні мови програмування

Станом на 2018 рік найбільш розробленою мовою, яка компілюється під Linux та Mac є QLC від Бернхарда Омера ². До QLC можна відзначити наступні мови: 1) Q-gol від Грега Бейкера (1996) ³; 2) qGCL від Паоло Зуліані (2000)[10]; 3) Quantum C Language від Стефана Блаха (2002); 4) QPAlg від Марі Лолір та Філіпа Жорранда (2004)[8], синтаксис та семантика цієї системи базується на численні процесів Мілнера CCS та формальній мові LOTOS; 5) CQP (Communication Quantum Processes) від Саймона Гея та Раджагопала Нагараджана (2004)[5]; 6) Q (DSL мова для C++) від Беттеллі, Каларко та Серафіні]. 7) LanQ від Гинека Млнаріка[9].

Визначення 23. (Перестановка).

```
cond qufunct Swap(quireg a,quireg b) {  
  int i; if #a!=#b { exit "arguments must equal"; }  
  for i=0 to #a-1 {  
    CNot(b[i],a[i]);  
    CNot(a[i],b[i]);  
    CNot(b[i],a[i]); } }
```

Визначення 24. (Зміна порядку квібітів).

```
cond qufunct flip(quireg q) {  
  int i; for i=0 to #q/2-1 { Swap(q[i],q[#q-i-1]); } }
```

Визначення 25. (Дискретне перетворення Фур'є, Коперсміт).

```
operator dft(quireg q) {  
  const n=#q; int i; int j;  
  for i=1 to n {  
    for j=1 to i-1 { V(pi/2^(i-j),q[n-i] & q[n-j]); }  
    H(q[n-i]); }  
  flip(q); }
```

²Bernhard Ömer. Structured Quantum Programming. PhD. TU Vienna. 2003. <http://tph.tuwien.ac.at/~oemer/doc/structqprog.pdf>

³Gregory David Baker. Qgol. A system for simulating quantum computations: Theory, Implementation and Insights. 1996. PhD. Macquarie University. <http://www.ifost.org.au/~gregb/q-gol/QgolThesis.pdf>

3.2 Функціональні мови програмування

Лямбда числення лежить в основі сучасних алгебраїчних систем та основ математики[4], воно може бути розглянуте не лише як мова програмування загального використання, але і фреймворк для судження про обчислювальні властивості програм.

Таксономію лямбда числень, яку виконав Хенк Барендрегт та подав у вигляді лямбда кубу[3], можна розширити квантовими мовними примітивами, вводити в вищі лямбда числення з гомотопічними типами, або в системи доведення теорем з екстрактом або лише верифікацією.

З огляду на природу квантових обчислень необхідним компонентом квантового лямбда числення є система лінійних типів, де за час існування змінної в області видимості під час виконання дозволяється звертання до неї тільки один раз. Така система типів уже використовується не тільки в експериментальних верифікаторах але і в сучасних системних мовах програмування (Rust), де завдяки верифікатору лінійних типів вдається обійтися без алгоритмів автоматичного вивільнення пам'яті під час виконання програми (схема пам'яті повністю моделюється під час компіляції програми).

Серед робіт присвячених мовам на базі лямбда числень можна відзначити наступні: 1) Найбільш класичне нетипизоване лямбда числення Андре ван Тондера[14]. Тут подається доведення ізоморфізму машині Тюрінга, повнота та звучання системи, що є формальною перевагою перед імперативними мовами, такими як QCL; 2) Quipper від Гріна, Люмсдейна, Росса, Селінжера та Валірона [11][12]. Це спроба вбудувати DSL для квантових обчислень в мову Haskell, симулятор мови побудований на генераторах групи Кліффорда, квантових операторах X, Y, Z, H, S . 3) Робота по лямбда численню від Arrigі та Доєка[2]. 4) QML для Haskell від Торстена Альтенкірха та Джонатана Граттажа[1][15][6].

Визначення 26. (Синтаксичне дерево O_H). Синтаксичне дерево O_H визначає лямбда числення з лінійними змінними поєднане з класичним нетипизованим лямбда численням. Тобто O_H є найпростішим операційним квантовим лямбда численням для середовищ виконання.

$$\begin{array}{l} c = \mathbf{0} \mid \mathbf{1} \mid \mathbf{H} \mid S \mid R_3 \mid \mathbf{CNot} \mid X \mid Y \mid Z \mid \dots \\ t = x \mid \lambda x . t \mid \mathbf{let} \ x = t \ \mathbf{in} \ t \mid t \ t \\ \mid (t, t) \mid t \mid c \mid !t \mid \lambda !x . t \end{array}$$

Тут даються примітиви лінійних типів, де доступ до змінної можливий лише раз в області визначення змінної. Для нелінійних або звичайних лямбда функції даються примітиви позначені $!$. В переліку квантових примітивів с даються: i) ортонормований базис $|0\rangle$ та $|1\rangle$; ii) H — оператор Адамара; iii) S — фазовий вентиль; iv) R_3 — $\pi/8$ вентиль; v) контрольований не вентиль $CNot$; vi) вентилі Паулі X , Y та Z .

Визначення 27. (Пара Айнштайна-Подольського-Розена).

$$\mathbf{EPR} = \mathbf{CNot} ((H \ 0), 0) \quad (1)$$

Визначення 28. (Квантова телепортація).

$$\begin{array}{l} \mathbf{teleport} \ x = \mathbf{let} \ (e_1, e_2) = \mathbf{EPR} \ \mathbf{in} \\ \quad \mathbf{let} \ (x', y') = \mathbf{alice} \ (x, e_1) \ \mathbf{in} \ \mathbf{bob} \ (x', y', e_2) \end{array} \quad (2)$$

$$\begin{array}{l} \mathbf{alice} \ (x, e_1) = \mathbf{let} \ (x', y') = \\ \quad \mathbf{CNot} \ (x, e_1) \ \mathbf{in} \ ((H \ x'), y') \\ \mathbf{bob} \ (x', y', e_2) = \mathbf{let} \ (y'', e'_2) = cX \ (y', e_2) \ \mathbf{in} \\ \quad \mathbf{let} \ (x'', e''_2) = cZ \ (x', e'_2) \ \mathbf{in} \ (x'', y'', e''_2) \end{array} \quad (3)$$

Визначення 29. (Дискретне перетворення Фур'є).

$$\mathbf{fourier} \ list = \mathbf{reverse} \ \mathbf{fourier}' \ list \quad (4)$$

$$\mathbf{fourier}' \ list = \mathbf{case} \ list \ \mathbf{of} \ \begin{cases} () \rightarrow () \\ h : t \rightarrow \mathbf{let} \ h' : t' = \mathbf{phases} \ (H \ h) \ t \ !2 \\ \quad \mathbf{in} \ h' : \mathbf{fourier}' \ t' \end{cases} \quad (5)$$

$$\begin{array}{l} \mathbf{phases} \ target \ controls \ !n = \\ \mathbf{case} \ control \ \mathbf{of} \ \begin{cases} () \rightarrow target \\ control : t \rightarrow \mathbf{let} \ (control', target') = \\ \quad (cR \ !n) \ (control, target) \ \mathbf{in} \\ \quad \mathbf{let} \ target'' : t' = \mathbf{phases} \ target' \ t \ !(\mathbf{succ} \ n) \ \mathbf{in} \\ \quad target'' : control' : t' \end{cases} \end{array} \quad (6)$$

4 Висновки

Як видно для реалізації семантики мови програмування для квантових комп'ютерів достатньо поєднати тензорне числення разом з π -численням процесів, або лінійними типами. На сьогоднішній день (2018) серед імперативних мов програмування найбільш завершена, повна та практична на думку автора є QLC від Бернхарда Омера, серед мов для лямбда числень немає жодної достатньо зрілої імплементації.

4.1 Мова PLQ

Відкрите питання в теорії типів, імплементація та дизайн мови з лінійними та залежними типами та квантовими примітивами. Тому ми пропонуємо як висновок після огляду мов та їх синтаксисів свій синтаксис такої мови, та показуємо з яких інгредієнтів її можна побудувати.

Мову квантових обчислень O_H можна розкласти до комбінації (прямої суми) трьох синтаксичних дерев: i) дерева O_λ — базового чистого лямбда числення з залежними типами (pure, імплементовану автором[13]); ii) дерева O_π — числення з лінійними типами, або просто інший вид стрілок (linear); iii) операторний базис квантового числення O_Q з конструкторами X, Y, Z, S, H, R_3 та конструктором визначення квабіта (quantum). Тоді формальний синтаксис мови для квантових комп'ютерів, записаний на **cubicaltt**[7] у вигляді індуктивного типу синтаксичного дерева, буде виглядати так:

```
data pure (lang: U)
  = star (n: nat) | var (l: nat) | pi (l: nat) (f: lang)
  | lambda (x: nat) (f: lang) | app (f a: lang)

data linear (lang: U)
  = star (n: nat) | var (l: nat) | pi (l: nat) (f: lang)
  | lambda (x: nat) (f: lang) | app (f a: lang)

data quantum (lang: U)
  = register (n: nat) | i (n: nat) | 0 | 1 | H (l: lang) | S (:lang)
  | X (f: lang) | Y (f: lang) | Z (f: lang) | CNot (f a: lang)
```

Результуюча авторська мова (назвемо її PLQ) виражається як взаєморекурсивна сума елементарних мовних синтаксисів.

```
data PLQ = Pure    (_: pure    PLQ)
          | Linear  (_: linear  PLQ)
          | Quantum (_: quantum PLQ)
```

З огляду на розмір реферату, семантику цієї мови наводити не будемо, однак, очевидно, що мову Андре ван Тондера можна продовжити до PTS з лінійними типами в категоріях з сімействами зі значеннями в симетричних моноїдальних категоріях, як було показано Матейсом Вакаром[?].

Список використаних джерел

- [1] Thorsten Altenkirch and Jonathan Grattage. In *A functional quantum programming language.*, page 249–258. IEEE, 2005.
- [2] Pablo Arrighi and Gilles Dowek. In *Operational semantics for formal tensorial calculus.*, 2004.
- [3] H. P. Barendregt. In *Lambda calculi with types.*, page 117–309, New York, NY, USA, 1992. Oxford University Press, Inc.
- [4] Andrej Bauer; Steve Awodey; Matthieu Sozeau; Michael Shulman; Dan Licata; Yves Bertot; Peter Dybjer; and Nicola Gambino. In *Homotopy Type Theory: Univalent Foundations of Mathematics.*, 2013.
- [5] Simon J. Gay and Rajagopal Nagarajan. In *Communicating quantum processes*, page 145–157, New York, NY, USA., 2005. ACM.
- [6] Jonathan Grattage. In *An overview of qml with a concrete implementation in haskell.*, volume 270(1):165–174. QPL/DCM, 2011.
- [7] Cyril Cohen; Thierry Coquand; Simon Huber; and Anders Mørthberg. In *Cubical Type Theory: a constructive interpretation of the univalence axiom*, 2017.
- [8] Marie Lalire and Philippe Jorrand. In *A process algebraic approach to concurrent and distributed quantum computation: Operational semantics.*, 2004.
- [9] Hynek Mlnarik. In *Operational semantics and type soundness of quantum programming language lanq.*, 2007.
- [10] J. W. Sanders and P. Zuliani. In *Quantum programming*. Springer-Verlag, 2000.
- [11] Alexander S Green; Peter LeFanu Lumsdaine; Neil J Ross; Peter Selinger; and Benoît Valiron. In *An introduction to quantum programming in quipper.*, page 110–124. Springer, 2013.
- [12] Alexander S Green; Peter LeFanu Lumsdaine; Neil J Ross; Peter Selinger; and Benoît Valiron. In *Quipper: a scalable quantum programming language.*, page 333–342. ACM, 2013.
- [13] Namdak Tonpa. In *The systems engineering of consistent pure language with effect type system for certified applications and higher languages.*, 2018.
- [14] Andre van Tonder. In *A lambda calculus for quantum computation.*, 2004.
- [15] Thorsten Altenkirch; Jonathan Grattage; Juliana K Vizzotto; and Amr Sabry. In *An algebra of pure quantum programming.*, volume 170:23–47, 2007.
- [16] Bernhard Mörtner. In *Structured quantum programming.*, 2003.