Метод построения алгоритма приведения по фиксированному модулю

Национальный авиационный университет

Аспирант кафедры БИТ: Мария Ковтун

Руководитель: доцент кафедры БИТ, к.т.н. Сергей Гнатюк

Содержание

- Введение
- Актуальность
- **Побитовый** метод приведения
- **Пословный** метод приведения
- **Построение пословного** метода
- Замеры производительности
- Выводы

Введение

Цель: разработка метода построения алгоритма приведения по фиксированному модулю

Объект: процесс приведения по фиксированному модулю

Предмет: неприводимые полиномы заданные в стандартах ДСТУ 4145:2002 и ДСТУ 7624:2014.

Актуальность

<table>
<tr><td>Криптопреобразования</td><td colspan="2">Зашифровывание/ расшифровывание</td><td colspan="2">Формирование и проверка цифровой подписи</td><td colspan="3">Обмен ключами</td></tr>
<tr><td rowspan="2">Арифметика в группе точек эллептической кривой</td><td colspan="5">Скалярное умножение точек эллиптической кривой</td><td colspan="2" rowspan="2">Генерация случайной точки</td></tr>
<tr><td colspan="3">Сложение точек</td><td colspan="2">Удвоение точки</td></tr>
<tr><td>Арифметика в поле GF(2^m)</td><td>Умноже-ние</td><td>Сложе-ние</td><td>Деление</td><td>Возведение в квадрат</td><td>Приведе-ние по модулю</td><td>Инверти-рование</td><td>Извлечения квадратного корня</td></tr>
<tr><td>Операции над массивами</td><td colspan="2">Сдвиг</td><td colspan="2">Сравнение</td><td>Сложение</td><td>Вычитание</td><td>Умножение</td></tr>
<tr><td>Команды CPU</td><td colspan="7">mov, mul, shr, shl, add, sub</td></tr>
</table>

Актуальность

Криптопреобразования	Режимы блочного симметричного шифрования (XTS, ...)				
Арифметика в поле **GF**(2^m)	Умножение	Сложение	Приведение по модулю	Возведение в степень	Инвертирование
Операции над массивами	Сдвиг	Сравнение	Сложение	Вычитание	Умножение
Команды CPU	mov, mul, shr, shl, add, sub				

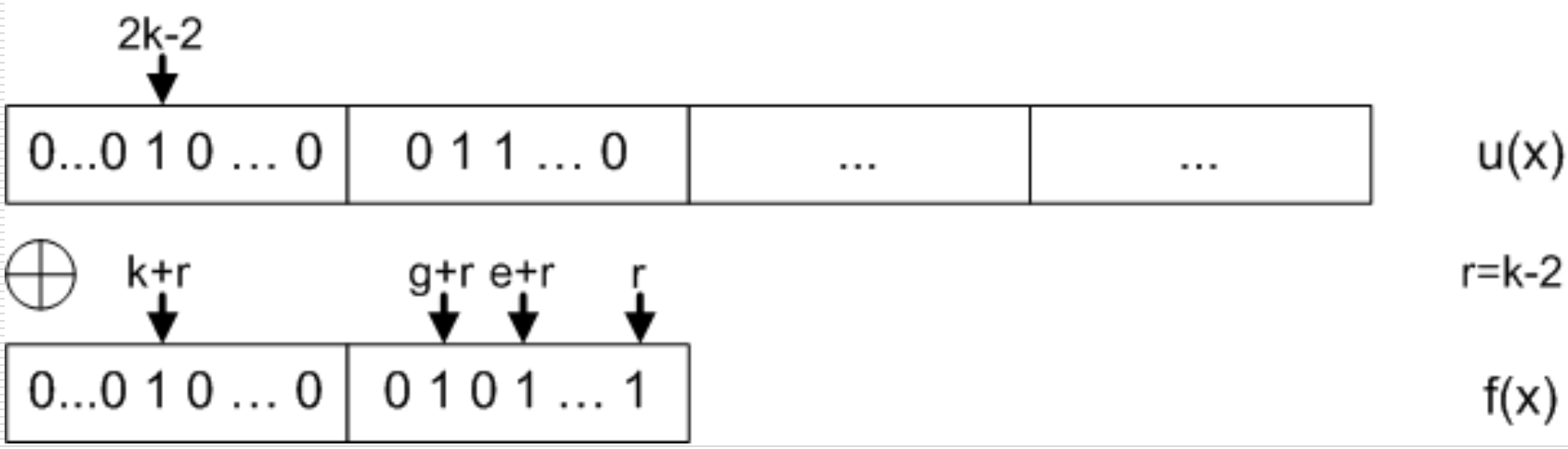
Актуальность

- Приведение по модулю
  - Универсальный (побитный)
    - Медленный
    - Произвольный модуль
  - Специализированный (пословный)
    - Быстрый
    - Фиксированный модуль
    - Отсутствует формализованное описание для построения (как «искусство»)

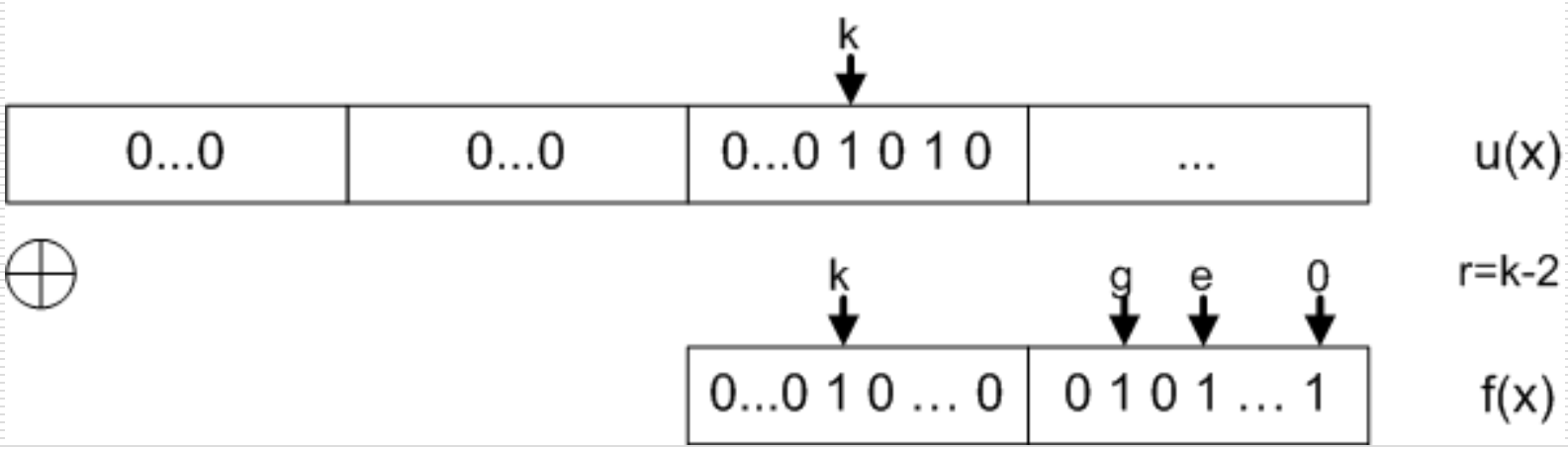
Известный метод

ПОБИТОВЫЙ МЕТОД ПРИВЕДЕНИЯ ПО МОДУЛЮ

Побитовое приведение по модулю



Побитовое приведение по модулю



Особенности

- Произвольный неприводимый полином
- Происходит:
  - k-2 проверки на ненулевой бит.
  - в среднем, (k-2)/2 операций сложения по модулю 2.
  - При каждом сложении по модулю 2, складывается $\lceil k/w \rceil$ машинных слов.

Метод-прототип

ПОСЛОВНЫЙ МЕТОД ПРИВЕДЕНИЯ ПО ФИКСИРОВАННОМУ МОДУЛЮ

Пословный метод (пример)

Вход: полином c(x), степени 324.

Выход: c(x) mod f(x), $f(x)=x^{163}+x^7+x^6+x^3+1$, w=32.

1. For i=10 downto 6
   1.1. T ← C[i].
   1.2. C[i-6] ← C[i-6]⊕(T<<29).
   1.3. C[i-5] ← C[i-5]⊕(T<<4)⊕(T<<3)⊕T⊕(T>>3).
   1.4. C[i-4] ← C[i-4]⊕(T>>28)⊕(T>>29).
2. T ← C[5] AND 0xFFFFFFF8.
3. C[0] ← C[0]⊕(T<<4) ⊕(T<<3) ⊕T⊕(T>>3).
4. C[1] ← C[1] ⊕(T>>28)⊕(T>>29).
5. C[5] ← C[5] AND 0x00000007.
6. Return ((C[5], C[4], C[3], C[2], C[1], C[0])).

Предложенный метод

МЕТОД ПОСТРОЕНИЯ АЛГОРИТМА ПРИВЕДЕНИЯ ПО ФИКСИРОВАННОМУ МОДУЛЮ

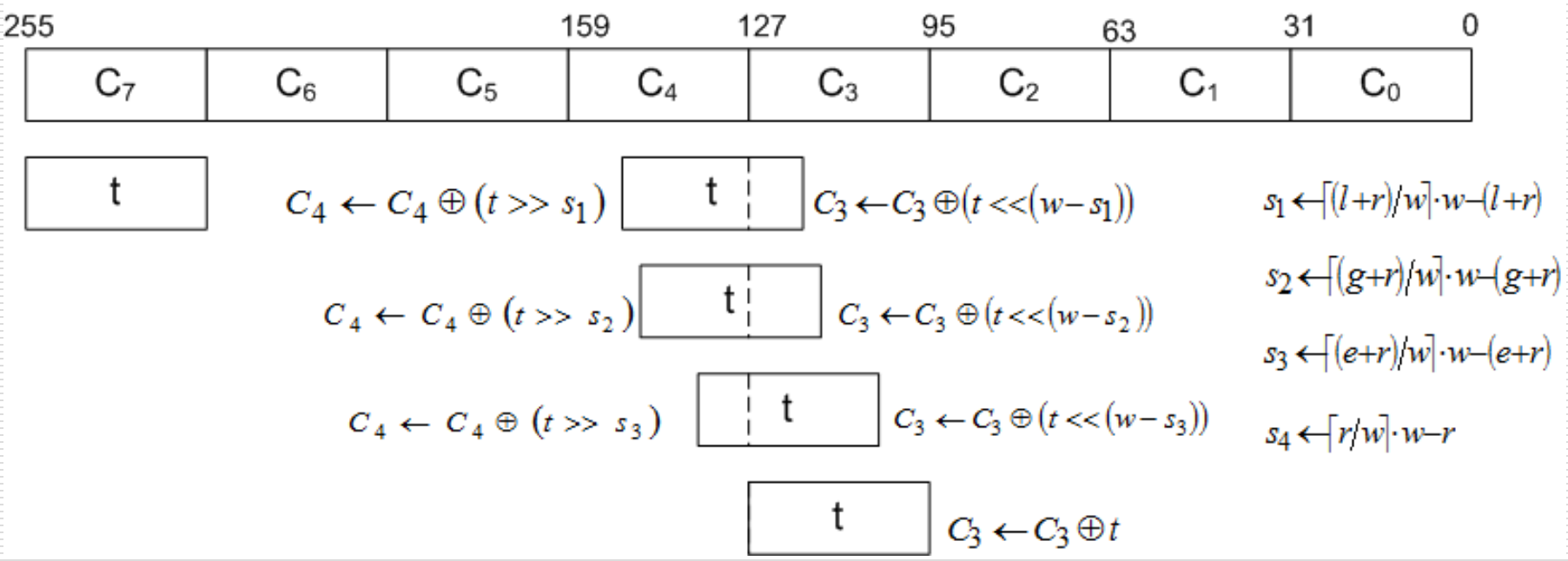
Предложенный метод построения

- $f(x) = x^k+x^l+x^g+x^e+1$, $k>l>g>e>1$
- Максимальная степень полнома 2k-2, который следует привести
- $r \leftarrow \lceil 2\cdot(k-1)/w \rceil \cdot w-k$ - размер сдвига, для выравнивания, w = 32.
- Диапазон битов $(x_{k+r}, x_{k+r-1},\dots, x_{k+r-(w-1)})$ складываем по модулю с укзанными битовыми диапазонами

$(x_{l+r}, x_{l+r-1},\dots, x_{l+r-(w-1)})$, $(x_{g+r}, x_{g+r-1},\dots, x_{g+r-(w-1)})$, $(x_{e+r}, x_{e+r-1},\dots, x_{e+r-(w-1)})$, $(x_r, x_{r-1},\dots, x_{r-(w-1)})$.

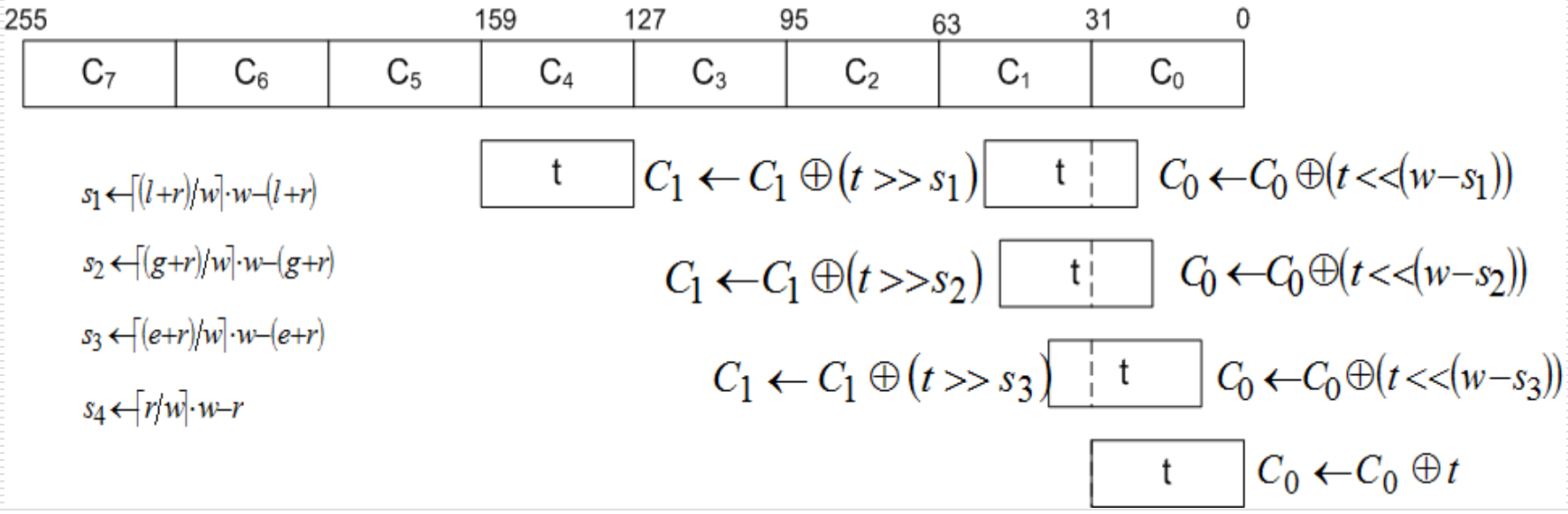
Предложенный метод построения

$f(x)=x^{128}+x^{7}+x^{2}+x^{1}+1$



Предложенный метод построения

$f(x)=x^{128}+x^7+x^2+x^1+1$



Замеры проивзодительности

СРАВНЕНИЕ

Условия экспериментов

- Язык: C++
- ОС: Microsoft Windows 7 SP1 x86-64
- Компилятор: Microsoft C++ x86/x86-64
- Число итераций: 1 млн.
- К1: Intel Core i5-3570 on Windows 7 SP1 x86-64
- К2: Core i5-4530 on Windows 7 SP1 x86-64

Замеры производительности

Алгоритм	Время, мкс	
	K1 (x86)	K2 (x86)
S [128]	0,009327	0,008259
U [128]	0,324494	0,287312
S [256]	0,015098	0,013414
U [256]	1,151691	1,062191
S [512]	0,027486	0,023108
U [512]	5,420564	4,417311

$f_{128}(x)=x^{128}+x^{7}+x^{2}+x+1$
$f_{256}(x)=x^{256}+x^{10}+x^{5}+x^{2}+1$
$f_{512}(x)=x^{512}+x^{8}+x^{5}+x^{2}+1$

Подведение итогов

ВЫВОДЫ

Выводы

- Разработан метод построения алгоритма приведения по фиксированному модулю.
- Метод может быть применим к различным полиномам, у которых ненулевое может быть произвольным, а не только младшее.
- Выигрыш в производительности пословного метода с фиксированным полиномом, по отношению к универсальному, составляет 36 раз для 128 бит, 95 раз для 256 бит и 200 раз для 512 бит.

Вопросы?

Спасибо за внимание!

Контакты

- Национальный авиационный университет
- Кафедра безопасности информационных технологий

- Мария Ковтун
- email: mg.kovtun@gmail.com

Алгоритм

Алгоритм приведения по модулю - фиксированному неприводим пятичлену.

Input: полином $c(x)$ степени не более $2(k-1)$.

Output: полином $d(x) \equiv c(x) \bmod f(x)$.

1. $m \leftarrow \lceil 2(k-1)/w \rceil - 1,\ n \leftarrow \lceil k/w \rceil,\ r \leftarrow \lceil 2(k-1)/w \rceil \cdot w - k.$
2. $s_1 \leftarrow \lceil (l+r)/w \rceil \cdot w - (l+r),\ z_1 \leftarrow m - (\lceil (l+r)/w \rceil - 1).$
3. $s_2 \leftarrow \lceil (g+r)/w \rceil \cdot w - (g+r),\ z_2 \leftarrow m - (\lceil (g+r)/w \rceil - 1).$
4. $s_3 \leftarrow \lceil (e+r)/w \rceil \cdot w - (e+r),\ z_3 \leftarrow m - (\lceil (e+r)/w \rceil - 1).$
5. $s_4 \leftarrow \lceil r/w \rceil \cdot w - r,\ z_4 \leftarrow m - (\lceil r/w \rceil - 1).$

Алгоритм

6. for $i \leftarrow m; i \geq n; i--$

6.1. $t \leftarrow c_i$.

6.2. $d_{i-z_1} \leftarrow c_{i-z_1} \oplus (t >> s_1)$, $d_{i-z_1-1} \leftarrow c_{i-z_1-1} \oplus (t << (w - s_1))$.

6.3. $d_{i-z_2} \leftarrow c_{i-z_2} \oplus (t >> s_2)$, $d_{i-z_2-1} \leftarrow c_{i-z_2-1} \oplus (t << (w - s_2))$.

6.4. $d_{i-z_3} \leftarrow c_{i-z_3} \oplus (t >> s_3)$, $d_{i-z_3-1} \leftarrow c_{i-z_3-1} \oplus (t << (w - s_3))$.

6.5. $d_{i-z_4} \leftarrow c_{i-z_4} \oplus (t >> s_4)$, $d_{i-z_4-1} \leftarrow c_{i-z_4-1} \oplus (t << (w - s_4))$.

7. $t \leftarrow c_{n-1}(x_{w-1}, x_{w-2}, \ldots, x_{(k \bmod w)}, 0_{(k \bmod w)-1}, \ldots, 0_0)$ // рассматриваются старшие биты, от $(w-1)$ до $(k \bmod w)$, остальные биты игнорируются.

8. if $(n - z_1) \geq 0$ then $d_{n-z_1} \leftarrow c_{n-z_1} \oplus (t >> s_1)$.

9. if $(n - z_1 - 1) \geq 0$ then $d_{n-z_1-1} \leftarrow c_{n-z_1-1} \oplus (t << (w - s_1))$.

Алгоритм

10. if $(n - z_2) \geq 0$ then $d_{n-z_2} \leftarrow c_{n-z_2} \oplus (t >> s_2)$.

11. if $(n - z_2 - 1) \geq 0$ then $d_{n-z_2-1} \leftarrow c_{n-z_2-1} \oplus (t << (w - s_2))$.

12. if $(n - z_3) \geq 0$ then $d_{n-z_3} \leftarrow c_{n-z_3} \oplus (t >> s_3)$.

13. if $(n - z_3 - 1) \geq 0$ then $d_{n-z_3-1} \leftarrow c_{n-z_3-1} \oplus (t << (w - s_3))$.

14. if $(n - z_4) \geq 0$ then $d_{n-z_4} \leftarrow c_{n-z_4} \oplus (t >> s_4)$.

15. if $(n - z_4 - 1) \geq 0$ then $d_{n-z_4-1} \leftarrow c_{n-z_4-1} \oplus (t << (w - s_4))$.

16. $d_{n-1} \leftarrow c_{n-1}(0_{w-1}, 0_{w-2}, \ldots 0_{(k \bmod w)}, x_{(k \bmod w)-1}, x_{(k \bmod w)-2}, \ldots, x_0)$. // рассматриваются младшие биты, от $(k \bmod w) - 1$ до 0, остальные биты - игнорируются.

17. Return $(d(x))$.