ДОКЛАДЫ РОССИЙСКОЙ АКАДЕМИИ НАУК. МАТЕМАТИКА, ИНФОРМАТИКА, ПРОЦЕССЫ УПРАВЛЕНИЯ, 2024, том 519, с. 22–27

МАТЕМАТИКА

УДК 519.178

ТОЧНЫЙ КВАДРАТИЧНЫЙ АЛГОРИТМ КРАТЧАЙШЕГО ПРЕОБРАЗОВАНИЯ ДЕРЕВЬЕВ

© 2024 г. К. Ю. Горбунов[1,*], В. А. Любецкий[1,2,**]

Представлено академиком РАН А. Л. Семеновым

Получено 24.01.2024 г.
После доработки 00.08.2024 г.
Принято к публикации 00.08.2024 г.

В статье предложен новый точный квадратичный по сложности алгоритм, который решает задачу кратчайшего преобразования (“выравнивания”) одного нагруженного дерева в другое с учетом произвольных цен операций над деревьями. Предложен набор из трех операций: добавление вершин-делеций к ребру или корню дерева и сдвиг поддерева с делециями.

Ключевые слова: дискретная оптимизация, кратчайшее преобразование дерева, операции над деревом, цена операции, точный алгоритм, квадратичной сложности алгоритм, выравнивание деревьев.

DOI: 10.31857/S2686954324050058, **EDN:** XEFREO

1. ВВЕДЕНИЕ И ПОСТАНОВКА ЗАДАЧИ

Широко используется расстояние Хэмминга между двумя словами одинаковой длины в фиксированном конечном алфавите. Часто рассматриваются слова не обязательно равной длины, как и операции, характерные для рассматриваемого приложения; операции последовательно преобразуют одно данное слово в другое; набор операций заранее выбран и фиксирован в зависимости от приложения. Расстоянием между двумя словами называют длину самой короткой последовательности допустимых операций, которая преобразует одно слово в другое. Более того, обычно каждой операции приписано строго положительное рациональное число, называемое ценой операции. Соответственно расстояние определяется как *минимум* суммы цен операций, которые последовательно преобразуют данные слова, одно в другое. Это расстояние не обязательно симметрично, и называется редакционным расстоянием или расстоянием Левенштейна. Задача кратчайшего преобразования состоит в построении эффективного алгоритма для нахождения этого минимума и, главное, самой последовательности операций, на которой минимум достигается. Эта задача решена многими алгоритмами в духе динамического программирования с квадратичным временем работы, [1, глава 11]. Цепочка операций, которая доставляет минимум, называется *кратчайшей*. Эффективность понимается как доказательство точности алгоритма (или нетривиальной верхней оценки его погрешности) вместе с доказательством низкой степени полиномиальной оценки времени его работы. В приложениях задача кратчайшего преобразования часто возникает для конечных графов, определяемых фиксированным свойством, вместо слов. Например, для ориентированных нагруженных графов, состоящих из цепей и циклов, рассмотренная в [2], и для нагруженных корневых деревьев, рассмотренная в данной работе. Набор операций, которые преобразуют граф, естественно, зависит от рассматриваемого приложения и от графов, которые рассматриваются.

В данной работе такой алгоритм построен для корневых деревьев, у которых все вершины помечены буквами или знаком “–”; эти пометки называются соответственно *типом* или *делецией* в той вершине, к которой они однозначно приписаны. Даны сами нагруженные деревья и матрица, состоящая из произвольных рациональных чисел, которые выражают *сходство* двух типов (соответствие: тип-тип) и *штрафы* за соответствия тип-делеция (и наоборот) и делеция-делеция. Сходство типов может выражаться любым числом, но штрафы обычно задаются неположительным числом. Штраф может зависеть от типа и положения вер-

[1] *Институт проблем передачи информации им. А. А. Харкевича РАН, Москва, Россия*
[2] *Московский государственный университет им. М. В. Ломоносова, Москва, Россия*
[*] *E-mail: gorbunov@iitp.ru*
[**] *E-mail: lyubetsk@iitp.ru*

шины на дереве. В биоинформатических приложениях такие деревья часто называют клеточными (cell lineage tree). В [3] для первых двух из рассмотренных ниже трех операций предложена компьютерная программа, названная mDELTA [4], которая решает задачу кратчайшего преобразования. Отсутствие третьей операции существенно: например, без нее лист преобразуется обязательно в лист, что не всегда выполняется в прикладных задачах. В данной работе, разрешая в задаче кратчайшего преобразования все три операции, мы опишем точный квадратичной сложности алгоритм, который существенно отличается от алгоритма в mDELTA. По аналогии с алгоритмами для слов, в задаче преобразования деревьев говорят, что алгоритм строит *выравнивание* двух исходных деревьев при данных ценах операций.

2. ОПРЕДЕЛЕНИЕ ОПЕРАЦИЙ НАД КОРНЕВЫМИ ДЕРЕВЬЯМИ

Над деревом разрешены следующие три операции, в которых x и y обозначают любые типы или делецию, рис. 1. Любая цепочка G операций — изоморфно вкладывает каждое предыдущее дерево в последующее.

1. На ребро (x, y) добавить вершину-делецию вместе с ее листом-делецией, рис. 1 (операция 1). Подчеркнем, что все операции состоят в расширении исходного дерева T некоторым числом *новых делеций*, приписанных *новым* вершинам в дереве T, которые расположены выше его корня r, ниже него и несравнимо с r (принято, что деревья растут вниз от корня).

2. Выше корня r добавить новый корень-делецию вместе с его листом-делецией, рис. 1 (операция 2). Ниже r ничего не меняется.

3. В вершине (включая корень или лист), помеченной x, пометку x заменить на делецию и инцидентно ей выше добавить новую вершину с пометкой x и инцидентным ей листом-делецией. Выше новой пометки x и ниже соответствующей новой делеции ничего не меняется, рис. 1 (операция 3).

Итак, задачу преобразования деревьев можно переформулировать как следующую **задачу выравнивания**. Даны два бинарных дерева T_1 и T_2. Указанными тремя операциями преобразовать T_1 и T_2 в бинарные деревья T_1' и T_2', $T_1 \subseteq T_1'$, $T_2 \subseteq T_2'$, которые без учета их пометок (т.е. топологически) изоморфны между собой так, что достигает *максимума* качество пары $\{T_1', T_2'\}$ или, иными словами, качество изоморфизма $f\colon T_1' \to T_2'$, которые обозначим $H(T_1, T_2) = H(f)$. *Качеством* называется сумма (по изоморфизму всех вершин в T_1' и T_2') сходств тип-тип плюс штрафы за тип-делеция (или наоборот) и делеция-делеция. В нашем алгоритме качество вычисляется по индукции. Эти две цепочки

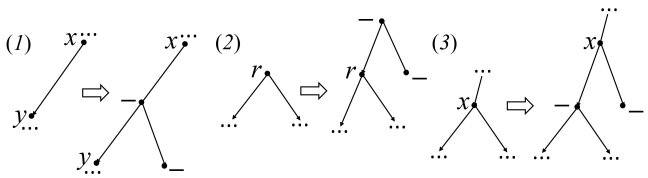

Рис 1. (1) На ребро (x, y) добавлена вершина-делеция вместе с ее листом-делецией. (2) Выше корня r добавлен новый корень-делеция вместе с его листом-делецией. Ниже r ничего не меняется. (3) В вершине (включая корень или лист), помеченной x, пометка x заменена на делецию и инцидентно ей выше добавлена новая вершина с пометкой x и инцидентным ей листом-делецией. Выше нового положения x и ниже соответствующей новой делеции ничего не меняется.

операций $T_1 \to T_1'$ и $T_2 \to T_2'$, расширения исходных деревьев T_1 и T_2 до T_1' и T_2' вместе с изоморфизмом $f\colon T_1' \to T_2'$ называются *выравниванием* деревьев T_1 и T_2. Решение задачи выравнивания тривиально влечет решение задачи преобразования: преобразуем T_1 в T_1', переходим по f к T_2' и удаляем все добавленные алгоритмом новые делеции и вершины, объединяя соответствующие ребра.

3. АЛГОРИТМ ВЫРАВНИВАНИЯ

Опишем алгоритм в предположении, что в матрице сходств делеция-делеция имеет нулевую цену, а тип-делеция (и наоборот) имеют одну цену, которая не зависит от типа и расположения на деревьях. Это не уменьшает общности алгоритма и доказательства. Результат замены в дереве T корневого типа на делецию назовем *упрощением* T и обозначим T^-.

Итак, даны два корневых бинарных дерева T_1 и T_2. Рассмотрим множество D_1 деревьев, которое состоит из всех поддеревьев в T_1 и их упрощений; аналогично определяется множество D_2. Пусть R, $S \in D_i$, $i = 1$ или 2; R *содержится* в S, $R \subseteq S$, если R — поддерево в S или упрощение поддерева из S. Для единообразия удобнее считать, что каждый лист каждого дерева в D_i дополнен двумя дочерними пустыми поддеревьями с пустыми ребрами к листу (на рисунках такие пустые продолжения листьев не показаны). *Деревом делеций* назовем любое дерево, в котором все пометки делеции, кроме пустых поддеревьев. *Определим* множество P (неупорядоченных) пар, состоящих из деревьев $R_1 \in D_1$ и $R_2 \in D_2$ вместе с *частичным порядком* на парах: $\{R_1, R_2\} \leqslant \{S_1, S_2\}$, если R_1 содержится в $S_1 \in D_1$ и R_2 содержится в $S_2 \in D_2$. Фиксируем любой *линейный порядок*, продолжающий частичный порядок на парах $\{R_1, R_2\}$. Алгоритм выполняет прямую и обратную индукции по этому линейному порядку.

Начало индукции. Обозначим $k(R)$ число вершин в дереве R, которые помечены типами, умноженное на штраф тип-делеция из матрицы сходств.

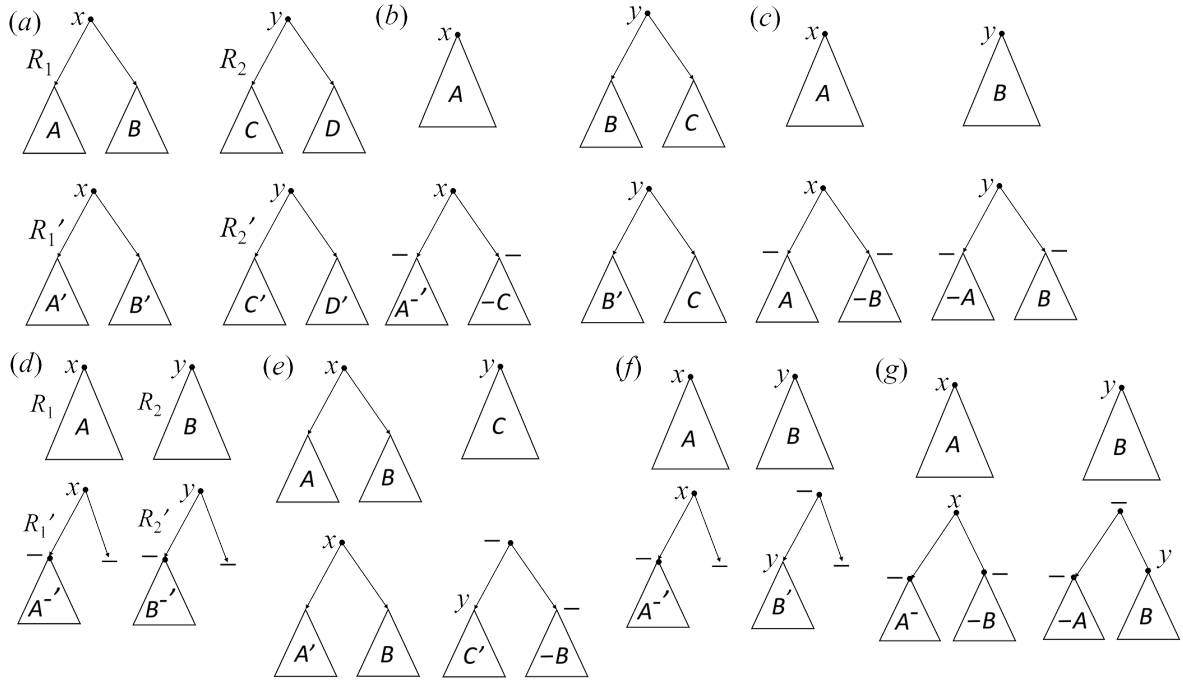

Рис 2. Преобразования $R_1 \to R_1'$ и $R_2 \to R_2'$ пары $\{R_1, R_2\}$, вычисления качеств $H(R_1, R_2)$ и ссылки, на каком преобразовании достигается максимум этих качеств.

Если R_1 и R_2 оба пустые, то качество $H(R_1, R_2) = 0$ и R_1', R_2' и их изоморфизм f пустые. Если одно из R_1 и R_2 — делеция с двумя пустыми поддеревьями и ведущими в них пустыми ребрами (пусть это R_1), а второе непустое, то качество $H(R_1, R_2) = k(R_2)$ и R_1' — дерево делеций, топологически изоморфное R_2, и $R_2' = R_2$. По индукции на прямом ходе вычисляются только качества и приводится ссылка, на каком преобразовании (см. ниже индуктивный шаг) достигается их максимум, а на обратном ходе строятся сами изоморфные расширения T_1' и T_2'.

Индуктивный шаг. Пусть $\{R_1, R_2\}$ текущая пара непустых деревьев в P. Алгоритм вычисляет качество $H(R_1, R_2)$ каждого из 7 *преобразований* $(R_1, R_2) \to (R_1', R_2')$, перечисленых ниже на рис. 2, и создает ссылку на то из них (включая его аргументы), на котором качество максимально до тех пор, пока выполняется $R_1 < T_1$ или $R_2 < T_2$. После этого, обратным ходом индукции от пары (T_1, T_2) до начала индукции алгоритм вычисляет сами деревья $R_1' = T_1'$ и $R_2' = T_2'$ и при желании их изоморфизм f.

Пусть x и y — пометки корней в R_1 и R_2. На рис.2 *сверху* показаны текущие деревья $\{R_1, R_2\}$, а *снизу* — их преобразования $\{R_1', R_2'\}$. *Обозначим* c_{xy} сходство пометок x и y в матрице сходств, включая штрафы. Результат $(\cdot)'$ определяется индукцией по указанному линейному порядку; у дерева $(S^-)'$ корень помечен делецией.

а) В R_1 и R_2 дочерние поддеревья суть A и B, C и D от корней в R_1 и R_2, соответственно, рис. 2a; поддеревья A, B, C, D все пустые или все непустые (отметим, что случай, когда, например, A и B пустые, а C и D непустые, применением к x операции 3 сводится к случаю b ниже). Положим $H(R_1, R_2) = H(A, C) + H(B, D) + c_{xy}$. Аналогично вычислим $H(A, D) + H(B, C) + c_{xy}$, и окончательно положим $H(R_1, R_2)$ равным максимуму по этим двум вариантам. Ссылка указывает, на каком из 7 преобразований и каких его аргументах, достигается максимум. По индукции известны $\{A', C'\}$ и $\{B', D'\}$, которые подвешиваем согласно рис. 2a. Изоморфизм f: $R_1' \to R_2'$ равен объединению изоморфизмов f_1 и f_2, дополненному сопоставлением $x \to y$. Эти данные не используются на прямом ходе. Все вершины в T_1 присутствуют в T_1' с теми же пометками; в вершинах T_1' ниже T_1-листьев и выше T_1-корня или несравнимо с ним присутствуют только делециии; аналогично для T_2 и всех 7 преобразований. Это используется в доказательстве Теоремы 1.

b) Для деревьев R_1 и R_2 положим $A = R_1$ и B, C — дочерние поддеревья в R_2 от его корня y, рис. 2b; поддеревья A, B, C непустые. Положим $H(R_1, R_2) = H(A^-, B) + c_{xy} + k(C)$. Аналогично вычислим $H(A^-, C) + c_{xy} + k(B)$ и два варианта качеств $H(R_2, R_1)$. Здесь используется *обозначение*: $\{R_2, R_1\}$ — пара, у которой R_2 есть новое A, подвешеное к y и объединение бывших B и C, а R_1 есть прежнее A с корнем x, разделенное на новые дочерние B и C. Окончательно положим $H(R_1, R_2)$ равным максимуму по этим четырем вариантам. Здесь к x применяется 3-я операция и подвешиваются $(A^-)'$ и $-C$, а к y подвешиваются B' и C. Изоморфизм f: $R_1' \to R_2'$ определяется как выше.

c) Для деревьев R_1 и R_2 положим $A = R_1$ и $B = R_2$, рис. 2c; деревья A и B непустые. Положим $H(R_1, R_2) = c_{xy} + k(A^-) + k(B^-)$. Здесь к x и y применяется 3-я операция и подвешиваются A^-, $-B$, $-A$, B^-. Изоморфизм f: $R_1' \to R_2'$ тривиален.

d) Пусть хотя бы одна из пометок x и y не делеция. Для деревьев R_1 и R_2 положим $A = R_1$ и $B = R_2$, рис. 2d; деревья A и B непустые. Положим $H(R_1, R_2) = H(A^-, B^-) + c_{xy}$. Здесь к x и y применяется 3-я операция и подвешиваются $(A^-)'$ и $(B^-)'$. Изоморфизм f: $R_1' \to R_2'$ определен по индукции.

e) Для деревьев R_1 и R_2 рассмотрим дочерние A, B от корня в x и положим $C = R_2$ с корнем в y, рис.2e; поддеревья A, B, C непустые. Положим $H(R_1, R_2) = H(A, C) + c_{x-} + k(B)$. Аналогично вычислим $H(B, C) + c_{x-} + k(A)$ и два варианта для $H(R_2, R_1)$. Здесь, как и в (b), используется обозначение: R_2, R_1 — пара, у которой R_2 есть прежнее C с корнем y, разделенное на новые дочерние A и B, а R_1 есть новое C, подвешеное к x и объеденение бывших A и B. Здесь к y применяется 2-я операция и подвешиваются A', B, C', C. Окончательно полагаем $H(R_1, R_2)$ равным максимуму по этим четырем вариантам. Изоморфизм f: $R_1' \to R_2'$ определен по индукции.

f) Пусть x не делеция. Для деревьев R_1 и R_2 положим $A = R_1$ и $B = R_2$, рис. 2f; деревья A и B непустые. Положим $H(R_1, R_2) = H(A^-, B) + c_{x-}$. Аналогично вычислим $H(A, B^-) + c_{x-}$, и окончательно положим $H(R_1, R_2)$ равным максимуму по этим двум вариантам. Здесь к x применяется 3-я операция, а к y применяется 2-я и подвешиваются $(A^-)'$ и B'. Изоморфизм f: $R_1' \to R_2'$ определен по индукции.

g) Для деревьев R_1 и R_2 положим $A = R_1$ и $B = R_2$, рис. 2g; деревья A, B непустые. Положим $H(R_1, R_2) = k(A) + k(B)$. Здесь к x применяется 3-я операция, к y применяется 2-я и подвешиваются A^-, $-B$, $-A$, B. Изоморфизм f: $R_1' \to R_2'$ тривиален.

По окончанию прямого хода алгоритма, обратным ходом алгоритма от $\{T_1 = R_1, T_2 = R_2\}$ согласно расставленным на прямом ходе ссылкам образуем $T_1' = R_1'$, $T_2' = R_2'$.

Замечание 1. Выравнивания связных частей исходных деревьев (не обязательно поддеревьев) выполнятся следующим образом. На прямом ходу любое отрицательное качество пары $p \in P$ заменим на число 0. После обратного хода из полученного изоморфизма f удалим все f-изоморфные поддеревья с качеством 0; и качество сужения изоморфизма f на связные части исходных деревьев равно исходному качеству.

4. ТОЧНОСТЬ АЛГОРИТМА И ПРИМЕР ЕГО РАБОТЫ

Теорема 1. *Алгоритм строит выравнивание T_1' и T_2' двух деревьев T_1 и T_2 с максимальным качеством изоморфизма. Деревья T_1' и T_2' — изоморфные расширения деревьев T_1 и T_2. Время работы алгоритма квадратично от размера исходных данных.*

Набросок доказательства. Индукцией по парам $\{R_1, R_2\}$ покажем, что наш алгоритм строит изоморфизм f_0: $R_1' \to R_2'$ максимального качества. Пусть f: $R_1'' \to R_2''$ — какой-то изоморфизм максимального качества и G_1, G_2 суть последовательности операций, расширяющих R_1 до R_1'' и R_2 до R_2''. Покажем, что $H(f) = H(f_0)$. Если одно из R_1 и R_2 — делеция с двумя пустыми поддеревьями, то равенство очевидно. Пусть r_1 и r_2 — корни в R_1 и R_2 и также вершины в R_1'' и R_2'', для которых $f(r_1) = r_2$. Поддеревья с корнями r_1 и r_2 изморфны, обозначим их соответственно также r_1 и r_2, дополнения поддеревьев $R_1'' \setminus r_1$ и $R_2'' \setminus r_2$, состоящие из одних делеций, также изоморфны. Было условлено, что соответствие делеция-делеция имеет штраф 0, поэтому $H(f) = H(f \restriction r_1)$; ограничение $f \restriction r_1$ обозначим снова f. Пусть u_1 и u_2, u_3 и u_4 — корни дочерних поддеревьев от вершин r_1 и r_2 и одновременно сами эти поддеревья в R_1'' и R_2''. В силу изоморфности $f(u_1) = u_3$ и $f(u_2) = u_4$ или наоборот (пусть выполняется первый случай). Получим изоморфизмы f_1: $R_1'' \restriction u_1 \to R_2'' \restriction u_3$ и f_2: $R_1'' \restriction u_2 \to R_2'' \restriction u_4$, т.е. f есть объединение f_1 и f_2 вместе с $r_1 \to r_2$.

Обозначим $v_i = G_1^{-1}(u_i)$ и $v_i = G_2^{-1}(u_i)$ (для $i = 1$ или 2 и $i = 3$ или 4). Пусть S_i — поддеревья в R_1 и R_2 от v_i. Последовательность операций G_i преобразует дерево S_i в дерево $R_1'' \restriction u_i$. Так что пары $\{S_1, S_3\}$ и $\{S_2, S_4\}$, которые строго меньше $\{R_1, R_2\}$, теми же G_i порождают $\{R_1''|u_1, R_2''|u_3\}$ и $\{R_1''|u_2, R_2''|u_4\}$. Изоморфизмы f_1: $R_1'' \restriction u_1 \to R_2'' \restriction u_3$ и f_2: $R_1'' \restriction u_2 \to R_2'' \restriction u_4$ имеют максимальное качество, так как иначе можно заменить S_1 и S_3 на другие поддеревья, которые породят изоморфные поддеревья с большим качеством вместо $R_1'' \restriction u_1$ и $R_2'' \restriction u_3$; и аналогично заменим S_2 и S_4. По предположению индукции $H(f_1)$ и $H(f_2)$ равны качествам изоморфизмов, построенных алгоритмом, на парах $\{R_1'|u_1, R_2'|u_3\}$ и $\{R_1'|u_2, R_2'|u_4\}$, откуда $H(f) = H(f_0)$.

Предположим $f(r_1) = d \neq r_2$. Возможны три случая.

1) Вершина d расположена выше r_2 и тогда является делецией. Снова обозначим u_i дочерние вершины от r_1 и d. Тогда у d в R_2'' ровно одно дочернее поддерево D (пусть с корнем u_4) — дерево делеций, которое изоморфно дочернему поддереву R от r_1 в R_2''. Качество этого изоморфизма f_2 равно $k(R)$. Пусть u_1 — дочерняя вершина от r_1, не лежащая в R, а u_3 — дочерняя вершина от d, не лежащая в D. Де-

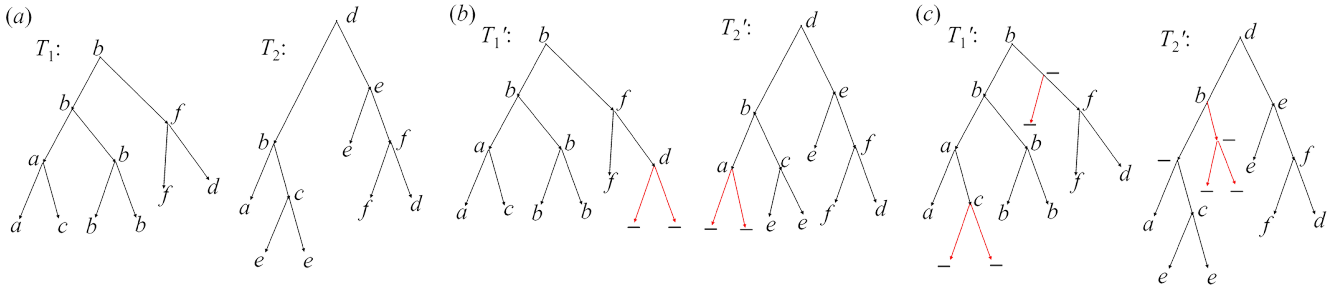

Рис 3. (a) Исходные деревья T_1 и T_2. (b) Сходства типов 1 и штрафы −1. Добавлены делеции и (красным) ребра, после чего получены топологически изоморфные деревья T_1' и T_2'. (c) Сходство типа с собою 4 и с другим типом −3, штраф за делецию −2. Добавлены делеции и (красным) ребра, после чего получены топологически изоморфные деревья T_1' и T_2'.

ревья от u_1 и u_3 f_1-изоморфны, а f есть объединение f_1 и f_2 вместе с $r_1 \to d$. Повторим рассуждения для случая $f(r_1) = r_2$, проведенные там для u_1, и получим соответствующее S_1, а $S_3 = R_2$. Здесь $\{S_1, S_3\}$ строго меньше $\{R_1, R_2\}$.

2) Вершина d не сравнима с r_2 и тогда является делецией. Тогда изоморфизм f переводит все вершины из R_1 в делеции и $H(f) = k(R_1) + k(R_2)$, как для преобразования (g) на рисунке 2g. Поскольку $H(f_0) \geqslant k(R_1) + k(R_2)$, получим $H(f) = H(f_0)$.

3) Вершина d ниже r_2. Изоморфизм f сохраняет отношение порядка на дереве, поэтому $f^{-1}(r_2)$ — делеция, расположенная выше r_1. Этот случай симметричен случаю 1.

Итак, $H(f) = H(f_0)$, т.е. алгоритм точен. Его время работы квадратично от размера исходных деревьев, поскольку таково число пар в P, а обработка каждой пары занимает константное время.

Примеры. 1) Даны деревья T_1 и T_2, показанные на рис. 3a, вершины которых помечены типами a, b, c, d, e, f (пустые поддеревья в листьях не показаны). Числа в матрице сходств равны 1 (тип-тип) и −1 (тип-делеция). Результат T_1' и T_2' работы алгоритма показан на рис. 3b. Качество изоморфизма 9 – 4 = 5.

2) Даны те же деревья T_1 и T_2. Сходство типа с собою 4 и с другим типом −3, штраф за тип-делеция (и наоборот) −2. Обратный ход алгоритма начинается с пары $\{T_1 = R_1, T_2 = R_2\}$ и (ранее полученной) ссылки a_1 (поддеревья A и C, B и D). На 2-м шаге для первой полученной пары $\{R_1 = A, R_2 = C\}$ была ссылка b_3 на новую пару $\{R_1, R_2\}$, где $R_1 = C$ (новое A) и R_2 — дерево с корнем a и листьями a и c (новое дочернее B). Для второй пары первого шага $\{R_1 = B, R_2 = D\}$ была ссылка e_4 на новую пару $\{R_1, R_2\}$, где R_1 — дерево с корнем f и листьями f и d (новое дочернее B) и $R_2 = B$ (новое C), которое в данном случае совпадает с R_1. Дальнейший ход алгоритма тривиален: везде была ссылка a_1. Результат алгоритма показан на рис. 3c: выравнивание (по изоморфизму) имеет 6 одинаковых типов, 1 неравный тип и 8 соответствий тип-делеция. Качество изоморфизма 24 – 3 – 16 = 5.

Замечание 2. Легко модифицировать алгоритм для политомических исходных деревьев так, что он будет выдавать пару их бинарных разрешений политомичности максимального качества. Для пары вершин $\{x, y\}$ соответственно из T_1 и T_2 (в порядке от листьев к корню) рассмотрим непустое множество X дочерних ребер для x и Y для y. Множество X порождает поддерево в T_1, являющееся частью T_1, расположенной ниже ребер из X, а если $|X| > 1$, то включающее и сами эти ребра с их верхним концом; аналогично для Y. Упрощением множества X назовем упрощение этого поддерева. Если X или Y одноэлементно, то поддеревья, порождаемые X и Y, имеют бинарные разрешения максимального качества, которые известны по индукции. Иначе разобьем эти два множества на непустые M_1 и M_2, M_3 и M_4. Рассмотрим ограничения T_1 и T_2 от вершин x и y соответственно на M_1 и M_2, M_3 и M_4. Алгоритм перебирает пары множеств X, Y и их упрощений в порядке возрастания мощности X, а при одинаковой мощности X — по возрастанию мощности Y; для фиксированных X, Y перебор выполняется в порядке (X^-, Y^-), (X^-, Y), (X, Y^-), (X, Y). Для каждой пары (X, Y) в любом порядке перебираются все пары их разбиений; остальные три пары рассматриваются аналогично. Пусть X^* обозначает X или X^-. По индукции известны бинарные разрешения максимального качества для пар ограничений с корнями от x и y на, соответственно, (M_1, M_3), (M_1, M_4), (M_2, M_3), (M_2, M_4), (M_1, Y^*), (M_2, Y^*), (X^*, M_3), (X^*, M_4), (X^-, Y^*), (X^*, Y^-). Выберем из всех пар разбиений, перебирая для каждого из них описанные в алгоритме преобразования, рис. 2, разбиение максимального качества и получим пару бинарных разрешений максимального качества для поддеревьев, порождаемых X и Y. Взяв для $\{x, y\}$ наибольшие X и Y, получим бинарное разрешение максимального качества поддеревьев от x и от y, и так дойдем до пары корней исходных деревьев T_1 и T_2.

Здесь перебираются четверки подмножеств, откуда возникает верхняя оценка 2^{4k} времени обработки одной пары вершин, где k — максимальное число дочерних вершин при вершине дерева. Квад-

ратичное время алгоритма для бинарного случая умножается на число порядка 2^{4k}.

Задача преобразования политомических упорядоченных деревьев (у нас деревья неупорядочены) широко изучалась, многочисленные ссылки приведены, например, в [5]. Для нее известен алгоритм решения, точный и с кубическим временем работы; в [5] показано, что при одном предположении это время нельзя существенно улучшить.

ИСТОЧНИК ФИНАНСИРОВАНИЯ

Исследование выполнено за счёт гранта Российского научного фонда № 24-44-00099, https://rscf.ru/project/24-44-00099/.

СПИСОК ЛИТЕРАТУРЫ

1. *Гасфилд Д.* Строки, деревья и последовательности в алгоритмах: Информатика и вычислительная биология (пер. с англ.). СПб.: Невский Диалект; БХВ-Петербург, 2003, 654 с.
2. *Горбунов К. Ю., Любецкий В. А.* Почти точный линейный алгоритм преобразования графов из цепей и циклов, с оптимизацией суммы цен операций // ДАН. 2020. Т. 494. № 1. С. 26–29. https://doi.org/10.31857/S2686954320050343
3. *Yuan M., Yang X., Lin J., Cao X., Chen F., Zhang X., Li Z., Zheng G., Wang X., Chen X., Yang J-R.* Alignment of Cell Lineage Trees Elucidates Genetic Programs for the Development and Evolution of Cell Types // iScience. 2020. V. 23. Art. 101273. https://doi.org/10.1016/j.isci.2020.101273
4. https://github.com/Chenjy0212/mdelta (Дата обращения: 20.01.2024).
5. *Bringmann K., Gawrychowski P., Mozes Sh., Weimann O.* Tree Edit Distance Cannot be Computed in Strongly Subcubic Time (Unless APSP Can) // ACM Trans. Algorithms. 2020. V. 16. № 4. Art. 48. https://doi.org/10.1145/3381878

AN EXACT QUADRATIC ALGORITHM FOR THE SHORTEST TREE TRANSFORMATION

K. Yu. Gorbunov[a], **V. A. Lyubetsky**[a,b]

[a] *Institute for Information Transmission Problems of the Russian Academy of Sciences (Kharkevich Institute), Moscow, Russia*

[b] *Lomonosov Moscow State University, Moscow, Russia*

Presendet by Academician of the RAS A. L. Semenov

The article proposes a new exact quadratic algorithm in complexity that solves the problem of the shortest transformation (“alignment”) of one loaded tree into another, taking into account arbitrary prices of operations on trees. Three operations are considered: adding vertex deletions to an edge or root of a tree and shifting a subtree with deletions.

Keywords: discrete optimization, shortest tree transformation, operations on a tree, operation price, exact algorithm, quadratic complexity algorithm, tree alignment.