Криптография в группе кос

Сергей Николенко

Криптография — АФТУ РАН, весна 2010

Outline

Группа кос

- Основной пример базовой группы для некоммутативной криптографии: группа кос B_n.
- Коса – это переплетённые ниточки, умножение – прикладывание двух натянутых наборов ниточек друг к другу.
- Её можно задать представлением Артина:

$$B_n = \left\langle \sigma_1, \ldots, \sigma_{n-1} \mid \begin{array}{ll} \sigma_i\sigma_j = \sigma_j\sigma_i, & |i-j| \geq 2, \\ \sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}, & i = 1..n-2 \end{array} \right\rangle.$$

Группа кос

- Группа Кокстера группы B_n – это перестановки S_n; можно от косы b оставить перестановку $\phi(b) = \pi \in S_n$ (получится, в частности, $\phi(\sigma_i) = \phi(\sigma_i^{-1}) = (i, i+1)$.
- Можно зафиксировать каноническое обратное отображение ϕ^{-1}, которое определит косы перестановок, или *простые* косы: представим перестановку как $\pi = (t_1, t_1 + 1) \dots (t_k, t_k + 1)$, выберем $\tilde{\pi} = \phi^{-1}(\pi) = \sigma_{t_1} \dots \sigma_{t_k}$.

Группа кос

- Положительные косы – моноид B_n^+, порождённый σ_i без σ_i^{-1}.
- Garside: две положительные косы эквивалентны в B_n iff они эквивалентны в B_n^+.
- Т.е. можно определить отношение порядка: $a \leq b$ iff $b = ac$ для $a, b, c \in B_n^+$, и отношение порядка даст решётку положительных кос.
- На решётке можно ввести sup и inf: $a \wedge b$, $a \vee b$.

Группа кос

- Фундаментальная коса: $\Delta_n = \tilde{\pi}_n$, где $\pi_n : i \mapsto n + 1 - i$. В образующих Артина

$$\Delta_n = (\sigma_1 \dots \sigma_{n-1})(\sigma_1 \dots \sigma_{n-2}) \dots (\sigma_1 \sigma_2)(\sigma_1).$$

- Свойства:
  - $\Delta_n \sigma_i = \sigma_{n-i} \Delta_n$;
  - для $n \geq 3$ центр группы B_n – это $\langle \Delta_n^2 \rangle$;
  - любая коса перестановки $\tilde{\pi}$ – левый фактор Δ_n в B_n^+, т.е. $\Delta_n = \tilde{\pi} c$ для некоторой простой косы $c \in B_n^+$ (т.е. Δ_n – наибольший элемент среди кос перестановок).

Группа кос

- Последовательность $A_1 A_2 \dots A_l$ простых кос $A_i \in \widetilde{S_n}$ *нормальна*, если для каждой соседней пары $A_i A_{i+1}$ коса A_i – максимальная простая коса, которая может появиться слева в любой другой эквивалентной форме $A_i A_{i+1}$.
- Т.е. мы просто протягиваем справа налево как можно больше переплетений, пока не появятся два переплетения одних и тех же ниточек.

Группа кос

- С обратными σ_i^{-1} можно разобраться по коммутативности: $\sigma_i^{-1} = \Delta_n^{-1} A$ для некоторой простой A, и можно все σ^{-1} превратить в Δ_n^{-1}, а потом их протянуть по коммутативности налево.
- В итоге получается каноническая форма

$$b = \Delta_n^u A_1 A_2 \dots A_l,$$

где Δ – фундаментальная коса, A_i – простые косы перестановок, образующие нормальную последовательность. l – *длина* косы.

Группа кос

- Другой набор образующих – band generators. Элемент a_{sr} – коса, где s и r перекрещены, s сверху, и каждая из них поверх всех остальных между ними, т.е.

$$a_{sr} = \sigma_{s-1} \dots \sigma_{r+1} \sigma_r \sigma_{r+1}^{-1} \dots \sigma_{s-1}^{-1}, \quad a_{i+1,i} = \sigma_i.$$

- Представление с такими образующими получается

$$B_n = \langle a_{sr} \mid a_{ts} a_{rq} = a_{rq} a_{ts}, \quad a_{ts} a_{sr} = a_{tr} a_{ts} = a_{sr} a_{tr} \rangle.$$

Группа кос

- В нём новая фундаментальная коса

$$\delta_n = a_{n,n-1} \dots a_{32} a_{21},$$

опять простые косы – это левые факторы по всем положительным разложениям δ_n, и новая нормальная форма

$$b = \delta_n^u A_1 A_2 \dots A_l.$$

- Тут получается разных возможных простых слов A_i меньше, меньше вариантов, и алгоритмы работают быстрее.
- Люди умеют делать все нужные операции за время $O(l[^2]n[\log n])$, где l – число кос перестановок.
- Т.е. есть P0, P1, P3, P4 тоже есть, а P2 предполагается.

Группа кос

- Есть естественно возникающие коммутирующие подгруппы:

$$LB_n = \left\langle \sigma_1, \dots \sigma_{\lfloor \frac{n}{2} \rfloor - 1} \right\rangle,$$
$$UB_n = \left\langle \sigma_{\lfloor \frac{n}{2} \rfloor + 1}, \dots \sigma_n \right\rangle.$$

Представления группы кос

- Всё можно представить в матрицах; представление Бурау (приводимое) $\rho : B_n \to GL(n-1, \mathbb{Z}[t, t^{-1}])$:

$$\rho(\sigma_1) = \begin{pmatrix} -t & 1 \\ 0 & 1 \end{pmatrix} \oplus I_{n-3},$$

$$\rho(\sigma_i) = I_{i-2} \oplus \begin{pmatrix} 1 & 0 & 0 \\ t & -t & 1 \\ 0 & 1 & 0 \end{pmatrix} \oplus I_{n-i-2},$$

$$\rho(\sigma_n) = I_{n-3} \oplus \begin{pmatrix} 1 & 0 \\ t & -t \end{pmatrix}$$

Представления группы кос

- Обобщённое представление Бурау – пусть ниточки разноцветные, и мы следим за цветами (перестановка) и за тем, что с ними происходит.
- Представление на множестве $CB_n = S_n \times GL\left(n-1, \mathbb{Z}[t_1, t_1^{-1}, \ldots, t_n, t_n^{-1}]\right)$ с умножением

$$(\pi_1, M_1)(\pi_2, M_2) = (\pi_1\pi_2, (\pi_2^{-1}(M_1))M_2),$$

где перестановки переставляют переменные t_i.

- $\rho_c(\sigma_i) = ((i, i+1), \rho(\sigma_i)(t_i))$.

Представления группы кос

- Представление Лоуренса–Краммера (Lawrence–Krammer):

$$\mathcal{K} : B_n \to GL\left(\frac{n(n-1)}{2}, \mathbb{Z}[q, t, q^{-1}, t^{-1}]\right),$$

- Без подробностей; очень эффективное, используется в атаках на криптосистемы.

Outline

Anshel-Anshel-Goldfeld

- Публикуем группу G и элементы $a_1, \ldots, a_k, b_1, \ldots, b_m \in G$.
- Алиса выбирает $x \in G$ как слово в $a_1, \ldots, a_k$ и посылает Бобу $b_1^x, \ldots, b_m^x$.
- Боб выбирает $y \in G$ как слово в $b_1, \ldots, b_m$ и посылает Алисе $a_1^y, \ldots, a_k^y$.
- Алиса вычисляет $x(a_1^y, \ldots, a_k^y) = x^y = y^{-1}xy$, а затем $[x, y] = xy^{-1}xy$. Боб вычисляет $y(b_1^x, \ldots, b_m^x) = y^x = x^{-1}yx$, а затем $[x, y] = xy^{-1}xy$. Получается общий ключ $[x, y]$.

Anshel-Anshel-Goldfeld

- AAG, как мы уже выясняли, работает на subgroup-related simultaneous conjugacy search problem: найти $x \in \langle a_1, \ldots, a_k \rangle$, для которого $a_1 = b_1^x, \ldots, a_k = b_k^x$.
- Т.е. врагу даны пары (x, axa^{-1}) для разных x из некоторой подгруппы, и ему надо найти a.
- Об этом позже.

Протокол типа Diffie-Hellman

- В B_n есть коммутирующие подгруппы LB_n и UB_n. Получается протокол:
  - Публикуется коса $a \in B_n$.
  - Алиса выбирает $x_1, x_2 \in LB_n$, посылает Бобу $x_1 a x_2$.
  - Боб выбирает $y_1, y_2 \in RB_n$, посылает Алисе $y_1 a y_2$.
  - Они вычисляют общий ключ $x_1 y_1 a x_2 y_2$.

Braid Diffie-Hellman Problem

- Он сводится к BDHP (Braid Diffie-Hellman Problem): по $a, b_1, b_2 \in B_n$, где $b_1 = x_1 a x_2$ и $b_2 = y_1 a y_2$ для некоторых $x_1, x_2 \in LB_n$, $y_1, y_2 \in RB_n$, найти $y_1 b_1 y_2 [= x_1 b_2 x_2]$.
- Можно на этом и криптосистему сделать, аналогичную криптосистеме Эль-Гамаля; пусть $H : \{0,1\}^M \to B_n$ – хеш-функция без коллизий.

Криптосистема на группе кос

- Генерация ключей:
  - выбрать достаточно сложную косу $a \in B_n$;
  - выбрать секретный ключ $(x_1, x_2) \in LB_n \times LB_n$;
  - выдать публичный ключ $(a, b) = (a, x_1 a x_2)$.
- Кодирование (сообщения $m \in \{0,1\}^M$):
  - выбрать $y_1, y_2 \in UB_n$;
  - выдать код $(c_1, c_2) = (y_1 a y_2, H(y_1 b y_2) \oplus m)$.
- Декодирование:
  - $m = H(x_1 c_1 x_2) \oplus c_2$.

Проблемы

- Но в последнее время с криптографией на группе кос начались проблемы.
- Мы их сейчас кратко опишем.

Атаки, основанные на длине

- Мы определяли *длину* на косах; для некоторой нормальной формы длина – это число простых кос в нормальной последовательности.
- Можно теперь определить *расстояние*:

$$d(a, b) = \mathrm{length}(ab^{-1}).$$

- Это расстояние будет удовлетворять неравенству треугольника.

Атаки, основанные на длине

- В протоколе AAG были две подгруппы $A = \langle a_1, \ldots, a_k \rangle$ и $B = \langle b_1, \ldots, b_m \rangle$.
- Идея: чем длиннее элементы, тем больше шансов, что они будут образовывать «переплетение» и редко будут сильно сокращаться друг с другом (иметь много общих факторов).
- Но генераторы должны быть достаточно длинными, иначе их легко будет распознать в a просто по виду.
- Враг знает пары (x, axa^{-1}); т.е., если a_i не есть левый фактор a, то мало шансов, что $\mathrm{length}\left(a_i(axa^{-1})a_i^{-1}\right)$ не увеличится.

Атаки, основанные на длине

- Отсюда простой алгоритм атаки: перебираем a_i, считаем $\mathrm{length}\left(a_i(axa^{-1})a_i^{-1}\right)$, выбираем самый короткий вариант, повторяем.
- Она хорошо работает против базового AAG.
- Модификация: возьмём короткие генераторы, а чтобы в a не распознали, будем вместо a выдавать хеш-функцию, т.е. ключ – это

$$E(b) = (\pi_b, M_b(\tau_1, \ldots, \tau_n)/\mathbb{F}_p),$$

где (π_b, M_b) – обобщённое представление Бурау, $\tau_i \in \mathbb{F}_p^*$.

- Но и этот взломали линейной атакой (будет позже).

Атаки на сопряжение

- Другой алгоритм основан на задаче сопряжения. Как её решить?
- По теореме о нормальной форме, для любого x есть максимальное k и минимальное r, для которых

$$\delta^k \leq x \leq \delta^{k+r}.$$

- Здесь $r = \mathrm{length}(x)$, обозначим ещё $\inf(x) = k$, $\sup(x) = r$.

Атаки на сопряжение

- Обозначим $\inf_s(x) = \max\{\inf(y) \mid y \in x^{B_n}\}$, $\sup_s(x) = \min\{\sup(y) \mid y \in x^{B_n}\}$.
- Множество $S_x = \{y \in x^{B_n} \mid \inf(y) = \inf_s(x),\ \sup(y) = \sup_s(x)\}$ – super summit set.
- Введём две легко вычислимые операции cycling и decycling: пусть $x = \delta^k A_1 \dots A_r$. Если $r = 0$, то $c(x) = d(x) = x$, иначе

$$c(x) = x^{A_1^{\delta^{-k}}}, \quad d(x) = x^{A_r^{-1}}.$$

Атаки на сопряжение

- Основные свойства этих операций и этого множества:
  1. S_x конечно и непусто;
  2. любой представитель S_x эффективно вычисляется конечным числом c и d из x;
  3. если $y \in S_x$, то $c(y) \in S_x$ и $d(y) \in S_x$;
  4. для всех $y \in B_n$, $(c(y))^\delta = c(y^\delta)$ и $(d(y))^\delta = d(y^\delta)$.

Атаки на сопряжение

- Вычислительные свойства:
  1. для любых $y, z \in B_n$ существует $u \in B_n^+$, для которого $y^u = z$;
  2. если для $y \in S_x$ и $u \in B_n^+$ $y^u \in S_x$, то $y^{\delta \wedge u} \in S_x$;
  3. для любых $y, z \in S_x$ существуют $y_0, \ldots, y_t \in S_x$ и $c_1, \ldots, c_t$ – простые, для которых $y_0 = y$, $y_t = z$, $y_{i-1}^{c_i} = y_i$.

Атаки на сопряжение

- Значит, можно посчитать S_x:
  - сначала посчитать $x' \in S_x$ по свойству 2;
  - затем сопрягать элементы S_x простыми косами и добавлять те, у которых inf и sup подходят;
  - когда новые перестанут добавляться, всё.

Атаки на сопряжение

- И можно проверить сопряжённость:
  - по x и y вычислить $x' \in S_x$, $y' \in S_y$;
  - если inf или sup не совпадают, значит, не сопряжённые;
  - начать вычислять S_x и S_y; x и y сопряжены iff $x' \in S_y$ (или $y' \in S_x$).
- Если по дороге запоминать, что делали, то и сам сопрягающий элемент получится.

Атаки на сопряжение

- Можно даже ещё лучше: не super summit set, а ultra summit set, т.е. только те элементы S_x, которые лежат внутри цикла по c:

$$U_x = \{y \in S_x \mid c^k(y) = y \text{ для некоторого } k > 0\}.$$

- Можно доказать, что и такими элементами будет достаточно сопрягать.

Атаки на сопряжение

- В итоге получился алгоритм, который в худшем случае работает за время и память, равные размеру S_x или даже U_x.
- S_x в среднем очень быстро растёт с ростом n, размер U_x обычно остаётся порядка $2r$, но всё равно, конечно, бывает так, что U_x очень большой.
- Но получается, что нужно как-то выбирать параметры, для которых U_x экспоненциальный.
- А таких оценок вообще не известно. :) В общем, сопряжение в B_n оказалось не такой сложной задачей, как казалось на первый взгляд.

Атаки линейной алгеброй

- Суть: выразить задачу в виде линейной системы, потом её решить.
- Косы представляются матрицами не слишком большого размера, так что это в принципе реально.
- Рассмотрим, например, Диффи-Хеллмана: даны x, $y_a = axa^{-1}$ и $y_b = bxb^{-1}$, найти $a \in LB_n$, $b \in RB_n$.

Атаки линейной алгеброй

- Представим $Y_a = \mathcal{K}(y_a)$, $Y_b = \mathcal{K}(y_b)$ в представлении Lawrence–Krammer.
- По модулю некоторого p и некоторых неприводимых многочленов по t и q решим систему:

$$\begin{aligned} Y_a A &= A Y_b, \\ \mathcal{K}(\sigma_i) A &= A \mathcal{K}(\sigma_i), \quad \sigma_i \in RB_n \end{aligned}$$

Атаки линейной алгеброй

- Тогда, даже если получится не совсем $\mathcal{K}(a)$, всё равно он сработает по коммутативности:

$$AY_bA^{-1} = \mathcal{K}(b)Y_a\mathcal{K}(b)^{-1} = \mathcal{K}(abxa^{-1}b^{-1}).$$

- Получается эффективная и разумная атака.

Спасибо за внимание!

- Lecture notes и слайды будут появляться на моей homepage:

  `http://logic.pdmi.ras.ru/~sergey/`

- Присылайте любые замечания, решения упражнений, новые численные примеры и прочее по адресам: `sergey@logic.pdmi.ras.ru`, `snikolenko@gmail.com`
- Заходите в ЖЖ  **smartnik**.