УДК 681.3.07

АЛГОРИТМ КРИПТОГРАФІЧНОГО СТИСНЕННЯ ІНФОРМАЦІЇ ЗА ДОПОМОГОЮ ДЕРЕВА ШТЕРНА-БРОККО

Л.Я. Глинчук

Київський національний університет ім. Т. Г. Шевченка,
43020, Україна, Волинська область, Луцьк, пр-т. Відродження 12, кв. 526.
Тел.: 80966773646, e-mail: Lyda5@bigmir.net

Досліджується підхід до побудови алгоритму і сам алгоритм, який поєднує стиснення і кодування. Алгоритм базується на системі числення Штерна-Брокко (дерево). Наводяться результати застосування даного алгоритму в порівнянні з іншими алгоритмами.

Approach to the construction of algorithm and algorithm which combines a compression and code is probed. An algorithm is based on the scale of notation of Shterna-Brokko (tree). Results over of application of this algorithm are brought as compared to other algorithms.

Вступ

Одна з проблем, що вирішується при розробці експертних систем [1] – це ефективність та надійність роботи з великими обсягами даних. Що більше достовірної інформації про предметну область – тим краща експертна система. Зазвичай дані розміщуються у деякій базі знань експертної системи, яка знаходиться окремо від, власне, програмного коду. Тому для правильної роботи системи слід забезпечити не тільки правильність коду програми, але цілісність і правдивість бази знань. Ефективна ж робота з даними може бути досягнута, наприклад, на шляху мінімізації бази знань. Ще однією проблемою, особливо актуальною в сучасних умовах, є захист від несанкціонованого доступу до даних.

В роботі пропонується один з підходів до вирішення проблеми ефективності та захищеності, який полягає у застосуванні до бази знань алгоритму криптографічного стиснення, що має на меті зробити можливим одночасно захист і зменшення обсягу даних.

Теоретичні відомості

Деякі поняття. Стиснення полягає в тому, щоб перетворити повідомлення так, щоб його довжина стала менша. Наприклад, використаємо метод стиснення RLE (англ. Run Length Encoding), який полягає в тому, що замість послідовності однакових символів записується один символ і його кількість, нехай маємо, вхідну послідовність АААААААББББББДДДДДДДД, після застосування RLE – алгоритму будемо мати такий запис: (А,8),(Б,6),(Д,8).

Під криптографічним захистом розуміють перетворення інформації, в результаті якого вона стає недоступною для осіб, що не мають на це повноважень. Наприклад, перетворимо ту саму послідовність АААААААББББББДДДДДДДД так: букву А замінимо на *, Б на !, Д на +, після заміни отримаємо: ********!!!!!!++++++++

Криптографічне стиснення має на меті зробити можливим одночасно захист інформації і зменшення обсягу. Наприклад, послідовність АААААААББББББДДДДДДДД запишемо так: (А,В,Д) – це буде результат стиснення і (8,6,8) – ключ, без якого вихідне повідомлення не можна буде перетворити до вхідного.

Що собою являє дерево Штерна-Брокко (або система числення)? Як відомо [2], дерево Штерна-Брокко являє собою сукупність всіх нескоротних позитивних дробів *m/n*. Будується так: починаючи з двох дробів 0/1 і 1/0, потрібно кілька разів виконати операцію вставки дробів $(m+m')/(n+n')$ між двома сусідніми дробами *m/n* і *m´/n´*. Після першого кроку виходить послідовність 0/1, 1/1, 1/0, після другого – 0/1, 1/2, 1/1, 2/1, 1/0 і так далі. Ось так будуть виглядати верхні рівні дерева (рис. 1).

Як дерево пов'язане з криптографічним стисненням? Рух по лівій вітці позначатимемо буквою *L*, а по правій – *R*. Наприклад, позначення LRRL відповідає дробу 4/7. Ідея алгоритму полягає в тому, щоб замість букв ставити "0" і "1". "0" – відповідає букві *L*, "1" – букві *R*. Тоді дробу 4/7 буде відповідати двійковий номер 0100. Дробу 3/7 буде відповідати – 0011, дробу 7/5 – 1001, дробу 5/4 – 1000 і т.д. Далі перетворюємо двійковий номер у десяткове число згідно правила переведення, отримаємо: $0100_2 = 0*2^3 + 1*2^2 + 0*2^1 + 0*2^0 = 4_{10}$ – це число у десятковій системі числення, тобто дробу 4/7 ми можемо поставити у відповідність число 4. Аналогічно для решти дробів.

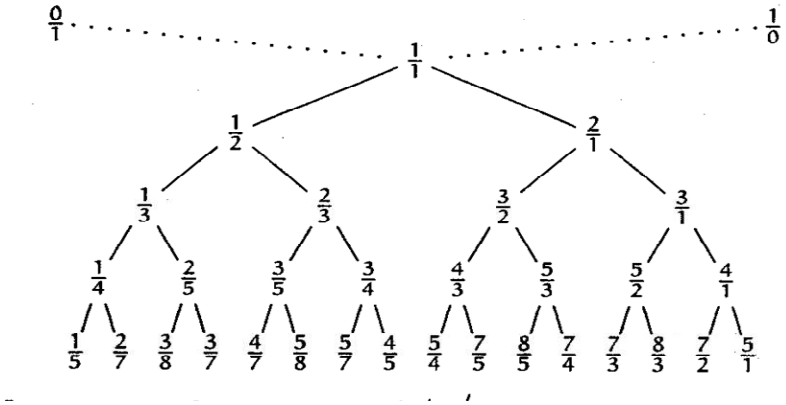

Рис. 1

Алгоритм криптографічного стиснення

Загальний алгоритм (для одного блоку байт, де блок = 2^n байт):

1. Вхідній послідовності символів (під символами розуміємо всі можливі символи, цифри, букви і ін.) A поставити у відповідність числа за якимось законом, правилом. Обов'язкова умова: вхідна послідовність має бути рівна 2^n байт. Кількість рівнів перетворення буде *n*.

2. Кількість кроків *k_krok=n*; кількість елементів вхідної послідовності *k_el*= 2^n .

3. $i := 1,$ пару сусідніх чисел A_i , A_{i+1} розглядати як дріб $\frac{A_i}{A_{i+1}}$ (або підібрати випадковим чином чисельник і знаменник, отримаємо новий дріб $\frac{A_i^{'}}{A_{i+1}^{'}}$).

4. $j := 1,$ від даних чисел відняти отримані, тобто $KL_{j,i} = A_i - A_i^{'}$, $KL_{j,i+1} = A_i - A_{i+1}^{'}$ (де $j = \overline{1,n}$, $i = \overline{1,2^n}$) і занести в матрицю *KL*, яка і буде матрицею-ключем (матриця *KL* має 2^n – стовпців, *n* – рядків, але вона не буде повністю заповнена: кожен її рядок буде містити у 2 рази менше елементів від попереднього, решту елементи – нулі).

5. Парі отриманих з п. 4 чисел (дробу) поставити у відповідність двійкову послідовність *Dv* за деревом Штерна-Брокко.

6. Двійкову послідовність *Dv* перетворити у десяткове число *Des* за алгоритмом переведення з двійкової системи числення у десяткову.

7. Десяткове число *Des* записати у проміжний масив, оскільки воно переходить у інший рівень

8. $i := i + 2.$

9. Виконати п.п. 3–8 *k_el=k_el/2* разів, тому, що кожен раз береться по 2 елементи.

10. Десяткові числа, що були записані у проміжний масив у п. 7 переходять як вхідна послідовність до п. 3.

11. $j := j + 1.$

12. Виконувати п.п. 3–11 *k_krok* разів.

13. Після виконання п.п. 1–10, отримаємо *REZULTAT* = одне число і матрицю-ключ *KL*.

14. Кінець алгоритму.

Розглянемо умови коректності роботи алгоритму, які сформулюємо у вигляді тверджень.

Твердження 1. Вхідна послідовність символів має бути рівна 2^n байт.

Пояснення: Оскільки в процесі алгоритму завжди береться по 2 числа, для того щоб утворити дріб, тоді кожен рівень перетворення повинен мати 2^n кількість вхідних символів, де *n* на кожному рівні стає на одиницю меншим. (Наприклад, нехай вхідна послідовність має $16 = 2^4$ символів, тоді на другому рівні послідовність стане в 2 рази меншою, тобто $8 = 2^3$, на третьому – $4 = 2^2$ і на 4-у – $2 = 2^1$; якщо вхідна послідовність має 17 (непарне) символів – вже маємо одне число лишнє, якщо вхідна послідовність = 6 – парна, тоді на другому кроці отримуємо лишнє число).

Твердження 2. Числа A_i , A_{i+1}, які розглядають як дріб $\frac{A_i}{A_{i+1}}$, мають задовольняти умовам:

$$A_i > A_{i+1} \ . \qquad (1)$$

$$\text{дріб } \frac{A_i}{A_{i+1}} \text{ – нескоротний, тобто НСД}(A_i,\ A_{i+1}) = 1. \quad (2)$$

Пояснення: а) ми ввели таке позначення “1” відповідає букві R, тобто ті дроби, які стоять по праву сторону дробу 1/1, починають свій двійковий запис з “1”. Якщо двійковий запис починається з одиниці, то при перетворені з десяткового числа у двійкове отримаємо точно попередній запис двійкового, чого не можна сказати про двійкові послідовності, які починаються з “0”. Наприклад, дробу 4/7 буде відповідати двійковий номер 0100, $0100_2 = 0*2^3 + 1*2^2 + 0*2^1 + 0*2^0 = 4$, тобто дробу 4/7 відповідає число 4. Нехай якомусь іншому дробу відповідає двійковий номер 00100, при перетворенні $00100_2 = 0*2^4 + 0*2^3 + 1*2^2 + 0*2^1 + 0*2^0 = 4$, знову дробу відповідає 4, тобто кількість нулів до одиниці не має значення. При дешифруванні нам потрібна однозначність, а якщо уважно подивитися як утворюється дерево Штерна-Брокко, то можна помітити, що ті дроби, які стоять по праву сторону дробу 1/1, мають особливість: чисельник більший від знаменника;

б) дерево Штерна-Брокко являє собою сукупність всіх нескоротних позитивних дробів *m/n*, тобто НСД(A_i, A_{i+1}) = 1.

Твердження 3. Для вхідного блоку розміром 2^n байт ключ буде

$$\le 2^n + 2^n \cdot \sum_{i=1}^{n-1} \frac{1}{2^i} = 2^n \left(1 + \sum_{i=1}^{n-1} \frac{1}{2^n}\right) \text{ байт.} \quad (3)$$

Пояснення: спочатку додаємо 2^n, тому, що вхідна послідовність має стільки ж символів, і якщо припустити, що для всієї послідовності умови (1–2) не виконуються, то всі числа міняються, відповідно ключ має 2^n елементів, а далі кожен раз додаємо кількість поділену на 2, тому, що на кожному рівні послідовність зменшується у 2 рази, додаємо *n*-1 раз, бо потрібно рахувати елементи ключа на кожному рівні, а один рівень ми включаємо додавши на початку 2^n.

Алгоритм дефрування-розпаковування

Загальний алгоритм:

1. Вхідний REZULTAT – число записати у проміжну матрицю TPROM. KX1:= 1 – кількість елементів матриці TPROM; KR:= кількість рядків матриці-ключа KL.
2. J:= 1 – кількість елементів вихідної матриці, KX:= 1 – біжуча кількість елементів матриці TPROM.
3. Перетворити елемент матриці TPROM з десяткової системи числення у двійкову, отримуємо Dv – двійкову послідовність.
4. Двійковій послідовності *Dv* поставити у відповідність дріб *m/n* за деревом Штерна-Брокко.
5. Утворити вихідний масив TVUX так: TVUX[J]:= *m* + *KL*[*KR*,*J*]; TVUX[*J*+1]: = *n* + *KL*[*KR*,*J*+1].
6. J:= J+2 – оскільки кожен раз утворюється два елементи матриці TVUX ; KX:= KX+1 – оскільки розглядаємо один елемент матриці TPROM.
7. Виконувати п.п. 3–6 доти KX ≤ KX1 (поки не розглянемо всі елементи матриці TPROM).
8. KX1:= J–1.
9. Переписати масив TVUX у TPROM.
10. KR:= KR–1.
11. Виконати п.п. 2–10 поки KR не стане рівним 1.
12. Перетворити вихідний масив за протилежним правилом, або законом який використовується в алгоритмі криптографічного стиснення.
13. Кінець алгоритму.

Програмна реалізація алгоритмів

Алгоритми реалізовані в програмній оболонці Shans 1.0, яка створена спеціально для організації і заповнення експертних систем діагностики. Для окремої експертної системи створюється 2 файли (база знань): один файл містить назву, власника, пароль, об’єкти конкретної області знань, можливі питання і відповіді; другий файл містить шифр діагнозу, який створюється тоді, коли користувач дає відповіді на питання, закладені в базу, і сам діагноз. Дуже важливо передбачити усі можливі діагнози. Кожен файл можна заповнити як вручну, так і за допомогою спеціальних передбачених форм. Система передбачає проведення діагнозу тільки для того користувача, який знає пароль. Файли експертних систем захищаються і стискаються за допомогою розроблених алгоритмів. При проведені тестування алгоритму криптографічного стиснення були використані файли різного розміру, результатом застосування першого алгоритму був файл чисел – з кожного вхідного блоку одне число, як і передбачалось, і матриця-ключ. Можна не розбивати файл на блоки, а застосовувати

алгоритм до цілого файлу відразу і отримати в результаті одне число і одну матрицю-ключ, питання лише в тому чи вистачить можливостей?

У таблиці наведено для порівняння дані інших криптографічних алгоритмів з секретним ключем [3] Про порівняння зі стисненням можна сказати тільки одне, що результатом стиснення є одне число для окремого блоку (≤ 4 байт або $\leq 4 * 8$ біт).

Таблиця

Алгоритм	Розмір вхідного блоку (біт)	Розмір вихідного блоку (біт)	Розмір ключа (біт)
BLOWFISH	64	64	448
DES	64	64	56
FEAL	64	64	64
IDEA	64	64	124
RC5	32 або 64 або 128	32 або 64 або 128	від 0 до 2040
Новий алгоритм (АКС)	$2^n \cdot 8$, де $n \geq 1$	≤ 32	$\leq (2^n + 2^n \cdot \sum_{i=1}^{n-1} \frac{1}{2^i}) \cdot 32$
АКС (конк. дані 1)	64 (n=3)	≤ 32	≤ 448
АКС (конк. дані 2)	128 (n=4)	≤ 32	≤ 960

Висновки

У даній роботі досліджено і розроблено алгоритм криптографічного стиснення і протилежний до нього алгоритм, на основі дерева (системи числення) Штерна-Брокко. Алгоритм має як свої переваги, так і недоліки.

Переваги:

- виконується одночасне стиснення і шифрування (захист), що економить час застосування;
- результатом завжди буде менша кількість байт (≤ 4 для одного блоку вхідної послідовності) у порівнянні, як з алгоритмами стиснення так і з криптографічними алгоритмами, тобто стискається максимально;
- для одного і того ж блоку кожен раз генерується інший ключ;
- ключ генерується випадковим чином;
- довжина ключа говорить про криптографічну стійкість алгоритму.

Недоліки:

- оскільки ключ шифрування $\geq$ довжини вхідного блоку, то для його зберігання слід також виділяти пам'ять;
- необхідно генерувати для кожного сеансу зв’язку новий ключ і своєчасно його передавати.

1. Экспертные системы. Принципы работы примеры. Под ред. Р. Форсайта. – М.: Радио и связь, 1987. – 224 с.
2. *Грэхем Р., Кнут Д., Паташник О.* Конкретная математика. Основание информатики. – М.: Мир, 1998.
3. *Грездов Г. Г.* Современные методы криптографической защиты информации (обзор по материалам открытой печати). – К.: 2002. – 32 с.