Math-Net.Ru

Общероссийский математический портал

В. В. Галатенко, Одна задача о расстановке меток в дереве и ее приложение к оптимальному приближению непрерывных функций, *Матем. заметки*, 2012, том 91, выпуск 1, 24–39

DOI: <https://doi.org/10.4213/mzm8827>

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением <http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 54.162.27.143

26 апреля 2023 г., 05:19:54

Одна задача о расстановке меток в дереве и ее приложение к оптимальному приближению непрерывных функций

В. В. Галатенко

В работе формулируется задача о расстановке меток в дереве, приводится алгоритм для решения этой задачи, обсуждаются некоторые сложные характеристики этого алгоритма. Кроме того, обсуждаются приложения полученных результатов, связанные с построением линейных комбинаций характеристических функций двоичных промежутков, приближающих непрерывную функцию с заданной точностью и содержащих минимально возможное число слагаемых. Также затрагиваются вопросы дискретизации сигналов с использованием деревьев.

Библиография: 10 названий.

1. Постановка задачи. Пусть K – произвольное натуральное число, T – корневое полное бинарное дерево (см., например, [1; п. 4.1]) с вершинами $v_{k,j}$ ($k = 0, 1, 2, \dots, K, j = 0, 1, 2, \dots, 2^k - 1$) и ребрами, идущими от вершин $v_{k,j}$ к вершинам $v_{k+1,2j}$ и $v_{k+1,2j+1}$. Вершину $v_{0,0}$ будем считать *корневой*, вершины $v_{1,0}$ и $v_{1,1}$ будем называть *вершинами первого яруса*, вершины $v_{2,0}$, $v_{2,1}$, $v_{2,2}$ и $v_{2,3}$ – *вершинами второго яруса*, и так далее. Вершины яруса K , т.е. вершины, из которых ребер не исходит, будем называть *концевыми вершинами*.

Будем приписывать вершинам дерева $v_{k,j}$ в качестве меток действительные числа: $c_{k,j} = c(v_{k,j})$. Если число $c_{k,j}$ нулевое, то будем говорить, что вершина $v_{k,j}$ не используется, в противном случае будем говорить, что эта вершина используется.

Для концевых вершин $v_{K,j}$ обозначим через $P_j = P(v_{K,j})$ множество всех вершин, лежащих на пути от корня до этой вершины, включая корень и саму вершину $v_{K,j}$. Далее, через $S_j = S(v_{K,j})$ обозначим сумму $\sum_{v \in P_j} c(v)$.

Будем считать, что с каждой концевой вершиной $v_{K,j}$ ассоциировано непустое множество действительных чисел X_j , которое будем называть *разрешенным* для данной вершины множеством.

Расстановку меток в дереве будем называть *допустимой*, если для всех индексов $j \in \{0, 1, 2, \dots, 2^K - 1\}$ справедливы включения $S_j \in X_j$.

Работа выполнена при поддержке Российского фонда фундаментальных исследований (гранты №№ 11-01-00476-а, 09-01-12173-офи_м), и программы “Ведущие научные школы РФ” (грант № НШ-979.2012.1).

Во введенных обозначениях рассматриваемая задача может быть сформулирована следующим образом. Требуется расставить в вершинах дерева метки таким образом, что полученная расстановка меток, во-первых, допустима, и, во-вторых, такова, что любая другая допустимая расстановка меток использует не меньшее количество вершин.

Легко видеть, что сформулированная задача всегда имеет решение (т.е. требуемая расстановка меток всегда существует), но решение это, вообще говоря, не единственно.

Отметим, что аналогичную задачу можно рассматривать необязательно для полных и необязательно для бинарных деревьев: можно считать, что ветвление в каждой вершине произвольное (возможно, разное для разных вершин), и что длина пути от корня к конечным вершинам может быть различной для различных конечных вершин. Рассмотрение именно полных бинарных деревьев делается исключительно для того, чтобы упростить описания и не концентрировать внимание на несущественных технических деталях.

2. Описание алгоритма. Опишем алгоритм, позволяющий построить требуемую расстановку меток. Для этого введем еще некоторые обозначения, аналогичные уже введенным. Для отличных от корня вершин $v_{k,j}$ обозначим через $P'_{k,j} = P'(v_{k,j})$ множество всех вершин, лежащих на пути от корня до этой вершины, включая корень, но не включая саму вершину $v_{k,j}$. Далее, через $S'_{k,j} = S'(v_{k,j})$ обозначим сумму $\sum_{v \in P'_{k,j}} c(v)$.

Алгоритм построения искомой разметки состоит из двух этапов.

На первом этапе осуществляется проход по дереву от конечных вершин к корню, и с каждой вершиной $v_{k,j}$ ассоциируется непустое множество $X_{k,j}$. Для конечных вершин $X_{K,j}$ полагается равным X_j . Далее, если множества $X_{k,j}$ построены для вершин $v_{m,2j}$ и $v_{m,2j+1}$ яруса m , то для (неконцевой) вершины $v_{m-1,j}$ яруса $m-1$ эти множества определяются следующим образом: $X_{m-1,j} = X_{m,2j} \cap X_{m,2j+1}$ в случае, если это пересечение непусто, и $X_{m-1,j} = X_{m,2j} \cup X_{m,2j+1}$ в противном случае. В случае неполных деревьев множества $X_{k,j}$ могут быть определены тем же способом, что и в случае полных деревьев.

Отметим, что в терминах многозначной функции Arg max определение множества $X_{m-1,j}$ может быть записано так:

$$X_{m-1,j} = \text{Arg max}(\chi_{m,2j}(x) + \chi_{m,2j+1}(x)),$$

где $\chi_{m,2j}(x)$ и $\chi_{m,2j+1}(x)$ – характеристические функции множеств $X_{m,2j}$ и $X_{m,2j+1}$ соответственно. В этой форме определение переносится на случай небинарных деревьев.

На втором этапе алгоритма осуществляется окончательное построение требуемых меток. Оно идет от корня к конечным вершинам. Сначала метка приписывается корню дерева: если множество $X_{0,0}$ содержит нуль, то корень не используется (т.е. $c_{0,0}$ полагается равным нулю), в противном случае в качестве $c_{0,0}$ берется произвольный элемент множества $X_{0,0}$. Далее, если метки $c_{k,j}$ приписаны уже всем вершинам ярусов $0, 1, \dots, m-1$ (в этом случае, в частности, при всех $j \in \{0, 1, 2, \dots, 2^m - 1\}$ определены величины $S'_{m,j}$), то метки вершинам яруса m приписываются следующим образом. Если $S'_{m,j} \in X_{m,j}$, то вершина $v_{m,j}$ не используется (соответственно,

$c_{m,j}$ полагается равным нулю). В противном случае в множестве $X_{m,j}$ выбирается произвольный элемент $\alpha_{m,j}$, и $c_{m,j}$ полагается равным разности $(\alpha_{m,j} - S'_{m,j})$.

3. Доказательство корректности. Покажем, что описанный алгоритм действительно решает поставленную задачу, т.е. строит допустимую расстановку меток, использующую при этом минимально возможное число вершин.

Допустимость полученной расстановки меток очевидна: при любом фиксированном значении $j \in \{0, 1, 2, \dots, 2^K - 1\}$, если $S'_{K,j}$ принадлежало $X_{K,j} = X_j$, то в качестве $c_{K,j}$ выбирался нуль, так что

$$S_j = S'_{K,j} + c_{K,j} = S'_{K,j} \in X_j;$$

в противном случае в качестве $c_{K,j}$ выбиралось такое число, что

$$c_{K,j} + S'_{K,j} \in X_{K,j} = X_j, \quad S_j = c_{K,j} + S'_{K,j}.$$

Доказательство минимальности числа используемых вершин начнем с нескольких технических лемм.

Обозначим через $S_j(\alpha)$ сумму $S_j + \alpha$ ($j = 0, 1, 2, \dots, 2^K - 1$). Через $N(\alpha)$ обозначим минимальное число вершин в дереве T , которое нужно использовать (т.е. которым нужно приписать ненулевые метки), чтобы для всех $j \in \{0, 1, 2, \dots, 2^K - 1\}$ добиться включений $S_j(\alpha) \in X_j$. Через $N_{\min}$ обозначим $\min_{\alpha \in \mathbb{R}} N(\alpha)$. Через X_{opt} обозначим $\text{Arg min}_{\alpha \in \mathbb{R}} N(\alpha)$.

ЛЕММА 1. Если $\alpha_0 \notin X_{\text{opt}}$, то $N(\alpha_0) = N_{\min} + 1$.

ДОКАЗАТЕЛЬСТВО. С одной стороны, справедливо неравенство $N(\alpha_0) > N_{\min}$, так как иначе $\alpha_0 \in \text{Arg min}_{\alpha \in \mathbb{R}} N(\alpha) = X_{\text{opt}}$. С другой стороны, имеет место оценка $N(\alpha_0) \leq N_{\min} + 1$. Действительно, пусть α_{opt} — произвольное число из множества X_{opt} . Из определений следует, что существует расстановка меток в дереве T , удовлетворяющая при всех j ограничениям $S_j(\alpha_{\text{opt}}) \in X_j$ и использующая ровно $N_{\min}$ вершин. Изменим в этой разметке метку корня, прибавив к исходному значению метки разность $(\alpha_{\text{opt}} - \alpha_0)$. Несложно видеть, что количество используемых вершин при этом изменении не увеличивается или увеличивается ровно на единицу, и что новая разметка при всех j удовлетворяет ограничениям $S_j(\alpha_0) \in X_j$. Лемма доказана.

Обозначим через T_0 и T_1 максимальные поддеревья дерева T с корнями $v_{1,0}$ и $v_{1,1}$ соответственно. Обозначим через $N_0(\alpha)$, $N_{\min}^0$ величины, аналогичные величинам $N(\alpha)$, $N_{\min}$, возникающие при ограничении рассматриваемой задачи на поддерево T_0 (при ограничении задачи на поддерево T_0 рассматриваются лишь расстановки меток, использующие только вершины поддерева T_0 и удовлетворяющие ограничениям на вхождение сумм $S_j(\alpha)$ в разрешенные множества X_j при $j \in \{0, 1, 2, \dots, 2^{K-1} - 1\}$). Также обозначим через X_{opt}^0 множество, аналогичное множеству X_{opt} , возникающее при ограничении рассматриваемой задачи на поддерево T_0 . Кроме того, ограничивая задачу на поддерево T_1 , определим аналогичным образом величины $N_1(\alpha)$, $N_{\min}^1$ и множество X_{opt}^1 .

ЛЕММА 2. Справедливы следующие утверждения:

- а) если $X_{\text{opt}}^0 \cap X_{\text{opt}}^1 = \emptyset$, то $N_{\min} = N_{\min}^0 + N_{\min}^1 + 1$ и $X_{\text{opt}} = X_{\text{opt}}^0 \cup X_{\text{opt}}^1$;
- б) если $X_{\text{opt}}^0 \cap X_{\text{opt}}^1 \neq \emptyset$, то $N_{\min} = N_{\min}^0 + N_{\min}^1$ и $X_{\text{opt}} = X_{\text{opt}}^0 \cap X_{\text{opt}}^1$.

ДОКАЗАТЕЛЬСТВО. Начнем со случая $X_{\text{opt}}^0 \cap X_{\text{opt}}^1 = \emptyset$. Покажем сначала, что для любого значения α справедливо неравенство $N(\alpha) > N_{\min}^0 + N_{\min}^1$. Действительно, пусть существует такое значение $\tilde{\alpha}$, что $N(\tilde{\alpha}) \leq N_{\min}^0 + N_{\min}^1$. Рассмотрим расстановку меток в дереве T , использующую ровно $N(\tilde{\alpha})$ вершин и удовлетворяющую при всех j ограничениям $S_j(\tilde{\alpha}) \in X_j$. Ограничим эту расстановку меток на поддеревья T_0 и T_1 и обозначим число используемых данной разметкой вершин в каждом из этих поддеревьев через n_0 и n_1 соответственно. Имеем:

$$N_{\min}^0 \leq N_0(\tilde{\alpha} + c(v_{0,0})) \leq n_0, \quad N_{\min}^1 \leq N_1(\tilde{\alpha} + c(v_{0,0})) \leq n_1.$$

При этом по крайней мере одно из этих двух неравенств строгое, так как число $\tilde{\alpha} + c(v_{0,0})$ не принадлежит хотя бы одному из множеств X_{opt}^0 и X_{opt}^1 в силу того, что пересечение этих множеств пусто. Кроме того, $n_0 + n_1 \leq N(\tilde{\alpha})$ (равенство имеет место, если и только если $c(v_{0,0}) = 0$, т.е. если расстановка меток не использует корень дерева T). Таким образом, получаем неравенство

$$N(\tilde{\alpha}) \leq N_{\min}^0 + N_{\min}^1 < n_0 + n_1 \leq N(\tilde{\alpha}).$$

Полученное противоречие доказывает, что $N_{\min} > N_{\min}^0 + N_{\min}^1$.

Установим теперь неравенство

$$N_{\min} \leq N_{\min}^0 + N_{\min}^1 + 1.$$

Пусть $\tilde{\alpha}$ – произвольный элемент множества X_{opt}^0 . Покажем, что

$$N(\tilde{\alpha}) \leq N_{\min}^0 + N_{\min}^1 + 1.$$

Возьмем расстановку меток в поддереве T_0 , использующую ровно $N_{\min}^0 = N_0(\tilde{\alpha})$ вершин и удовлетворяющую при всех j ограничениям $S_j^0(\tilde{\alpha}) \in X_j$ (здесь $S_j^0(\alpha)$ – суммы α и меток для поддерева T_0 , т.е. суммы, включающие слагаемые не от корня $v_{0,0}$ дерева T , а от корня $v_{1,0}$ поддерева T_0). Также возьмем расстановку меток в поддереве T_1 , использующую ровно $N_{\min}^1 + 1 = N_1(\tilde{\alpha})$ вершин и удовлетворяющую при всех j ограничениям $S_j^1(\tilde{\alpha}) \in X_j$ (здесь $S_j^1(\alpha)$ – суммы α и меток для поддерева T_1). Построим расстановку меток в дереве T , объединив разметки поддеревьев T_0 , T_1 и приписав корню $v_{0,0}$ нулевую метку. Полученная расстановка меток использует в точности $N_{\min}^0 + N_{\min}^1 + 1$ вершину и удовлетворяет при всех j ограничениям $S_j(\tilde{\alpha}) \in X_j$.

В результате установлено, что

$$N_{\min} = N_{\min}^0 + N_{\min}^1 + 1.$$

Кроме того, показано, что для любого числа $\alpha \in X_{\text{opt}}^0$ выполняется равенство

$$N(\alpha) = N_{\min}^0 + N_{\min}^1 + 1,$$

т.е. $X_{\text{opt}}^0 \subset X_{\text{opt}}$. Аналогично показывается, что $X_{\text{opt}}^1 \subset X_{\text{opt}}$.

Для завершения доказательства утверждения а) осталось показать, что для любого значения $\tilde{\alpha} \notin X_{\text{opt}}^0 \cup X_{\text{opt}}^1$ выполняется неравенство

$$N(\tilde{\alpha}) > N_{\min} = N_{\min}^0 + N_{\min}^1 + 1.$$

Зафиксируем произвольное значение $\tilde{\alpha} \notin X_{\text{opt}}^0 \cup X_{\text{opt}}^1$. Рассмотрим расстановку меток в дереве T , использующую ровно $N(\tilde{\alpha})$ вершин и удовлетворяющую при всех j ограничениям $S_j \in X_j$. Ограничим эту расстановку меток на поддеревья T_0 , T_1 и обозначим число используемых данной разметкой вершин в каждом из этих поддеревьев через n_0 и n_1 соответственно. Если в исходной разметке корень дерева T не используется, то

$$n_0 \geq N_0(\tilde{\alpha}) > N_{\min}^0, \quad n_1 \geq N_1(\tilde{\alpha}) > N_{\min}^1,$$

т.е. $n_0 \geq N_{\min}^0 + 1$ и $n_1 \geq N_{\min}^1 + 1$, откуда

$$N(\tilde{\alpha}) = n_0 + n_1 \geq N_{\min}^0 + N_{\min}^1 + 2.$$

Если же в исходной разметке корень используется, то

$$n_0 \geq N_0(\tilde{\alpha} + c(v_{0,0})) \geq N_{\min}^0, \quad n_1 \geq N_1(\tilde{\alpha} + c(v_{0,0})) \geq N_{\min}^1,$$

причем в силу того, что $X_{\text{opt}}^0 \cap X_{\text{opt}}^1 = \emptyset$, по крайней мере одно из этих неравенств строгое. Тогда

$$N(\tilde{\alpha}) = n_0 + n_1 + 1 > N_{\min}^0 + N_{\min}^1 + 1.$$

Утверждение а) доказано.

Доказательство утверждения б) проведем по аналогичной схеме. Покажем сначала, что в случае $X_{\text{opt}}^0 \cap X_{\text{opt}}^1 \neq \emptyset$ справедлива оценка $N_{\min} \geq N_{\min}^0 + N_{\min}^1$. Зафиксируем произвольное действительное $\tilde{\alpha}$. Рассмотрим расстановку меток в дереве T , использующую ровно $N(\tilde{\alpha})$ вершин и удовлетворяющую при всех j ограничениям $S_j(\tilde{\alpha}) \in X_j$. Ограничим эту расстановку меток на поддеревья T_0 и T_1 и обозначим число используемых данной разметкой вершин в каждом из этих поддеревьев через n_0 и n_1 соответственно. Имеем:

$$n_0 \geq N_0(\tilde{\alpha} + c(v_{0,0})) \geq N_{\min}^0, \quad n_1 \geq N_1(\tilde{\alpha} + c(v_{0,0})) \geq N_{\min}^1, \quad N(\tilde{\alpha}) \geq n_0 + n_1.$$

Следовательно, справедливо неравенство

$$N(\tilde{\alpha}) \geq n_0 + n_1 \geq N_{\min}^0 + N_{\min}^1.$$

В силу произвольности $\tilde{\alpha}$ получаем отсюда требуемую оценку

$$N_{\min} \geq N_{\min}^0 + N_{\min}^1.$$

Далее, пусть $\tilde{\alpha} \in X_{\text{opt}}^0 \cap X_{\text{opt}}^1$. Покажем, что $N(\tilde{\alpha}) \leq N_{\min}^0 + N_{\min}^1$ (отсюда и из предыдущей оценки следуют, в частности, что $N_{\min} = N_{\min}^0 + N_{\min}^1$ и $N(\tilde{\alpha}) = N_{\min}$). Возьмем расстановку меток в поддереве T_0 , использующую ровно $N_{\min}^0 = N_0(\tilde{\alpha})$ вершин и удовлетворяющую при всех j ограничениям $S_j^0(\tilde{\alpha}) \in X_j$. Также возьмем расстановку меток в поддереве T_1 , использующую ровно $N_{\min}^1 = N_1(\tilde{\alpha})$ вершин и удовлетворяющую при всех j ограничениям $S_j^1(\tilde{\alpha}) \in X_j$. Построим расстановку меток в дереве T , объединив разметки поддеревьев T_0 , T_1 и приписав корню $v_{0,0}$ нулевую метку. Полученная расстановка меток использует в точности $N_{\min}^0 + N_{\min}^1$ вершин и удовлетворяет при всех j ограничениям $S_j(\tilde{\alpha}) \in X_j$.

Наконец, покажем, что если $\tilde{\alpha} \notin X_{\text{opt}}^0 \cap X_{\text{opt}}^1$, то $N(\tilde{\alpha}) > N_{\min}^0 + N_{\min}^1$. Зафиксируем произвольное значение $\tilde{\alpha} \notin X_{\text{opt}}^0 \cap X_{\text{opt}}^1$. Рассмотрим расстановку меток в дереве T ,

использующую ровно $N(\tilde{\alpha})$ вершин и удовлетворяющую при всех j ограничениям $S_j(\tilde{\alpha}) \in X_j$. Ограничим эту расстановку меток на поддеревья T_0 , T_1 и обозначим число используемых данной разметкой вершин в каждом из этих поддеревьев через n_0 и n_1 соответственно. Если в исходной разметке корень дерева T не используется, то

$$n_0 \geq N_0(\tilde{\alpha}) \geq N_{\min}^0, \quad n_1 \geq N_1(\tilde{\alpha}) \geq N_{\min}^1,$$

причем хотя бы одно из неравенств строгое, так как $\tilde{\alpha}$ не принадлежит по крайней мере одному из множеств X_{opt}^0 , X_{opt}^1 . Следовательно, выполняется оценка

$$N(\tilde{\alpha}) = n_0 + n_1 > N_{\min}^0 + N_{\min}^1.$$

Если же в исходной разметке корень используется, то

$$n_0 \geq N_0(\tilde{\alpha} + c(v_{0,0})) \geq N_{\min}^0, \quad n_1 \geq N_1(\tilde{\alpha} + c(v_{0,0})) \geq N_{\min}^1,$$

откуда получаем оценку

$$N(\tilde{\alpha}) = n_0 + n_1 + 1 > N_{\min}^0 + N_{\min}^1.$$

Лемма 2 доказана.

ЛЕММА 3. *Справедливы следующие утверждения:*

- а) *если $\tilde{\alpha} \in X_{\text{opt}}$, то существует расстановка меток, удовлетворяющая при всех значениях j ограничениям $S_j(\tilde{\alpha}) \in X_j$ и использующая ровно $N_{\min} = N(\tilde{\alpha})$ вершин, в которой корень не используется, т.е. $c(v_{0,0}) = 0$;*
- б) *если $\tilde{\alpha} \notin X_{\text{opt}}$, а α – произвольный элемент множества X_{opt} , то существует расстановка меток, удовлетворяющая при всех значениях j ограничениям $S_j(\tilde{\alpha}) \in X_j$ и использующая ровно $N(\tilde{\alpha})$ вершин, в которой корню приписана метка $c(v_{0,0}) = (\alpha - \tilde{\alpha})$.*

ДОКАЗАТЕЛЬСТВО. Рассмотрим сначала случай $\tilde{\alpha} \in X_{\text{opt}}$ (утверждение а)). Возьмем произвольную расстановку меток, использующую ровно $N(\tilde{\alpha}) = N_{\min}$ вершин и удовлетворяющую при всех j ограничениям $S_j^0(\tilde{\alpha}) \in X_j$. Покажем, что в этой разметке корень не используется. Действительно, предположим, что метка корня $c = c(v_{0,0})$ в этой разметке отлична от нуля. Построим новую разметку, получаемую из рассматриваемой заменой метки корня нулем. Новая разметка, с одной стороны, использует ровно $(N_{\min} - 1)$ вершину, а, с другой, удовлетворяет при всех j ограничениям $S_j(\tilde{\alpha} + c) \in X_j$. Соответственно,

$$N(\tilde{\alpha} + c) \leq N_{\min} - 1 < N_{\min},$$

что противоречит определению величины $N_{\min}$. Полученное противоречие доказывает, что в исходной разметке корень не используется.

Далее, пусть $\tilde{\alpha} \notin X_{\text{opt}}$ и α – произвольный элемент множества X_{opt} (утверждение б)). Возьмем произвольную расстановку меток, удовлетворяющую при всех j ограничениям $S_j(\alpha) \in X_j$ и использующую при этом ровно $N(\alpha) = N_{\min}$ вершин. Согласно показанному выше (при доказательстве утверждения а)), корень в этой расстановке меток не используется. Построим новую расстановку меток, заменив в рассматриваемой расстановке метку корня на разность $(\alpha - \tilde{\alpha})$. Тогда новая разметка использует ровно $N_{\min} + 1 = N(\tilde{\alpha})$ вершин. При этом суммы $S_j(\tilde{\alpha})$ новой

разметки совпадают с суммами $S_j(\alpha)$ исходной разметки и, значит, для новой разметки при всех j выполняются ограничения $S_j(\tilde{\alpha}) \in X_j$.

Лемма 3 доказана.

ЗАМЕЧАНИЕ 1. Отметим, что в случае тривиальных деревьев, т.е. деревьев, состоящих только из корня, утверждения леммы 3 также верны. Этот факт сразу же вытекает из определений используемых величин.

Используя утверждения лемм 2 и 3, докажем, что описанный алгоритм действительно строит допустимую расстановку меток, использующую минимально возможное число вершин.

Введем некоторые обозначения, аналогичные введенным ранее. Для вершин v дерева T через $T(v)$ обозначим максимальное поддерево дерева T с корнем v . Легко видеть, что $T = T(v_{0,0})$, $T_0 = T(v_{1,0})$, $T_1 = T(v_{1,1})$.

Пусть $\tilde{v}$ – произвольная вершина дерева T , а j – такой индекс, что концевая вершина $v_{K,j}$ принадлежит поддереву $T(\tilde{v})$. Обозначим через $P(\tilde{v}, v_{K,j})$ множество всех вершин поддерева $T(\tilde{v})$, лежащих на пути от вершины $\tilde{v}$ до вершины $v_{K,j}$, включая сами вершины $\tilde{v}$ и $v_{K,j}$. Через $S(\tilde{v}, v_{K,j})$ обозначим сумму $\sum_{v \in P(\tilde{v}, v_{K,j})} c(v)$. Через $S(\tilde{v}, v_{K,j}; \alpha)$ обозначим сумму $S(\tilde{v}, v_{K,j}) + \alpha$.

Через $N(\tilde{v}; \alpha)$ обозначим минимальное число вершин в дереве $T(\tilde{v})$, которое нужно использовать (т.е. которым нужно приписать ненулевые метки), чтобы для всех j таких, что концевая вершина $v_{K,j}$ принадлежит поддереву $T(\tilde{v})$, добиться включений $S(\tilde{v}, v_{K,j}; \alpha) \in X_j$. Через $N_{\min}(\tilde{v})$ обозначим $\min_{\alpha \in \mathbb{R}} N(\tilde{v}; \alpha)$. Через $X_{\text{opt}}(\tilde{v})$ обозначим $\text{Arg} \min_{\alpha \in \mathbb{R}} N(\tilde{v}; \alpha)$. Легко видеть, что

$$X_{\text{opt}}(v_{0,0}) = X_{\text{opt}}, \quad X_{\text{opt}}(v_{1,0}) = X_{\text{opt}}^0, \quad X_{\text{opt}}(v_{1,1}) = X_{\text{opt}}^1.$$

Отметим, что для индексов $j \in \{0, 1, 2, \dots, 2^K - 1\}$ поддерево $T(v_{K,j})$ состоит из одной вершины $v_{K,j}$ и $S(v_{K,j}, v_{K,j}; \alpha) = c(v_{K,j}) + \alpha$. Таким образом, если $\alpha \in X_j$, то для достижения включения $S(v_{K,j}, v_{K,j}; \alpha) \in X_j$ можно пометить вершину $v_{K,j}$ нулем, т.е. не использовать ее. Если же $\alpha \notin X_j$, то для достижения включения $S(v_{K,j}, v_{K,j}; \alpha) \in X_j$ вершину $v_{K,j}$ обязательно нужно использовать. Таким образом, $N_{\min}(v_{K,j}) = 0$, а $X_{\text{opt}}(v_{K,j}) = X_j = X_{K,j}$.

Применяя лемму 2 для поддеревьев

$$T(v_{K-1,0}), \quad T(v_{K-1,1}), \quad \dots, \quad T(v_{K-1,2^{K-1}-1}),$$

получаем, что для всех $j \in \{0, 1, 2, \dots, 2^{K-1} - 1\}$ справедливы равенства $X_{K-1,j} = X_{\text{opt}}(v_{K-1,j})$. Продолжая двигаться по ярусам от концевых вершин к корню, получаем, что для всех $k \in \{0, 1, 2, \dots, K\}$ и всех $j \in \{0, 1, 2, \dots, 2^k - 1\}$ справедливы равенства $X_{k,j} = X_{\text{opt}}(v_{k,j})$.

Далее, рассмотрим все допустимые разметки дерева T , т.е. разметки, для которых при всех j имеет место включение $S_j = S_j(0) \in X_j$. В силу определений минимально возможное число используемых вершин в таких разметках равно $N(0)$. Согласно утверждению а) леммы 3, если 0 является элементом множества $X_{\text{opt}} = X_{0,0}$, то найдется допустимая разметка, использующая минимально возможное число вершин, в которой корень не используется, т.е. $c_{0,0} = 0$. В противном же случае в соответствии с утверждением б) леммы 3 для произвольного элемента $\alpha \in X_{\text{opt}} = X_{0,0}$ найдется допустимая разметка, использующая минимально возможное число вершин и приписывающая корню дерева метку α .

После того, как корню дерева T метка уже приписана, задача расстановки меток сводится к приписыванию меток вершинам поддеревьев T_0, T_1 , причем ограничения приобретают вид

$$\begin{aligned} S_j &= S(v_{1,0}, v_{K,j}) + c_{0,0} = S(v_{1,0}, v_{K,j}; S'_{1,0}) \in X_j \quad \text{для } j \in \{0, 1, 2, \dots, 2^{K-1} - 1\}, \\ S(v_{1,1}, v_{K,j}; S'_{1,1}) &\in X_j \quad \text{для } j \in \{2^{K-1}, 2^{K-1} + 1, \dots, 2^K - 1\}. \end{aligned}$$

При этом требуется минимизировать число используемых вершин, и расстановка меток в одном из поддеревьев не влияет на выполнение или невыполнение ограничений в другом поддереве. Вновь применяя лемму 3, на этот раз к поддеревьям T_0, T_1 , получаем, что существует допустимая расстановка меток, минимизирующая число используемых вершин, в которой метки, приписанные вершинам $v_{1,0}, v_{1,1}$, совпадают с метками, приписываемыми алгоритмом. Продолжая двигаться по ярусам от корня к конечным вершинам и применять лемму 3 (а для самих конечных вершин – замечание после леммы 3), получаем, что существует допустимая разметка дерева T , минимизирующая число используемых вершин, в которой всем вершинам $v_{k,j}$, $k \in \{0, 1, 2, \dots, K\}$, $j \in \{0, 1, 2, \dots, 2^k - 1\}$, приписаны метки, совпадающие с метками, приписанными алгоритмом. Иными словами, алгоритм строит одну из расстановок меток, которые являются допустимыми и минимизируют среди всех допустимых расстановок меток число используемых вершин.

4. Оценки сложности. Сначала получим оценки на количество операций, осуществляемых алгоритмом, в абстрактном случае, когда никаких предположений о структуре множеств X_j не делается, а операции разделены на следующие виды: пересечение множеств, объединение множеств, проверка принадлежности заданного элемента множеству, получение произвольного элемента множества и сравнение множества с пустым, арифметические операции (а именно, сложение и вычитание). Получим оценки на количество операций в терминах количества конечных вершин 2^K .

На первом этапе алгоритма для определения множеств $X_{k,j}$ для всех неконцевых вершин требуется не более 2^K операций объединения множеств, не более 2^K операций пересечения множеств, а также не более 2^K операций сравнения множества с пустым (не более одной операции каждого вида на каждую неконцевую вершину, а неконцевых вершин всего $2^K - 1$).

На втором этапе алгоритма приписываются метки всем вершинам дерева, число которых $2^{K+1} - 1 < 2 \cdot 2^K$. Для уменьшения числа арифметических операций при вычислении величин $S'_{k,j}$ не будем каждый раз осуществлять суммирование по всем вершинам, лежащим на пути от корня к вершине $v_{k,j}$, а по мере движения по ярусам от корня к конечным вершинам будем накапливать и хранить сумму меток на пути от корня до вершины – значение этой суммы для вершины $v_{k,j}$ и будет равно значениям $S'_{k+1,2j}$ и $S'_{k+1,2j+1}$. При обработке вершины $v_{k,j}$ для получения значения соответствующей суммы после определения метки $c_{k,j}$ требуется только одна арифметическая операция – достаточно к уже сохраненной сумме, соответствующей вершине $v_{k-1,j/2}$ (где $j/2$ – целая часть от деления j на два), прибавить метку $c_{k,j}$. При таком подходе при обработке каждой вершины $v_{k,j}$ к началу обработки величины $S'_{k,j}$ уже известна. При приписывании каждой метки используется не более одной проверки принадлежности заданного элемента множеству, не более одной операции

получения произвольного элемента множества и не более двух арифметических операций (одна – получение разности $\alpha_{k,j} - S'_{k,j}$, вторая – получение “текущей суммы”, т.е. суммы $S'_{k,j} + c_{k,j}$). В результате получаем, что операций проверки принадлежности заданного элемента множеству требуется не более $2 \cdot 2^K$, операций получения произвольного элемента множества также требуется не более $2 \cdot 2^K$, а арифметических операций требуется не более $4 \cdot 2^K$.

Таким образом, число операций каждого вида, осуществляемых алгоритмом, допускает оценку, линейную относительно числа концевых вершин. При этом оценка не зависит от того, как соотносятся множества X_j .

Легко видеть, что линейные по числу концевых вершин оценки справедливы и в случае произвольного ветвления (если считать, что максимальная степень ветвления ограничена).

Операции с множествами (объединение и пересечение множеств, проверка принадлежности заданного элемента множеству) могут отличаться по сложности от арифметических операций. Задав ограничения на структуру множеств X_j , получим оценки на число операций, выполняемых алгоритмом, в которых вместо количества операций над множествами оценивается количество арифметических операций и операций сравнения чисел. Также в этом случае получим оценку на объем памяти, который использует алгоритм.

Рассмотрим сначала случай, когда все множества X_j конечны и заданы перечислением

$$X_j = \{x_0^j, x_1^j, \dots, x_{q(j)-1}^j\}, \quad \text{где } x_0^j < x_1^j < \dots < x_{q(j)-1}^j.$$

Для хранения всех множеств X_j в этом случае требуется объем памяти, линейный относительно $Q = q(0) + q(1) + \dots + q(2^K - 1)$. Отметим, что если мощность всех множеств X_j не превышает некоторой константы $q_{\max}$, то Q оценивается сверху через количество концевых вершин, умноженное на $q_{\max}$.

Количество операций (сравнение чисел), необходимых для выполнения объединения и пересечения заданных таким образом множеств A_0 и A_1 , допускает оценку, линейную относительно суммы $|A_0| + |A_1|$, где $|A_0|$ и $|A_1|$ – мощности множеств A_0 и A_1 соответственно (см., например, [2; п. 4.4]). Действительно, для выполнения объединения (с учетом того, что результат также должен быть записан упорядоченно) можно идти по множествам слева направо (т.е. от меньшего элемента к большему), добавляя к результирующему множеству тот из текущих элементов первого или второго множества, который меньше, и переходя в множестве, элемент которого добавлен, к следующему элементу. В случае, когда текущие элементы совпадают, делается добавление к результирующему множеству соответствующего числа и осуществляется переход к следующему элементу сразу в двух множествах. Операция пересечения может быть реализована аналогично, разница заключается лишь в том, что к результирующему множеству нужно добавлять элементы только в случаях, когда текущие элементы двух множеств совпадают. Аналогичные линейные относительно суммы мощностей оценки имеют место и в случае, когда рассматриваются объединения и пересечения не пары, а произвольного набора множеств (в этом случае мультипликативная константа в оценке зависит от количества множеств в наборе).

Соответственно, для получения множеств $X_{K-1,j}$ ($j = 0, 1, 2, \dots, 2^{K-1} - 1$) требуется выполнить не более $C(q(2j) + q(2j + 1))$ операций сравнения, причем мощность

множества $X_{K-1,j}$ не превосходит $q(2j) + q(2j + 1)$ (C здесь сумма мультипликативных констант из оценок на число операций, необходимых для осуществления пересечения и объединения множеств). Суммируя эти оценки, получаем, что для получения всех множеств $X_{K-1,j}$ яруса $K - 1$ требуется не более CQ операций, причем сумма мощностей этих множеств не превосходит Q . Это же рассуждение применимо и при последующих переходах от яруса k к ярусу $k - 1$. Таким образом, верхняя оценка и на число операций, и на объем памяти, необходимой для первого этапа алгоритма, линейна относительно произведения $QK = Q \log_2(2^K)$. В частности, если зафиксирована максимально возможная мощность $q_{\max}$ множеств X_j , то оценка линейна относительно $2^K \cdot K = 2^K \log_2(2^K)$.

Основной операцией второго этапа алгоритма, связанной с множествами, является проверка принадлежности заданного элемента множеству. Тривиальной верхней оценкой на число операций сравнения для осуществления проверки здесь является оценка, линейная относительно числа элементов множества (полный перебор). (При использовании определенных структур, например, массивов, для хранения множеств $X_{k,j}$ легко получить и логарифмическую оценку.) Таким образом, при осуществлении операций проверки для всех вершин яруса k число операций будет оцениваться сверху линейно через сумму мощностей множеств $X_{k,j}$ ($j = 0, 1, 2, \dots, 2^k - 1$). Но, как уже отмечалось, эта мощность не превосходит Q . В результате получаем, как и в случае первого этапа алгоритма, верхнюю оценку, линейную относительно $QK = Q \log_2(2^K)$, а в случае зафиксированного ограничения на максимальную мощность множеств X_j — линейную относительно $2^K \cdot K = 2^K \log_2(2^K)$.

Таким образом, в случае, когда множества X_j конечны и заданы перечислением, верхние оценки и на число операций (сложение, вычитание, сравнение чисел), и на объем требуемой памяти линейны относительно $Q \cdot 2^K = Q \log_2(2^K)$, а в случае дополнительного ограничения на максимальную мощность множеств X_j — линейны относительно $2^K \cdot K = 2^K \log_2(2^K)$.

Рассмотрим теперь случай, когда все множества X_j представляют собой отрезки $[a_j; b_j]$ (либо все множества X_j представляют собой интервалы $(a_j; b_j)$). В этом случае, как и в предыдущем, имеют место верхние оценки на число операций и на объем памяти, линейные относительно $2^K \cdot K = 2^K \log_2(2^K)$. Действительно, каждое из множеств $X_{k,j}$ в рассматриваемом случае представляет собой конечное объединение попарно не пересекающихся отрезков (интервалов). Соответственно, для задания этого множества достаточно хранить концы этих отрезков (интервалов). При этом затраты на объединение и пересечение множеств такого вида (в терминах операций сравнения) линейны относительно числа промежутков в объединяемых или пересекаемых множествах: аналогично предыдущему, объединяем конечные упорядоченные множества из границ промежутков, дополнительно приписывая границам их тип (начало, конец, или одновременно начало и конец промежутка), затем проходим слева направо (от наименьшей границы к наибольшей) и, учитывая типы границ, формируем результат объединения или пересечения. Кроме того, количество граничных точек у объединения (пересечения) множеств рассматриваемого вида не превышает суммарное число граничных точек у объединяемых (пересекаемых) множеств. С учетом этих тривиальных замечаний, для получения указанных линейных оценок достаточно фактически повторить рассуждения, использовавшиеся при обосновании оценок в предыдущем случае.

5. Приложение к приближению непрерывной функции. Обозначим через $\Delta_{k,j}$ двоичный полуотрезок $[j/2^k; (j+1)/2^k)$, через $\chi_{k,j}$ – характеристическую функцию $\Delta_{k,j}$ ($k = 0, 1, 2, \dots$, $j = 0, 1, 2, \dots, 2^k - 1$). Для полуотрезка $\Delta_{k,j}$ значение k будем называть *рангом* или *номером пачки*.

Пусть f – действительнoзначная равномерно непрерывная на полуотрезке $[0; 1)$ функция, $\varepsilon_{\max}$ – некоторое положительное число. Рассмотрим задачу построения конечной линейной комбинации функций $\{\chi_{k,j}\}_{k,j}$, которая приближает функцию f в метрике $\rho(g, h) = \max_{x \in [0; 1)} |g(x) - h(x)|$ с точностью не хуже $\varepsilon_{\max}$ и содержит среди всех таких линейных комбинаций минимальное число членов. Легко видеть, что требуемая линейная комбинация всегда существует, но единственности, вообще говоря, нет.

Эта задача близка к широко изучающейся задаче поиска наилучшего n -членного приближения (см., например, обзорные статьи [3], [4] и, в частности, [3; п. 3.4]). Разница заключается в том, что в задаче об n -членном приближении фиксируется число слагаемых в приближающей линейной комбинации и минимизируется погрешность приближения. В рассматриваемой же задаче фиксируется другой параметр, описывающий приближающую линейную комбинацию, а именно, максимальная разрешенная погрешность, и минимизируется число используемых слагаемых.

В терминах сжатия информации с потерей части данных фиксация числа слагаемых, возникающая в задаче построения наилучшего n -членного приближения, содержательно означает, что ограничивается объем памяти, необходимый для хранения данных после сжатия, и в рамках этого ограничения осуществляется сжатие с минимально возможными потерями. Ограничение, возникающее в рассматриваемой задаче, в терминах сжатия данных означает, что фиксируется наибольшая допустимая погрешность, и в рамках этого ограничения осуществляется максимально возможное сжатие.

В настоящее время алгоритмы, позволяющие в рассматриваемом (или каком-либо более общем) случае эффективно строить наилучшие n -членные приближения, или даже n -членные приближения, гарантированно близкие к наилучшим, неизвестны. Вместе с тем, сформулированная задача о построении приближающей линейной комбинации характеристических функций может быть легко переформулирована в терминах задачи о расстановке меток в дереве. Соответственно, применение описанного в первой части работы алгоритма позволяет эффективно строить требуемую приближающую комбинацию для произвольной пары $(f, \varepsilon_{\max})$. Отметим, что алгоритм обеспечивает построение не близкой к оптимальной, а именно оптимальной линейной комбинации, причем обеспечивает это не для отдельных классов функций, а для каждой индивидуальной приближаемой функции.

ЗАМЕЧАНИЕ 2. Следует отметить, что рассматриваемая задача близка также к задаче, описанной в [3; п. 3.3], где, как и в рассматриваемой задаче, фиксируется максимальная допустимая погрешность $\varepsilon_{\max}$, но дополнительно требуется, чтобы промежутки, характеристические функции которых входят в линейную комбинацию, образовывали разбиение (т.е. не перекрывались). Это дополнительное требование является существенным, оно может приводить к значительному росту минимально возможного числа слагаемых в линейных комбинациях, обеспечивающих допустимую погрешность приближения. Чтобы это показать, достаточно рассмотреть пример приближаемой функции $f(x) = 2\chi_{K,0}(x) - \chi_{0,0}(x)$, подробно обсуждаемый в последней части статьи (см. также [3; п. 3.4]).

Опишем сведение задачи о построении требуемой линейной комбинации к рассмотренной задаче об оптимальной расстановке меток в дереве. Заметим сначала, что при построении линейной комбинации достаточно рассматривать характеристические функции не всех двоичных промежутков, а только промежутков начальных пачек. Более точно, имеет место следующая лемма.

ЛЕММА 4. Пусть колебание функции f на всех двоичных полуотрезках ранга K не превосходит $2\varepsilon_{\max}$. Тогда существует требуемая линейная комбинация (т.е. линейная комбинация, приближающая функцию f с погрешностью не более $\varepsilon_{\max}$ и имеющая среди всех таких линейных комбинаций минимальное число слагаемых), в которую входят только характеристические функции двоичных промежутков ранга не выше K .

ДОКАЗАТЕЛЬСТВО. Рассмотрим произвольную линейную комбинацию характеристических функций двоичных промежутков, приближающую функцию f с погрешностью, не превосходящей $\varepsilon_{\max}$, и содержащую минимально возможное число слагаемых. Если эта линейная комбинация содержит характеристические функции промежутков ранга, превосходящего K , построим по этой линейной комбинации новую так, что число слагаемых не увеличивается, погрешность приближения по-прежнему не превосходит $\varepsilon_{\max}$ и все слагаемые построенной линейной комбинации соответствуют двоичным промежуткам ранга не выше K . Рассмотрим последовательно все значения j , для которых в исходной линейной комбинации присутствует хоть одно слагаемое, соответствующее функции $\chi_{k,m}$ с $k > K$ и $\Delta_{k,m} \subset \Delta_{K,j}$. Найдем такую константу $C_{K,j}$, что при всех $x \in \Delta_{K,j}$ значение $f(x)$ отличается от этой константы не более, чем на $\varepsilon_{\max}$ (это можно сделать, так как колебание f на $\Delta_{K,j}$ не превосходит $2\varepsilon_{\max}$). Уберем из линейной комбинации все слагаемые, соответствующие функциям $\chi_{k,m}$ с $k \geq K$ и $\Delta_{k,m} \subset \Delta_{K,j}$, и добавим слагаемое $C_{K,j}\chi_{K,j}$.

Полученная линейная комбинация приближает f с погрешностью не более $\varepsilon_{\max}$. Действительно, если для некоторого значения j описанные изменения, связанные с $\chi_{K,j}$, были сделаны, то оценка погрешности приближения на $\Delta_{K,j}$ следует из выбора константы $C_{K,j}$. Если же для этого значения j изменения, связанные с $\chi_{K,j}$, не потребовались, то оценка погрешности приближения на $\Delta_{K,j}$ следует из того, что для исходной линейной комбинации такая оценка была справедлива. Кроме того, полученная линейная комбинация содержит не больше слагаемых, чем исходная, а исходная содержала минимально возможное число слагаемых.

Зафиксируем значение K такое, что на всех двоичных полуотрезках ранга K колебания функции f не превосходят $2\varepsilon_{\max}$. Существование такого значения K следует из равномерной непрерывности функции f .

Для каждого $j \in \{0, 1, 2, \dots, 2^K - 1\}$ обозначим через I_j образ промежутка $\Delta_{K,j}$: $I_j = f(\Delta_{K,j})$. Из непрерывности функции f следует, что множества I_j являются промежутками, а из выбора K — то, что длина каждого из этих промежутков не превосходит $2\varepsilon_{\max}$. Через X_j обозначим множество всех таких действительных чисел x , что для любого $y \in I_j$ справедливо неравенство $|x - y| \leq \varepsilon_{\max}$. Легко видеть, что множество X_j — это отрезок (возможно, вырожденный) $[\beta_j - \varepsilon_{\max}; \alpha_j + \varepsilon_{\max}]$, где α_j и β_j — соответственно левый и правый концы промежутка I_j .

Опишем взаимно-однозначное соответствие между всевозможными линейными комбинациями характеристических функций двоичных промежутков ранга не выше K и всевозможными разметками корневого полного бинарного дерева высоты K .

Линейной комбинации поставим в соответствие следующую разметку: если слагаемое, соответствующее $\chi_{k,j}$ ($k = 0, 1, 2, \dots, K$, $j = 0, 1, 2, \dots, 2^k - 1$), в этой линейной комбинации отсутствует, то полагаем $c_{k,j} = 0$ (т.е. не используем вершину $v_{k,j}$); если же слагаемое, соответствующее $\chi_{k,j}$, в этой линейной комбинации присутствует, то в качестве $c_{k,j}$ возьмем коэффициент при $\chi_{k,j}$. Линейная комбинация характеристических функций двоичных промежутков представляет собой кусочно-постоянную функцию, и легко видеть, что для каждого $j \in \{0, 1, 2, \dots, 2^K - 1\}$ значение линейной комбинации на промежутке $\Delta_{K,j}$ совпадает со связанной с построенной разметкой суммой S_j . Таким образом, учитывая определение множеств X_j , получаем, что задача о построении линейной комбинации характеристических функций, приближающей заданную функцию f с погрешностью не больше $\varepsilon_{\max}$ и содержащей при этом минимально возможное число слагаемых, эквивалентна задаче о расстановке меток в дереве, где в качестве разрешенных множеств выступают в точности множества X_j .

ЗАМЕЧАНИЕ 3. Абсолютно аналогичное рассмотрение (с сохранением всех результатов) естественно проводить не на полуотрезке $[0, 1)$, а на модифицированном отрезке $[0, 1]^*$, или, эквивалентно, на двоичной топологической группе (см., например, [5; § 1.2]). В этом случае и приближаемая функция, и характеристические функции двоичных промежутков являются непрерывными.

ЗАМЕЧАНИЕ 4. В случае, когда при переходе к следующей пачке полуотрезки делятся не на две части, а на произвольное (не обязательно всюду одинаковое) число частей, задача абсолютно аналогично сводится к задаче о расстановке меток в дереве, но дерево в этом случае уже не является бинарным. Это же справедливо и в случае, когда функция определена не на промежутке прямой, а на прямоугольнике в $\mathbb{R}^2$ (и при переходе к следующей пачке прямоугольник дробится, например, на 4 подобных), или на параллелепипеде в $\mathbb{R}^3$ (и при переходе к следующей пачке параллелепипед дробится, например, на 8 подобных). Кроме того, абсолютно аналогичные рассуждения применимы и в случае, когда функция определена на более сложных множествах в $\mathbb{R}^n$, и дробление осуществляется необязательно на одинаковые подмножества.

ЗАМЕЧАНИЕ 5. Один из стандартных методов, связанных с приближением кусочно-постоянными функциями – использование разложений по системе Хаара (см., например, [6; гл. 3], а также [3; п. 3.5, 3.6]). Однако в рассматриваемой постановке использование разложения именно по характеристическим функциям представляется более естественным. Одна из основных причин этого заключается в следующем. При переходе от разложения по системе Хаара к разложению по характеристическим функциям число слагаемых возрастает не более чем вдвое. При обратном же переходе количество слагаемых может вырасти сколь угодно сильно. Это следует, например, из рассмотрения случая, когда в роли приближаемых функций выступают характеристические функции полуотрезков вида $[0, 1/2^k)$.

6. Приложения к компактному хранению информации. Пусть сигнал f представляет собой определенную на отрезке $[0, 1]$ действительную функцию. В приложениях (в частности, при компьютерной обработке информации) работа осуществляется не с самим сигналом, а с его дискретизацией. Один из простейших способов дискретизации заключается в следующем. Фиксируется натуральное число K , и сигнал при обработке заменяется значениями функции f на равномерной

сетке, состоящей из 2^K элементов. Иными словами, сигнал f заменяется на вектор

$$(f_0, f_1, f_2, \dots, f_{2^K-1}), \quad \text{где} \quad f_j = f\left(\frac{j}{2^K}\right).$$

Вместо значений функции в точке в качестве f_j могут браться средние значения функции на двоичном отрезке $[j/2^K; (j+1)/2^K]$ или другие усреднения.

Для того чтобы результат дискретизации был репрезентативен (например, чтобы результаты дискретизации далеких сигналов существенно отличались), значение K при описанном подходе должно быть велико. В то же время, если сигнал f на определенных подотрезках изменяется медленно, то дробить эти подотрезки можно менее сильно, чем те подотрезки, на которых функция f быстро изменяется. Это замечание часто используется для того, чтобы сократить объем памяти, требуемой для хранения дискретизованного сигнала. Один из стандартных способов такого сокращения – сопоставление сигналу бинарного дерева. Это сопоставление осуществляется следующим образом (см., например, [3; п. 3.3]). Если замена сигнала f на одно число (т.е. замена сигнала одной константой) уже дает допустимую погрешность, то сигналу сопоставляется дерево, состоящее лишь из корня, и корню приписывается именно это число. В противном случае в дерево добавляются корень и вершины первого яруса $v_{1,0}$ и $v_{1,1}$. Если замена сигнала на отрезке $[0; 1/2]$ на одно число (константу) дает допустимую погрешность, то вершина $v_{1,0}$ объявляется концевой, и ей приписывается это число. В противном случае в дерево добавляются вершины $v_{2,0}$ и $v_{2,1}$. Аналогично, если замена сигнала на отрезке $[1/2; 1]$ на одно число (константу) дает допустимую погрешность, то вершина $v_{1,1}$ объявляется концевой, и ей приписывается это число. В противном случае в дерево добавляются вершины $v_{2,2}$ и $v_{2,3}$. Процесс продолжается, пока в дереве присутствуют листья, не объявленные концевыми вершинами: если замена сигнала на соответствующем вершине дерева промежутке на одно число (константу) дает допустимую погрешность, то вершина объявляется концевой, и ей приписывается это число, в противном случае в дерево добавляются две вершины следующего яруса, соединенные ребрами с рассматриваемой вершиной. Таким образом строится бинарное дерево, в котором концевым вершинам приписаны числа, а неконцевые вершины не используются (т.е. им не приписаны никакие числа). Приближенное восстановление сигнала по такой дискретизации основывается на том, что всем вершинам, включая концевые, естественным образом сопоставляются двоичные отрезки, приписанное концевой вершине число есть некоторое усреднение сигнала по соответствующему двоичному отрезку, и при этом для различных концевых вершин соответствующие этим вершинам отрезки не перекрываются и объединение всех таких отрезков дает носитель сигнала – отрезок $[0; 1]$.

Полученное таким образом дерево не является полным: количество ребер в пути от корня до концевой вершины может быть различным для различных концевых вершин. Структура дерева адаптирована под сигнал: зонам, где сигнал изменяется медленно, соответствуют низкие поддеревья, а в зонах быстрого изменения сигнала соответствующие поддеревья высокие. При этом разница в расстоянии от корня до различных концевых вершин для многих сигналов является большой, так что общее количество вершин в построенном дереве (и, тем более, количество концевых вершин) существенно меньше, чем количество концевых вершин в полном дереве

той же высоты, или, эквивалентно, чем количество чисел, необходимых для удовлетворительной дискретизации сигнала в случае равномерного дробления отрезка $[0; 1]$. Это позволяет, несмотря на дополнительные расходы, связанные с появлением неконцевых вершин и вообще с дополнительной структурой дерева, обеспечить существенное сокращение объема памяти, требуемого для хранения дискретизации сигнала.

Аналогичным образом устроена ситуация в случае, когда сигнал представляет собой функцию, определенную на прямоугольнике (например, графическое изображение), параллелепипеде (например, томограммы) или брусах в пространствах большей размерности, а также на более сложных множествах. Разница заключается лишь в способах разбиения и степени ветвления. Например, в двумерном случае часто осуществляется разбиения прямоугольника на четыре подобных, так что кратность ветвления равна четырем, и получаемые деревья иногда называют квадродеревьями. В трехмерном случае кратность ветвления часто равна восьми (разбиение параллелепипеда на восемь подобных), и получаемые деревья иногда называют октодеревьями. Подробнее о квадродеревьях, октодеревьях и некоторых их приложениях см., например, [7].

В рамках рассматриваемой структуры дерева представляется естественным со-держательно использовать не только концевые вершины, но и остальные вершины. А именно, представляется естественным разрешить приписывать числа и неконцевым вершинам, и при восстановлении сигнала ассоциировать с концевыми вершинами $v_{k,j}$ не приписанные этим вершинам числа $c_{k,j}$, а соответствующие вершинам суммы $S_{k,j}$. При этом во многих приложениях разумно хранить лишь используемые вершины (вершины с ненулевыми метками), сохраняя при этом информацию, позволяющую восстановить структуру дерева, например, сохраняя для вершин значения индексов, указывающих на соответствующий вершине двоичный подотрезок (в случае больших размерностей – прямоугольник, параллелепипед, многомерный брус, или другое соответствующее вершине множество). При таком подходе для определения результата дискретизации естественно строить расстановку меток в дереве, минимизирующую число используемых вершин.

Очевидно, что при приписывании меток и неконцевым вершинам количество используемых вершин не будет больше, чем в случае приписывания меток лишь концевым вершинам (в первом случае строится минимизирующая число используемых вершин расстановка меток в классе, включающем как подмножество разметки, использующие лишь концевые вершины). При этом уменьшение числа используемых вершин может быть очень велико. Чтобы показать это, рассмотрим следующий пример (см. также [3; п. 3.4]). Пусть сигнал f равен 1 на промежутке $[0; 2^{-K}]$ ($K \in \mathbb{N}$) и равен -1 на промежутке $(2^{-K}; 1]$. Дискретизацию будем считать удовлетворительной, если она позволяет получить приближение сигнала с погрешностью не больше $\varepsilon_{\max}$, где $\varepsilon_{\max} \in (0; 1)$, в метрике L^∞ (т.е. будем игнорировать значения сигнала на множестве меры нуль, а в рассматриваемом случае – лишь в одной точке 2^{-K}). В таком случае при подходе, связанном с приписыванием меток только концевым вершинам, будет построено дерево, содержащее $(K + 1)$ концевую вершину:

$$v_{K,0}, v_{K,1}, v_{K-1,1}, v_{K-2,1}, \dots, v_{1,1}$$

(первой вершине будет приписана единица, всем другим – минус единица). В то же время, в подходе, позволяющем использовать и неконцевые вершины, количе-

ство используемых вершин будет равно двум: например, могут быть использованы вершины $v_{0,0}$ и $v_{K,0}$ с метками -1 и 2 соответственно.

ЗАМЕЧАНИЕ 6. Известны модификации построения дискретизации с использованием деревьев, обеспечивающие экономию памяти за счет деления на каждом шаге отрезков (прямоугольников, параллелепипедов, брусков) не на равные части, а на различные (см., например, [8; п. 2], [9; разд. 4.4], [10]). При этом метки, как и в стандартном способе, приписываются только конечным вершинам, а выбор новых границ на очередном шаге разбиения осуществляется так, чтобы в итоге по возможности минимизировать число конечных вершин. Для многих приложений такие модификации дают хорошие результаты, но в общем случае они обладают рядом недостатков, к числу которых можно отнести дополнительные вычислительные затраты, связанные как раз с подзадачей выбора новой границы, а также свойственное многим прикладным алгоритмам использование эвристик при выборе новых границ разбиения. В случае деления на неравные части содержательное использование не только конечных, но и внутренних вершин дерева также может применяться для сокращения объема требуемой памяти.

Автор выражает искреннюю благодарность рецензенту за ценные замечания и рекомендации.

СПИСОК ЦИТИРОВАННОЙ ЛИТЕРАТУРЫ

- [1] О. Оре, *Теория графов*, Наука, М., 1980.
- [2] А. В. Ахо, Дж. Э. Хопкрофт, Дж. Д. Ульман, *Структуры данных и алгоритмы*, Издательский дом “Вильямс”, М., 2007.
- [3] R. A. DeVore, “Nonlinear Approximation”, *Acta Numer.*, **7** (1998), 51–150.
- [4] V. N. Temlyakov, “Nonlinear methods of approximation”, *Found. Comput. Math.*, **3:1** (2003), 33–107.
- [5] Б. И. Голубов, А. В. Ефимов, В. А. Скворцов, *Ряды и преобразования Уолша. Теория и применения*, Наука, М., 1987.
- [6] Б. С. Кашин, А. А. Саакян, *Ортогональные ряды*, Изд-во АФЦ, М., 1999.
- [7] H. Samet, *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*, Addison-Wesley Publ., Reading, MA, 1990.
- [8] P. Binev, R. DeVore, “Fast computation in adaptive tree approximation”, *Numer. Math.*, **97:2** (2004), 193–217.
- [9] M. Pharr, G. Humphreys, *Physically Based Rendering: From Theory to Implementation*, Morgan Kaufman Publ., San Francisco, CA, 2004.
- [10] W. Hunt, W. R. Mark, G. Stoll, “Fast kd-tree construction with an adaptive error-bounded heuristic”, 2006 *IEEE Symposium on Interactive Ray Tracing*, Salt Lake City, UT, 2006, 81–88.

В. В. Галатенко

Московский государственный университет
им. М. В. Ломоносова
E-mail: vvgalatenko@yahoo.com

Поступило

24.06.2010

Исправленный вариант

16.10.2010