О реализации декодера одного класса алгебро-геометрических кодов на проективных кривых с использованием алгоритма Сакаты

Пеленицын А. М.

Научные руководители: доцент, к. ф.-м. н. Деундяк В. М., асс. Маевский А. Э.
Факультет Математики, механики и компьютерных наук ЮФУ

В данной работе рассматривается обобщение алгоритма Берлекемпа—Месси, предложенное Сакатой [1]. Оно может быть использовано в конструкции декодера одного класса алгебро-геометрических кодов, которые можно назвать алгебро-геометрическими кодами типа Рида—Соломона. Как и в классическом варианте кодов Рида—Соломона [2], использование этого алгоритма позволяет уменьшить теоретическую сложность процесса декодирования, более того, именно эта часть является определяющей для сложности всего процесса [3].

Первоначальной задачей, поставленной в работе, была программная реализация алгоритма, который может быть встроен в имеющийся декодер, использовавший до того менее эффективные методы декодирования. Однако анализ [1] показал, что актуальной проблемой является формализация изложенных там идей и получение строгого, замкнутого в себе описания алгоритма, которое, в свою очередь, может быть использовано для построения программной реализации. Одной из сопутствующих задач стало выделение общих для обеих версий алгоритма Берлекемпа—Месси, обобщённой и классической, особенностей и идей. Приведём здесь полученное описание алгоритма.

Предварительные сведения. Пусть задано поле Галуа K, введём обозначения для следующих множеств: $\mathbb{N}_0=\{0,1,2,\ldots\}$, $\Sigma_0=\mathbb{N}_0^2$, компоненты точек из Σ_0 будем обозначать, как обычно, нижними индексами. Введём два отношения порядка на Σ_0:

- $\forall n,m\in\Sigma_0$: $n<m$ тогда и только тогда, когда $n_1<m_1\wedge n_2<m_2$;
- $\forall n,m\in\Sigma_0$: $n<_T m$ тогда и только тогда, когда
$$n_1+n_2<m_1+m_2 \ \vee\ (n_1+n_2=m_1+m_2)\wedge(n_2<m_2)$$

Первое является частичным порядком, а второе — полным и позволяет задать операцию инкремента:

$$\forall n\in\Sigma_0:\ n+1=\begin{cases}(n_1-1,n_2+1), & \text{если } n_1>0\\ (n_2+1,0), & \text{если } n_1=0\end{cases}$$

Естественным образом определяются «нестрогие неравенства» $\leqslant,\leqslant_T$. Введём теперь множество: $\forall n,m\in\Sigma_0$: $\Sigma_n^m=\{p\in\Sigma_0 | n\leqslant p\wedge p<_T m\}$.

Далее будем рассматривать конечную двумерную последовательность u длины $p\in\Sigma_0$ над полем K, которая понимается как отображение $u:\Sigma_0^p\to K$. Полиномом от двух переменных $x=(x_1,x_2)$ называется выражение $f(x)=\sum_{m\in\Gamma_f} f_m\cdot x^m$, где $\Gamma_f=\{m\in\Sigma_0 | f_m\neq 0\}$ ($|\Gamma_f|<\infty$), $x^m=(x_1^{m_1},x_2^{m_2})$. Полный порядок на Σ_0 позволяет ввести понятие (старшей) степени полинома f, которую будем обозначать $LP(f)$. Определим следующую операцию: $f[u]_n=\sum_{m\in\Gamma_f} f_m\cdot u_{m+n-s}$, где $n\in\Sigma_s^p$, s — степень полинома f. Будем писать $f\in VALPOL(u)$ тогда и только тогда, когда $\forall n\in\Sigma_s^p$: $f[u]_n=0$ либо $p\leqslant_T s$. Последнее понятие, необходимое для корректного определения задачи, решаемой алгоритмом, понятие Δ-множества. Будем говорить, что упорядоченный набор $\{s^{(i)}\}_{i=0}^l$ точек из Σ_0 «определяет Δ-множество», если:

$$s_1^{(1)}>s_1^{(2)}>\ldots>s_1^{(l)}=0,\ \ 0=s_2^{(1)}>s_2^{(2)}>\ldots>s_2^{(l)}$$

Пусть ещё $\forall n\in\Sigma_0$: $\Sigma_n=\{p\in\Sigma_0 | n\leqslant p\}$. Наконец, $\Delta=\Sigma_0\setminus(\Sigma_{s^{(1)}}\cup\Sigma_{s^{(2)}}\cup\ldots\cup\Sigma_{s^{(l)}})$ — *Δ-множество, определяемое* $\{s^{(i)}\}_{i=0}^l$. Если степени полиномов из некоторого конечного набора F определяют Δ-множество, то последнее мы будем обозначать $\Delta(F)$.

Минимальное множество полиномов $F=\{f^{(1)},\dots f^{(l)}\}$ для последовательности *u* длины *p*:

(1) $F\subset VALPOL(u)$;

(2) степени элементов *F* определяют Δ-множество;

(3) $\forall g:\ g\in VALPOL(u)\to LP(g)\notin\Delta(F)$.

Алгоритм строит минимальное множество полиномов для данной последовательности *u* длины *p*. Перед тем как описать шаги алгоритма, нужно дать определение тех специфических операций и объектов, с которыми он работает. Алгоритм имеет итеративный характер, на каждом шаге $n\in\Sigma_0$ имеются: $F=\{f^{(i)}(x)\}_{i=1}^{l}$ — минимальное множество для «первых *n*» элементов последовательности, $S=\{s^{(i)}=LP(f^{(i)})\}_{i=1}^{l}$, вместо $\Delta(F)$ будем писать просто Δ; $G=\{g^{(i)}(x)\}_{i=1}^{l-1}$ — «вспомогательное множество» полиномов и связанные с ним множества $T=\{t^{(i)}=LP(g^{(i)})\}_{i=1}^{l-1}$, $PG=\{p^{(i)}\}_{i=1}^{l-1}$, $DG=\{d^{(i)}\}_{i=1}^{l-1}$. Определим операции, которые используют перечисленные параметры алгоритма и, таким образом, зависят от текущего шага:

- $\forall i,j\in\mathbb{N}_0:\ BP\langle i,j\rangle=\begin{cases}f^{(i)}, & j=0\\ x^{r-s^{(i)}}\cdot f^{(i)}-(d/d^{(j)})\,x^{r-n+p^{(j)}-t^{(j)}}\cdot g^{(j)}, & \text{иначе}\end{cases}$,

где $d=f^{(i)}[u]_n$, $r_k=max\{s_k^{(i)},t_k^{(j)}+n_k-p_k^{(j)}\}$, $k=1,2$;

- $\forall i\in\mathbb{N}_0:\ SP_k\langle i\rangle=x_k^{n_k-s_k^{(i)}+1}\cdot f^{(i)}$, $k=1,2$;
- $\forall n,m\in\Sigma_0:\ inSD(n,s)=\begin{cases}j, & n_k<s_k+s_k^{(j+k-1)},\ k=1,2\\ 0, & \text{иначе}\end{cases}$.

Вход: двумерная последовательность *u* длины $p\in\Sigma_0$ над полем *K*.

Выход: минимальное множество *F* для *u*.

Шаг 0. Положить $n=(0,0)$, $F=\{1\}$, $G=DG=PG=\Delta=\varnothing$.

Шаг 1. $\forall i\in[1,l]_{\mathbb{N}}$: *if* $\neg(s^{(i)}\leqslant n)\vee(f^{(i)}[u]_n=0)$ *then* $f^{(i)}\in F_V$ *else* $f^{(i)}\in F_N$.

Шаг 2. Положить $aux\in\mathbb{N}_0^l$, $aux=\bar{0}$. $\forall f^{(i)}\in F_N:\ aux[i]=inSD(n,s^{(i)})$.

if $\forall f^{(i)}\in F_N:\ aux[i]\neq 0$ *then* [$\forall f^{(i)}\in F_N:\ f^{(i)}=BP\langle i,aux[i]\rangle$; *goto* Шаг 6].

Шаг 3. Определим Δ_{new}, включая в него последовательно точки каждого из следующих четырёх типов при выполнении условий:

(1) $(s_1^{(i)},s_2^{(i)})$, если $f^{(i)}\in F_V\vee aux[i]\neq 0$;

(2) $(n_1-s_1^{(i)}+1,n_2-s_2^{(i+1)}+1)$, если $\exists i\in[1,l-1]_{\mathbb{N}}:\ f^{(i)},f^{(i+1)}\in F_N$;

(3) $(n_1-s_1^{(i)}+1,s_2^{(j)})$, если
$\exists f^{(i)},f^{(j)}\in F_N:\ (n\geqslant s^{(i)}+s^{(j)})\wedge\forall k\in[i,l]:\ n_2<s_2^{(k)}+s_2^{(j)}$;

(4) $(s_1^{(i)},n_2-s_2^{(j)}+1)$, если
$\exists f^{(i)},f^{(j)}\in F_N:\ (n\geqslant s^{(i)}+s^{(j)})\wedge\forall k\in[1,j]:\ n_1<s_1^{(k)}+s_1^{(i)}$.

Шаг 4. Построить новое *F*, добавляя в него новый элемент для каждой точки Δ_{new} по определённому правилу для точек каждого типа:

(1) $(s_1^{(i)},s_2^{(i)})$ — $BP\langle i,aux[i]\rangle$;

(2) $(n_1-s_1^{(i)}+1,n_2-s_2^{(i+1)}+1)=t$ — $BP\langle k,i\rangle$, где $k:\ f^{(k)}\in F_N\wedge s^{(k)}<t$;

(3) $(n_1-s_1^{(i)}+1,s_2^{(j)})$ — $\begin{cases}BP\langle j,i\rangle, & i\neq l\\ SP_1\langle j\rangle, & \text{иначе}\end{cases}$;

(4) $(s_1^{(i)},n_2-s_2^{(j)}+1)$ — $\begin{cases}BP\langle i,j-1\rangle, & j\neq 1\\ SP_2\langle i\rangle, & \text{иначе}\end{cases}$.

Шаг 5. Построить G_{new}, исходя из условий: $G_{new}\subset G\cup F_N$ и $|G_{new}|=|F|-1$. Пополнить $PG=\{p^{(i)}\}_{i=1}^{l-1}$ и $DG=\{d^{(i)}\}_{i=1}^{l-1}$:

$$\forall g^{(k)}\in G_{new}\cap F_N:\ PG=PG\cup\{p^{(k)}=n\},\ DG=DG\cup\{d^{(k)}=g^{(k)}[u]_n\}$$

Заменить G на G_{new}. Обновить T, исходя из его определения. Перенумеровать элементы G и соответствующие им элементы PG, DG и T так, чтобы:

$$\forall i \in [1, l-1]_{\mathbb{N}}: \; s_k^{(i+k-1)} = p_k^{(i)} - t_k^{(i)} + 1, \; k = 1,2$$

Шаг 6. $n = n+1$ *; if* $n = p$ *then exit else goto* Шаг 1 .

Литература.

1. S. Sakata, «Finding a minimal set of linear recurring relations capable of generating a given finite two–dimensional array,» J. Symb. Comp., vol. 5, pp. 321–337, 1988.
2. Р. Блейхут, «Теория и практика кодов, контролирующих ошибки»: Пер. с англ. — М.: Мир, 1986.
3. J. Justesen, K. J. Larsen, H. E. Jensen, and T. Hoholdt, «Fast decoding of codes from algebraic plane curves,» IEEE Trans. Inform. Theory, vol. 38. DD. 111-119. Jan. 1992.