Міністерство освіти і науки України
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

ЛАВРЕНЮК Юлія Олександрівна

«Алгоритми кодування інформації на основі заміни символьних пар / Algorithms for information encoding based on the character pairs replacement»

Студент групи КІзм – 21
ЛАВРЕНЮК Юлія Олександрівна
_____

Науковий керівник
к.т.н., Ю.М. Батько
_____

Тернопіль – 2020

РЕЗЮМЕ

Кваліфікаційна робота на тему “ Алгоритми кодування інформації на основі заміни символьних пар ” зі спеціальності 123 «Комп’ютерна інженерія» освітнього ступеня «магістр» написана обсягом 89 сторінок і містить 26 ілюстрації, 9 таблиць, 2 додатків та 51 джерел за переліком посилань.

Метою роботи є розробка алгоритму кодування інформації на основі заміни символьних пар.

Методи досліджень. Для розв’язання поставлених задач у кваліфікаційній роботі використано: методи: обробки цифрової інформації (для кодування інформації); математичної статистики (для оцінки рівня інформації); об’єктно-орієнтованого програмування (для проектування програмних засобів кодування/декодування цифрової інформації).

Результати дослідження: алгоритми кодування цифрової інформації на основі заміни пар символів, програмний модуль кодування та декодування цифрової інформації для систем обробки інформації.

Результати роботи можуть бути використані в створенні нових систем отримання, обробки, обміну та зберігання інформації, при проектуванні та розробці програмних складових систем безпеки, для наукових досліджень та в навчальному процесі.

Орієнтовні напрямки розвитку досліджень: розроблення програмних систем зберігання та передачі інформації в реальному часів, створення алгоритмів кодування даних без надлишкової інформації.

КЛЮЧОВІ СЛОВА: КОДУВАННЯ, ДЕКОДУВАННЯ, ІНФОРМАЦІЯ, ПРОГРАМНА СИСТЕМА.

RESUME

Graduate qualification work on “ Algorithms for information encoding based on the replacement pairs character ” specialty 123 - Computer Engineering is 89 pages long and contains 26 illustrations, 9 tables, 2 appendices and 51 references.

The aim of the work is to develop an algorithm for encoding information based on the replacement of character pairs.

Research methods. To solve the tasks in the qualification work used: methods: digital information processing (for encoding information); mathematical statistics (to assess the level of information); object-oriented programming (for designing software for encoding / decoding digital information).

Research results: algorithms for encoding digital information based on the replacement of symbol pairs, software module for encoding and decoding digital information for information processing systems.

The results of the work can be used in the creation of new systems for obtaining, processing, exchanging and storing information, in the design and development of software components of security systems, for research and in the educational process.

Approximate directions of research development: development of software systems for storage and transmission of information in real time, creation of data coding algorithms without redundant information.

KEYWORDS: CODING, DECODING, INFORMATION, SOFTWARE SYSTEM.

ЗМІСТ

ВСТУП

Актуальність роботи. У 2019 році обсяг глобального ринку програмного забезпечення для шифрування склав 2,98 млрд. Доларів США. За прогнозами, з 2020 по 2025 р. CAGR становитиме 16,8%. Проблеми безпеки даних зростають із зростанням тенденції інтернету речей та мобільних пристроїв серед підприємств. Це також призвело до збільшення кібератак, комерційного шпигунства, порушення даних та крадіжок та втрат у компаніях, що готується до ескалації необхідності захисту конфіденційних даних та забезпечення відповідності.

Досягнення мобільних технологій з точки зору апаратного та програмного забезпечення, їх розповсюдження серед підприємств та зростаюче проникнення смартфонів, швидше за все, спровокують попит на програмне забезпечення для шифрування до 2025 року. Зростаюче проникнення мобільних пристроїв в організації збільшує ризик втрати даних серед підприємств, що зробило виконання програмного забезпечення для шифрування необхідним для безпечної передачі даних. Крім того, оскільки підприємства все частіше рухаються до хмарних обчислень, зростає потреба в захисті конфіденційних даних, що призводить до більшого розгортання програмного забезпечення.

Кілька галузевих галузей, таких як охорона здоров'я, повинні дотримуватися суворих норм, що вимагає впровадження рішення щодо захисту даних. Це означає більший попит на програмне забезпечення для шифрування по всьому світу.

Крім того, швидка оцифровка та зростання використання Інтернету, інтелектуальної власності підприємств та користувачів стали сприйнятливі до крадіжок та порушень. Завдяки цим ризикам безпеки компанії зобов’язані впроваджувати рішення для захисту даних. Програмне забезпечення для шифрування дозволяє організаціям захищати свою інтелектуальну власність та

інші конфіденційні дані, що, як очікується, збільшить її попит протягом прогнозованого періоду.

Північна Америка домінувала на ринку в 2019 році. Наявність добре налагодженого IT та телекомунікаційного сектору в регіоні та створення величезного обсягу даних, які необхідно захищати, сприяють домінуванню в регіоні. Очікується, що раптовий сплеск зашифрованого інтернет-трафіку через прийняття HTTPS провідними компаніями, включаючи Facebook, Twitter та Netflix, спрацює на користь ринку протягом прогнозованого періоду.

Як тільки зловмисники отримують доступ до мережі або даних під час транзиту, найкращим способом захисту захищеної конфіденційної інформації є її нерозбірливість. Це робиться за допомогою програмного забезпечення для шифрування, яке захищає інформацію, що зберігається, отримується та надсилається. Доступ до даних, захищених шифруванням, можна отримати лише за допомогою пароля, додавши додатковий життєво важливий рівень безпеки. Тому задача розробки алгоритму кодування інформації на основі заміни символьних пар є актуальною.

Метою роботи є розробка алгоритму кодування інформації на основі заміни символьних пар.

Для досягнення даної мети ставились наступні завдання:

− провести огляд видів інформаціїї в системах обробки інформації, її властивості та сфери використання;

− провести класифікацію інформаційних систем обробки даних;

− провести дослідження існуючих програмних кодування/декодування інформації;

− проаналізувати існуючі методи та алгоритми кодування/декодування інформації;

− розробити алгоритм кодування інформації на основі аміни символьних пар;

− реалізувати програмну систему кодування інформації, провести її тестування та порівняти з програмами аналогами.

Об’єкт дослідження – процес обробки та аналізу цифрових даних.

Предмет дослідження – методи і алгоритми кодування/декодування даних в системах обробки та зберігання інформації.

Наукова новизна одержаних результатів визначається наступним чином:

– проведено комплексний аналіз та класифікацію алгоритмів кодування/декодування інформації, що дозлолило виділити їх переваги та недоліки та розробити власний алгоритм кодування інформації на основі заміни символьнихпар;

– розроблено алгоритм алгоритм кодування інформації на основі заміни символьнихпар, що дозволило зменшити обчислювальну складність процесу обробки інформації.

Практична цінність одержаних результатів полягає в тому, що:

– розроблено та проведено моделювання програмної системи обробки інформації з використанням елементів теорії кодування, що дозволило в подальшому програмно реалізувати та провести дослідження запропонованих алгоритмів;

– реалізовано програмне забезпечення для кодування/декодування цифрової інформації на основі мови програмування високого рівня та з використанням алгоритмів кодування даних.

Публікації та апробація до випускної кваліфікаційної роботи. За результатами наукових досліджень, проведених у випускній кваліфікаційні роботі, підготовлено тези [1,2] доповіді «Аналіз алгоритмів кодування інформації в системах обробки даних» обсягом 1 сторінка на ііі науково-практичній конференції молодих вчених і студентів «інтелектуальні комп’ютерні системи та мережі», а також «Аналіз алгоритмів розпізнавання цифрової інформації на зображеннях» обсягом 1 сторінка на ііі науково-практичній конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі».

1 ПРОГРАМНІ, АПАРАТНІ ТА ГІБРИДНІ СИСТЕМИ ОБРОБКИ ЦИФРОВОЇ ІНФОРМАЦІЇ

1.1 Поняття інформації, основні поняття та функції

При дослідженні або спостереженні будь-якого фізичного процесу, об'єкта або явища можна ввести поняття інформації. Існує досить багато визначень поняття інформація, що істотно розрізняються в залежності від сфери діяльності в застосуванні до якої дається конкретне визначення цього поняття. Такий стан, при якому існує безліч різних визначень і відсутня єдина загальноприйнята і точне визначення цього поняття, випливає з того, що поняття інформації, поряд з такими фундаментальними поняттями як матерія і енергія у фізиці або як безліч або точка в математики, є первинним поняттям і тому не може мати чіткого формалізованого визначення, тобто не може бути визначено через більш прості відомі об'єкти, що мають чіткі визначення. Тому при визначенні основних фундаментальних первинних понять використовують інтуїтивний підхід, визначаючи поняття через сукупність властивих йому властивостей. Дійсно, виходячи з практичного досвіду, інтуїтивно, можна представити інформацію як сукупність змістовних відомостей, укладених в тому або інший об'єкт або явище, і перерахувати її основні властивості, тим самим інтуїтивно, виходячи з цих властивостей, її визначити.

Основні властивості інформації.

1. Інформація приносить знання про навколишній світ, яких в даній точці простору не було до її отримання.

2. Інформація не матерія, а властивість організованої матерії. Інформація таке ж невід'ємна властивість матерії як маса і енергія, однак, на відміну від них вона не підпорядковується законам збереження, подібним законам збереження маси і енергії.

3. Інформація не матеріальна, але вона проявляється у вигляді матеріальних носіїв - символів і сигналів. Причому символи - це реальні

помітні одержувачем матеріальні об'єкти (букви, цифри, зображення), а сигнали - це динамічні процеси, тобто змінюються в часі або просторі значення будь-якого фізичного величини.

4. Інформація може бути укладена як в самих символах, так і в їх взаємне розташування (наприклад, символи Т, Р, С, О можуть принести інформацію: торс, сорт, трос тощо).

Символи і сигнали несуть інформацію тільки для одержувача, здатного їх розпізнати, тобто поставити у відповідність прийнятим символам і сигналам об'єкти реального світу і їх відносини.

Окреме питання це представлення та зберігання інформації в системах обрбки даних, огляд основних моделей наведено на рисунку 1.1.

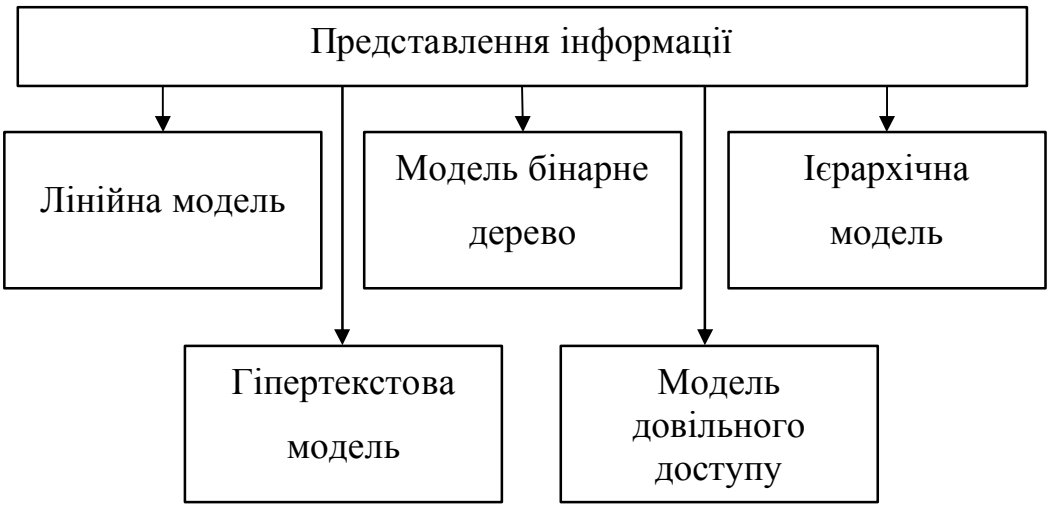

Рисунок 1.1 – Моделі представлення інформації

Виходячи з перерахованих вище властивостей, інформацію можна розглядати як відомості (знання), отримані в результаті моделювання (опису) реального світу або його досліджуваної частини, які є об'єктом деяких операцій: передачі, розподілу, перетворення, зберігання або безпосереднього використання.

Інформація міститься в повідомленнях, які генеруються джерелами повідомлень.

Джерело повідомлень - будь-який процес, об'єкт або явище, який мають здатністю змінювати свій стан у часі або в просторі.

Для практичного використання поняття інформації необхідно навчитися її вимірювати. Це можна зробити за аналогією з методикою чисельного вимірювання інших фундаментальних понять, таких як матерія (маса), енергія або простір, для цього потрібно встановити, що приймається за міру кількісної оцінки інформації і що приймати за одиницю виміру цього заходу. Під мірою кількісної оцінки розуміють деякий явище або об'єкт, які однозначно (пропорційно) пов'язані з визначеним первинним поняттям і які можуть характеризувати кількісний вміст цього поняття.

Традиційно склалося три основні підходи до вибору міри кількісної оцінки інформації (рисунок 1.2). Дані підходи в начній мірі групують відомі на сьогоднішній день алгоритми та дозволяють отримати оцінку деяку інформції, в тому числі і кулькісно для роботи з цифровими системами.

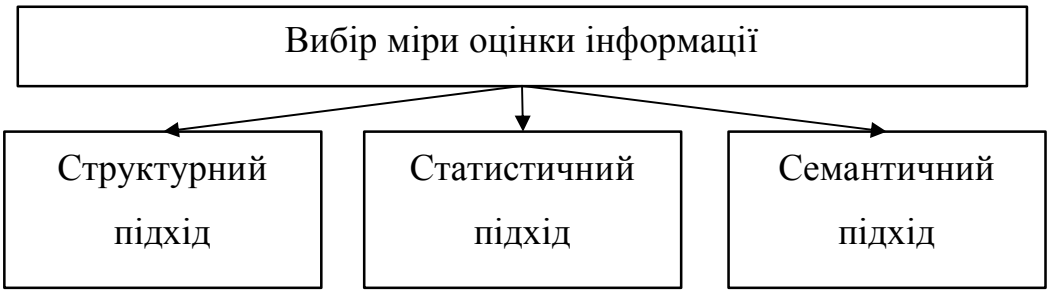

Рисунок 1.2 – Класивікація підходів до оцінки інформації

1. Структурний підхід, при якому кількісна оцінка інформації про подію оцінюється шляхом визначення об'єктивної можливості цієї події, що входить в деяку повну групу подій.

2. Статистичний підхід, при якому кількісна оцінка інформації про прийняте сполученні провадиться на основі міри невизначеності, що знімається з досліджуваного інформаційного процесу (події) при отриманні даного повідомлення.

3. Семантичний підхід, який в основному враховує цінність отриманої інформації з точки зору конкретного одержувача цієї інформації.

Очевидно, що, для точного, технічного, об'єктивного використання, семантичний підхід не прийнятний, так як він суто суб'єктивний і не може дати загальноприйнятою кількісної міри інформації, хоча цей підхід і може бути використаний в сфері гуманітарних і суспільних наук.

Стосовно до точних і технічних наук для визначення міри кількісної оцінки інформації використовують структурний і статистичний підходи.

Вибір критерію для кількісної оцінки інформації, незалежно від обраного півходу, повинен відповідати умовам, що випливають з практичного досвіду:

− повідомленню більшої довжини (при одному і тому ж обсязі алфавіту) відповідає більшу кількість інформації;

− більшу кількість інформації міститься в тих повідомленнях (однакової довжини), які складені з символів більшого алфавіту;

− символи в повідомленні можуть з'являтися з різними можливостями і можуть бути статистично залежними.

З огляду на це, міру кількісної оцінки інформації можна ввести виходячи з наступних міркувань. Припустимо, що якась подія має m рівно можливих результатів, наприклад, поява будь-якого символу з алфавіту, що містить m таких символів. Виміряти кількість інформації, що міститься в повідомленні з n таких символів можна, визначивши число N всіх можливих повідомлень, які можуть бути складені з символів цього алфавіту. Якщо повідомлення формується з одного символу, то $N=m$, якщо з двох, то $2*N = m*m = m^2$, якщо з n символів, то $N=m^n$. Отриману міру кількісної оцінки інформації можна розуміти як міру невизначеності отримання конкретного заданого повідомлення, що складається з n символів алфавіту. Однак цей захід кількісної оцінки інформації не зовсім зручна. Дійсно, при m=1 (тобто алфавіт складається з одного символу) невизначеності не існує і поява цього символу не несе ніякої інформації, проте значення N в цьому випадку (N=1n) Не звертається до нуль. Крім цього, з практичних міркувань, доцільно вважати, що кількість

інформації, отримане від двох незалежних джерел, дорівнює сумі кількостей інформації, одержуваних від кожного джерела, а пропонована міра кількості інформації дає в цьому випадку значенн

$$N=m^{n1+n2}+m^{n1}*m^{n2}=N_1*N_2$$

де N_1, N_2 - число можливих повідомлень від двох джерел повідомлень.

Ці незручності легко переборні, якщо в якості запобіжного кількісної оцінки інформації (I) взяти логарифм по якого-небудь підстави від загального числа можливих повідомлень (N).

При описі інформаційних процесів часто користуються такими інформаційними характеристиками.

Продуктивність джерела повідомлень (Р) - середня кількість інформації, генерується джерелом в одиницю часу $P=I/t$.

Одиницею виміру продуктивності джерела повідомлень є бот. Інформаційна ємність повідомлення (R) - це середня кількість інформації, що міститься в повідомленні одиничної тривалості:

$$R = \frac{I}{t},$$

де I – кількість інформації, що міститься в повідомленні;

t - тривалість повідомлення.

Швидкість створення (генерації) інформації (Q) - це середня кількість інформації, генерується джерелом повідомлень за одиницю часу:

$$Q = W * H,$$

де W – швидкість передачі символів (символ / сек);

H – середня кількість інформації припадає на один символ (біт/символ).

Одиницею вимірювання швидкості генерації інформації та інформаційної ємності повідомлення служить бот. Як приклад, визначимо інформаційну ємність найпростішого дискретного повідомлення, що складається із сукупності імпульсів і пауз, причому тривалість імпульсів і пауз однакова, а амплітуда імпульсів постійна, тобто повідомлення будується на використанні лише двох символів (його обсяг алфавіт дорівнює 2).

Очевидно, якщо надмірність відсутня, то на кожен символ такого повідомлення доводиться одна біт (1 біт), а на все повідомлення, що складається з n символів, доводиться n біт інформації.

Розумною мірою інформації, що міститься в повідомленні є міра, монотонно пов'язана з витратами на передачу повідомлення.

Припустимо, що повідомлення є деякі випадкові події. Розглянемо в якості джерела довільне дискретну множину X і кожній букві X поставимо у відповідність ймовірність $p(x)$.

Сформулюємо вимоги до деякій мірі $\mu(x)$, визначеної для всіх X, яку слід прийняти як міру інформації, що міститься в повідомленнях ансамблю $X=\{x,p(x)\}$.

Оскільки передбачається, що цей захід буде визначати витрати, пов'язані з передачею або зберіганням повідомлень, міра повинна бути невідємною.

Оскільки для нас несуттєво, яким чином будуть інтерпретовані і використані передані повідомлення, міра повинна однозначно визначатися ймовірністю повідомлення. Тому замість $\mu(x)$ будемо писати $\mu(p(x))$.

Встановимо характер залежності заходи від ймовірності повідомлень. Нехай є безліч з двох рівноймовірно повідомлень і стоїть завдання кодування повідомлень безлічі двійковими символами алфавіту $A=\{0, 1\}$.

Різні методи кодування широко використовуються в практичній діяльності людини з незапам'ятних часів. Наприклад, десяткова позиційна система числення – це спосіб кодування натуральних чисел. Інший спосіб кодування натуральних чисел - римські цифри, причому цей метод більш наочний і природний, дійсно, палець - I, п'ятірня - V, дві п'ятірні - X. Однак при

цьому способі кодування важче виконувати арифметичні операції над великими числами, тому він був витіснений способом кодування заснованому на позиційної десяткової системою числення. З цього прикладу можна зробити висновок, що різні способи кодування мають властивими тільки їм специфічними особливостями, які в залежності від цілей кодування можуть бути як гідністю конкретного способу кодування, так і його недоліком.

Широко відомі способи числового кодування геометричних об'єктів і їх положення в просторі: декартові координати і полярні координати. І ці способи кодування відрізняються притаманними їм специфічними особливостями. До XX століття методи і засоби кодування грали допоміжну роль, але з появою комп'ютерів ситуація радикально змінилася. Кодування знаходить найширше застосування в інформаційних технологіях і часто є центральним питанням при вирішенні найрізноманітніших завдань таких як:

− подання даних довільної природи (чисел, тексту, графіки) в пам'яті комп'ютера;

− оптимальна передача даних по каналах зв'язку;

− захист інформації (повідомлень) від несанкціонованого доступу;

− забезпечення завадостійкості при передачі даних по каналах зв'язку;

− стиснення інформації.

З точки зору теорії інформації кодування - це процес однозначного зіставлення алфавіту джерела повідомлення і деякої сукупності умовних символів, здійснюване за певним правилом, а код (кодовий алфавіт) - це повна сукупність (безліч) різних умовних знаків (символів коду), які можуть використовуватися для кодування вихідного повідомлення і які можливі при даному правилі кодування. Число ж різних кодових символів складових кодовий алфавіт називають об'ємом коду або об'ємом кодового алфавіту. Очевидно, що обсяг кодового алфавіту не може бути менше обсягу алфавіту кодованого вихідного повідомлення. Таким чином, кодування - це перетворення вихідного повідомлення в сукупність або послідовність кодових символів, що відображають повідомлення, передане по каналу зв'язку.

Кодування може бути числовим (цифровим) і нечислове, в залежності від виду, в якому представлені кодові символи: числа в будь-якій системі числення або інші якісь об'єкти або знаки відповідно.

У більшості випадків кодові символи являють собою сукупність або послідовність деяких найпростіших складових, наприклад, послідовність цифр в кодових символах числового коду, які називаються елементами кодового символу.

Місцезнаходження або порядковий номер елемента в кодовому слові визначається його позицією. Число елементів символу коду, що використовується для представлення одного символу алфавіту вихідного джерела повідомлень, називають значности коду. Якщо значность коду однакова для всіх символів алфавіту вихідного повідомлення, то код називають рівномірним, в іншому випадку - нерівномірним. Число елементів входять до кодовий символ іноді називають довжиною кодового символу.

З точки зору надмірності всі коди можна розділити на ненадлишкових коди і надлишкові. У надлишкових кодах число елементів кодових символів може бути скорочено за рахунок більш ефективного використання елементів, що залишилися, в ненадлишкових же кодах скорочення числа елементів в кодових символах неможливо. Завдання кодування при відсутності перешкод і при їх наявності істотно різні.

Тому розрізняють ефективне (статистичне) кодування і коригуючий кодування. При ефективному кодуванні ставиться завдання добитися представлення символів алфавіту джерела повідомлень мінімальним числом елементів кодових символів в середньому на один символ алфавіту джерела повідомлень за рахунок зменшення надмірності коду, що веде до підвищення швидкості передачі повідомлення. А при корекції (перешкодостійкі) в кодуванні ставиться завдання зниження ймовірності помилок в передачі символів вихідного алфавіту шляхом виявлення та виправлення помилок за рахунок введення додаткової надмірності коду.

Окремо стоїть завданням кодування є захист повідомлень від несанкціонованого доступу, спотворення і знищення їх. При цьому виді кодування кодування повідомлень здійснюється таким чином, щоб навіть отримавши їх, зловмисник не зміг би їх розкодувати. Процес такого виду кодування повідомлень називається шифруванням, а процес декодування – розшифрування. Саме кодоване повідомлення називають шифрованим, а застосовуваний метод кодування - шифром.

Досить часто в окремий клас виділяють методи кодування, які дозволяють побудувати коди повідомлень, що мають меншу довжину в порівнянні з вихідним повідомленням. Такі методи кодування називають методами стиснення або упаковки даних. Якість стиснення визначається коефіцієнтом стиснення, який зазвичай вимірюється в відсотках і який показує на скільки відсотків кодоване повідомлення коротше вихідного.

При автоматичній обробці інформації з використанням персонального комп’ютера як правило використовують числове (цифрове) кодування, при цьому, природно, виникає питання обгрунтування використовуваної системи числення. Дійсно, при зменшенні підстави системи числення спрощується алфавіт елементів символів коду, але відбувається подовження символів коду. З іншого боку, чим більше підставу системи числення, тим менше число розрядів потрібно для представлення одного символу коду, а, отже, і менше час для його передачі, але з ростом підстави системи числення істотно підвищуються вимоги до каналів зв'язку і технічних засобів розпізнавання елементарних сигналів, що відповідають різним елементам символів коду. Зокрема, код числа, записаного в двійковій системі числення в середньому приблизно в 3,5 рази довше десяткового коду. Так як у всіх системах обробки інформації доводиться зберігати великі інформаційні масиви у вигляді числової інформації, то одним з істотних критеріїв вибору алфавіту елементів символів числового коду (тобто підстави використовуваної системи числення) є мінімізація кількості електронних елементів в пристроях зберігання, а також їх простота і надійність.

При визначенні кількості електронних елементів необхідних для фіксації кожного з елементів символів коду необхідно виходити з практично виправданого припущення, що для цього потрібно кількість найпростіших електронних елементів (наприклад, транзисторів), рівне основи системи числення *a*. Тоді для зберігання в деякому пристрої *n* елементів символів коду потрібно *M* електронних елементів:

$$M=a*n.$$

Найбільша кількість різних чисел, яке може бути записано в цьому пристрої *N*:

$$N=a^n.$$

На основі проведеного аналізу можна зробити висновки, що в сучасному світі інформація має велике значення та цінність, а отже для її опрацювання , зберігання та передачі необхідно розробляти нові методи, алгоритми та програмні системи.

1.2 Інформційні системи, класифікація та сфери застосування

У сучасний інформаційно-комунікаційний вік постійно згадуються інформаційні системи та управління інформаційними системами. У цифрову епоху дані, зберігання та пошук здійснюються за допомогою різних систем та інтерфейсів.

Інформаційна система. Отже, інформаційну систему можна визначити як сукупність скоординованої мережі компонентів, які діють спільно на виробництво, розповсюдження та обробку інформації. Важливим фактором

комп'ютерної інформаційної системи є точність, яка може не стосуватися інших типів систем.

У системі мережа компонентів працює задля єдиної мети, якщо відсутність координації між компонентами, це призводить до контрпродуктивних результатів. Система може мати такі особливості.

Адаптованість: деякі системи є адаптивними до зовнішнього середовища, тоді як деякі системи не адаптуються до зовнішнього середовища. Наприклад, антиблокувальна гальмівна система в автомобілі реагує залежно від дорожніх умов, де як музична система в машині не залежить від інших подій, що відбуваються з автомобілем.

Обмеження: кожна система має заздалегідь визначені межі або межі, в яких вона функціонує. Ці межі або межі можуть бути визначені законом або сучасним станом техніки.

Загальним визначенням інформації є дані. Однак дані не є справжньою інформацією. Дані отримують своє значення і значення, якщо тільки це інформація. Інформація представлена даними, символами та літерами.

Однією з ключових властивостей інформації є її об’єктивність. Об'єктивна інформація є ключовою складовою будь-якого сучасного наукового дослідження.

Набір інформації, яка є корисною для науки, може бути абстрактною або неактуальною для інших. Тому інформація також є суб'єктивною.

Іінформація є тимчасовою при кожному оновленні бази даних.

Інформація представлена за допомогою даних, цифр, букв або символів. Інформація сприймається так, як вона представляється. Десяткова система та двійкова система - це два способи подання інформації. Бінарні схеми комп'ютерів призначені для роботи в двох станах (0,1).

Спосіб організації інформації безпосередньо впливає на спосіб управління та отримання інформації. Найпростіший спосіб організації інформації – це лінійна модель. У цій формі дані структуровані одна за одною, наприклад, на магнітних стрічках, музичних стрічках тощо.

У моделі бінарних дерев дані розташовуються у перевернутому форматі дерева, де вони приймають два значення.

Модель ієрархії отримана з бінарної моделі дерева. У цій моделі гілка може приймати багатозначні дані, наприклад, в операційній системі UNIX ця модель використовується для своєї файлової системи.

Модель гіпертексту інший спосіб організації інформації; Всесвітня павутина є прикладом цієї моделі.

Модель довільного доступу - ще один спосіб організації інформації. Ця модель використовується для оптимального використання доступного місця для зберігання даних на комп'ютері. Тут дані зберігаються у вказаному місці під керівництвом операційної системи.

Безпека інформації, а також інформаційна система є критично важливою. Резервне копіювання даних відбувається на шляху захисту інформації. Управління безпекою мережі та інформаційної системи відрізняється різними установками, такими як дім, малий бізнес, середній бізнес, великий бізнес, школа та уряд.

Інформаційна система - це інтегрована та координована мережа компонентів, які поєднуються разом, перетворюючи дані в інформацію.

Складові інформаційних систем. Інформаційна система, по суті, складається з п'яти компонентів:

- апаратного забезпечення;
- програмного забезпечення;
- бази даних;
- мережі;
- люди.

Ці п'ять компонентів інтегруються для здійснення введення, обробки, виведення, зворотного зв'язку та управління.

Апаратне забезпечення складається з пристрою введення/виведення, процесора, операційної системи та медіа-пристроїв. Програмне забезпечення складається з різних програм та процедур. База даних складається з даних,

упорядкованих у необхідну структуру. Мережа складається з концентраторів, засобів зв'язку та мережевих пристроїв. Люди складаються з операторів пристроїв, адміністраторів мережі та системних спеціалістів.

Обробка інформації складається з введення; обробка даних, зберігання, виведення та контроль даних. На етапі введення інструкції даних надходять до систем, над якими на етапі процесу працюють програмні програми та інші запити. На вихідному етапі дані подаються у структурованому форматі та звітах.

Класифікація інформаційних системи. У будь-якій даній організації інформаційна система може бути класифікована на основі використання інформації. Отже, інформаційну систему в організації можна розділити на систему підтримки операцій та систему підтримки управління.

Система підтримки операцій. В організації введення даних здійснюється кінцевим користувачем, який обробляється для формування інформаційних продуктів, тобто звітів, які використовуються внутрішніми або зовнішніми користувачами. Така система називається системою підтримки операцій.

Мета системи підтримки операцій - полегшити ділові операції, контролювати виробництво, підтримку внутрішньої, а також зовнішньої комунікації та оновлення центральної бази даних організації. Система підтримки операцій далі поділяється на систему обробки транзакцій, систему управління обробкою та систему співпраці підприємств.

Система обробки транзакцій (TPS). В організації виробництва існує кілька видів операцій між відділами. Типовими організаційними підрозділами є продаж, бухгалтерія, фінанси, завод, інжиніринг, людські ресурси та маркетинг. Протягом якої наступної операції може відбутися замовлення на продаж, повернення продажів, надходження грошових коштів, продаж кредитів; кредитні білети, облік матеріалів, управління запасами, облік амортизації тощо.

Ці транзакції можна класифікувати за пакетною обробкою, обробкою одиничної транзакції та обробкою транзакцій у реальному часі.

Система управління процесами. У виробничій організації певні рішення приймаються комп’ютерною системою без будь-якого ручного втручання. У цьому типі системи критична інформація подається до системи в режимі реального часу, що дозволяє керувати процесом. Цей тип систем називають системами управління процесами.

Система співпраці підприємств. Останнім часом спостерігається більший наголос на колективні зусилля чи співпрацю між різними функціональними командами. Система, що забезпечує спільні зусилля шляхом поліпшення зв'язку та обміну даними, називається системою корпоративної співпраці.

Система підтримки прийняття рішень. Менеджерам потрібна точна інформація у певному форматі для прийняття організаційного рішення. Система, що полегшує ефективний процес прийняття рішень менеджерами, називається системою підтримки управління.

Системи підтримки прийняття рішень по суті класифікуються як інформаційна система управління, система підтримки прийняття рішень, експертна система та інформаційна система бухгалтерського обліку.

Управлінська інформаційна система надає інформацію керівнику, полегшуючи рутинний процес прийняття рішень. Система підтримки прийняття рішень надає інформацію керівнику, сприяючи вирішенню конкретних проблем.

Інформаційну систему можна класифікувати на основі діяльності за системою стратегічного планування, тактичною інформаційною системою та оперативною інформаційною системою.

Застосування концепції, згідно з якою інформаційна система знаходиться в суворій компетенції IT-відділу, може призвести до негативної ситуації для компанії. Тому організації важливо визнати внесок інформаційних систем в ефективність бізнесу.

Розвиток інформаційних систем приніс можливості, але й загрози. Організація покладається на виявлення можливостей та їх реалізацію. Організація повинна розробити стратегії, які можуть найкраще

використовувати інформаційні системи для підвищення загальної продуктивності.

Найбільш поширеною практикою стосовно інформаційних систем є автоматизація. Хоча автоматизація є корисною, інновації, що використовують інформаційні системи, дають організації конкурентну перевагу.

Організації цілком усвідомлюють, що розповсюдження інформаційних систем зменшило життєвий цикл товару, зменшило націнку та привело нові товари. За такого сценарію задоволення споживачів одного буде недостатньо, організація повинна прагнути до задоволення клієнтів. Інформаційні системи з можливістю зберігання даних та аналітики можуть допомогти організації збирати відгуки клієнтів та розробляти продукти, що перевищують очікування споживачів. Це задоволення клієнтів призведе до лояльної клієнтської бази та посла бренду.

Організаціям потрібні різні типи інформаційних систем, щоб пом'якшити відмінні процеси та вимоги. Ефективні системи ділових операцій роблять організацію продуктивною. Системи ділових операцій гарантують, що рутинний процес фіксується та ефективно діє, наприклад, транзакція продажу, готівкова операція, нарахування заробітної плати тощо.

Крім того, для прийняття виконавчого рішення необхідні інформаційні системи. Для вироблення стратегії організації вищому керівництву потрібна точна внутрішня та зовнішня інформація. Системи підтримки прийняття рішень призначені для виконання саме цієї функції.

Завдяки впровадженню електронної пошти, відеоконференцій та спільної роботи в дошках серед організацій та підрозділів зросло. Це посилене співробітництво забезпечує безперебійне виконання та реалізацію різних проектів у різних регіонах та регіонах.

Проаналізувавши сучасні інформаційні системи можна зробити висновок, що вони широко використовуються у всіх галузях і однією з головних задач таки систем є збір та обробка інформації, при цьому вагому частку операцій займає саме передача інформації, для якої необхідне кодування.

1.3 Програмні системи для кодування/декодування даних

Програмне забезпечення для шифрування шифрує дані, що читаються, використовуючи алгоритми в ключах шифрування, і перетворює їх на закодовану інформацію. Відкриті ключі шифрування використовуються для початкового скрембування та захисту інформації. Потім приватний ключ, який утримується уповноваженим користувачем, використовується для дешифрування даних і повернення їх до читабельного формату. Сучасні ключі шифрування дотримуються Advanced Encryption Standard (AES), який використовує 128-бітові та 256-бітові довжини ключів, які є надзвичайно довгими рядками чисел, для скремблювання інформації. У деяких випадках використовується навіть 4096-бітна довжина ключа.

Шифрування допомагає забезпечити збереження та передачу даних (у спокої та в русі) у безпеці та не розшифровці, але врешті-решт до цієї інформації потрібно отримати доступ. Це відкриває вікно для хакерів, щоб знайти та вкрасти цю інформацію. Гомоморфне шифрування було розроблено, щоб дозволити обчислення зашифрованих даних, що використовуються, тому воно залишається конфіденційним, поки можна виконувати деякі завдання. Це може бути корисним для додаткової безпеки, але не всі завдання можна виконати під час роботи з гомоморфно зашифрованими даними.

Усі організації повинні зберігати та передавати дані, такі як особиста інформація (ФІО) або фінансові дані. Особливо це стосується величезних обсягів даних, якими керують корпоративні організації. Шифрування є критично важливим для захисту інформації, яка виявляється, коли інше програмне забезпечення безпеки на передовій не працює.

Шифрування даних не тільки допомагає захищати конфіденційну інформацію, але й допомагає зменшити ймовірність дорогих судових зборів та репутації організації. Без належних заходів безпеки, включаючи шифрування,

організації ризикують потрапити під загрозу порушення правил конфіденційності даних.

Важливо знати різницю між шифруванням файлів та шифруванням на повному диску, щоб уникнути створення критичних дир у інфраструктурі безпеки. Повнодискове шифрування корисно для захисту окремих пристроїв. Його випадки використання обмежені, оскільки він не може зашифрувати дані, що надсилаються або отримуються цим пристроєм.

Шифрування файлів - більш комплексне рішення. Він може шифрувати всі окремі файли та фрагменти даних, що зберігаються на пристрої або на сервері, а також шифрувати дані в дорозі.

Шифрування на повному диску та шифрування файлів мають свої програми. Що найкраще для вашої організації, залежить від ваших потреб у безпеці. Цей список включає обидва типи продуктів шифрування.

Є кілька ключових особливостей, на які слід звернути увагу при покупці рішення для шифрування. Показники надійності пароля повинні бути пріоритетними. Занадто багато співробітників використовують однакові прості, легко запам'ятовуються паролі майже для всього. Індикатори надійності пароля допоможуть зменшити будь-які вразливості, спричинені слабкими паролями.

Можливості управління паролями також можуть допомогти в цьому, надійно зберігаючи та вводячи паролі автоматично, так що кожен працівник може мати довгі, складні паролі, не потребуючи їх запам'ятовувати. Крім того, шукайте функції віртуального подрібнення документів. Це гарантує, що будь-які дані, які видаляються, насправді викорінюються і не можуть бути викреслені з вашого диска.

Сучасні алгоритми шифрування вимагають стільки обробної потужності, щоб зламатися, що вони практично захищені від хакерів. Але з появою потужних квантових обчислень це може бути вже не так.

Квантова криптографія, яку також називають квантовим шифруванням, застосовує принципи рідинних станів квантової механіки для вирішення

набагато більшої кількості проблем з однаковою швидкістю обробки, щоб не відставати від хакерів, що використовують квантові обчислення.

В таблиці 1.1 наведено перелік сучасних програмних засобів для кодування/декодування інформації.

Таблиця 1.1 – Порівняння програмних засобів кодування інформації

Назва додатка	Шифрування диска	Шифр-ня файлів	Ключові особливості	Розгортання	Ціна
IBM Guardium Data Encryption	Так	Так	Можливості, що відповідають вимогам маскування даних використанняхмарних ключів	SaaS / Web / Cloud	Спеціальна ціна
AxCrypt Premium	Немає	Так	- Безпечний обмін за допомогою криптографії з відкритим ключем - Безпечне видалення файлів - Безпечне онлайн-зберігання паролів	Програмне забезпечення - безстрокова ліцензія	$ 9,92 / місяць передплата
VeraCrypt	Немає	Так	- Шифрування розділів - Підтримує UEFI та MBR для Windows	Завантажити безкоштовну утиліту з відкритим кодом	Безкоштовно / з відкритим кодом
CertainSafe Digital Safety Deposit Box	Немає	Так	-Аутентифікує користувача на сервері і навпаки - Надійно зберігає попередні версії файлів	SaaS	Спеціальна ціна
NordLocker	Немає	Так	- Простий інтерфейс перетягування - Зашифровані файли можна переглядати через додаток без шифрування	SaaS	Спеціальна ціна
Kruptos 2	Немає	Так	- Безшовне хмарне шифрування - Подрібнення даних -Вбудований безпечний редактор приміток	Клієнт програмного забезпечення	$ 39,95 / одноразова покупка

Шифрування може бути одним із найпотужніших інструментів у вашій архітектурі безпеки, але це не самостійне рішення. Її все одно слід поєднувати з іншими рішеннями, такими як антивірусне програмне забезпечення , брандмауери та VPN служби щоб охопити всі кінцеві точки.

Після шифрування або копіювання версії файлу оригінальну незашифровану версію завжди слід повністю видалити з вашої системи. Дані можуть існувати на диску навіть після їх видалення, і їх можна відновити за допомогою спеціалізованих інструментів. Використання віртуального подрібнювача або функції безпечного видалення забезпечить його повне знищення.

1.4 Постановка задач дослідження

В даному розділі провдено дослідження та класифікацію різних видів цифрової інформації та розглянуто основні сфери її використання. Проведено та класифікацію сучасних інформаційних систем для обробки цифрових даних, а також виділено їх основні характеристики. Проаналізовано програмні комплекси для кодування/декодування цифрової інформації.

Для досягнення поставленої мети необхідно розв’язати наступні задачі.

− провести огляд видів інформаціїї в системах обробки інформації, її властивості та сфери використання;

− провести класифікацію інформаційних систем обробки даних;

− провести дослідження існуючих програмних кодування/декодування інформації;

− проаналізувати існуючі алгоритми кодування/декодування інформації;

− розробити алгоритм кодування інформації на основі аміни символьних пар;

− реалізувати програмну систему кодування інформації, провести її тестування та порівняти з програмами аналогами.

1.5 Висновки до розділу

Проведено дослідження та клафивікацію властивостей та сфер застосування цифрових даних, що дозволило виділити основні напрямки та алгоритми які використовуються кодуванні інформації.

Проаналізовано алгоритми обробки цифрових зображень, що дозволило вигначити групу алгоритмів для проведення пертворень цифрових зображень, а також їх перевоги та недоліки.

Проведено аналіз програмних додатків обробки цифрових дани який дозволив виділити головні структурні блоки, а встановити інтерфейси передачі даними між ними.

2 МЕТОДИ ТА АЛГОРИТМИ КОДУВАННЯ ЦИФРОВОЇ ІНФОРМАЦІЇ

2.1 Методи та алгоритми запису та зберігання інформації

У процесі кодування джерело інформації представляє повідомлення в алфавіті, який називається первинним, далі це повідомлення потрапляє в пристрій, що перетворює і представляє його у вторинному алфавіті.

Код - правило, яке описує відповідність знаків (або їх поєднань) первинного алфавіту знаком (їх поєднаннями) вторинного алфавіту. Код – це правило, яке описує відповідність знаків або їх поєднань одного алфавіту знаків або їх сполученням іншого алфавіту; знаки вторинного алфавіту, використовувані для представлення знаків або їх поєднань первинного алфавіту.

Кодування - операція отожествления символів або груп символів одного коду з символами або групами символів іншого коду.

Кодування інформації - процес формування певного уявлення інформації. У більш вузькому сенсі під терміном «кодування» розуміють перехід від однієї форми подання інформації до іншої, більш зручною для зберігання, передачі або обробки.

Декодування - операція, зворотна кодуванню, тобто відновлення інформації з закодованого виду (відновлення в первинному алфавіті за отриманою послідовності кодів).

Кодер - пристрій, що забезпечує виконання операції кодування.

Кодер - програміст, що спеціалізується на кодуванні - написанні вихідного коду по заданих специфікацій.

Кодер - одна з двох компонент кодека (пари кодер - декодер).

Декодер - деякий ланка, яке перетворює інформацію з зовнішнього вигляду в вид, застосовуваний усередині вузла. У програмному забезпеченні: модуль програми або самостійний додаток, яке перетворює файл або

інформаційний потік із зовнішнього виду на вид, який підтримує інше програмне забезпечення.

Декодер - пристрій, що виробляє декодування.

Шифрування - різновид кодування.

Шифр - код, значення і правила використання якого відомо обмеженому колу осіб.

Операції кодування і декодування називаються оборотними, якщо їх послідовне застосування забезпечує повернення до вихідної інформації без будь-яких її втрат.

Прикладом оборотного кодування є представлення знаків у телеграфному коді і їх відновлення після передачі. Прикладом кодування незворотного може служити переклад з однієї природної мови на іншу - зворотний переклад, взагалі кажучи, не відновлює вихідного тексту. Безумовно, для практичних завдань, пов'язаних із знаковою виставою інформації, можливість відновлення інформації по її коду є необхідною умовою застосування коду, тому в подальшому викладі обмежимо себе розглядом тільки оборотного кодування.

Таким чином, кодування передує передачі та зберігання інформації. При цьому зберігання пов'язане з фіксацією деякого стану носія інформації, а передача - зі зміною стану з плином часу (тобто процесом). Ці стани або сигнали будемо називати елементарними сигналами - саме їх сукупність і складає вторинний алфавіт.

Будь-код повинен забезпечувати однозначне читання повідомлення (надійність), так і, бажано, бути економним (використовувати в середньому трохи менше символів на повідомлення).

Можливість відновити текст означає, що в мові є певна надмірність, за рахунок якої ми відновлюємо відсутні елементи за рештою. Ясно, що надмірність знаходиться в можливостях букв і їх комбінаціях, їх знання дозволяє підібрати найбільш ймовірний відповідь.

Комп'ютер може обробляти тільки інформацію, представлену в числовій формі. Вся інша інформація (звуки, зображення, показання приладів тощо) Для

обробки на комп'ютері повинна бути перетворена в числову форму. За допомогою комп'ютерних програм можна перетворювати отриману інформацію, в тому числі - текстову. При введенні в комп'ютер кожна буква кодується певним числом, а при виведенні на зовнішні пристрої (екран або друк) для сприйняття людиною по цих числах будуються зображення букв. Відповідність між набором літер і числами називається кодуванням символів. Як правило, всі числа в комп'ютері представлені за допомогою нулів і одиниць, тобто словами, комп'ютери працюють в двійковій системі числення, оскільки при цьому пристрої для їх обробки виходять значно простішими.

В процесі перетворення інформації з однієї форми подання (знакової системи) в іншу здійснюється кодування. Засобом кодування служить таблиця відповідності, яка встановлює взаємно однозначну відповідність між знаками або групами знаків двох різних знакових систем. У процесі обміну інформацією часто доводиться проводити операції кодування і декодування інформації. При введенні знака алфавіту в комп'ютер шляхом натискання відповідної клавіші на клавіатурі виконується його кодування, тобто перетворення в комп'ютерний код. При виведенні знака на екран монітора або принтер відбувається зворотний процес - декодування, коли з комп'ютерного коду знак перетворюється в графічне зображення.

Кодування інформації розпадається на етапи:

1) визначення обсягу інформації, що підлягає кодуванню;

2) класифікація та систематизація інформації;

3) вибір системи кодування і розробка кодових позначень;

4) безпосереднє кодування.

Хоча природною для органів почуттів людини є аналогова форма, універсальної все ж слід вважати дискретну форму подання інформації за допомогою деякого набору знаків. Зокрема, саме таким чином представлена інформація обробляється комп'ютером, передається по комп'ютерних і деяким іншим лініях зв'язку. Повідомлення - послідовність знаків алфавіту. При їх передачі виникає проблема розпізнавання знака: яким чином прочитати

повідомлення, тобто за отриманими сигналами встановити вихідну послідовність знаків первинного алфавіту. В усному мовленні це досягається використанням різних фонем (основних звуків різного звучання), за якими ми і відрізняє знаки мови. У писемності це досягається різним шрифтом букв і подальшим аналізом написаного. Можна реалізувати деяку процедуру, за допомогою якої виділити з повідомлення той чи інший знак. Але поява конкретного знака (літери) в конкретному місці повідомлення - подія випадкове. Отже, впізнавання (ототожнення) знака вимагає отримання деякої порції інформації. Можна пов'язати цю інформацію з самим знаком і вважати, що знак несе в собі деяку кількість інформації.

Алфавіт сукупність графічних знаків - букв або складових знаків, розташованих в традиційно встановленому порядку.

Знак - угода (явне або неявне) про приписуванні чому-небудь (що означає) якого-небудь певного сенсу (означаемого). Знак - це матеріально виражена заміна предметів, явищ, понять в процесі обміну інформацією в колективі. Знаком також називають конкретний випадок використання такої угоди для передачі інформації. Знак може бути складовим, тобто складатися з кількох інших знаків. Букви і слова людської мови є знаками. Цифри і числа є знаками. Наука про знакові системи називається семіотикою.

Символи (в комп'ютері) - цифри, букви, ієрогліфи і тощо. Для подання цифрової інформації використовується деякий алфавіт. Однак однозначна відповідність між інформацією та алфавітом відсутня. Іншими словами, одна і та ж інформація може бути представлена за допомогою різних алфавітів. У зв'язку з такою можливістю виникає проблема переходу від одного алфавіту до іншого, причому, таке перетворення не повинно призводити до втрати інформації.

Алфавіт, за допомогою якого подається інформація до перетворення називається первинним; алфавіт кінцевого уявлення - вторинним.

В рамках математичної постановки задачі кодування введемо позначення: A - первинний алфавіт, що складається з N знаків з середньою інформацією на

знак I_A; B - вторинний з M знаків з середньою інформацією на знак I_B. Повідомлення в первинному алфавіті містить n знаків, а закодоване - m знаків. $I_s(A)$ - інформація у вихідному повідомленні; $I_f(B)$ - інформація в закодованому повідомленні.

Кодування сигналу - це його уявлення в певній формі, зручній для подальшого використання сигналу, тобто це правило, яке описує відображення одного набору знаків в інший набір знаків. Тоді відображається набір знаків називається вихідним алфавітом, а набір знаків, який використовується для відображення, - кодовою алфавітом, або алфавітом для кодування. При цьому кодування підлягають як окремі символи вихідного алфавіту, так і їх комбінації. Аналогічно для побудови коду використовуються як окремі символи кодового алфавіту, так і їх комбінації. Наприклад, відповідності між натуральними числами трьох систем числення. Вихідний алфавіт – десяткові цифри від 0 до 9, а кодові алфавіти – це 0 і 1 для двійкової системи; цифри від 0 до 9 і символи {*A, B, C, D, E, F*} – для шістнадцятковій.

Кодовою комбінацією (кодом) називається сукупність символів кодового алфавіту, що застосовуються для кодування одного символу (або однієї комбінації символів) вихідного алфавіту. При цьому кодова комбінація може містити один символ кодового алфавіту. Вихідним символом називається символ (або комбінація символів) вихідного алфавіту, якому відповідав би кодова комбінація. Наприклад, оскільки 8 = 10002 і 8 є вихідним символом, 1000 - це кодова комбінація, або код, для числа 8. У той же час 8 - це вихідний символ. Сукупність кодових комбінацій називається кодом. Взаємозв'язок символів (або комбінацій символів, якщо кодуються не окремі символи) вихідного алфавіту з їх кодовими комбінаціями складає таблицю відповідності (таблицю кодів). Зворотна процедура отримання вихідних символів за кодами символів називається декодуванням. Очевидно, для виконання правильного декодування код повинен бути однозначним, тобто одному вихідному символу повинен відповідати точно один код і навпаки.

Залежно від цілей кодування, розрізняють наступні його види які візуально відображеня на рисуноку 2.1.

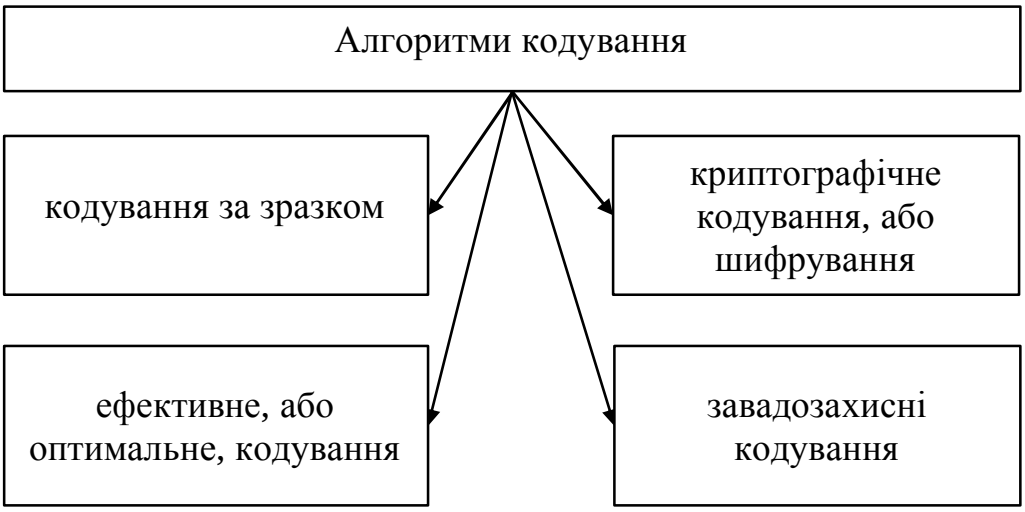

Рисунок 2.1 – Види алгоритмів кодування

Розглянемо більш детально кожний з видів кодування та сіери їх застосування:

– кодування за зразком - використовується щоразу при введенні інформації в комп'ютер для її внутрішнього уявлення;

– криптографічне кодування, або шифрування, - використовується, коли потрібно захистити інформацію від несанкціонованого доступу;

– ефективне, або оптимальне, кодування - використовується для усунення надмірності інформації, тобто зниження її обсягу, наприклад, в архіваторах;

– завадозахисні кодування - використовується для забезпечення заданої достовірності в разі, коли на сигнал накладається перешкода, наприклад, при передачі інформації по каналах зв'язку.

Нехай первинний алфавіт A містить N знаків з середньою інформацією на знак, визначеної з урахуванням ймовірностей їх появи, $I_1^{(A)}$ (нижній індекс відображає ту обставину, що розглядається перше наближення, а верхній індекс

у дужках вказує алфавіт). Вторинний алфавіт B нехай містить M знаків з середньою інформаційної ємністю $I_1^{(A)}$. Нехай також вихідне повідомлення, представлене в первинному алфавіті, містить n знаків, а закодоване повідомлення - m знаків. Якщо вихідне повідомлення містить $I^{(A)}$ інформації, а закодоване - $I^{(B)}$, то умова оборотності кодування, тобто неісчезновенія інформації при кодуванні, очевидно, може бути записане таким чином:

$$I^{(A)} \leq I^{(B)}.$$

Сенс якого в тому, що операція оборотного кодування може збільшити кількість формальної інформації в повідомленні, але не може його зменшити. Однак кожна з величин в даному нерівності може бути замінена твором числа знаків на середню інформаційну ємність знака, тобто:

$$nI^{(A)} \leq mI^{(B)},$$

або

$$I^{(A)} \leq (m/n)I^{(B)}.$$

Ставлення m/n, очевидно, характеризує середнє число знаків вторинного алфавіту, яке доводиться використовувати для кодування одного знака первинного алфавіту - будемо називати його довжиною коду або довжиною кодової ланцюжка і позначимо $K^{(B)}$ (верхній індекс вказує алфавіт кодів).

Введемо відносну надмірність коду (Q):

$$Q = 1 - \frac{I^{(A)}}{I^{(B)}}$$

Дана величина показує, наскільки операція кодування збільшила довжину вихідного повідомлення. Очевидно, чим менше Q (тобто чим ближче вона до 0 або, що те ж, $I^{(B)}$ ближче до $I^{(A)}$, тим більш вигідним виявляється код і більш ефективної операція кодування. Вигідність кодування при передачі і зберіганні - це економічний фактор, оскільки більш ефективний код дозволяє витратити на передачу повідомлення менше енергії, а також часу і, відповідно, менше займати лінію зв'язку; при зберіганні використовується менше площі поверхні (обсягу) носія. При цьому слід усвідомлювати, що вигідність коду ідентична временно́й вигідності всього ланцюжка кодування - передача - декодування; можлива ситуація, коли за використання ефективного коду при передачі доведеться розплачуватися тим, що операції кодування і декодування будуть займати більше часу і інших ресурсів (наприклад, місця в пам'яті технічного пристрою, якщо ці операції проводяться з його допомогою).

Розглянутий вираз поки слід сприймати як співвідношення оціночного характеру, з якого неясно, якою мірою при кодуванні можливо наближення до рівності його правої і лівої частин. З цієї причини для теорії зв'язку найважливіше значення мають дві теореми, доведені Шенноном.

2.2 Алгоритми кодування інформації в системах обробки даних

Характерною особливістю більшості «класичних» типів інформації, з якими працюють люди, є їх надмірність.

Для української мови характерні слова, однозначно не прочитувані в разі «втрати» деяких букв. Наприклад, ко_а («коса» або «кора»), _ак («мак», «гак» або «лак») тощо. Крім того, маючи текст українською мовою з «втраченими» буквами, людина, досить добре володіє українською мовою, може однозначно відновити його. Наприклад, без труднощів можна відновити текст наступного

змісту «Тар_с Ше_вче_ко – вида_ний п_е_». Однак якщо ця пропозиція буде читати іноземець, ледь знає українську мову та українську поезію, то він не зможе його зрозуміти. Носії мови, можуть з легкістю відновити закінчення, пропущені букви в складах, підібрати відповідні слова. Для носія мови звичайний зв'язний текст на його рідній мові містить надлишкову інформацію - її можна видалити, але зміст тексту для нього збережеться.

Одним із прикладів прояви надмірності інформації та її стискування є використання математичних позначень. Наприклад, сума чисел 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40 записується так:

$$2+4+6+8+10+12+14+16+18+20+22+24+26+28+30+32+34+36+38+40$$

Ця сума складена з усіх парних чисел від 2 до 40. Використовуючи математичну нотацію, цю суму можна записати набагато коротше:

$$\sum_{i=2}^{20} 2*i\,.$$

Отже, ввівши нові позначення, ми зуміли скоротити запис математичного виразу, а значить, стиснути інформацію. Однак якщо перша форма запису зрозуміла кожному, то друга зрозуміла тільки тим, хто знає, як слід інтерпретувати цей запис.

Приклад з сумою дозволяє продемонструвати ще одну особливість методів стиснення - ступінь стиснення вхідних даних залежить від самих стискаються даних. Так, щоб скористатися позначенням суми, треба знайти формулу, яка має складаються числа через індекс підсумовування. І хоча якусь формулу можна вивести для будь-якої кінцевої послідовності (наприклад, за допомогою інтерполяційної формули Лагранжа), але ось компактну формулу підібрати вдається далеко не завжди.

Формальне визначення надмірності інформації.

Кодування інформації є надлишковим, якщо кількість біт в отриманому коді більше, ніж це необхідно для однозначного декодування вихідної інформації.

Ступінь надмірності залежить від типу інформації: у відеоінформації вона в кілька разів більше, ніж у графічної інформації, а ступінь надмірності останньої в кілька разів більше, ніж текстової інформації. Взагалі, ступінь надмірності природної інформації досить велика. Клод Шеннон, дослідивши надмірність літературної англійської мови, встановив, що вона становить близько 50%. Це означає, що якщо в англійському тексті навмання стерти близько половини букв, то за рештою буквах людина, яка знає англійську мову, майже напевно зможе відновити текст. Надмірність мови виконує дуже важливу функцію - забезпечує людині надійність її сприйняття, особливо в несприятливих умовах (перегляд телепередач при наявності перешкод, читання тестів в умовах недостатньої освітленості, розмова в вагоні метро тощо).

Зберігання та передача інформації вимагають певної витрати ресурсів. Стиснення даних (перед збереженням або передачею по каналах зв'язку) дозволяє зменшити ці витрати. На практиці такі витрати можна навіть висловити в грошовому еквіваленті: наприклад, на скачування з Інтернету стисненого музичного файлу буде потрібно менше часу, а значить, доведеться заплатити менше грошей за користування Інтернетом.

Всі методи стиснення можна поділити на два великі класи (рисунок 2.2). Одні алгоритми тільки змінюють спосіб представлення вхідних даних, приводячи їх формі, яка більш компактно кодується. Такі алгоритми прийнято називати оборотними, оскільки для них існують зворотні алгоритми, здатні точно відновити вихідні дані з стисненого масиву. Інші алгоритми виділяють у вхідних даних істотну інформацію і ту частину, якою можна знехтувати і видалити, після чого залишилися «істотні» дані стискаються. Такі алгоритми прийнято називати алгоритми з регульованою втратою інформації.

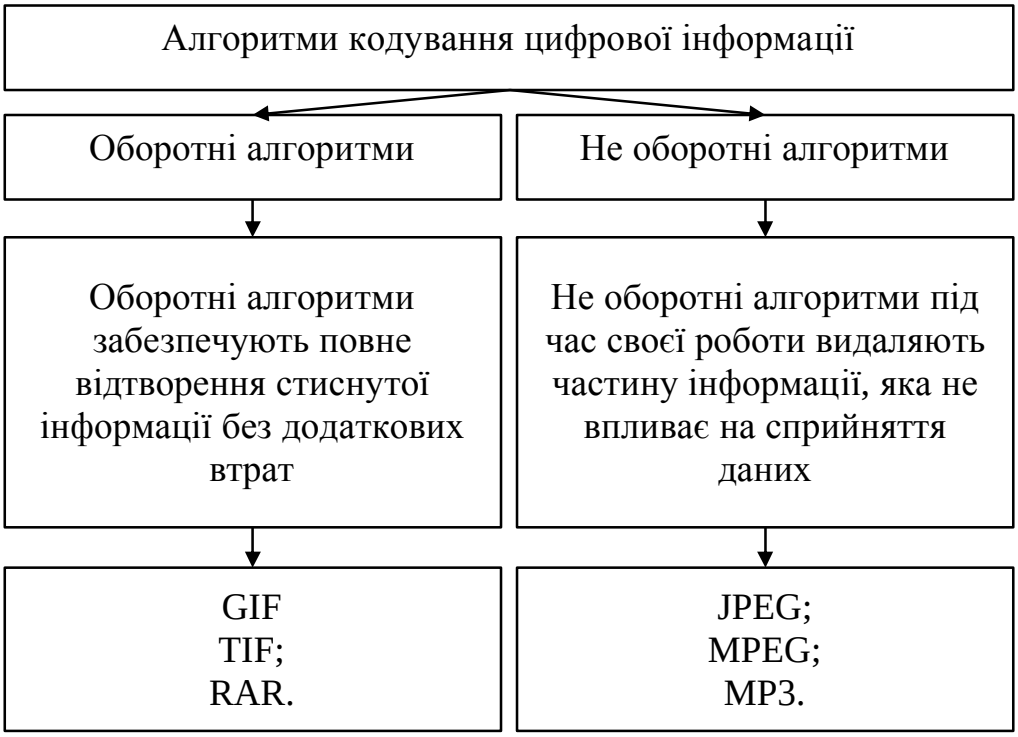

Рисунок 2.2 – Класифікація алгоритмів кодування

Метод стиснення називається оборотним, якщо з даних, отриманих при стисканні, можна точно відновити вихідний масив даних.

Оборотні методи можна застосовувати для стиснення будь-яких типів даних. Характерними форматами файлів, що зберігають стислу без втрат інформацію:

– GIF, TIF, PCX, PNG - для графічних даних;

– AVI - для відеоданих;

– ZIP, ARJ, RAR, LZH, LH, CAB - для будь-яких типів даних.

Існує досить багато оборотних методів стиснення даних, однак в їх основі лежить порівняно невелика кількість теоретичних алгоритмів, які проаналізовано більш докладно.

Метод упаковки полягає в зменшенні кількості біт, що відводяться для кодування символів, якщо в стисливому масиві даних є тільки невелика частина використовуваного алфавіту.

Для прикладу, припустимо, вхідний текст складається з десяткового запису цілих чисел і знаків «мінус», розділених пробілу (наприклад, 280 - 1 296 48 40 365 - 159 13 777). Масив символів, що зустрічаються в такому тексті, складається всього з 12 символів (цифри від «0» до «9», знак «-» (мінус) і пробіл). Для кодування такої кількості символів достатньо всього чотирьох біт, цілого байта для цього багато. Якщо упакувати коди цих символів в 4 біта (наприклад, так: «0» → «0000», «1» → «0001», ... «9» → «1001», «-» → »1110», пробіл → «1111 »), то можна буде кодувати по два символи вхідного тексту одним байтом в вихідному масиві. В результаті отримаємо дворазове стиснення даних. Формат запису чисел, при якому число, записується в десятковій системі, а цифри кодуються 4-бітовими кодами, називається BCD - форматом (Binary Coded Decimal, або двійковій-десятковий запис). BCD- формат нерідко використовується для зберігання цілих чисел, наприклад в базах даних.

Розглянемо інший приклад, Вхідний текст «коса косив косо» містить всього 7 різних символів (к, о, с, а, в, и та пробіл), отже, кожен символ може бути закодований трьома бітами. Всього в початковому тексті 15 символів, так що буде потрібно 15*3=45 біта. Округливши це значення з надлишком до цілого числа байт, отримаємо розмір стисненого масиву - всього 6 байт. Коефіцієнт стиснення дорівнює 18/6 = 3.

Одна з переваг методу упаковки полягає в тому, що будь-який фрагмент стислих даних можна розпакувати, абсолютно не використовуючи попередні дані. Дійсно, знаючи номер потрібний символ N і довжину кодів символів M, можна обчислити місцеположення коду символу в стислому масиві даних:

– номер байт, в якому починається код символу, обчислюється за такою математичною формулою як $L[M \cdot N/8]$;

– номер першого біта коду (в межах цього байта) До дорівнює залишку від ділення $M \cdot N$ на 8.

Метод упаковки дає хороші результати тільки, якщо масив використовуваних символів невелика. Наприклад, якщо в тесті

використовуються тільки прописні кириличні букви і розділові знаки, то текст може бути стиснутий всього на 25%: 33 українські літери плюс пробіл і знаки пунктуації - 40 символів. Для їх кодування досить 6 біт. При упаковці текст зменшується до 6/8 = 3/4 від початкового об'єму.

Слабке місце методу упаковки полягає в тому, що символи кодуються бітовими послідовностями однакової довжини. Наприклад, будь-який текст, що складається тільки з двох букв «а» і «в», стискається методом упаковки в вісім разів. Однак якщо до такого тексту додати всього лише одну букву, наприклад, «с», то ступінь стиснення відразу зменшиться вдвічі, причому незалежно від довжини тексту і кількості доданих символів «с».

Покращення ступеня стиснення можна досягти, кодуючи часто зустрічаються символи короткими кодами, що рідко зустрічаються - довшими. Алгоритм Хаффмана виявився простим, швидким і оптимальним: серед алгоритмів, що кодують кожен символ окремо і цілим кількістю біт, він забезпечував найкраще стиснення.

Алгоритм Хаффмана стискає дані за два проходи: на першому проході читаються все вхідні дані і підраховуються частоти народження всіх символів. Потім за цими даними будується дерево кодування Хаффмана, а по ньому - коди символів. Після цього, на другому проході, вхідні дані читаються ще раз і при цьому генерується вихідний масив даних.

Для тексту «кол_около_колокола» код тексту буде займати 39 біт або 5 байт. Коефіцієнт стиснення дорівнює 18/5 = 3,6.

Код Хаффмана є префіксним. Це означає, що код кожного символу не є початком коду будь-якого іншого символу. Код Хаффмана однозначно відновимо, навіть якщо не повідомляється довжина коду кожного переданого символу. Одержувачу пересилають тільки дерево Хаффмана в компактному вигляді, а потім вхідна послідовність кодів символів декодується їм самостійно без будь-якої додаткової інформації.

В основу алгоритмів RLE (Run-Length Encoding - кодування шляхом обліку кількості повторень) покладено принцип виявлення повторюваних

послідовностей даних і заміною їх простою структурою. Алгоритми Лемпеля-Зива тим краще стискають текст, чим більше розмір вхідного масиву. Характерною особливістю зворотних алгоритмів LZ77 і LZ78 є те, що крім самих стислих даних, ніякої додаткової інформації їм не потрібно! Почавши працювати, ці алгоритми по вже розпакованої частини відновлюють інформацію, необхідну для розпакування наступних частин стислих даних. Для порівняння: в алгоритмі Хаффмана разом із стисненими даними потрібно зберігати древо Хаффмана, інакше розпакування буде неможлива. В даний час існує півсотні модифікацій цього алгоритму. Всі вони називаються методами стиснення зі словником. Ці алгоритми виявилися настільки швидкі й ефективні, що зараз займають лідируюче місце серед використовуваних на практиці алгоритмів стиснення.

Алгоритми кодування є оборотними: будь-який масив вхідних даних при розпакуванні відновлюється абсолютно точно. Тому оборотні алгоритми можна застосовувати для стиснення інформації будь-якого роду. Але, як виявилося, для аудіо- та відеоінформації абсолютно точне відновлення зовсім не обов'язково. Деякі особливості людського сприйняття дозволяють істотно збільшити ступінь стиснення звуковий, графічної та відеоінформації. Наприклад, очей людини найбільш чутливий до зеленого кольору, чутливість до червоного нижче в 4 рази, а до синього - в 10 разів. А це означає, що на зберігання інформації про червоною і синьою складових кольору можна було б відводити менше біт. Але в більшості форматів графічних файлів це не так - колірні компоненти кодуються однаковою кількістю біт. Цей приклад показує, що традиційні способи представлення відеоінформації мають дуже великим ступенем надмірності, за умови, що мова йде про відтворення відеоінформації для людини.

До середини 1990-х були розроблені спеціальні високоефективні методи стиснення аудіо- та відеоінформації, що враховують особливості людського слуху і зору. Характерною рисою цих методів є можливість регульованого видалення маловажной (для людського сприйняття) інформації. Тому такі

алгоритм стиснення називають алгоритмами з регульованою втратою інформації. За рахунок видалення частини інформації вдається домогтися дуже великій мірі стиснення даних при суб'єктивно незначної втрати якості аудіо- і відеоданих.

Алгоритми з регульованою втратою інформації неуніверсальність, вони не можуть використовуватися для стиснення будь-яких даних, оскільки повне відновлення вихідної інформації неможливо. Найбільш відомими методами стиснення з регульованою втратою інформації є:

– JPEG - метод стиснення графічних даних;

– MPEG - група методів стиснення відеоданих;

– MP3 - метод стиснення звукових даних.

Алгоритм JPEG використовується для стиснення статичних зображень. Крім стискається зображення, алгоритму передається також бажаний коефіцієнт стиснення - цей параметр регулює частку інформації, яка буде видалена при стисненні.

Власне стиснення JPEG здійснюється в кілька етапів: спершу кольору пікселів переводяться з RGB-вистави в YUV-уявлення (відповідної йому колірної моделі YCdCr колір представляється компонентами «яскравість» *Y*, «різниця зелений - червоний» *Cr* і «різниця зелений-синій» *Cb*. Потім в кожному другому рядку і кожному другому стовпці матриці пікселів інформація про колірних компонентах *Cb* і *Cr* просто видаляється, що миттєво зменшує обсяг даних вдвічі. Решта даних піддаються спеціальній процедурі «згладжування», при якому обсяг даних не змінюється, але потенційна ступінь їх стисливості різко збільшується. На цьому етапі враховується бажаний коефіцієнт стиснення. Потім дані стискаються алгоритмом Хаффмана.

Алгоритм MP3 є частиною стандарту MPEG і описує стиснення аудіоінформації. Крім стискається звукового фрагмента алгоритму передається також бажаний бітрейт (bitrate) - кількість біт, що використовуються для

кодування однієї секунди звуку. Цей параметр регулює частку інформації, яка буде видалена при стисненні.

Стиснення MP3 також здійснюється в кілька етапів: звуковий фрагмент розбивається на невеликі ділянки - фрейми (frames), а в кожному фреймі звук розкладається на складові звукові коливання, які у фізиці називають гармоніками. З точки зору математики, звук розкладається на групу синусоїдальних коливань з різними частотами і амплітудами. Потім починається психоакустична обробка - видалення маловажной для людського сприйняття звукової інформації, при цьому враховуються різні особливості слуху. Бажаний бітрейт визначає, які ефекти будуть враховуватися при стисненні, а також кількість що видаляється. На останньому етапі залишилися дані стискаються алгоритмом Хаффмана.

Алгоритм MP3 дозволяє стискати звукові файли в кілька разів. При цьому навіть найбільший бітрейт 320 Кбіт/с стандарту MP№ забезпечує чотириразове стиснення аудіоінформації в порівнянні м форматом Audi XD, при такому ж суб'єктивному якості звуку. Формат MP3 став стандартом де-факто для поширення музичних файлів через Інтернет.

MPEG - ціле сімейство методів стиснення відеоданих. У них використовується дуже велика кількість прийомів стиснення. Вони спираються на кілька базових ідей, а розрізняються конкретною реалізацією алгоритмів. Одна з основних ідей стиснення відео - метод «опорного кадру» - записує не цілком відеокадри, а тільки зміни кадрів. Наприклад, у фільмі є сцена розмови героїв в кімнаті. При цьому від кадру до кадру змінюються лише вирази облич, а велика частина зображення нерухома. Закодувавши перший кадр сцени і відмінності інших е кадрів від першого, можна отримати дуже велику ступінь стиснення. Ще один спосіб зменшення кодованої інформації полягає в тому, щоб швидко змінювані ділянки зображення кодувати з якістю, яке набагато нижче якості статичних ділянок, - людське око не встигає розглянути їх достатньо детально.

Крім того, формат MPEG дозволяє зберігати в одному файлі декілька потоків даних. Так, в основному потоці можна зберегти фільм, в іншому - логотип, в третьому – субтитри тощо. Потоки даних накладають один на одного тільки при відтворенні. Такий спосіб дозволяє, наприклад, зберегти субтитри у вигляді тексту замість зображень букв, логотип зберегти всього один раз, а не в кожному кадрі тощо.

Різновиди формату MPEG відрізняються один від одного за можливостями, якістю відтвореного зображення і максимальному ступені стиснення:

– MPEG-1 - використовувався в перших Video CD (VCD-I);

– MPEG-2 - використовувався в DVD і Super Video CD (SVCD, VCD-II);

– MJPEG - формат стиснення відео, в якому кожен кадр стискається за методом JPEG;

– MPEG-4 - популярний ефективний формат стиснення відео;

– DivX, XviD - поліпшені модифікації форматаMPEG-4.

Виявлення помилок - дія, спрямована на контроль цілісності даних при записі / відтворенні інформації. Виправлення помилок - процедура відновлення інформації після читання її з пристрою зберігання. Для виявлення помилок використовують коди виявлення помилок, для виправлення - коригувальні коди. В процесі зберігання даних і передачі інформації по мережах зв'язку неминуче виникають помилки. Контроль цілісності даних і виправлення помилок - важливі завдання на багатьох рівнях роботи з інформацією.

Існує кілька стратегій боротьби з помилками:

1) виявлення помилок в блоках даних і автоматичний запит повторної передачі пошкоджених блоків - цей підхід застосовується в основному на канальному і транспортному рівнях;

2) виявлення помилок в блоках даних і відкидання пошкоджених блоків - такий підхід іноді застосовується в системах потокового мультимедіа, де важлива затримка передачі і немає часу на повторну передачу;

3) виправлення помилок застосовується на фізичному рівні.

Коригувальні коди - коди, службовці для виявлення або виправлення помилок, що виникають при передачі інформації під впливом перешкод, а також при її зберіганні.

Для цього під час запису (передачі) в корисні дані додають спеціальним чином структуровану надлишкову інформацію (великі червоні літери), а при читанні (прийомі) її використовують для виявлення або виправлення помилки. Число помилок, яке можна виправити, обмежена і залежить від конкретного застосовуваного коду. З кодами, виправляють помилки, тісно пов'язані коди виявлення помилок. За способом роботи з даними коди, що виправляють помилки діляться на блокові, що ділять інформацію на фрагменти постійної довжини і обробні кожен з них окремо, і свёрточние, що працюють з даними як з безперервним потоком. Хороший код повинен задовольняти, як мінімум, наступними критеріями:

1) здатність виправляти якомога більше число помилок;

2) як можна менша надлишковість;

3) простота кодування і декодування.

Практично всі використовувані коди є лінійними. Це пов'язано з тим, що нелінійні коди значно складніше досліджувати, і для них важко забезпечити прийнятну легкість кодування і декодування.

Лінійний блоковий код - такий код, що безліч його кодових слів утворює *k*-мірний лінійне підпростір в *n*-вимірному лінійному просторі, изоморфное простору *k*-бітних векторів. Це означає, що операція кодування відповідає множенню вихідного *k*-бітного вектора на невироджених матрицю, яка називається породжує матрицею.

Коди Хеммінга - найпростіші лінійні коди з мінімальним відстанню 3, тобто здатні виправити одну помилку.

При передачі інформації по каналу зв'язку ймовірність помилки залежить від відношення сигнал/шум на вході демодулятора, тому при постійному рівні шуму вирішальне значення має потужність передавача. У системах

супутникового або мобільного зв'язку гостро стоїть питання економії енергії, а в телефонному зв'язку необмежено підвищувати потужність сигналу не дають технічні обмеження. Оскільки завадостійке кодування дозволяє виправляти помилки, при його застосуванні потужність передавача можна знизити, залишаючи швидкість передачі інформації незмінною. Енергетичний виграш визначається як різниця відносин сигнал/шум при наявності і відсутності кодування.

Коди, що виправляють помилки, застосовуються:

1) в системах цифрового зв'язку, в тому числі: супутникової, радіорелейного, стільникового, передачі даних по телефонних каналах;

2) в системах зберігання інформації, в тому числі магнітних і оптичних.

Коди, що виявляють помилки, використовуються в мережевих протоколах різних рівнів.

2.3 Алгоритм кодування інформації на основі символьних пар

Проводячи аналіз рограмних додатків кодування інформації з різноманітими законами розподілу ваг для розрядів монолітного коду, можна зауважити, що в окремих ситуаціях розподіл ваг монолітного код є занадто надлишковим, оскільки для одного і того ж числа відображення можна подати за допомогою набору різних кодів, у вигляді кодових комбінацій позиційного коду. При вирішені даної задачі, виконують пошук оптимальної для даної ситуації комбінаторного варіанту значень ваг для розрядів монолітного коду, при якому вхідне число можна було б представити в монолітному коді , причому він буде унікальним.

При цьому може зявитись проблема яким чином обрати оптимальний масив ваг розрядів, основна роль якого полягає в тому, щоб обраному масиву

кодових комбінацій монолітного коду, обов'язково була поставлена у відповідність,причому однозначну, інший масив чисел натурального ряду.

При проведені досліджень, розглядались системикодування, які базуються в своїй основі на використанні комбінаторних властивостей числових послідовностей. Числова послідовність – це по суті, алгебраїчна структура, створена на деякій послідовності N цілих додатних чисел, причому значення яких, як і відповідно значення сум поруч розташованих чисел, повторюються счисленну кількість разів R. Елементи числових послідовностей розміщуються у вигляді лінійки або кільця.

Потужність для кільцевого монолітного коду, що був реалізований на числових послідовностях n-го порядку, обчислюється на основі загальної кількісті варіантів утворення кодових слів:

$$N_{(K)}=n(n-1)+1.$$

Завадостійкість такого монолітного коду слід оцінити як результат відношення кількості отриманих помилкових кодових комбінацій, що були отримані M, до загальної кількості комбінацій які взагалі були отримані для заданої розрядності.

При побудові кільця монолітного коду необхідно виконати таку послідовність крокі:

Крок 1. За потужністю кільця монолітного коду вибирають числові лінійні посділовності порядку N кратності R .

Крок 2. Формується множина із N комірок, які повинні бути пронумеровані за зростанням.

Крок 3. В перші R комірок множини заноситься число, з якого стартує відлік у числовій лінійній посділовності, в $(R+1)$ – наступну, всі інші комірки заповнюються «0».

Крок 4. При першому попаданні число A оголошується як збільшене на «1» найбільше число серед найкоротшого ряду відповідної послідовності чисел,

що утворене в масиві сум, що визначені на всіх окремих послідовностях, причому вони повинні нажежати множині. При подальших співпадіннях *A* з тим самим значенням необхідно оголосити як збільшене на «1» наступне за найбільшим числом відповідного найкоротшого ряду послідовності чисел. При наявності вільних комірок множини, число *A* буде збережене у вільній комірці з найменшим порядковим значенням.

Крок 5. На даному кроці необхідно вирахувати нове значення суми усіх елементів множиини числових лінійних послідовностей. Якщо у множині присутні вільні комірки, то необхідно знайти всі суми на всіх послідовностях, а при умові їхньої відсутності – всі можливі лінійні суми для єдиної послідовності:

а) якщо кожна зі виділених сум потрапляє не більше ніж значення *R* разів, а також є вільні комірки, то необхідно здійснити перехід до кроку 4. При умові відсутності вільних комірок, а також при здійсненні умови, що нове значення суми буде не більшим від значення попереднього, то отримується варіант числової лінійної послідовності, після чого перехід на крок 7. В іншому випадку перехід на крок 6;

б) При умові, що хоча б одна зі визначених сум зустрінеться більше ніж *R* разів, то перехід на крок 6.

Крок 6. Визначаєтсья максимальне число *B*, після чого перевіряють, чи є вільна комірка з номером, що є більшим, від того, де розміщене число *B*; у випадку наявності комірки, число *B* переміщається з комірки з меншим номером у вільну комірку з більшим номером, і після цього відбувається перехід на крок 5; у іншій ситуації перехід на крок 7.

Крок 7. Очищується комірка з числом *B* та відбувається перехід на крок 6. Ознакою завершення обрахунків при прорахунку повної сім'ї числової лінійної послідовності є зустріч числа, яке є стартовим для числової лінійної послідовності в комірці $\frac{N+3}{2}$, при чому якщо воно відсутнє в попередніх

комірках при непарних значень N і або при подібній ситуації в комірці $\frac{N+2}{2}$ для парних значень N.

Алгоритм кодування за допомогою числових лінійок-послідовностей базується на визначені монолітного коду та кодуванні числа ASCII-формату. Запропонований алгоритм умовно поділити на три етапи:

– отримання даних та проведення ініціалізації;

– генерування кодової комбінації;

– збереження обчислени кодів чисел.

Серед особливостей, які можна відмітити, у запропонованомі алгоритмі можна відзначити поєднання алгоритмів аналізу та кодування цифрової інформації, що дозволили обримати відмінні результати, при невеликих часових затратах.

Запропонований алгоритм базується та припущеннях, що в тудь якій посділовності присутні ділянки, що мають повторювану структуру. Тобто декілька ділянок просто дублюються. А отже послідовності що повторюються можна замінити якимось меншим кодом, що дозволить зменшити кількість памяті для зберігання іноформації, а також спричинить її «спотворення», для того щоб даної інформацією не зміг скористатись сторонній користувач бз прав доступу.

Даний перелік кроків дозволяє повноцінно провести обробку та кодування цифрової інформації на основі заміни послідовності пар символів, при цьому часові затрати є мінімальними.

Серед переваг запропонованого алгоритму слід відмітити швидкодію та простоту реалізації. До недоліків: необхідність проведення процесу налаштування програми пред її роботою, оскільки для коректного та швидкого функціоування необхідно встановити початкові параметри роботи алгоритму.

2.5 Висновки до розділу

Проведно аналіз алгоритмів кодування цифрової інформції, що дало можливість обрати метематичне обгрунтування для проведення кодування інформції на основі заміни символьних пар.

Розроблено алгоритм проведення кодування інформції на основі заміни символьних пар, що дозволило розробити структуру програмної системи кодування/декодування цифрової інформації.

3 ПРОГРАМНА СИСТЕМА КОДУВАННЯ/ДЕКОДУВАННЯ ЦИФРОВОЇ ІНФОРМАЦІЇ

3.1 Структура програмної кодування/декодування інформації

Підприємства можуть інвестувати в сучасні засоби захисту, такі як брандмауери нового покоління, мікросегментація та інші інструменти, але навіть найкращі інструменти припускають, що порушення даних трапляються і мають на меті обмежити шкоду. А надсилання та отримання даних створює потенціал для ще більшої вразливості, оскільки зловмисники можуть перехоплювати передачу даних. Придбання та впровадження цих рішень вимагає величезних витрат, що може стримувати ріст ринку. Більше того, варіанти розгортання цих рішень трудомісткі та складні. Однак переваги, які вони пропонують, часто перевершують труднощі, пов'язані з хитросплетінням та часом, пов'язаним з їх розгортанням.

Підходячи до проектування та розробки кожної з програмнох систем прш за все її необхідно поділити на окремі функціональні модулі для спрощення процесу програмування, перевірки та тестування окремих функцій, та можливості розпарамелення самого процесупрограмування. Для цього таку програму розробляють вроздріб, які називаються програмними модулями. А сам такий метод розробки програмних додатків - модульним програмуванням. Програмний модуль - це будь-який фрагмент коду програми або опису процесу, що був оформлений як самостійний програмний блок, що може бути придатний для використання в описах процесу. Це означає, що кожен програмний модуль програмується, компілюється та налагоджували окремо від інших модулів програми, а отже він може бути фізично розділеним з іншими програмними блоками системи. Більш того, кожен розроблений програмний модуль може включатися до складу різних програм, якщо виконані умови його використання, декларовані в документації з цього модулю. Таким чином, програмний модуль може розглядатися і як засіб боротьби зі складністю програм, і як засіб

боротьби з дублюванням в програмуванні (тобто як засіб накопичення та багаторазового використання програмістських знань).

Модульне програмування є втіленням в процесі розробки програм обох загальних методів боротьби зі складністю: і забезпечення незалежності компонент системи, і використання ієрархічних структур. Для втілення першого методу формулюються певні вимоги, яким повинен задовольняти програмний модуль, тобто виявляються основні характеристики "хорошого" програмного модуля. Для втілення другого методу використовують деревоподібні модульні структури програм (включаючи дерева зі зрослими гілками).

При проектування структурної архітектури програмної системи був використаний ієрархічно-модульний підхід, оскільки даний підхід надає можливості максимально швидко проводити маніпуляції з окремими модулями системи: додавання, корегування, заміни та виділення обкремих структурних елементів програмної системи без страти її працездатності в цілому.

DRUID Coder - механізм шифрування інформації. Основна ідея шифрування полягає в тому, що при кодуванні інформації кожен раз виходить є повторюваною комбінація символів при використанні одного і того ж ключа. Також шифрування неможливо без наявності сертифіката. Сертифікат створюється унікальний, і може бути прив'язаний до накопичувача, тобто програма зможе правильно інтерпретувати інформацію, використовуючи прив'язаний сертифікат перебуваючи на одному і тому ж накопичувачі з сертифікатом, і якщо сертифікат був прив'язаний до даного накопичувача.

GraveStone - блочно-потоковий алгоритм шифрування. Розроблено та оптимізовано для шифрування великих обсягів інформації. Спектр шифрованого файлу близький до спектру білого шуму (залежить від обраного режиму алгоритму). Довжина ключа будь-яка. Ключів безліч. Колізія в ключах неможлива.

Спрощену структуру програмної системи наведено на рисунку 3.1.

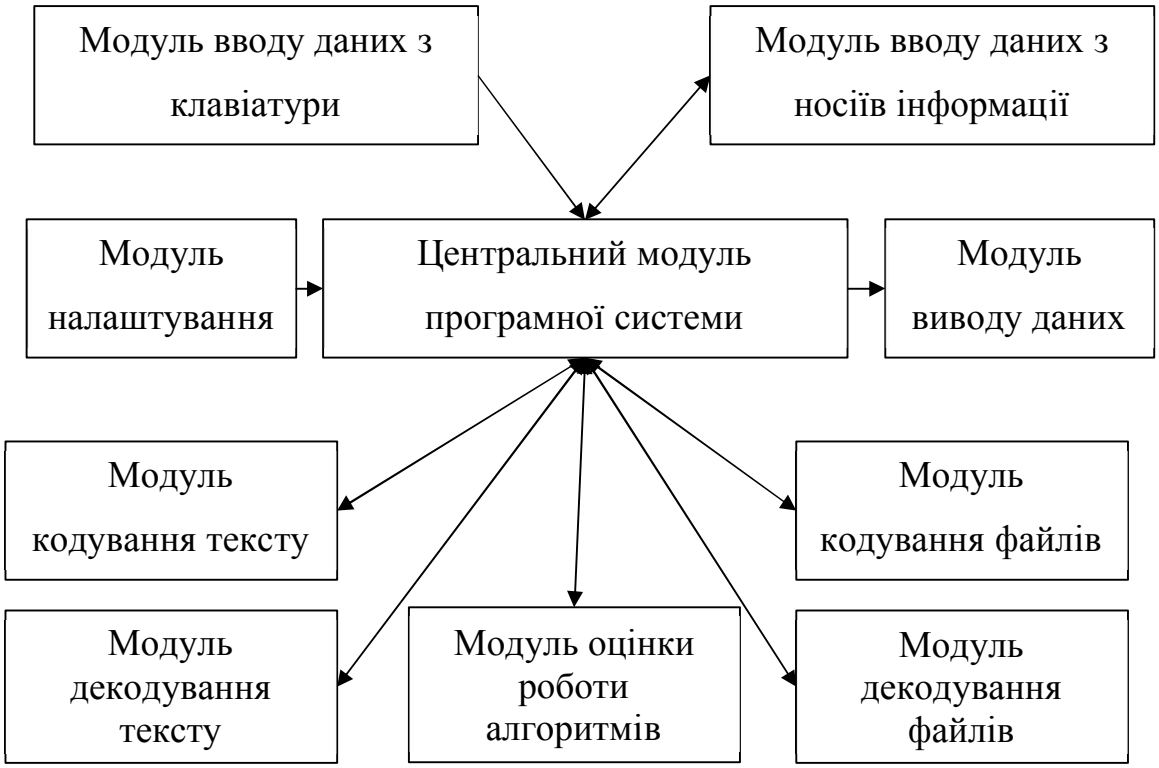

Рисунок 3.1 – Узагальнена структура програмного додатка аналізу та перетворення зображень

Запропонована структура є включає усі стандартні програмні модуля, які характерні дла програмних розробок даного класу. Серед основних модулів слід відмітити такі:

Центральний модуль програмної системи – даний програмний модуль забезпечує взаємодію між окремими структуриними елементами реалізованої програмної системи, до його функцій відносяться функції перевірки цілісності програмного додатку, наявність оновлень чи відключення пошкоджених блоків програми, контроль та переконвертація даних під час передачі інформації між окремими мобілями системи. Вхідними даними даного модуля є вектор параметрів роботи програми. Вихідними – корекційні правки пакетів обміну даних, повідомленя для користувача при необхідності внести тих чи інших змін в робту програмного додатку в цілому.

Модуль вводу даних з клавіатури – даний модуль має низький приорітет важливості, оскільки містить функції для отримання вхіних даних

безпосередньо в ручному режимі під час роботи програми. В ньому зібрані функції візуалізації набраного на клавіатурі тексту для його подальшого кодування.

Модуль вводу даних з насіїв інформації – дани програмний блок забезпечує можливість кодувати інформацію в файлаї які були створенні перед використанням програмного додатку та позволяє здійснити такі функції як: пеервірку наявності відповідного файлу, перевірку прав доступу до дадного файлу, можливість створення шифрованих файлів тощо.

Модуль налаштування – невеликий модуль функції якого забезпечую можливість налаштування програмної системи під потребу користувача, встановлення роботи самої системи та окремих її алгоритмів. Серед параметрів, які можна встановити слід відмітити: тип та розмір шрифтів інтерфейсу та модулі виводу звітів. Однією з головних його задач є створення та завантаження файлів з сертифікатами для моливості отримання унікальних ключів для кодування та декодування вхідної інформації.

Модуль кодування тексту – даний модуль маємаксимальний приорітет важливості в програмній системі, оскільки саме він проводить безпосередньо процес кодування інформації. На перших етапах роботи проводитсья перевірка та аналіз наяних параметрів кодування та перевірка коректності введеного тексту. Після чого вібуваютсья спроби знаходження відповідних парсимволів, щоб провести процедуру кодування. Час на даний етап робіт є невеликий, тому доцільність використання розпаралелення прироботі з програмною системою не викликає потреби. Оскільки в ручному режимів користувачі вводять невиликі обєми тексту. Результатом роботи даного модуля є закодована текстова інформація, яка будображена у відповідном текстовому полі.

Модулі кодування/декодування файлів – дана пара моділів приначена для роботи з файлами, що були створені різними користувачами позамежами роботи з програмним додатком. Особливістю є те, що даний модуль наперед має модливість для обробки великих масивів даних, оскільки кількість інформації, яка зерігається у тому чи іншому файлі є невідомою, а тому в для

робти цих модулів щоцільно продумти додатково алгоритми розпаралелення роботи. Після того як вхідний файл буде ідентифікаовано необхідно розбити його не менші блоки, щоб отримати можливість опрацювання файла в декілька потоків. Після того як кожен окремий блок буде опрацьовано вони всі об’єднуються в один файл, який може бути переданий засобами компютерних мереж або іншимиконалами передачі даних. При потребі отримати вміст файла у початковому форматі необхідно застосувати розроблений програмний додаток ще раз. При цьому слід памятати, що для успішної процедури декодування необхідно ввести тіж самі параметри, що були введені під час процесу кодування. Вкщо набір параметрів буде відмінний, то вхідний файл зазнає повторного кодування і втратить всю інформацію, що була закодована в нього.

Модуль перевірки роботи алгоритмів кодування/декодування – під час виконання поставлених задач, користувач може контролювати результати проміжних та кінцевих моментів робти програми. Для цього розроблена ціла система повідомлень та візуальних перемикачів, для своєчасної сигналізації про проблеми, що можуть виникнути. Окрім того після когдного етапу кодування або декодування програма виводить на екран спеціальні ключі для аналізу отриманих файлів з подальшою перевіркою даної інформації та моливістю корекції, якщо отриманий відправлений та отриманий файл з однаковим вмістом відрізняються за кодовим кючем.

Модуль виводу даних – призначений для відображення результатів роботи програми у вигляді графіків та додаткових сиситемних повідомлень. При цьому створюється лог-файл в якому занотовуються усі процеси, що відбулись в програмі, це дозволяє проаналізувати ефективність роботи самої програми.

Спроектована та розроблена ієрархічно-модульна структура програмного додатку аналізу та перетворення зображень у мовній мірі формує цілісну систему та має всі можливості для виконання поставлених завдань по отриманні, перетворенні,, аналізі попередній обробці та виводі результатів

обробки вхіної інформації. Окрім того використаний модульний підхід дозволяє реагувати на зміни в робті програмної системи додаючи наві модулі чи модіфікуючи вже присутні, при цьому цілісність та роботоздатність системи не буде пошкоджена.

Перед процесом реалізації розглянутих алгоритмів було проведено моделювання розробленого прогламного додатку за допомогою сучасних програмних пакетів. Серед переваг використання універсальної мови моделювання слід відмітити:

– можливість отримати графічне уявлення на високому рівні про те, як працює система з різних точок зору;

– оцінити доцільність використання тих чи інших модулів в структрі системи;

– проаналізувати інтерфейси між різними програмними модулями та можливість зменшення кількомті інформації, яка буде обмінюватись між ними.

– UML можна використовувати для моделювання практично будь-якого типу додатків, що працюють на будь-якому типі та комбінації апаратного забезпечення, операційної системи, мови програмування та мережі в UML.

– UML можна використовувати для моделювання проміжного програмного забезпечення, і це ефективно для моделювання великих, складних програмних систем

– Побудована на метамоделі Microsoft Operating Framework (MOF) для об’єктно-орієнтованого моделювання.

– Профілі UML (тобто підмножини UML, розроблені спеціально для конкретних цілей) допомагають природним чином моделювати транзакційні системи, системи в режимі реального часу та системи з толерантністю до несправностей.

Недоліки: найкраще працює з тим, що називається "великим заздалегідь дизайном", де дизайн та вимоги визначені до початку проекту - він не працює так само з підходом до гнучкого дизайну

– часові затрати. Потрібно багато часу, щоб діаграма була розумною і синхронізована з фактичним кодом. Діаграми UML не запускаються, але вимагають багато часу. Тому вони корисні тільки в тому випадку, якщо розробник може ними керувати.

На першому кроці моделювання проведено аналіз набору функцій та можливих дій користувача при роботі з розробленою системою при необхідності закодувати текстову інформацію. Приклад діаграми прецеденів наведено на рисунку 3.2.

Рисунок 3.2 – Діаграма прецедентів програмної системи

Як видно з наведено діаграми користувачу надаються обмежені можливості для проведення кодування в даному підході. Оскільки режим кодування текстової інформції є більше ознаюмлювальним та носить навчальний заракетр, тому кількість функціональних можливостей користувача обмежені. Проте це у повній мрі доволяє провести усі необхідні маніпуляції та

попередньо оцінити якість обраних параметрів для проведення кодування даних. Окрім того користувач може оцінити швидкість обробки даних та спрогнозувати час для оброкби файлу в цілому.

Для отримання більш детальної оцінки про роботу програми проведемо додаткове модулювання сиситеми при необхідності кодування цілого файла. Результати процесу аналізу та моделювання у вигляді діаграми прецедентів наведено на рисунку 3.3.

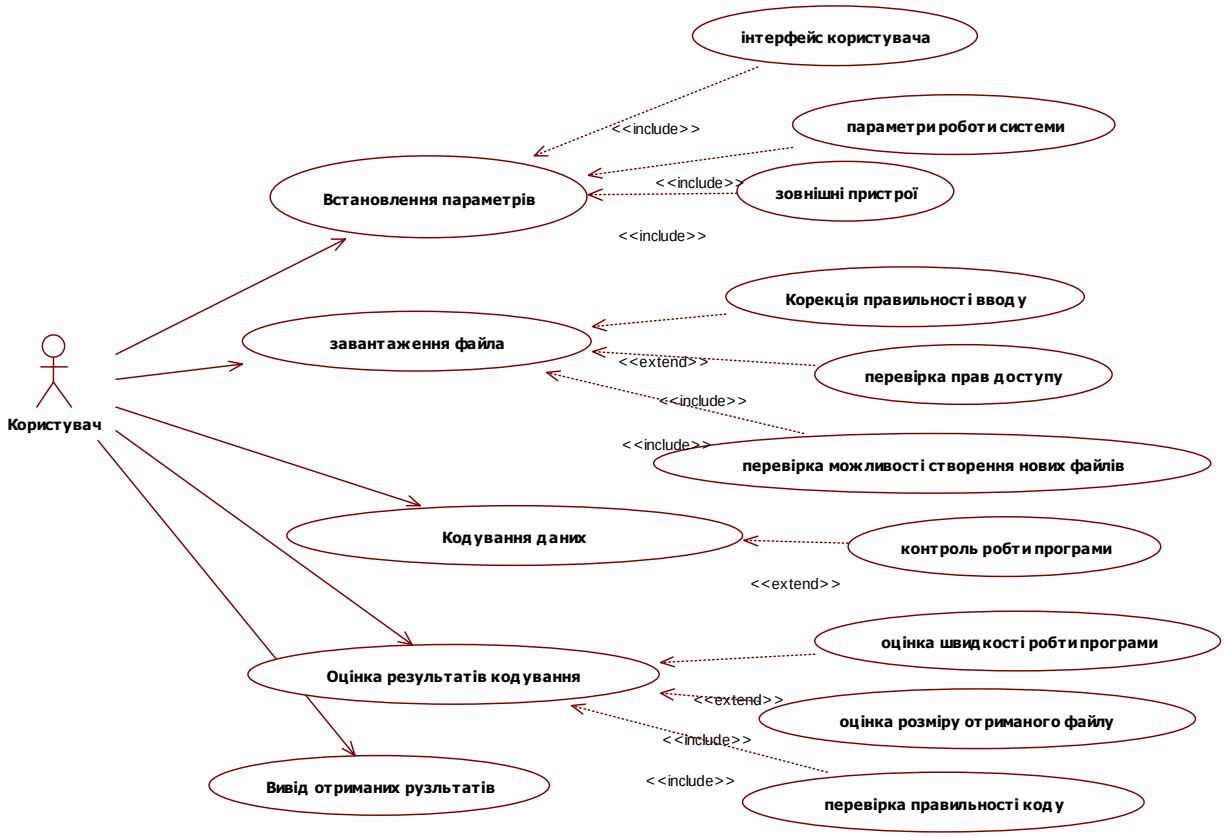

Рисунок 3.3 – Діаграма прецедентів кодування файлів

Як видно з наведених результатів моделювання в даному режимі робти користувчеві надається моливість більшої взаємодії з програмним засобом, оскільки для проведення перекодування програмного файлу необхідні знані ресурси які може надати програмна система, а крім того додатково необхідні ресурси, що може на дати операційна система (права доступу до файлової

системи, права доступу до користування компютерної мережею чи зованішніми носіями інформації).

Проте, достатньо високий рівень автоматизації роботи програмного додатку забезпечує можливість використання її для роботи та навчання користувачів без досвіду роботи. А сам процес опанування програмою є достатньо короткий та не ввимагає від нових кристувачів грибоких попередніх знань комп'ютерної техніки.

На другому кроці моделювання було проведено аналіз послідовності взаємодії між основними група процесів та об'єктів, що будуть відбуватись в процесі роботи програми. Результат моделювання наведено на рисуноку 3.4.

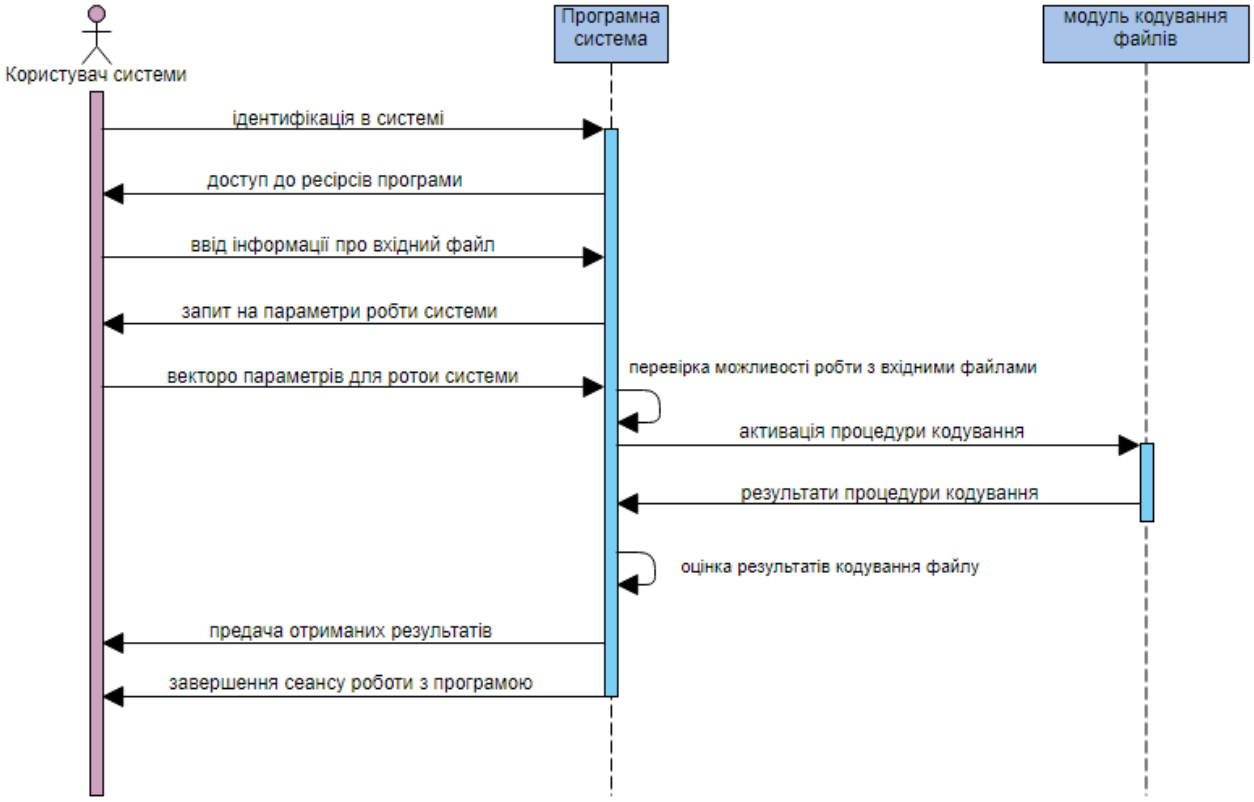

Рисунок 3.4 – Діаграма послідовності програмної системи

Як видно з наведеної діаграми, процес функціонування програмного додатку дмає послідовний характер та виконує усі дії послідовно, що з однієї сторони сповільнює процес опрацювання інформації, проте запропонований

алгоритм позволяє при наявності відповідних апаратних засобів провести процес розпаралелення задачі кодування/декодування даних та прискорити процедуру обробки інформації. Проте в даній реалізації для використання програмного додатку для його роботи буде достатньо робочої станції з середніми технічними показниками

На етапі моделювання було розглянуто основні фактори. Які можлуть впливати на рообту системиа також функції, що має виконати користувач для досягнення результату.

Іншим етапом проектування та розробки програмної системи є етам розробки користувацького інтерфейсу програмної системи. Під час проектування дизайну головного вікна програми були враховуні усі можливості для користувачів до швидкого доступу до окремих функцій та пунктів меню. А з іншої сторони вдалось уникнути значного завантаження головного вікна програми. При цьому основний простір на головному вікні було віддано областям для відображення вхідних даних та область для відображення результатів роботи програми. Результат проектування графічного інтерфейсу користувача наведено на рисунку 3.5.

При проектуванні зовнішнього інтерфейсу було враховано те, що програмна система повинна працювати з мінімальним втручанням ззовні, через те для користувача доступна мінімальна кількість активних функціональних елементів. Основне функції, що необхідно виконувати в процесі налаштування робjти програми реалізовані у вигляді інтерактивних перемикачів, що можуть змінювати налаштування робти програми. Мінімалізація кількості активних компонентів на екрані дозволила збільшити корисну площу на головному вікні програми, що в свою чергу позитивно вплинуло на зручність при роботі з програмним додатком. Іншою перевагою такого дизайну є можливість встановлення початкових параметрів роботи прямо на головному вікрі програми, тобто в користувача відпадає потреба переглядати усі пункти меню, для встановлення початкових значень робти програми, оскільки вони всі є одразу «під рукою» та можуть бути швидко налаштовані.

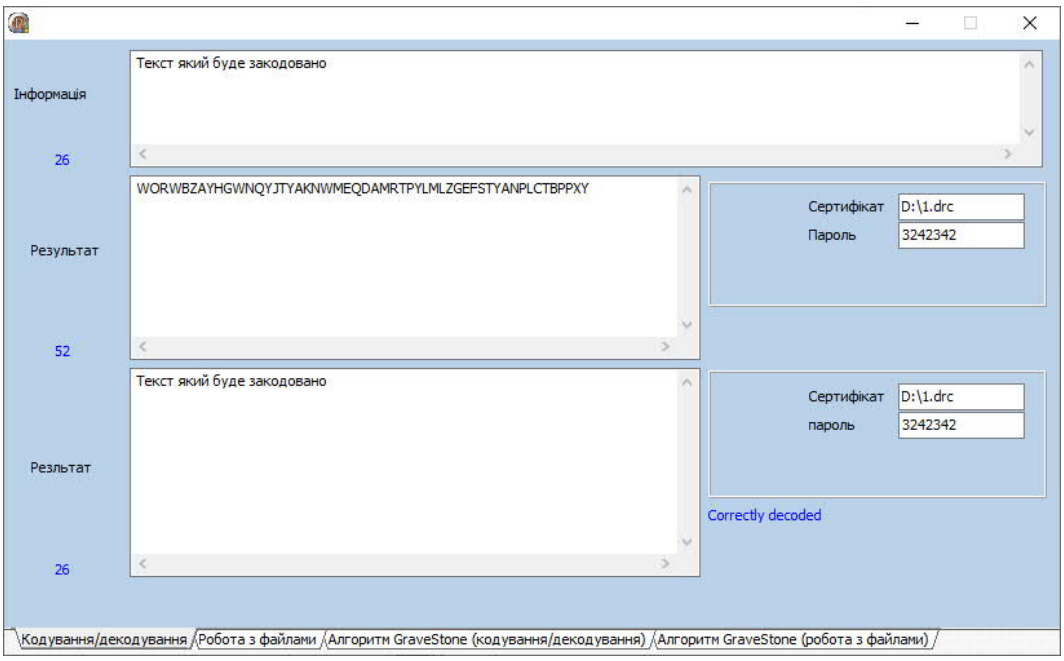

Рисунок 3.5 – Зовнішній вигляд програмної системи

Сам інтерфейс складається з деікількох складивих елементів:

– головного вікна програмної системи для візуалізації вхідних зображень. Дана область розділена на групу візуалізуючих компонентів, за допомого ю яких користувач може взаємодіяти з програмною системою та отримувати результат в залежності від поставлених задач. Дана область займає більшу часниту головного вікна, оскільки саме воне несе максимальне інформаційне навантаження при виоконанні функціональних задач;

– вікно налаштування та візуалізації параметрів рооботи системи та відповідних алгоритмів. В даній області розміщені основні перемикачі, активатори та поля для вводу пераметрів робти програми. Всі вони мають компактний вигляд, проте за допомогою них користувчеві надаєтсья можливість повністю вприлвати на результати роботи програми.

Даний зовнішній вигляд програмної системи достатньо зручний та швидкий в опануванні, що є безперечною перевагою під час роботи з програмним продуктом.

Для роботи з програмою користувачеві необхідно виконати просту послідовність кроків, візуально відображену на рисунку 3.6:

Рисунок 3.5 – Послідовність дій користувача при роботі з програмною системою кодування інформації

Як показано зі схеми дій користувача, при роботі користувач повинен виконати три основі дії з програмною системою, що включають активацію програмного додатку шляхом завантаження вхідних даних, встановдення параметрів роботи системи, оцінки отриманих результатів та перегляду отриманих результатів в процесі кодування інформації. Окрім того користувач може виконати ще дві додаткові дії, при необхідності внесення змін при негативному або не достатьо хорошому результаті роботи програми.

3.2 Програмні модулі системи кодування/декодування інформації

На сьогоднішній день на ринку присутня велика кількість програмних бібліотек для кодування інформації, що містять велику кількість функцій необхідних для провдення процедури обробки на різних стаціях його аналізу. Проте однією з найбільш підходящих для наших задач є бібіліотека Druid.

Бібліотека druid_gravestone.dll містить 2 повноцінних алгоритму шифрування: DRUID Coder і GraveStone.

DRUID Coder - алгоритм шифрування, призначений для генерації серійних ключів. Результатом кодування є кожен раз є повторюваною комбінація латинських букв і цифр, які можна ввести з клавіатури. Механізм

може бути налаштований таким чином, щоб його працездатність зберігалася тільки на обраному накопичувачі інформації. Робота алгоритму неможлива без наявності файлу сертифіката. Кожен сертифікат унікальний.

Приклади легкого шифрування слова «Барбаросса» по одному паролю і одному сертифікату наведено для прикладу можливостей даної бібліотеки та представлено на рисунку 3.6.

GRPEQPDKRRWSMRVHBNNQ

GRCROVDAOFBKNTFQRDHT

JAEYKTXBOKXEBAQHKPSJ

GRCRNZLXVPGCYRDXKQDY

JAEYKTXBOKWXQWBNTLQD

Рисунок 3.6 – Приклади шифрування текстової інформації

Для легкого режиму шифрування існує $6.2*10^{24}$ комбінацій паролів. Для посиленого режиму – $3.14*10^{89}$ комбінацій паролів. Тобто в ключах не виключена колізія.

У силу особливостей цього алгоритму на дешифрування входить час у більшій кількості, чим на шифруванні.

Серед недоліків даної реалізації:

– об'єм зашифрованої інформації у 2 рази більше вихідної;
– невисока швидкість алгоритма.

До перваг слід віднести:

– висока надійність;
– гнучкість використання;
– широка сфера застосування;
– ідеально для шифрування пісем.

GraveStone – блочно-поточний алгоритм шифрування. Розроблений та оптимізований для шифрування більших об'ємів інформації. Спектр

шифрованого файлу поблизу до спектру білого шуму (залежить від вибранного режиму алгоритма). Довжиина ключа довільна. Ключів необмежений масив. Колізія в ключах неможлива.

В алгоритмі передбачено 15 методів шифрування, різні за функціональністю і продуктивності. Інструментарій алгоритму містить функцію вичислення контрольної суми файлів. Недоліки:

– результат шифрування тільки бінарний.

Переваги реалізації:

– висока надійність і швидкість роботи;

– гнучкість

Зображення: на описаних вище алгоритмах можна побудувати систему будь-якої сложності для захисту інформації.

Для більш детального аналізу робти програмної розробки розглянемо декілька функції які були реалізовані. Функція crypt_file:

```
bool crypt_file_druid(char * input,char * output,char * File_cert, char * password,uc heavy_lite,uc direction,bool attached,bool process_mess,short *progress,int block);
```

Шифрує/дешифрує вказаний файл. Повертає false, якщо виникла помилка відкриття/збереження файлу, завантаження сертифіката або операції шифрування. Дана функція є головною в проекті оскільки виконує всі основні завдання, що ставляться користувачами пред системою. Окрім того для її функціонування необхідно активувати ще декілька додаткових (підготовчих) функцій, які будуть виконувати поставлені задачі по попередньому аналізу вхідних даних та встановлені параметрів роботи програми для отримання максимальної ефективності робти програми. Серед такиж функцій можна відмітити функції перевірки наявності відповідних файлів та прав доступу до них. Для оцінки її можливостей та аналізу вхідних параметрів проведемо огляд усії їх параметрів з описом функціонального призначення (таблиця 3.1).

Таблиця 3.1 – Параметри функції crypt_file

Параметр	Опис
char * input	Вказівник на ім'я вхідного файлу
char * output	Вказівник на ім'я вихідного файлу
char * File_cert	Вказівник на ім'я файлу сертифіката
char * password	Вказівник на ключ
uc heavy_lite	PW_HARD / PW_LITE
uc direction	КОДУВАТИ / ДЕКОДУВАТИ
bool attached	Фактор привязки до накопителю
bool process_mess	Вивести паралельні операції в додатках (істина)
short *progress	Указатель на процес виконання (прогрес задається від 0 до 1000)
int block	Розмір блока даних

```
bool druid_go(uc *source,uc *dest,long int len_source, char * password,uc heavy_lite,uc direction,bool process_mess);
```

Низькорівнева функція кодуваннння інформації. Повертає true, якщо операція пройшла успішно (таблиця 3.2).

Таблиця 3.2 – Параметри функції druid_go

Параметр	Опис
unsigned char * source	Вказівник на источник информации
unsigned char * dest	Вказівник на приемник информации
long int len_source	Довжина джерела інформації
char * password	Вказівник на ключ
uc heavy_lite	PW_HARD / PW_LITE
uc direction	КОДУВАТИ / ДЕКОДУВАТИ
bool process_mess	Вивести паралельні операції в додатках (істина)

Функція set_certificate_current_hard. Низькорівнева функція встановлення активного сертифіката для посиленого кодування. Немає параметрів, нічого не повертає.

Функція set_certificate_current_basic. Низькорівнева функція встановлення активного сертифікату для легкого кодування. Немає параметрів, нічого не повертає.

Низькоуровневі функції необхідні для того, щоб виключити часто повторювані операції при виклику функцій. Наприклад, щоб організувати підбір паролів, необхідно світи до мінімуму затрати на вичислення та отримання лішніх інструкцій.

3.3 Тестування та аналіз реалізованої системи

Для провдення оцінки розробленої програмної системи було проведено тестування розробленої програмної системи кодування/декодування цифрової інформації та порівняно результати з програмами аналогами. В процесі тестування використовувалась робоча станція з наступними технічними характеристиками (Таблиця 3.3):

Таблиця 3.3 – Основіні технічні параметри комп’ютера для тестів

Параметр	Значення
matherboard	Asus B350M-E 90
videocard	Asus PH-GTX1060-3G
МП	AMD Ryzen 3 2200G BOX 120
ОЗУ	8Gb 2400GHz
HDD	WD Caviar 1TB
Корпус + живлення	Zalman Z1 Black + Chieftec APS550SB

Анаранті параметри дисплея тестової робочої станції мають наступний перелік значень:

– діагональ монітора 19";
– тип матриці, що використовується- TN;
– роздільна здатність дисплея 1920x1080;
– наявні інтерфейси HDMI;
– частота відображення кадрів 65 Гц;
– відношення сторін дисплея 16: 9.

Технічні параметри комп’ютера в повній мірі забезпечують можливість проведення тестування спроектованого та реалізованого програмного додатку та дозволить отримати тестові вибірки для проведення аналізу коректності роботи програмної системи та порівняння її з програмами аналогами.

При проведднні тестування було здійснено два варіанта перевірки якості робти запропонованих алгоритмів, на першому етапі тестування проводилось на сонові введеної з клавіатури інформації. Даний етап був цікавий тим, що можна було оцінити роботу програми зневеекими об’ємати даних. Приклад роботи програми наведено на рисуноку 3.7.

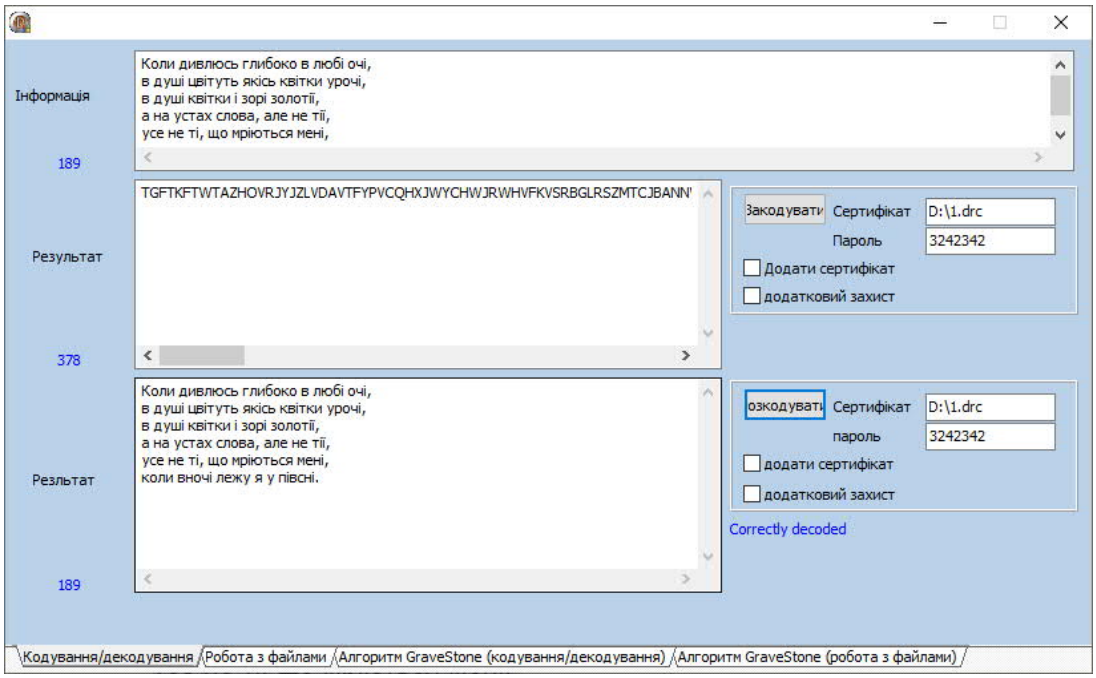

Рисунок 3.7 – Приклад кодквання/декодування текстової інформації

Як бачимоз начедених результатів кількість вхідних символів рівне 189, після кодування розмір набраного тексту збільшився майже у два рази та став 378 символів, проте післяя декодування було отримано туж саму взхідну інформацію.

Для перевірки можливості роботи алгоритму з інформацією написаною на різних мовах та з використанням інших символів, було поведено іншу групу експериментів. Приклад результатів даної групи наведено на рисунку 3.8.

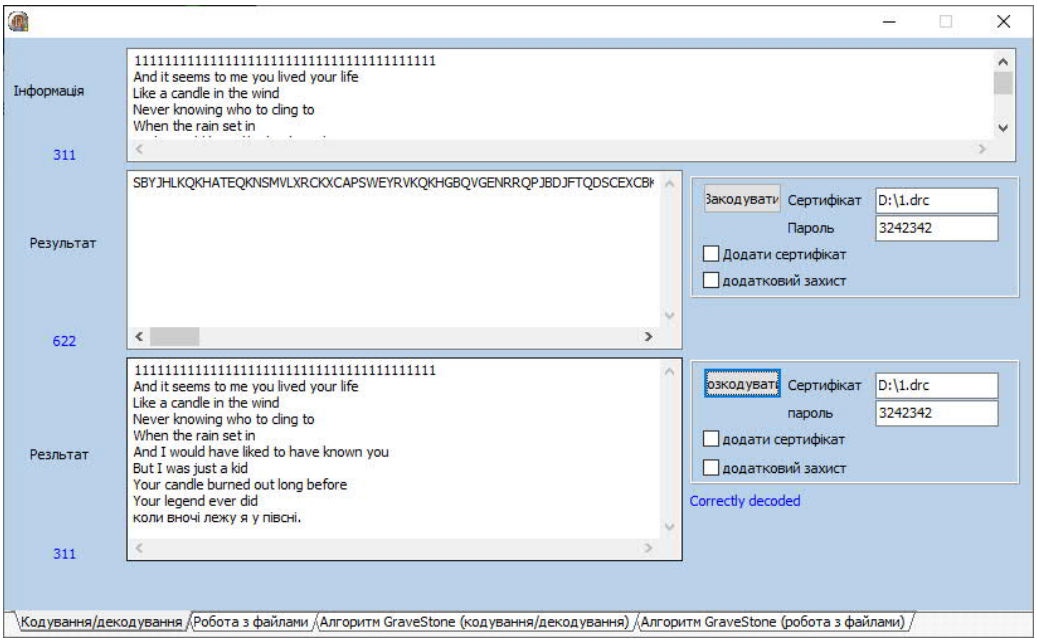

Рисунок 3.8 – Приклад кодквання/декодування текстової з використанням цифрових символів та букв іншого алфавіту

На основі даного експерименту можна зробити висновки, що запропоновані та реалізовані алгоритми у повній мірі виконують поставлені на них завдання. А збільшення обєму інформації повязане виключно з принципами, що були закладені в сам алгоритм. Прицьому аналіз часових затрат показав, що запропоновані алгоритми виконують поставлені завдання з прийнятні проміжки часу.

Для отримання інформації про можливості використання розроблених програмних модулів для кодування/декодування файлів було проведено ряд

експериментів, в якості вхідних даних використовуватись файли різного типу. Для простоти експерименту першими були протестовані файли з текстовою інформаціїю (рисунок 3.9).

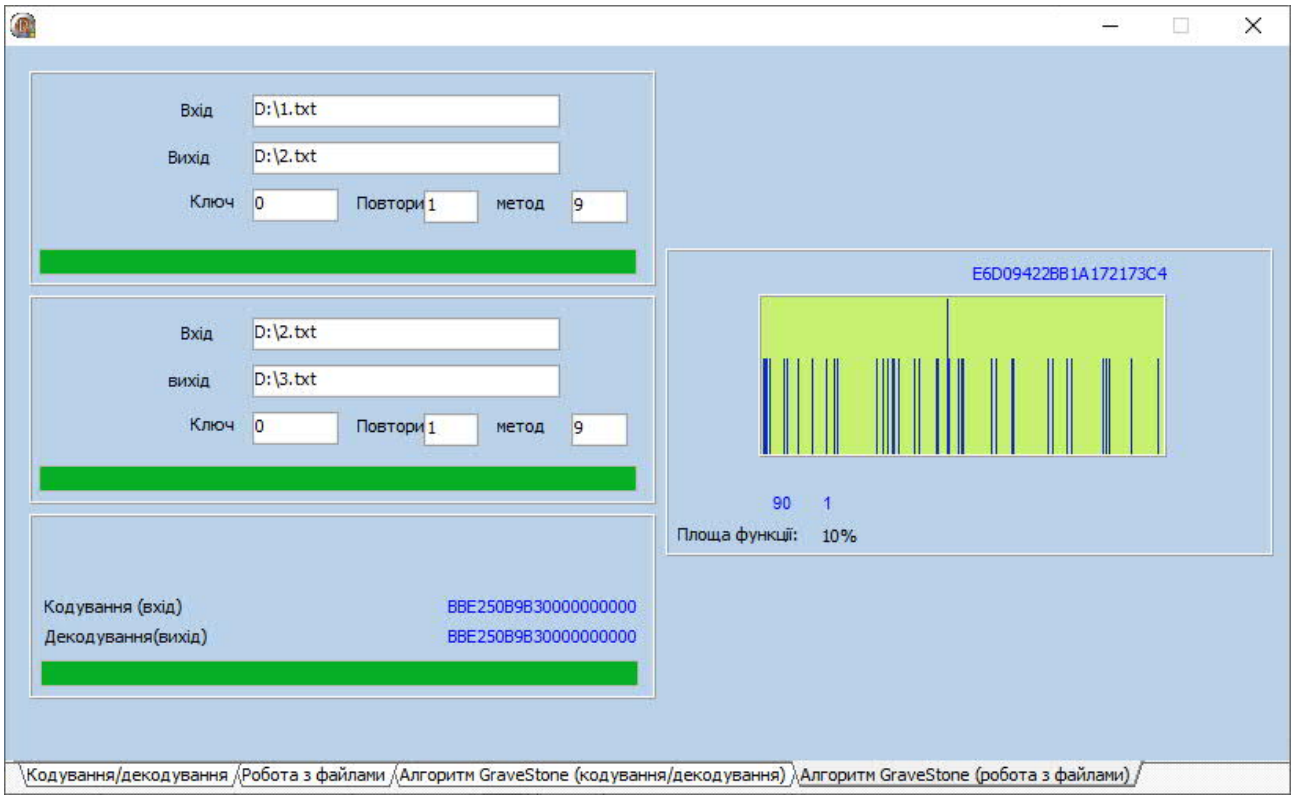

Рисунок 3.9 – Приклад кодквання/декодування текстової з використанням цифрових символів та букв іншого алфавіту

Для того щоб персвідчитись, що процес кодування декодування пройшов успішно перевіримо вміст файлів (рисунок 3.10).

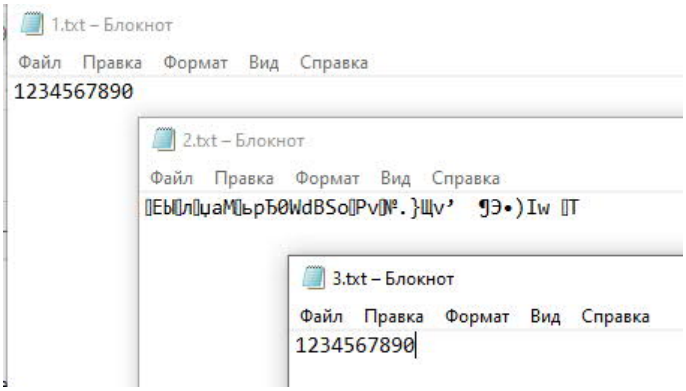

Рисунок 3.10 – Вміст файлів

В результаті візуальної перевірки було підтверджено, що дана реалізація запропонованих алгоритмів виконує завдання з необхідною точністю та дозволяє проводити зворотнє кодування без втрати інформації. Оскільки вміст першого та третього файдів є ідентичним. В другому файлі зберігається інформація про вигляд кодованого файла. Окрівм того аналізуючи резулльтати роботи лагоритмів, окрім візуального підтвердження, що відповдіні файли є ідентичними, додатково можна в цьому пересвідситись превіривши додаткові коди (рисунок 3.11).

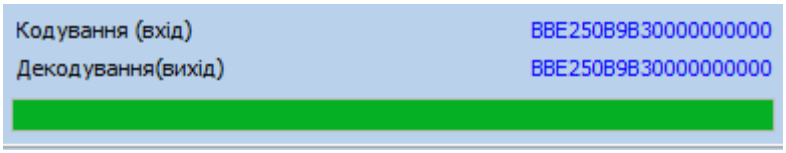

Рисунок 3.11 – Перевірка нонтрольних сум перед кодуванням та після декодування

З наведеного прикладу видно, що контрольні суми є ідентичними, що є додатковим свідченням, що файл перед кодуванням та після є тим самим. А оттже поставлена задача виконана в повній мірі.

Ще однією інформацією для користувачів при кодуванні інформації є гістограма розподілу частоти зустрічань повторювальних пар символів. Дану гістограму можна побачити після етапу аналізу вхідних даних (рисунок 3.12).

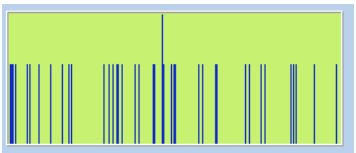

Рисунок 3.12 – Гістограма розподілу частори повторювальни елементів вхіднго файлу

Як видно з наведених вище прикладів, програмна система успішно виконує аналіз та обробку вхідних даних, проводить прорахунок та встановлює

параметри роботи. Додатково надає можливість користувачеві проводити налаштування процесу перетворення в ручному режімі.

В результаті проведених експерементів та на основі отриманих рещультатів можна стверджувати, що програмна система успішно проводить процеси кодування/декодування даних. Отримані результати підтвердили припущення, які були зроблені на етапі моделювання, що запропонований алгоритм перетворення має високу швидкодію, та пороводить обробку даних незадежно від їх типу, при цьому забезпечуючи відповідну якість результатів роботи.

В результаті проведених експериментів було виділено пари зображень дя проведення процесу морфінгу, а також встановлено параметри автоматичного налаштування початкових характеристик робти програмного додатку.

3.4 Висновки до розділу

Розроблено та проведено моделювання програмної системи кодування/декодування цифрової інформації, що дозволило програмно реалізувати систему обробки даних.

Проведно тестування розробленої програмної системи кодування/декодування, на базі різних тестових груп, що надало можливість проаналізувувати результати роботи алгоритмів та порівняти розробку з програмами аналогами.

ВИСНОВКИ

На основі аналізу сучасних програмних систем обробки цифрової інформації та аналізу алгоритмів кодування/декодування даних на основі заміни символьних пар можна розбити висновки:

1. Проведено дослідження та клафивікацію властивостей та сфер застосування цифрових даних, що дозволило виділити основні напрямки та алгоритми які використовуються кодуванні інформації.

2. Проаналізовано алгоритми обробки цифрових зображень, що дозволило вигначити групу алгоритмів для проведення пертворень цифрових зображень, а також їх перевоги та недоліки.

3. Проведено аналіз програмних додатків обробки цифрових дани який дозволив виділити головні структурні блоки, а встановити інтерфейси передачі даними між ними.

4. Проведно аналіз алгоритмів кодування цифрової інформції, що дало можливість обрати метематичне обгрунтування для проведення кодування інформції на основі заміни символьних пар.

5. Розроблено алгоритм проведення кодування інформції на основі заміни символьних пар, що дозволило розробити структуру програмної системи кодування/декодування цифрової інформації.

6. Розроблено та проведно тестування програмної системи кодування/декодування, на базі різних тестових груп, що надало можливість проаналізувувати результати роботи алгоритмів та порівняти розробку з програмами аналогами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Лавренюк Ю.О., Савчук В.В. Аналіз алгоритмів кодування інформації в системах обробки даних. Збірник тез III Науково-практична конференція молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі»., Тернопіль, 26 листопада 2020 р. с. 41.

2. Лавренюк Ю.О., Савчук В.В Аналіз алгоритмів розпізнавання цифрової інформації на зображеннях. Збірник тез III Науково-практична конференція молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі»., Тернопіль, 26 листопада 2020 р. с. 42.

3. Бартлетт Н. Программирование на Delphi Путеводитель. The Coriolis Group,Inc.,1996, Издательство НИПФ "ДиаСофт Лтд.",2016. 116с.

4. Вебер Дж. Технология С++в подлиннике. QUE Corporation, 2016, "BHV-Санкт-Петербург",2017. 256с.

5. Волш А. И. Основы программирования на С++для World Wide Web. IDG Books Worldwide,Inc.,1996, Издательство "Диалектика",2016. 458с.

6. Марков А. С. «Базы данных. Введение в теорию и методологию. Финансы и статистика». 2016. С.24-35.

7. Абрамов С. А. Задачи по программированию. М.: Наука, 2018. 256с.

8. Березин Б.И., Начальный курс Delphi. М.: ДИАЛОГ-МИФИ, 2016. 331с.

9. Бондарев В.М. Основы программирования. Харьков: Фолио, Ростов н/Д: Феникс, 2017. 446с.

10. Вирт Н. Алгоритмы и структуры данных. М.: Мир, 2019. 345с.

11. Гладков В. П. Задачи по информатике на вступительном экзамене в вуз и их решения: Учебное пособие. Пермь: Перм. техн. ун-т, 2014. 516с.

12. Грогоно П. Программирование на языке Delphi. М.: Мир, 2012. 216с.

13. Дагене В.А. 100 задач по программированию. М.: Просвещение, 2013. 106с.

14. Джамса К. Библиотека программиста Java. Jamsa Press, 2016, ООО "Попурри", 2016. 656с.

15. Марков А. С. «Базы данных. Введение в теорию и методологію. Финансы и статистика». 2016. Р. 24-35.

16. Заварыкин В.М. Основы информатики и вычислительной техники. М.: Просвещение, 2019. 556с.

17. Касаткин В. Н. Информация. Алгоритмы. ЭВМ. М.: Просвещение, 2011. 219с.

18. Кен А. Язык программирования Delphi. Addison-Wesley Longman,U.S.A.,1996, Издательство "Питер-Пресс", 2017. 378с.

19. Керниган Б. Язык программирования Delphi. Пер. с англ. М.: Финансы и статистика, 2012. 391с.

20. Ляхович В.Ф. Руководство к решению задач по основам информатики и вычислительной техники. М.: Высшая школа, 2014. 127с.

21. Мейнджер Дж. Delphi Основы программирования. McGraw-Hill,Inc.,1996, Издательская группа BHV, Киев, 2017. 346с.

22. Миков А. И. Информатика. Введение в компьютерные науки. Пермь: Изд-во ПГУ, 2018. 442с.

23. Могилев А. В. Информатика: Учеб. пособие для студ. пед. Вузов. М.: Изд. центр «Академия», 2019. 629с.

24. Нотон П. JAVA:Справ.руководство. М.:БИНОМ:Восточ.Кн.Компания, 2016: Восточ.Кн.Компания. 447с.

25. Нотон П. Полный справочник по Java. McGraw-Hill,1997, Издательство "Диалектика",2017. 556с.

26. Ренеган Э.Дж. 1001 адрес WEB для программистов :Новейший путеводитель программиста по ресурсам World Wide Web:Пер.. Минск:Попурри, 2017. 512с.ил.

27. Родли Дж. Создание Java-апплетов. The Coriolis Group,Inc.,1996, Издательство НИПФ "ДиаСофт Лтд.", 2016. 466с.

28. Секреты программирования для Internet на Java. Ventana Press, Ventana Communications Group, U.S.A., 2016, Издательство "Питер Пресс", 2017. 396с.

29. Семакина И. Г. Информатика. Задачник-практикум: В 2 т.. М.: Лаборатория Базовых Знаний, 2019. 476с.

30. Сокольский М.В. Все об Intranet и Internet. М.:Элиот, 2018. 254с.

31. Тассел Д. Стиль, разработка, эффективность, отладка и испытание программ. М.: Мир, 2011. 56с.

32. Тюрин Ю.Н. Анализ данных на компьютере. М.: ИНФРА-М, Финансы и статистика, 2015. 384с.

33. Флэнэген Д. Java in a Nutshell. O'Reilly & Associates, Inc., 1997, Издательская группа BHV, Киев, 2018. 473с.

34. Чен М.С. Программирование на С++:1001 совет:Наиболее полное руководство по Java и Visual J++ :Пер.с англ. Минск:Попурри, 2017. 640с.ил.

35. Эферган М. С++: справочник. QUE Corporation, 2017, Издательство "Питер Ком", 1998. 256с.

36. G. Yang. «Human face detection in a complex background. Pattern Recognition », 27 (1): 2014. P.53-63.

37. Kotropoulos C. «Acoustics, Speech, and Signal Processing», 2017. ICASSP-97, 2017. IEEE International Conference on pp.2537-2540 v. 4

38. Leung TK. «Finding Faces in Cluttered Scenes Using Random Labeled Graph Matching» 2015.p P.83-95.

39. Yow KC. Feature-based human face detection. Image and vision computing 15 (9), 2017. P.713-735.

40. Sinha, P. Perceiving and Recognizing threedimensional forms. PhD thesis, Massachusetts Institue of Technology, 2016. 278p.

41. Lanitis, A «Image Anal. Classifying variable objects using a flexible shape model »Image Processing and its Applications, 2015., P.70-74.

42. Viola P. «Rapid Object Detection using a Boosted Cascade of Simple Features», proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2001), 2011., vol. 1, 518p.

43. Jones MJ. «Robust real-time face detection», International Journal of Computer Vision, vol. 57, no. 2, 2014., P.137-154.

44. Buchatskiy AN. «Selection of the Optimal Color Space for Reducing False Positives Rate in the Viola-Jones Method», Актуальні проблеми інфотелекомунікацій в науці та освіті, II Міжнародна науково-технічна та науково-методична конференція. Санкт-Петербург, 2013.

45. Ethan R. ORB: an efficient alternative to SIFT or SURF. Computer Vision (ICCV), IEEE International Conference on. IEEE, 2011. P. 2564–2571.

46. Stefan L. BRISK: Binary Robust Invariant Scalable Keypoints. Computer Vision (ICCV), 2011. P. 2548–2555.

47. Pablo F. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In British Machine Vision Conference (BMVC), 2013.

48. Martin A. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. Of the ACM24: 2001. P.381–395,.

49. Патин М.В. Сравнительный анализ методов поиска особых точек и дескрипторов при группировке изображений по схожему содержанию. Молодой ученый. 2016. №11. С. 214-221.

50. Гонсалес Р. Цифровая обработка изображений в среде MATLAB //М.: Техносфера. 2016. 616с.

51. Березький О.М., Дубчак Л.О., Мельник Г.М. Методичні рекомендації до виконання кваліфікаційної роботи з освітнього ступеня “Магістр”. Спеціальність: 123 - Комп’ютерна інженерія. Магістерська програма - Комп’ютерна інженерія"/ Під ред. О.М. Березького. Тернопіль:ЗУНУ,2020.32 с.

52. Гураль І.В., Дубчак Л.О. Методичні вказівки до оформлення курсових проектів, звітів про проходження практики, випускних кваліфікаційних робіт для студентів спеціальності «Комп’ютерна інженерія»/Під ред. О.М. Березького. Тернопіль: ТНЕУ, 2019. 33 с.