УДК 004.056.5 **DOI:10.30837/rt.2024.3.218.03**

О.І. ФЕДЮШИН, канд. техн. наук, Є.В. ГОЛОВКО, А.О. СМІРНОВ, канд. техн. наук, В.М. СУХОТЕПЛИЙ, О.В. ЧЕЧУЙ, канд. техн. наук

МЕТОДИ ЗАХИСТУ ІНФОРМАЦІЇ НА ОСНОВІ КВАНТОВОЇ СТЕГАНОГРАФІЇ ЗОБРАЖЕНЬ

Вступ

Стеганографія – це процес, в якому конфіденційні дані в тій чи іншій формі ховаються в зображенні, яке може бути будь-яким звичайним зображенням, наприклад, кота або дерева. Зображення, яке використовується в цьому процесі, називається стего-зображенням. Цей процес не змінює жодної видимої риси вихідного контейнера. Таким чином, будь-який зловмисник, який хоче отримати доступ до даних, навіть не знає, що в ньому приховані дані. Доступ до даних може отримати лише особа, яка має спеціальний ключ, в той час як для будь-якої іншої особи він навіть не існує.

Безпека даних дуже важлива, коли мова йде про обмін інформацією між користувачами. Наш мотив полягає в тому, щоб заховати дані в зображенні та отримати їх за допомогою алгоритму вилучення, і зробити це за допомогою квантових обчислень. У квантовій обробці зображень [1] квантове представлення зображень відіграє ключову роль, яка визначає типи завдань обробки і те, наскільки добре вони можуть бути виконані.

Існують різні методи представлення зображень: Qubit Lattice [2], Entangled image, Real Ket [3], Flexible Representation of Quantum Images (FRQI) [1], Novel Enhanced Quantum Image Representation (NEQR) [4].

FRQI використовує нормалізовану суперпозицію для зберігання всіх пікселів зображення, однакові операції можна виконувати одночасно над усіма пікселями, і тому FRQI може полегшити обчислювальну проблему обробки зображень. Обмеження FRQI полягає в тому, що він використовує лише один кубіт для зберігання інформації про відтінки сірого для кожного пікселя зображення, тому деякі операції цифрової обробки зображень, такі як складні операції з кольором, не можуть бути виконані на основі FRQI.

Модель NEQR використовує лінійний незалежний базовий стан кубітової послідовності для зберігання значення відтінку сірого для кожного пікселя. Таким чином, для зберігання цифрового зображення з використанням квантової механіки в NEQR використовуються дві переплетені кубітні послідовності, які представляють інформацію про відтінки сірого та положення всіх пікселів на зображенні. У представленні FRQI інформація про відтінки сірого зображення кодується за допомогою одного кубіта, тоді як у NEQR інформація про відтінки сірого кодується в базисних кубітних станах. Оскільки кожен базисний кубітний стан є лінійно незалежним, завдання обробки зображення стає набагато простішим, ніж у FRQI. Крім того, часова складність підготовки квантового зображення NEQR зменшується приблизно в квадратичному відношенні порівняно з FRQI. Використовуючи особливості зберігання даних в різних форматах можна організувати стеганографічне вбудовування даних. Актуальним завданням роботи є проведення наукового пошуку ефективних методів квантової стеганографії. В подальшому планується провести якісний аналіз переваг та недоліків, перспектив та труднощів їх практичного впровадження та розглянути ефективні методи для моделювання подібних систем.

Представлення зображень у квантових станах

Квантова обробка зображень займається представленням зображень і зберіганням даних про зображення у квантових станах, а також перетворенням цих станів для досягнення поставленої мети. Першим кроком при обробці зображень у квантових комп'ютерах є переведення пікселів зображення у квантові стани. Розглянемо різні підходи до цього процесу.

Квантовий комп'ютер оперує так званими квантовими бітами. Можна визначити квантовий біт, або скорочено q-біт (кубіт), як квантово-механічну систему, що має два стани, які позначаються відповідно, як $|0\rangle$ і $|1\rangle$.

Однак на відміну від класичного випадку, у квантовій механіці ці два стани можуть перебувати у стані суперпозиції, тобто загальний стан квантового біта може бути записаний як:

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad (1)$$

де α і β – комплексні коефіцієнти.

Іншими словами, можна сказати, що закони квантової механіки допускають інші значення кубіта, які називаються станами суперпозиції. Таким чином стани суперпозиції являють собою значення між екстремумами 0 і 1, а квантовий біт може приймати нескінченно багато значень. Кубіт можна визначити, як вектор одиничної довжини у двовимірному гільбертовому просторі над полем комплексних чисел.

Стани $|0\rangle$ і $|1\rangle$ разом являють собою базисні вектори. Як і всі вектори, вони вказують напрямок і мають величину.

Для запису двох станів кубітів можна використовувати позначення бра ($\langle\ |$) і кет ($|\ \rangle$) – позначення Дірака. Вектори виду $|\ \rangle$ називаються кет-векторами, а виду $|\ \rangle$ бра-векторами.

Формула (1) для хвильової функції $|\varphi\rangle$ описує, в якій пропорції нескінченна множина всіх варіантів значень квантового стану містить варіанти базисних станів $|0\rangle$ і $|1\rangle$.

Візуалізація стану кубіту можлива за допомогою спеціального інструменту, названого сферою Блоха. Сфера Блоха – це сфера з одиничним радіусом, при цьому точка на її поверхні відповідає стану кубіта.

На рис. 1 зображено побудову Сфери Блоха з використанням бібліотек Qiskit від IBM та Python [5].

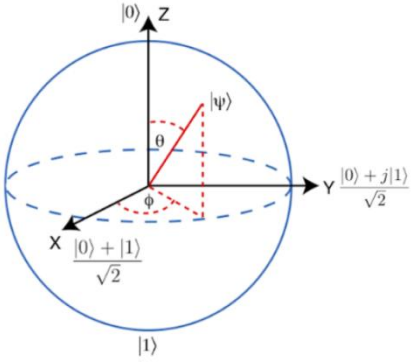

Рис. 1. Модель Сфера Блоха. Стан у верхній частині сфери представляє $|0\rangle$, а стан у нижній частині сфери представляє $|1\rangle$

Квантові решітки (Quantum Lattice).

Група пікселів збирається разом, щоб сформувати зображення. Ці пікселі містять такі властивості, як положення та інтенсивність кольору. Першим методом, який дозволив перетворити значення пікселів у квантовий стан, була модель "Квантової решітки". Дослідження забезпечило аналогове перетворення у квантові стани. Для зображення розміром $N \times N$ використовується $N \times N$ кубітів, які розташовані у матричному форматі, де кожен кубіт еквівалентний кожному пікселю. Дослідження було зосереджено на представленні пікселя у вигляді частот, а не лінійної комбінації колірної моделі RGB. Частоти, представлені пікселем, обчислюються за допомогою гіпотетичної машини, яка перетворює електромагнітні хвилі світла в ініціалізовані кубіти.

Всю систему можна представити у вигляді

$$A: F \to \varphi, \tag{2}$$

де φ представляє кубіт, який ініціалізується частотою F, зчитаною машиною A.

Процес перетворення показаний на рис. 2.

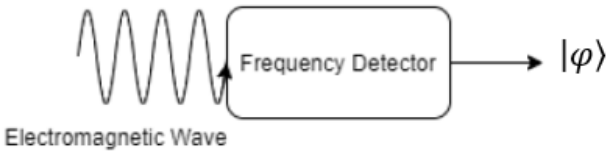

Рис. 2. Модель Qubit Lattice

Структура "решітки кубітів" формується шляхом розміщення ідентичних $K-1$ кубітів за кожним кубітом, які представляють частоту пікселя. Ці частоти кодуються в θ – кут відхилення кубіта в суперпозиції $|0\rangle$ та $|1\rangle$ (див. рис.1). Основним завданням зберігання частоти в ідентичному кубіті була можливість повернення до значень пікселів, щоб реконструювати зображення з квантового стану. Кількість часу, необхідного для отримання даних, залежить від кількості пікселів/кубітів, задіяних у процесі. Кількість кубітів прямо пропорційна точності вилучення зображень з його квантового стану. Нехай кубіт має стан α і β.

Позначимо кількість вимірювань для цих станів через M_α і M_β:

$$\cos^2\frac{\theta}{2} = M_\alpha \Big/ \left(M_\alpha + M_\beta\right) \tag{3}$$

Тоді частоту можна знайти, якщо розв'язати рівняння для тета-квантів.

Модель Real Ket.

З введенням кубітної решітки [2], Латорре у 2005 р. представив заплутану модель представлення зображень [3], де кожне зображення було розділене на 4 квадранти, де кожен квадрант був пронумерований, починаючи зліва направо з верхнього рядка. Потім ці квадранти були знову поділені на ще 4 квадранти з такими ж номерами. Цей поділ продовжувався до тих пір, поки ми не отримали один піксель. Модель ділить зображення на менші зображення (розбиваючи зображення на 4 частини) і створює структуру Quadtree з коефіцієнтами, як на зображенні у відтінках сірого на рис. 3 (тут реалізовано стиснення та заплутування зображення).

І Real Ket, і квантова решітка надавали способи представлення зображення, але вони мали свої обмеження.

Решітчаста модель вимагала більше кубітів для представлення зображення, чого практично неможливо досягти. Модель Real Ket забезпечувала краще стиснення зображення, і виявилася швидшою за модель квантової решітки, але була обмежена випадковістю, яка була необхідна пікселям для ефективної роботи коли ми розглядаємо реальне зображення, оскільки пікселі пов'язані між собою.

Модель FRQI.

У 2010 р. було запроваджено модель FRQI (Flexible Representation of Quantum Images – гнучкого представлення квантових зображень), яка мала перевагу над іншими розглянутими моделями, оскільки вимагає меншу кількість кубітів для представлення зображення. У FRQI потрібно лише $2n+1$ кубіт для зображення розміром $2^n \times 2^n$.

Таке представлення використовує переваги суперпозиції між кубітами і дозволяє кодувати інтенсивність кольору та положення пікселів у нормалізованому стані кубіта [1].

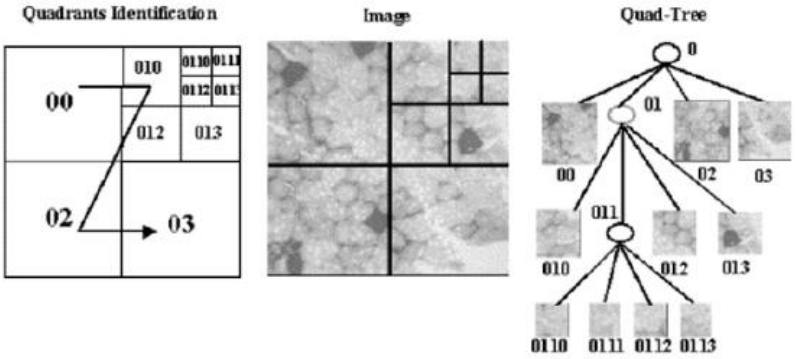

Рис. 3. Модель Real Ket

Зображення представлено в залежності від θ у рівняннях:

$$\left|I(\theta)\right\rangle = \frac{1}{2^n}\sum_{i=0}^{2^{2n-1}}\left(\cos\theta\left|0\right\rangle + e^{i\phi}\sin\theta\left|1\right\rangle\right)\otimes\left|i\right\rangle, \tag{4}$$

де $\theta \in \left[0, \frac{\pi}{2}\right], i = 0,1,2,3,\ldots,2^n - 1$.

Коефіцієнт тета дозволяє нам кодувати інтенсивність кольору, в той час як положення пікселя представлено через $\left|I\right\rangle$.

Зображення перетворюються з початкового стану ($\left|0\right\rangle^{\otimes 2n}$) кубіта у стан FRQI за допомогою унітарного перетворення (позначається P), яке включає в себе гейт Адамара (позначається H) для створення суперпозицій початкового стану кубіта з наступними контрольованими обертаннями (позначаються R) для створення FRQI. Використовуючи гейт H на керовані обертання навколо осей X та Y, результатом є стан FRQI, представлений як $P = HR$. Приклад представлення показано на рис. 4.

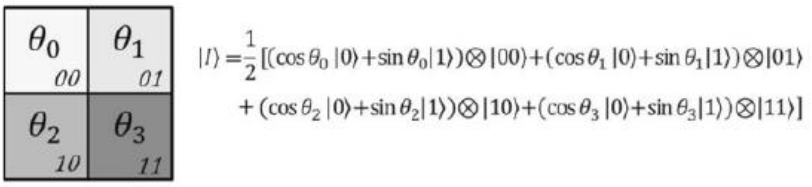

Рис. 4. Приклад представлення зображення в моделі FRQI

Перевагою FRQI є суперпозиція послідовності кубітів, що дозволяє нам трансформувати всі пікселі, змінивши лише один кубіт. Недоліком цього методу представлення зображень є кількість гейтів, необхідних для підготовки зображення у квантовому форматі FRQI. Пряма реалізація цього методу вимагає 2^{2n} керованих обертань і $2n$ гейтів Адамара. Загальна обчислювальна складність зростає до квадратичної (2^{4n}).

Іншим недоліком, який пов'язаний зі станом FRQI, є те, що його можна застосовувати лише до квадратних зображень.

Модель NEQR.

Нова модель NEQR (Novel Enhanced Quantum Image Representation – Нове покращене квантове представлення зображень) спирається на переваги, які надає FRQI, і використовує 2

заплутаних кубіти для зберігання пікселя і властивостей кольору пікселя. Метод зменшує загальну обчислювальну складність з (2^{4n}) до (2^{2n}) [4]. Він зосереджується на недоліках моделі FRQI і зберігає інформацію на основі кубітової послідовності, що дозволяє вдвічі зменшити обчислювальну складність та покращити коефіцієнт стиснення в 1,5 рази. Існує набагато більше методів, які дослідники опублікували для представлення зображення у квантових станах. Але FRQI і NEQR є найпоширенішими, з яких NEQR має більшу застосовність в поточному сценарії роботи для задач стеганографії.

Колірна схема зображення складається з трьох значень інтенсивності, відомих як RGB-значення зображення, інтенсивність кожного кольору може змінюватися від 0, де 0 означає чорний, а 255 – білий. Для кодування кожної інтенсивності ($2^q = 255$), де q – кількість кубітів, необхідних для кодування різних інтенсивностей певного кольору, а для кодування позиції нам потрібен інший набір кубітів. Оскільки ми будемо представляти двовимірне (2x2) піксельне зображення, ми будемо визначати позицію зображення через його рядок і стовпець, Y та X відповідно, а колір – через формулу

$$f(Y,X) = C_{YX}^{q-1} C_{YX}^{q-2} \ldots C_{YX}^{1} C_{YX}^{0}, C_{YX}^{q} \in [0,1], f(Y,X) \in \left[0, 2^q - 1\right] \qquad (5)$$

Промоделюємо представлення зображення в моделі NEQR за допомогою середовища Qiskit Jupyter [5].

Qiskit – фреймворк Python з відкритим вихідним кодом, наданий IBM, який використовується для маніпулювання, написання квантових програм, а потім реалізації їх на реальному квантовому комп'ютері або симуляторах, наданих на сайті IBM з квантових досліджень. Дані необхідні для побудови зображення наведено в табл. 1.

Таблиця 1

Положення пікселя	Бінарне представлення	Інтенсивність в градаціях сірого
$\|00\rangle$	$\|00000000\rangle$	0 – Black
$\|01\rangle$	$\|01100100\rangle$	100 – Darkshade
$\|10\rangle$	$\|11001000\rangle$	200 – Lightshade
$\|11\rangle$	$\|11111111\rangle$	255 – White

Етапи обробки зображень за допомогою NEQR є наступними:

Крок 1: Генеруємо класичне зображення у відтінках сірого (2×2) за допомогою python.

Крок 2: Квантовий ланцюг зображення формується на основі налаштування кольору зображення за допомогою NEQR-коду. Цей крок передбачає зберігання класичних даних у вигляді квантового стану, як показано на рис. 5.

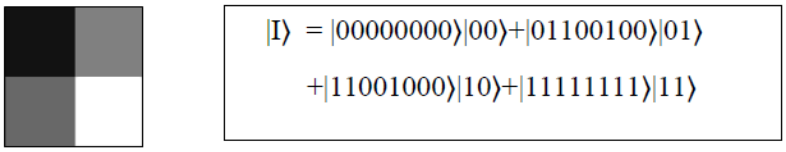

Рис. 5. Приклад представлення зображення в моделі NEQR

Код виконується в блокноті Jupyter на мові Python, а потім на симуляторах Quantum, наданих IBM. Кодується класичне зображення розміром 2×2 пікселі, як показано на рис. 5, зі значенням шкали сірого в діапазоні 0-255 у схему квантового зображення за допомогою коду NEQR. Схема, яка перетворює дані зображення в дані Quantum, показана на рис. 6. Значення всіх чотирьох пікселів розділені за допомогою бар'єрів (показані пунктирними вертикальними лініями).

На рис. 6 перші вісім кубітів (0-7) використовуються для кодування інтенсивності пікселів, а інші два кубіти (8 і 9) використовуються для кодування інформації про положення пікселів. Можливі положення пікселів отримуються за допомогою перетворення Адамара двопозиційних кубітів, яке дає нам всі чотири можливі стани положення.

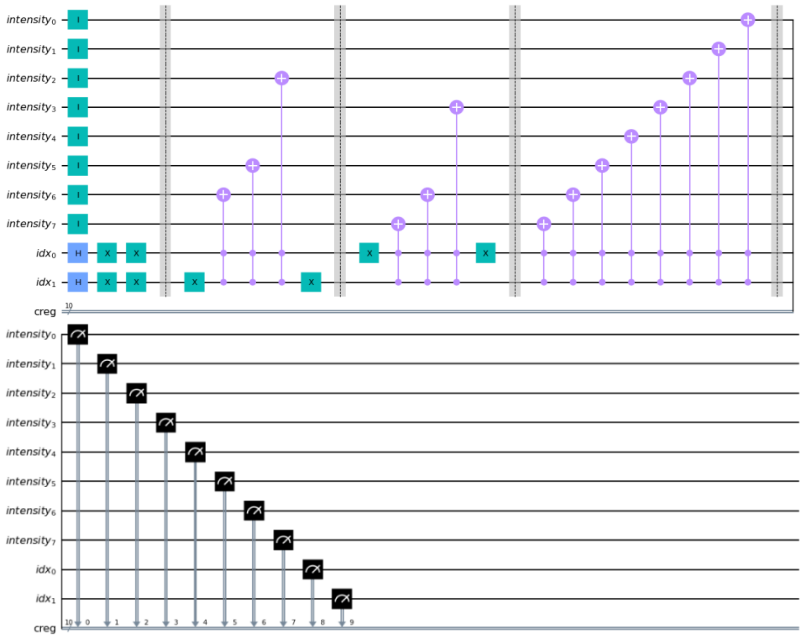

Рис. 6. Квантова схема для перетворення зображення в модель NEQR

Перший піксель на рис. 6 має інтенсивність 0, яка в NEQR представлена бітовим рядком |00000000⟩. Це кодується за допомогою восьми кубітів. Аналогічно, значення інших пікселів задаються за допомогою гейтів CC-NOT у різних позиціях схеми залежно від значення інтенсивності пікселів. В кінці схеми десять класичних регістрів використовуються для вимірювання вихідного сигналу. Всі пікселі кодуються відповідними значеннями інтенсивності. Вихід схеми – десятикубітний бітовий рядок, в якому перші два біти представляють інформацію про положення, а решта вісім бітів – інформацію про колір пікселів.

Технології безпеки на основі квантової обробки зображень

Технологію квантової обробки зображень побудовано на розширенні цифрової обробки зображень до області квантових обчислень, що призводить до реалізації безпечних, ефективних і передових технологій для криптографії та приховування інформації.

На рис. 7 представлено загальну схему квантових технологій захисту зображень в рамках цих двох широких областей.

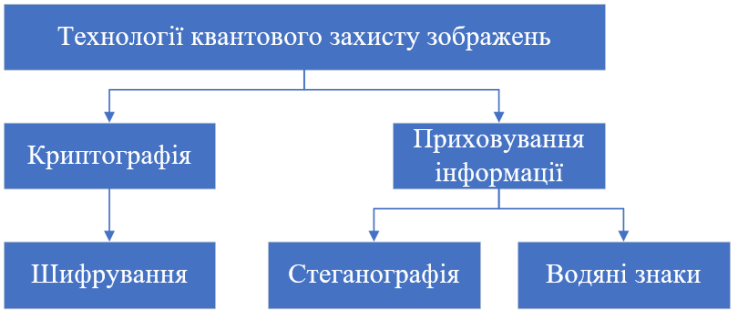

Рис. 7. Технології квантового захисту зображень

В науці криптографії шифрування розглядається як процес прямого приховування інформації, щоб зробити її нечитабельною без спеціальних знань. Зазвичай це робиться для

збереження таємниці і, як правило, для конфіденційних комунікацій. Криптографія спрямована на захист змісту повідомлень, тоді як приховування інформації фокусується на приховуванні самого факту їхнього існування. Приховування інформації за допомогою таких стратегій, як стеганографія та водяні знаки, видається більш безпечною, оскільки такі методи не так легко помітити зловмисникам. Серед основних обмежень цих методів є лімітований об'єм інформації для передачі. Адже кількість інформації залежить від розмірів контейнера-носія і алгоритму вбудовування, також зображення-носій після додавання прихованого повідомлення не повинно мати видимих спотворень.

Алгоритми для квантової стеганографії зображень

Як було сказано раніше, стеганографія зображень – це метод приховування інформації орієнтований на приховування секретного повідомлення в зображенні-носії [6]. На рис. 8 наведено загальну схему протоколів квантової стеганографії зображень, а решта цього розділу висвітлює деякі досягнення на їх основі.

Рис. 8. Схема квантової стеганографії зображень

У 2014 р. Цзян та ін. запропонували стратегію стеганографії зображень NEQR на основі муарових моделей [7]. Стратегія була розроблена як стеганографічний алгоритм з відповідними квантовими схемами для приховування двійкового зображення у відтінках сірого.

Алгоритм вбудовування починається з вибору початкової решітки муару, тобто стохастичного зображення, як зображення прикриття. Потім початкова муарова решітка модифікується відповідно до секретного зображення, і деформована розглядається як муаровий шаблон. Зрештою, зображення муару змінюється для отримання стего-зображення.

Після цього дослідження була розроблена вдосконалена версія з використанням двох сліпих алгоритмів стеганографії з використанням найменшого значущого біта (LSB) на основі NEQR представлення [7]. Перший алгоритм базується на стандартному (або простому) LSB, який використовує кубіти повідомлень для безпосередньої заміни LSB пікселів.

Хоча стандартна система стеганографії LSB є простою, її стійкість є низькою. Інший алгоритм – блоковий LSB, вбудовує кубіт повідомлення в декілька пікселів, які належать до одного блоку зображення. Стеганографічна схема блочного LSB має на меті покращити стійкість стандартної схеми LSB. Це досягається шляхом розбиття зображення обкладинки на блоки, кожен з яких приховує один кубіт повідомлення замість пікселя. Експериментальні результати, представлені в цьому дослідженні, демонструють що невидимість є хорошою, а баланс між пропускною здатністю і надійністю можна регулювати відповідно до потреб додатків.

Алгоритм простого LSB вбудовування

Припустимо, що зображення контейнеру є $2^n \times 2^n$ квантовим зображенням $|I\rangle$ з діапазоном сірого 2^q (як визначено у рівнянні (5)), а повідомлення – $2^n \times 2^n$ двійкове квантове зображення $|M\rangle$, як показано нижче:

$$|M\rangle = \frac{1}{2^n} \sum_{i=0}^{2^{2n}-1} |m_i\rangle \otimes |i\rangle, \tag{6}$$

де $m_i \in \{0,1\}, i = 0,1,\ldots,2^{2n}-1$.

Схема вбудовування простого алгоритму LSB представлена на рис. 9, в якій $2n$ вентилів CNOT використовуються для перевірки того, чи збігається інформація про положення $|I\rangle$ та $|M\rangle$. Якщо інформація про позицію ідентична, то інформація про позицію $|M\rangle$ змінюється на $|00\ldots0\rangle$. Таким чином, під їхнім контролем LSB для $|I\rangle$ (тобто, $\left|C_i^0\right\rangle$) міняється місцями з кубітом повідомленням $|m_i\rangle$ для отримання стего-зображення $|I'\rangle$.

Схема вилучення показана на рис. 9, *б*, де $2n$ вентилів Адамара використовуються для перетворення початкового стану (тобто послідовності $|0\rangle$) у порожнє зображення. Аналогічно зі схемою вбудовування, коли інформація про положення $|I\rangle$ та $|M\rangle$ однакова, LSB для $|I\rangle$ міняється місцями з повідомленням кубіта $|m_i\rangle$ для отримання повідомлення $|M\rangle$.

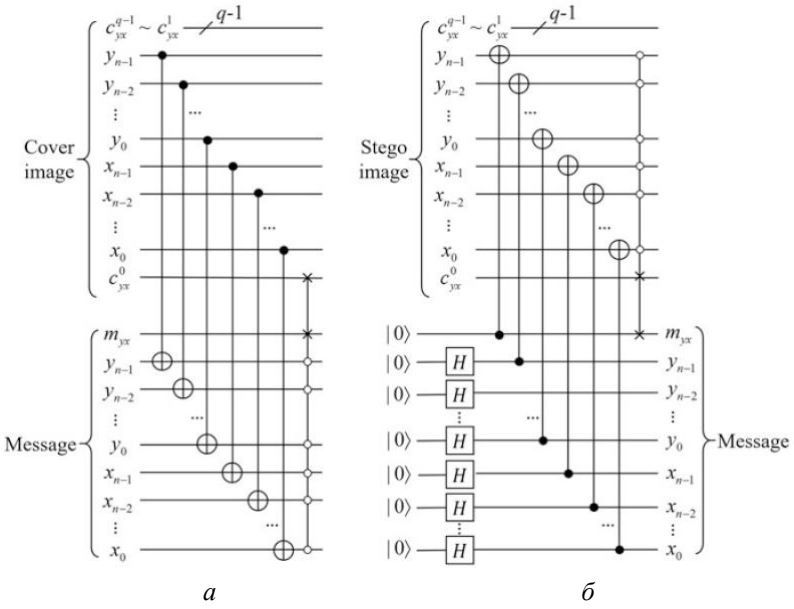

Рис. 9. Квантові схеми реалізації LSB стеганографії (*а* – вбудовування, *б* – вилучення)

Алгоритм блочної стеганографії LSB

Хоча алгоритм простої LSB-стеганографії простий, він має низьку стійкість [8–10]. Для покращення стійкості та непомітності схеми LSB, алгоритм блокової стеганографії розбиває зображення контейнера на блоки, в кожному з яких (замість кожного пікселя) ховається одне повідомлення довжиною в кубіт. Насправді, звичайну LSB-стеганографію можна розглядати як окремий випадок блокової LSB, в якій кожен блок вміщує лише один піксель.

Для реалізації схеми вбудовування та вилучення даних потрібно використовувати додатково квантовий лічильник та компаратор.

Квантовий лічильник.

Схему квантового лічильника [8] показано на рис. 10, де $|b\rangle$ – кубіт на вході і $b \in \{0,1\}$. $|a_{n-1}\ldots a_1 a_0\rangle$ – лічильник з початковим значенням $|00\ldots0\rangle$. Якщо вхідний кубіт $|b\rangle$ дорівнює $|1\rangle$, то $|a_{n-1}\ldots a_1 a_0\rangle$ збільшується на 1, інакше $|a_{n-1}\ldots a_1 a_0\rangle$ залишається без змін.

Квантовий компаратор.

Схему квантового компаратора [9] показано на рис. 11. Компаратор порівнює a та b, де $|a\rangle = |a_{n-1}\ldots a_1 a_0\rangle$ та $|b\rangle = |b_{n-1}\ldots b_1 b_0\rangle$, $a_i, b_i \in \{0,1\}$, $i = 0,1,\ldots,n-1$. Кубіти $|e_1\rangle$ та $|e_0\rangle$ є вихідними даними. Якщо $e_1 e_0 = 10$, то a > b; якщо $e_1 e_0 = 01$, то $a < b$; і якщо $e_1 e_0 = 00$, то $a = b$.

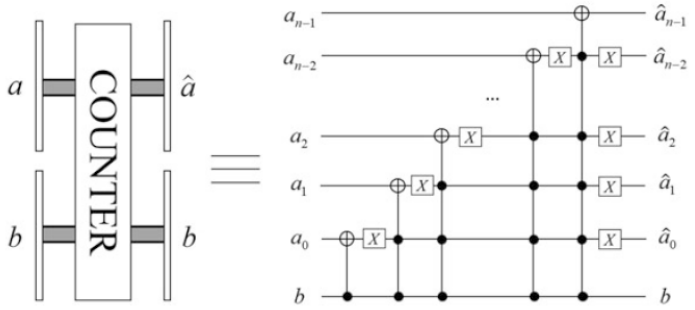

Рис. 10. Квантова схема реалізації лічильника

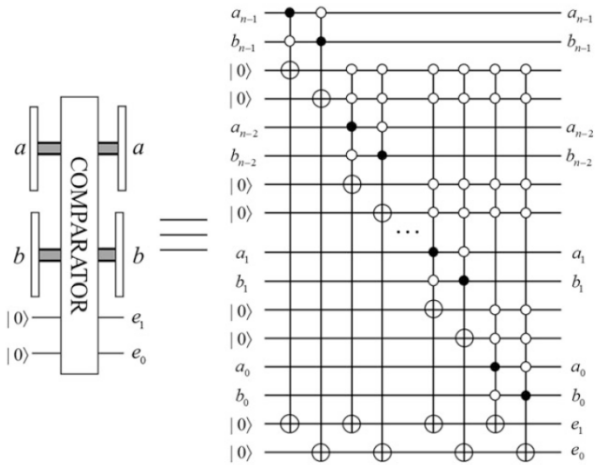

Рис. 11. Квантова схема реалізації компаратора

Процедура вбудовування блоків.

У блоковій схемі LSB зображення контейнеру $|I\rangle$ розміром $2^n \times 2^n$ має бути розбите на блоки $2^{n-p_1} \times 2^{n-p_2}$, де кожен блок має розмір $2^{p_1} \times 2^{p_2}$, де $p_1, p_2 \in \{0,1\}$. Блок зображення $|B\rangle$ можна визначити наступним чином:

$$\left|B_{k,l}\right\rangle = \frac{1}{2^n} \sum_{k=0}^{2^n-1} \sum_{l=0}^{2^n-1} \left|b_{k,l}\right\rangle \otimes |kl\rangle, \tag{7}$$

де $|k\rangle = \left|y_{n-1}, y_{n-2}, \ldots, y_{p_1}\right\rangle$, $|l\rangle = \left|x_{n-1}, x_{n-2}, \ldots, x_{p_2}\right\rangle$.

Припустимо, що повідомлення є двійковим квантовим зображенням, як показано в рівнянні (6). Його розмір $2^{n-p_1} \times 2^{n-p_2}$, а інформація про колір $p_1, m_{k,l} \in \{0,1\}$, де

$|k\rangle = |y_{n-1}, y_{n-2}, \ldots, y_{p_1}\rangle$ та $|I\rangle = |x_{n-1}, x_{n-2}, \ldots, x_{p_2}\rangle$. Процедура вбудовування полягає у наступному [7]:

К р о к 1 : Зображення контейнеру $|I\rangle$ зашифровано, щоб підвищити його непомітність у схемі. Для цього використовується квантовий алгоритм скремблювання зображень Гільберта.

К р о к 2 : Якщо інформація про положення $|y_{n-1}, y_{n-2}, \ldots, y_{p_1}\rangle$ $|x_{n-1}, x_{n-2}, \ldots, x_{p_2}\rangle$ в $|I\rangle$ дорівнює інформації $|M\rangle$, то операція вбудовування міняє місцями LSB $|I\rangle$ (тобто, $|C^0_{yx}\rangle$ і кубіт повідомлення $|m_{k,l}\rangle$.

К р о к 3 : Для відновлення зашифрованого зображення використовується обернене гільбертово скремблювання зображення.

Процедура вилучення блоків.

Після вбудовування кожного кубіта повідомлення 2^p разів (де $p = p_1 + p_2$) стегозображення може бути атаковано зловмисниками, що може змінити деякі LSB значення. Це призведе до того, що сума LSB всіх пікселів, що належать до одного блоку буде дорівнювати не 0 або 2^p, а якомусь значенню між ними.

Визначення того, чи дорівнює витягнутий кубіт повідомлення 0 або 1 відповідно до значення суми полегшується встановленням порогового значення. Якщо сума більша або дорівнює порогу, то кубіт повідомлення дорівнює 1, інакше він дорівнює 0. Процедура вилучення відбувається наступним чином [7]:

К р о к 1 : Повторює Крок 1 процедури вбудовування.

К р о к 2 : Схема керування використовується для розділення стегозображення на $2^{n-p_1} \times 2^{n-p_2} = 2^{n-p_1-p_2}$ блоків. Крім того, схема включає $2^{n-p_1-p_2}$ керуючих шарів, кожен з яких відповідає одному блоку зображення.

К р о к 3 : Квантові лічильники використовуються для підсумовування всіх LSB пікселів, які належать до одного блоку. Блок містить $2^{n-p_1-p_2}$ лічильників, а числа підрахунку представлені у вигляді $a_{y_{n-1}, y_{n-2}, \ldots, y_{p_1}\ x_{n-1}, x_{n-2}, \ldots, x_{p_1}}$.

К р о к 4 : Оскільки кожен блок складається з 2^p пікселів, число, отримане на кроці 3, слід порівняти з 2^{p-1}, що є порогом T, який встановлюється за допомогою квантового компаратора. Якщо $a_{y_{n-1}, y_{n-2}, \ldots, y_{p_1}\ x_{n-1}, x_{n-2}, \ldots, x_{p_1}} \geq 2^{p-1}$, то вилучене повідомлення дорівнює 1, інакше витягнуте повідомлення дорівнює 0.

П р и к л а д

Розглянемо просте зображення контейнеру розміром 4×4 та 8-бітне повідомлення 00110110, яке є ASCII-кодом символу "6" (див. рис. 12), зображення контейнеру розбивається на вісім блоків розміром 1×2 (у цьому випадку у рівнянні (7) $n = 2$, $p_1 = 0$, $p2 = 1$), і повідомлення перебудовується у двійкове зображення 4×2, як показано на рис. 12.

На рис. 13 показано схему вбудовування блоку LSB, яка складається з трьох частин, що відповідають трьом етапам, описаним вище. Перша та третя частини виконують гільбертове скремблювання зображення та його зворотну операцію.

Блок-схема вилучення LSB (на рис. 14) складається з чотирьох частин, які відповідають чотирьом етапам, описаним вище. Частина гільбертового скремблювання є такою ж, як і в операції вбудовування. Модуль розділення – це схема керування, яка визначає, який лічильник $C^0_{y_1 y_0 x_1 x_0}$ використовується. Наприклад, якщо керуючим значенням є 000, то $C^0_{000 x_0}$ міняється місцями з першим допоміжним кубітом $|0\rangle$, тобто входить у перший лічильник.

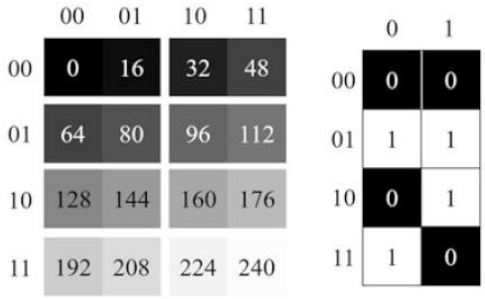

Рис. 12. Зображення-контейнер та повідомлення

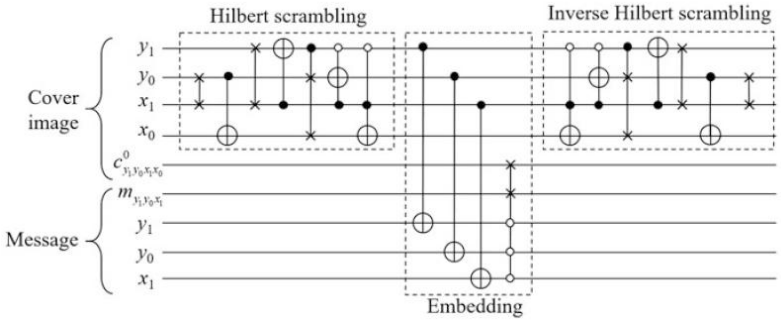

Рис. 13. Приклад блочного LSB вбудовування

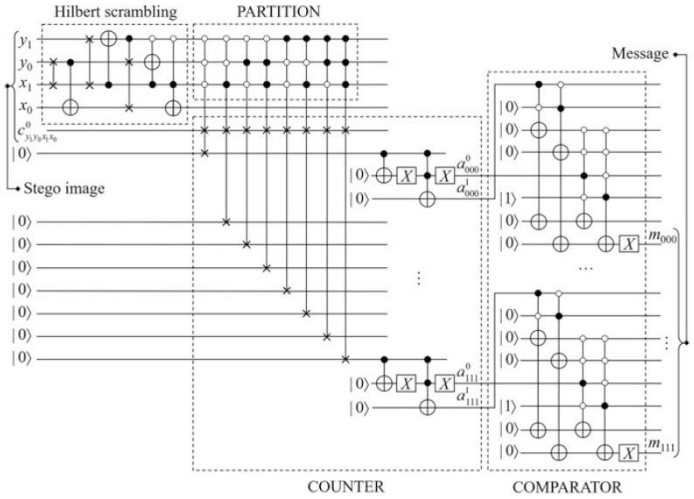

Рис. 14. Приклад вилучення даних для блочного LSB

Лічильний модуль складається з $2^{n-p_1} \times 2^{n-p_2} = 8$ лічильників, які відповідають восьми блокам. Кожен лічильник $a_{y_1y_0x_1}$ підсумовує LSB пікселів блоку $B_{y_1y_0x_1}$. Оскільки кожен блок має два пікселі, максимальне значення $a_{y_1y_0x_1}$ дорівнює 2, тому достатньо двох кубітів, тобто $\left|a_{y_1y_0x_1}\right\rangle = \left|a^1_{y_1y_0x_1} a^0_{y_1y_0x_1}\right\rangle$.

Крім того, частина порівняння містить $2^{n-p_1} \times 2^{n-p_2} = 8$ компараторів, які порівнюють $a_{y_1y_0x_1}$ з порогом $2^{p-1} = (01)_2$. Як показано вище, якщо $a_{y_1y_0x_1} > 01$, то молодші два кубіти кожного компаратора дорівнюють 10, і так далі. Отже, потрібно лише інвертувати нижній кубіт, щоб отримати кубіт повідомлення $\left|m_{y_1y_0x_1}\right\rangle$.

Висновки

Квантові водяні знаки – це область квантового приховування інформації, що швидко розвивається. Квантові зображення забезпечують міцну основу для цієї галузі. Різні моделі представлення квантових зображень мають різні переваги. Вибір відповідної моделі представлення квантових зображень для конкретних процесів квантової обробки зображень може суттєво вплинути на ефективність та результативність. В роботі ми представили декілька основних моделей квантових зображень, розглянули їх переваги та недоліки відносно використання в задачах стеганографії.

Детально зупинились на двох алгоритмах стеганографії на основі LSB для квантових зображень. Вони відрізняються тим, чи вбудовується кубіт повідомлення в піксель або блок контейнера-зображення. Обидва алгоритми є сліпими, тобто процедура вилучення не потребує оригінального зображення або оригінального повідомлення. Аналіз і моделювання на основі експериментальних результатів показують, що невидимість алгоритмів є гарною, і існує природний компроміс між їхньою пропускною здатністю та стійкістю.

Список літератури:

1. Le P. Q., Dong F. and Hirota K. A flexible representation of quantum images for polynomial preparation, image compression, and processing operations // Quantum Information Processing. 2010. Vol. 10. P. 63–84.
2. Venegas-Andraca S. and Bose S. Storing, processing, and retrieving an image using quantum mechanics // Proc. SPIE Conf. Quantum Information and Computation. 2003. P. 134–147.
3. Latorre J. I. Image Compression and Entanglement. [Електронний ресурс] Режим доступу: arXiv: quant-ph/0510031, 2005.
4. Zhang Y., Lu K., Gao Y. and Wang M. Neqr: a novel enhanced quantum representation of digital images // Quantum Information Processing. 2013. Vol. 12. P. 2833–2860.
5. Zulehner A., Wille R. Simulation and Design of Quantum Circuits // Ulidowski I., Lanese I., Schultz U.P., Ferreira C. (eds) Reversible Computation: Extending Horizons of Computing. Lecture Notes in Computer Science. 2020. Vol 12070. Springer, Cham. https://doi.org/10.1007/978-3-030-47361-7_3.
6. Gill S.S., Kumar A., Singh H., et al. Quantum computing: A taxonomy, systematic review and future directions // Softw: Pract Exper. 2022; 52(1):66-114. doi:10.1002/spe.3039.
7. Jiang N., Zhao N., Wang L. LSB based quantum image steganography algorithm // Theoret. Phys. 2016. 55(1). P.107–123.
8. Ma L., Lu J. Construction of controlled quantum counter // Chin. J. Quantum Electr. 2003. 20(1). P. 47–50.
9. Wang D., Liu Z., Zhu W., Li S. Design of quantum comparator based on extended general Toffoli gates with multiple targets // Comput. Sci. 2012. 39(9). P. 302–306.
10. Zhou RG., Luo J., Liu X. et al. A Novel Quantum Image Steganography Scheme Based on LSB // Theor Phys. 2018. 57. P.1848–1863. https://doi.org/10.1007/s10773-018-3710-x.

Надійшла до редколегії 10.09.2024

Відомості про авторів:

Федюшин Олександр Іванович – канд. техн. наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри безпеки інформаційних технологій, факультет комп'ютерної інженерії та управління; Україна; e-mail: oleksandr.fediushyn@nure.ua; ORCID: http://orcid.org/0000-0002-3600-405X

Головко Євген Вікторович – Харківський національний університет радіоелектроніки, аспірант кафедри безпеки інформаційних технологій; Україна; e-mail: yevhen.holovko1@nure.ua; ORCID: https://orcid.org/0009-0000-9684-7369

Смірнов Антон Олександрович – канд. техн. наук, Харківський національний університет радіоелектроніки, доцент кафедри безпеки інформаційних технологій, факультет комп'ютерної інженерії та управління; Україна; e-mail: anton.smirnov@nure.ua, ORCID: https://orcid.org/0000-0003-4121-3902

Сухотеплий Владислав Миколайович – Харківський національний університет Повітряних Сил імені Івана Кожедуба, старший викладач кафедри радіоелектронних систем пунктів управління Повітряних Сил, Україна; e-mail: vladislav181168@gmail.com; ORCID: https://orcid.org/0000-0002-2366-4167

Чечуй Олександр Вікторович – канд. техн. наук, доцент, Харківський національний університет Повітряних Сил імені Івана Кожедуба, доцент кафедри радіоелектронних систем пунктів управління Повітряних Сил; Україна; e-mail: alche1972@ukr.net; ORCID: https://orcid.org/0000-0002-7584-4457