ПРОБЛЕМЫ ПЕРЕДАЧИ ИНФОРМАЦИИ

Том 53 **2017** **Вып. 1**

УДК 621.391 : 519.1

© 2017 г. **К.Ю. Горбунов, В.А. Любецкий**

ЛИНЕЙНЫЙ АЛГОРИТМ МИНИМАЛЬНОЙ ПЕРЕСТРОЙКИ СТРУКТУР[1]

Предлагается линейный по времени и используемой памяти алгоритм, строящий минимальную последовательность операций, которая преобразует одну структуру (ориентированный граф из циклов и цепей) в другую. Структуры в такой последовательности могут иметь переменное множество ребер, список операций фиксирован и включает удаление и вставку участка структуры. Приводится полное доказательство точности алгоритма, т.е. того, что он находит соответствующий минимум.

§ 1. Введение

В § 2 приведены формулировки и определения, относящиеся к рассматриваемой в статье и близкой к ней задачам. Этим задачам посвящено много публикаций, среди которых отметим основополагающую работу [1]. Коснемся ниже только нескольких работ, наиболее примыкающих к настоящей статье, см. также [2]. Эти задачи навеяны биологической и медицинской тематикой, поэтому в их постановке принято использовать некоторые слова из этой тематики. Однако мы рассматриваем лишь математическую составляющую этих задач; полученные решения применимы, например, и в технических вопросах.

В работе [3] предложены операции для преобразования хромосомных структур (см. § 2); далее они называются *стандартными* и являются частью нашего более общего набора операций. В [3] приведен алгоритм вычисления числа операций в *минимальной* по числу операций последовательности, которая преобразует одну структуру в другую, если структуры состоят только из цепей (“линейных хромосом”); время работы алгоритма близко к линейному, явная оценка времени отсутствует. Эти операции, применяемые только к цепям, соответствуют ранее рассмотренным в [4] операциям инверсии, транслокации, склейки и разреза. В случае, если и все промежуточные структуры состоят из линейных хромосом, алгоритм вычисления минимального числа операций приведен также в [4]. Общий случай постоянного генного состава и *одинаковых цен операций* решен в [5] с использованием идей из [3, 4].

В случае *переменного генного состава* нельзя обойтись без *дополнительных* операций: удаления и вставки участка особых генов, которые были предложены в [6, 7].

В работах [6–8] используется граф смежности: его вершины – отождествления (склейки) краев генов, которые принадлежат обеим структурам, и еще края исходных цепей. Две склейки из разных структур соединяются ребром, если в них представлен один и тот же ген. Кроме того, край исходной цепи считается склеенным с “пустым краем”; между склеенными краями *общих* генов двух структур может располагаться участок генов, принадлежащих только одной структуре (*особые* гены),

[1] Исследование выполнено в ИППИ РАН за счет гранта Российского научного фонда (проект №14-50-00150).

тогда они приписываются этой вершине. Такой граф, очевидно, отличается от определяемого ниже графа. В этих работах предложен линейный по времени алгоритм, строящий минимальную последовательность перестроек одной структуры в другую теми же операциями, что и в настоящей статье. В [6,7] цены всех операций равны 1; в [8] цены стандартных операций равны 1, а цены операций вставки и удаления одинаковы и не превосходят 1. Неясно, можно ли связать алгоритм из этих работ с нашим алгоритмом. Авторам неизвестно, где приведено доказательство точности алгоритма из [6–8]; замечания, которые там содержатся, не позволяют его получить.

В работах [9–11] предлагается линейный алгоритм, основанный на пополнении обеих исходных структур особыми генами. Таким образом, задача сводится к случаю равного генного состава, а суммарное число генов увеличивается на $k + t$, где k и t – количества особых генов в исходных структурах. Используется граф, который дополнительно включает по два края от каждого особого гена, что приводит к увеличению графа, с которым работает их алгоритм, по сравнению с графом, предложенным в [3, 12]. Граф и алгоритм, предложенные в [2, 12], другие, чем в [9–11]. В [11] описывается обобщение алгоритма из [9, 10] на случай, в котором все хромосомы циклические, цены стандартных операций равны 1, цены операций вставки и удаления равны между собой. Доказательство точности в [11] содержит, по нашему мнению, некоторые пробелы.

Отметим работы [13, 14]. В работе [13] рассматривается обобщение двойной переклейки на случай произвольного числа разрывов склеек с переклейкой их краев; другие операции не рассматриваются. Описываются алгоритмы вычисления минимальной последовательности, если структуры имеют постоянный набор генов и только циклические хромосомы. Один из алгоритмов имеет время $O(n^{0,5k-2})+O(n)$, где n – размер исходных структур, k – максимальное число разрывов, другой – линейное от n время при условии предобработки, занимающей экспоненциальное от k время. Используется граф, который в этом специальном случае совпадает с нашим графом. В работе [14] рассматривается задача вычисления длины минимальной последовательности переклеек, переводящей одну структуру в другую, если они имеют постоянный набор генов и только линейные хромосомы. Здесь алгоритм не предлагается, но приводятся явные нижние оценки на длину, как и результаты, показывающие, что они достаточно точные. Здесь задача решается сведением к случаю структур лишь с циклическими хромосомами; для этого добавляются склейки краев генов. Авторы не знали о работах [13,14] и, к сожалению, не рассматривали такие переклейки.

В данной статье для ориентированного графа, который состоит из любого множества цепей и циклов (“хромосом”), и всех упомянутых операций кроме обобщенных переклеек из [13,14], впервые, насколько известно авторам, приводится полное доказательство точности предлагаемого нами линейного по времени и памяти алгоритма в случае переменного состава генов и одинаковых цен операций. Случай разных цен операций рассмотрен нами в [2].

§ 2. Задача и основные понятия

2.1. Постановка задачи. Приведем все необходимые определения. *Структура* – конечный набор ориентированных графов (“компонент”), каждая компонента – линейная цепь или цикл, включая петлю. Ребро называют *геном*; компоненту – *хромосомой*. Гену приписан *номер* – натуральное число, которое, вообще говоря, может повторяться. В этой статье повторы не допускаются, случай повторов (“паралогов” генов) рассмотрен в [2, 15].

Компонент обычно много. Всю структуру можно рассматривать как ориентированный граф специального вида (иногда его называют CC-графом).

Пусть даны две структуры a и b и фиксирован набор операций, которые преобразуют одну структуру в другую. Каждой операции приписано положительное рациональное число, ее *цена* (иногда называемая *весом*). В этой статье все цены предполагаются единичными (или, что то же самое, одинаковыми).

Задача состоит в поиске *кратчайшей* последовательности операций, которая переводит структуру a в структуру b. "Кратчайшая", естественно, означает последовательность, у которой суммарная цена всех операций минимальная. Эту суммарную цену назовем *кратчайшей ценой*. В рассматриваемом здесь случае говорят о *минимальной* последовательности и кратчайшую цену называют *минимальной длиной*. *Приведение* a к b означает использование кратчайшей последовательности операций, преобразующих a в b; саму последовательность называют *приводящей*. Если $a = b$, то формально приведение выполняется "пустой последовательностью операций", которой приписывается цена и длина 0.

Обычно набор операций содержит обратную к каждой из операций, поэтому неважно, приводить a к b или b к a. Цены прямой и обратной операций могут различаться.

Эта задача в частном случае, когда структуры a и b имеют одинаковый набор генов и цены всех операций равны 1, решена в [5] более сложным алгоритмом, чем в [12].

Ниже на строго математическом уровне поставленная задача решена при переменном генном составе. Некоторые технические детали вычислений, использованных в доказательствах теорем 1, 2, 5, приведены в Приложении на странице `http://lab6.iitp.ru/ru/pr_chromo/`. Полученный результат докладывался на конференциях [15–18], включая описание алгоритмов и доказательств, а также описан в [2].

Упомянем результаты, которые имеют место в случае неравных цен. Здесь приходится накладывать специфические условия на цены, так как задача предполагается NP-трудной, и поэтому в общем виде не может быть решена не только линейным, но и любым полиномиальным алгоритмом. На упомянутых конференциях докладывались линейные по времени и памяти алгоритмы ее решения для неравных цен, см. также [2]. Первые два варианта цен таковы: $c_2 \leqslant c_1 \leqslant c'_1 \leqslant c_{1,5}$ (*циклический*) и $c_1 \leqslant c'_1 \leqslant c_{1,5} \leqslant c_2$ (*линейный*), в которых указаны соотношения между ценами операций разреза c_1, склейки c'_1, полуторной переклейки $c_{1,5}$, двойной переклейки c_2, при этом генный состав постоянный. Третий вариант – цена операции вставки равна $1 + \varepsilon$, где $0 \leqslant \varepsilon \leqslant 1$, цены других операций равны 1, при этом генный состав переменный. Для третьего варианта соответствующий алгоритм строит последовательность, суммарная цена которой отличается от минимально возможной не более чем на ε.

2.2. Понятия общего графа. Наш оригинальный подход основан на предлагаемом нами понятии общего графа; фактически используется и определение его качества [12].

Края генов обозначим через i_j, где i_1 – начало гена с номером i, а i_2 – конец гена с тем же номером; петля в структуре означает ген, у которого начало совпадает (отождествлено) с концом. Понятие общего графа сначала определим в частном случае, когда структуры a и b имеют одинаковые наборы генов, т.е. их ребрам приписаны одинаковые наборы натуральных чисел без повторений, например, все числа от 1 до n. *Общим графом* структур a и b назовем (уже неориентированный) граф $a+b$, у которого вершинами являются края i_j всех генов из a (или из b, наборы генов и краев одинаковы), а ребра соединяют края, отождествляемые в a или в b; каждое ребро помечено именем структуры, в которой произошло отождествление, т.е. именами a или b. В графе $a + b$ некоторые ребра могут быть параллельными: одно ребро из a, другое – из b, это – циклы длины 2, которые будем называть 2-*циклами*.

Легко видеть, что компоненты общего графа $a + b$ – чередующиеся (a, b)-цепи и (a, b)-циклы. Как мы увидим в дальнейшем, алгоритм нахождения минимальной длины не использует пометки i_j, приписанные вершинам.

Теперь рассмотрим нетривиальное определение $a + b$ для случая, когда a и b имеют разные наборы генов. Гены, принадлежащие обеим структурам a и b, назовем *общими*, остальные – *особыми*. Максимальные по включению участки в a или b, состоящие лишь из особых генов, назовем *блоками*; *именем* блока служит множество имен генов, входящих в него. Вершины графа $a + b$ – края i_j всех общих генов (*обычные* вершины) и еще *особые* вершины, соответствующие блокам и помеченные их именами. Особую вершину и ее имя не различаем. Кроме того, особую вершину *помечаем* именем a или b в зависимости от принадлежности блока.

Определим ребра в $a + b$. Обычные вершины соединяются, если они отождествлены в a или b (как и в случае одинаковых наборов генов). Если блок с именем A – циклическая компонента (включая петлю), то проводится петля в изолированную вершину A. Эту петлю назовем *особой*. Если блок A – линейная компонента, то A остается изолированной вершиной. Изолированные особые вершины считаем нечетными цепями *длины* -1, изолированные обычные вершины считаем четными цепями *длины* 0.

Если линейный блок A заключен между двумя краями общих генов (т.е. блок – не вся цепь и не весь цикл), то эти края соединяются ребрами с особой вершиной A. Если линейный блок A соединяется в структуре лишь с одним краем общего гена, то этот край в общем графе соединяется ребром с особой вершиной A (*висячее* ребро). Ребро, инцидентное двум обычным вершинам, называется *обычным*; ребро, инцидентное особой вершине, называется *особым*. ▲

Назовем *финальным* (*финального вида*) общий граф с обычными ребрами, состоящий из 2-циклов и изолированных обычных вершин.

Операция выполняется над одной или двумя компонентами (хромосомами), принадлежащими структуре. Операция над парой структур $\langle a, b\rangle$ определяется как операция над одной из компонент, вторая остается без изменения. Операция над парой $\langle a, b\rangle$ естественным образом переносится на их общий граф, название операции сохраняется; важно, что соответствующая диаграмма коммутативна.

Рассматриваемая задача на языке общего графа формулируется следующим образом. Найти кратчайшую последовательность, которая приводит наперед заданный общий граф $a + b$ к финальному виду; при этом в качестве цены операции, применяемой к b, берется цена обратной к ней операции. Операции позволяют различать применение к a или b.

Общий граф $a + b$ удобно хранить в массиве M, индексы которого меняются от $-n$ до n; отрицательные индексы соответствуют началам одноименных генов, положительные – концам в структурах a и b. Значение $M[i]$ – пара индексов краев, с которыми край i склеен в структурах a или b (если не склеен, то значение 0). Такой способ хранения обеспечивает линейные время и память алгоритма построения графа $a + b$ по структурам a и b, а также быстрый просмотр его компонент и обратный переход от операции над $a + b$ к операции над a или b.

2.3. Исходные операции над структурой и общим графом. Над структурой рассматриваются следующие *операции*: расклейка двух склеенных краев генов и переклейка четырех краев по-другому (“двойная переклейка”); расклейка двух склеенных краев и склеивание одного края с каким-то *свободным* (т.е. несклеенным) краем (“полуторная переклейка”); расклейка двух склеенных краев с образованием двух свободных краев и обратная к ней операция склейки двух свободных краев (“разрез” и “склейка”). Эти операции, иногда называемые *стандартными*, предложены в [3] и более четко в [5]. В случае разного генного состава нельзя обойтись без

дополнительных операций: удаление или вставка участка особых генов. В удобной для нас форме они описаны в начале п. 3.1.

Соответственно, к общему графу $a + b$ применяются следующие аналогичные стандартным *операции*. Двойная переклейка: удаление двух одинаково помеченных ребер и соединение четырех их концов двумя новыми неинцидентными ребрами с теми же пометками; полуторная переклейка: удаление ребра и соединение (ребром с той же пометкой, например, a) одного из его концов с обычной вершиной, не инцидентной ребру с этой пометкой, или с особой вершиной степени не больше 1 с этой пометкой; склейка: добавление ребра (допустим, с пометкой a) между вершинами – обычной, не инцидентной ребру с пометкой a, или особой степени не больше 1 с пометкой a; разрез: удаление любого ребра. Если в результате какой-либо операции образуется ребро с особыми концами (оба относятся к a или оба к b), то оно заменяется особой вершиной с именем, равным объединению имен концов (число особых вершин уменьшается на 1).

§ 3. Приведение структур в случае переменного набора генов и равных цен операций

Напомним, что наборы генов в двух структурах a и b могут различаться, и кроме четырех стандартных операций могут использоваться дополнительные операции вставки и удаления. *Цены всех операций одинаковы.* Требуется найти минимальную ("приводящую") последовательность от a к b. Напомним (в несколько более общем виде): в произвольной структуре максимальный по включению участок, состоящий из особых генов, все из которых принадлежат только a или только b, называется *блоком*.

Удобно представлять себе, что задача состоит в поиске структуры c, к которой независимо приводятся a и b, суммарно минимальными последовательностями (тогда a приводится к b через c), что эквивалентно приведению $a+b$ к $c+c$, а последний граф уже имеет финальный вид.

Внешние края/гены – примыкают к блоку. Напомним, что обычное ребро соединяет обычные вершины, особое ребро соединяет крайнюю в цепи особую вершину с обычной вершиной ("висячее") или внутреннюю особую вершину в цепи или цикле с обычной.

Длиной цепи или цикла назовем число обычных ребер в ней (в нем) сложенное с половиной числа особых не висячих ребер. Нечетные (четные) цепи – цепи нечетной (четной) длины.

Общий граф назовем *особым* (*особого вида*), если в нем нет петель и все ребра, не входящие в 2-циклы, особые. 2-циклы могут содержать особые и обычные ребра, изолированные вершины могут быть особыми и обычными.

Приведение графа к особому виду – первый шаг нашего алгоритма, второй состоит в приведении особого графа к финальному виду. Суть в том, что особые ребра лучше удалять совместно, тем самым, экономя число операций, к чему мы стремимся; а обычные ребра нужно удалять по отдельности, в любом порядке.

Операции применяются друг за другом, начиная с данной структуры, поэтому каждая операция рассматривается вместе со структурой, к которой она применяется. Назовем операцию *обычной* (*особой*), если в результате ее выполнения число особых вершин не меняется (строго уменьшается). Из определений в п. 2.3 видно, что особая операция уменьшает это число ровно на 1. Действительно, в результате двойной и полуторной переклеек или склейки может возникнуть не больше одного ребра, соединяющего особые вершины, которые при этом сливаются в одну. Операция удаления также удаляет лишь одну особую вершину.

Идея алгоритма следующая: сначала применяем только обычные операции и удаляем обычные ребра, пока доля особых ребер не достигнет максимума; возникает граф особого вида; к нему применяем в основном особые операции, чтобы изгнать особые вершины; возникает граф финального вида.

3.1. Дополнительные операции удаления и вставки. Возможность обойтись без одной из них. *Подблоком* назовем связный участок блока (т.е. условие максимальности опущено). Назовем его *a-подблоком*, если он состоит из генов, которые принадлежат только a. Аналогично определим b-подблок.

Операция удаления – удаление a-подблока, которая допускается, если подблок:

а) находился строго внутри линейной или циклической хромосомы, два конца внешних генов склеиваются;

б) находился с края линейной хромосомы, конец внешнего гена становится не склеенным;

в) являлся отдельной хромосомой.

Операция вставки – вставка b-подблока, которая допускается, если подблок присоединяется:

а′) строго внутри линейной или циклической хромосомы, место вставки расклеивается;

б′) с края линейной хромосомы, если подблок – линейная цепь;

в′) как отдельная хромосома.

Жесткими называются удаления и вставки, которые определяются в пунктах *а*, *полужесткими* – в пунктах *б*, *свободными* – в пунктах *в*.

a-блок состоит из генов структуры a. Операция вставки *разбивает* a-блок, если он линейный, и подблок вставляется между двумя его генами. Операции двойной или полуторной переклейки, или разреза *разбивают* a-блок, если они включают одну или две расклейки внутри линейного a-блока, или две расклейки внутри циклического a-блока. Операция удаления *разбивает* a-блок, если удаляется лишь его часть.

Теорема 1. *Среди минимальных последовательностей, приводящих a к b, существует такая, у которой*:

1) *никакая операция не разбивает a-блок,*

2) *все удаления происходят раньше всех вставок.*

Доказательство. Некоторые, по сути, очевидные выкладки вынесены в Приложение на странице `http://lab6.iitp.ru/ru/pr_chromo/`.

Для краткости a-блоки будем называть *блоками*, так как другие блоки не рассматриваются. *Обозначим* через $r(a, b)$ минимальную длину для структур a и b.

Лемма 1. *Если структура d' получается из структуры d удалением из блока одного гена g, то $r(d, b) \geqslant r(d', b)$.*

Доказательство. Индукция по величине $r(d, b)$. Если $r(d, b)$ равно 1, то b получается из d одной операцией удаления и $r(d', b) = r(d, b)$. В противном случае рассмотрим первую операцию в минимальной последовательности от d к b, обозначим ее через o. Ей соответствует такая операция o' над структурой d' (возможно, пустая), что структура $o'(d')$ совпадает со структурой $o(d)$ или получается из нее удалением гена g. По предположению индукции $r(o(d), b) \geqslant r(o'(d'), b)$. Отсюда $r(d, b) = r(o(d), b) + 1 \geqslant r(o'(d'), b) + 1 \geqslant r(d', b)$. ▲

Лемма 2. *Если структура d' получается из структуры d добавлением в блок гена g, не принадлежащего b, то $r(d, b) = r(d', b)$.*

Доказательство. Индукция по величине $r(d, b)$. Если $r(d, b) = 1$, то структура b получается из d и d' операцией удаления и утверждение доказано. В противном случае рассмотрим первую операцию в минимальной последовательности от d к b,

обозначим ее через o. Ей соответствует такая операция o' над структурой d', что $o'(d')$ получается из $o(d)$ добавлением гена g в некоторый блок. По предположению индукции $r(o(d), b) = r(o'(d'), b)$. Отсюда $r(d, b) = r(o(d), b) + 1 = r(o'(d'), b) + 1 = = r(d', b)$. ▲

Операцию, не разбивающую блок, назовем *цельной*. Последовательность из цельных операций назовем *цельной*. *Сужением* блока назовем удаление генов из него, а расширением блока – добавление генов в него, не принадлежащих структуре b. Структуру d' назовем *упрощением* структуры d, если d' получается из d сужением или расширением блоков (т.е. в d' не может появиться блок там, где его не было в d).

Следствие 1. *Существует минимальная последовательность операций от a к b, в которой все операции удаления цельные.*

Доказательство. Рассмотрим в произвольной минимальной последовательности первую нецельную операцию удаления. Полученную после ее применения структуру обозначим через d. Заменим эту операцию на цельное удаление: удалим весь блок и получим структуру d'. По лемме 1 имеем $r(d, b) \geqslant r(d', b)$. Таким образом, можно удлинить начало минимальной последовательности, не содержащее нецельных удалений. ▲

Следствие 2. *Существует минимальная последовательность операций от a к b, у которой все операции удаления и вставки цельные.*

Доказательство. По следствию 1 существует минимальная последовательность S, в которой все операции удаления цельные. Рассмотрим в ней первую нецельную операцию вставки, разбивающую некоторый блок r. Полученную после ее применения структуру обозначим через d. Заменим место вставки: вставим тот же отрезок на любой край блока r. Полученная структура d' – упрощение структуры d, поскольку получается из нее удалением одного блока (любой части разбитого блока) и расширением другого блока (другой части разбитого блока). По леммам 1 и 2 имеем $r(d, b) \geqslant r(d', b)$, а по следствию 1 существует минимальная последовательность операций от d' к b, содержащая только цельные операции удаления. Таким образом, можно удлинить начало минимальной последовательности, не содержащее нецельных вставок. ▲

Все операции, кроме удаления и вставки, назовем *переклейками*.

Лемма 3. *Пусть к некоторой структуре d применяется переклейка o, разбивающая блок r. Существует цельная операция переклейки o' (возможно, пустая), для которой структура $o'(d)$ – упрощение структуры $o(d)$.*

Схема доказательства. Для каждой операции o указывается "заменяющая" ее операция o', которая не разбивает блок: разбивающая блок расклейка в o переносится на край блока. ▲

Подробное доказательство леммы 3 вынесено в Приложение на странице `http://lab6.iitp.ru/ru/pr_chromo/`.

Следствие 3. *Существует минимальная цельная последовательность операций от a к b.*

Доказательство. По следствию 2 существует минимальная последовательность S, в которой все удаления и вставки цельные. Рассмотрим в ней первую нецельную операцию переклейки o; структуру, полученную применением этой операции, обозначим через d. По лемме 3 (с учетом лемм 1, 2) существует такая цельная переклейка o', что $r(o'(d), b) \leqslant r(o(d), b)$. По следствию 2 существует минимальная последовательность операций, переводящая $o'(d)$ в b, в которой все удаления и вставки цельные. Преобразуем последовательность S, убирая из нее нецельные переклейки и сохраняя цельность удалений и вставок. Следствие 3, а вместе с ним и п. 1 теоремы 1 доказаны. ▲

Следующая лемма позволяет перемещать особые операции в начало минимальной последовательности.

Лемма 4. *Пусть d – структура, и пусть $f = o_2(o_1(d))$, где операция o_1 – обычная, а операция o_2 – особая, причем обе они цельные. Существуют цельные операции o_3 и o_4 (одна из них может быть пустой), такие что структура $o_4(o_3(d))$ – упрощение структуры f, причем операция o_4 – обычная.*

Схема доказательства. Рассмотрим любые операции o_1 и o_2. Поскольку o_1 – обычная операция, она сохраняет все блоки. Сначала можно склеить те два блока, которые склеивала o_2, а затем "доделать" то, что делала o_1. ▲

Подробное доказательство леммы 4 вынесено в Приложение на странице `http://lab6.iitp.ru/ru/pr_chromo/`.

Минимальную цельную последовательность операций, приводящих a к b, назовем *канонической*, если в ней все особые операции предшествуют всем обычным.

Следствие 4. *Существует каноническая последовательность операций, приводящая a к b.*

Доказательство. По следствию 3 существует минимальная цельная последовательность S, приводящая a к b. Если в ней обычная операция идет перед особой, то по лемме 4 поменяем их местами. После конечного числа таких замен последовательность S преобразуется к нужному виду. ▲

В канонической последовательности операций все удаления осуществляются до всех вставок, что доказывает п. 2 теоремы 1. ▲

Благодаря теореме 1 можно далее рассматривать только пять операций; по существу, они рассматриваются только над общим графом. Итак, над общим графом разрешены четыре исходные операции (п. 2.3) и операция удаления. Это удаление особой вершины (блока): если ее степень 2, то она удаляется и инцидентные ей ребра сливаются в одно ребро с той же пометкой; если эта вершина степени 1, то она удаляется вместе с инцидентным ей ребром; если эта вершина степени 0 или с петлей, то вершина и петля удаляются.

Следствие 5. *Задача поиска минимальной последовательности, приводящей a к b, сводится к задаче приведения общего графа $a + b$ к финальному виду $c + c$ исходными операциями и только одной дополнительной операцией удаления блока.*

На самом деле эти задачи эквивалентны, но нам достаточно сформулированной односторонней импликации.

Доказательство. Решением второй задачи является пара последовательностей (S_1, S_2), первая из которых преобразует структуру a в некоторую структуру c, а вторая – структуру b в ту же структуру c. Длиной решения называем сумму количества операций в S_1 и S_2. Соединение последовательности S_1 и последовательности, обращенной к S_2, дает последовательность той же длины, преобразующую a в b. Осталось доказать, что она минимальна. Для этого достаточно показать, что решение второй задачи имеет не большую длину, чем решение первой. По теореме 1 первая задача имеет решение, удовлетворяющее свойствам 1 и 2 этой теоремы. Возьмем в соответствующей последовательности S структуру c, полученную после всех удалений, но до всех вставок. Тогда все операции слева от нее не содержат вставок и не разбивают a-блоки. Рассмотрим часть последовательности S между c и b, записанную в обратном порядке. Это минимальная последовательность, преобразующая b в c. По теореме 1 существует такая последовательность той же длины, которая преобразует b в c, не содержит вставок и операций, разбивающих b-блоки. Вместе с начальной частью S она дает решение второй задачи той же длины, что и решение первой. ▲

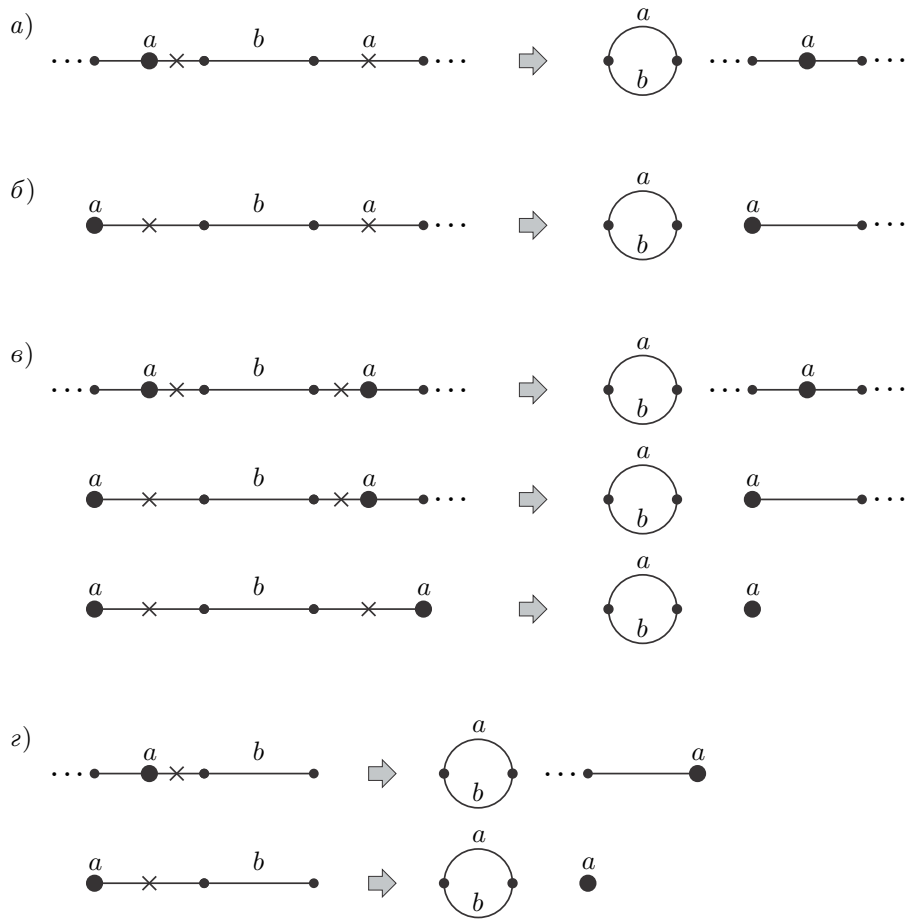

Рис. 1. К алгоритму приведения общего графа к особому виду: *а* – удаление пары обычных ребер (основной случай). Особые вершины обозначены большими полужирными кружочками; *б* – удаление пары обычных ребер (дополнительный случай); *в* – удаление обычного ребра (три случая). В третьем случае две особые вершины сливаются в одну изолированную особую вершину; *г* – удаление крайнего обычного ребра (два случая)

В общем графе *сегментом* назовем максимальный не входящий в 2-цикл отрезок из обычных ребер, возможно, цикл. Сегмент назовем *нечетным* (*четным*), если в нем нечетное (четное) число ребер. Сегмент назовем *крайним*, если в нем имеется ребро, инцидентное не более одному ребру графа. Отметим, что изолированная вершина сегментом не является, изолированное обычное ребро – крайний сегмент.

3.2. Алгоритм приведения общего графа к особому виду. Удаляются все особые петли. Для каждого нециклического сегмента s в графе $a + b$ выполняется: пока в s имеется ребро e, инцидентное двум не висячим ребрам графа, хотя бы одно из которых обычное, зацикливаем e двойной переклейкой, возникает цикл из двух обычных ребер. При этом два соседних ребра сливаются в одно (особое, если одно из них было особым, см. рис. 1,*а*). Эта операция обычная и соответствует вырезанию из s двух соседних обычных ребер.

В результате в s число ребер равно 1 или 2 (если s не исчезло). Если оно равно 2, то s образует цепь длины 2 без особых ребер (она зацикливается полуторной пере-

клейкой, см. [12, рис. 3,*б*]) или в s существует ребро, расположенное между другим ребром из s и висячим ребром. В последнем случае оно зацикливается двойной переклейкой в 2-цикл. При этом соседнее ребро становится висячим (см. рис 1,*б*). Эта операция обычная и соответствует вырезанию из s двух соседних обычных ребер, после чего s исчезает.

Пусть число ребер в s равно 1. Если сегмент s не крайний, то его единственное ребро расположено между двумя особыми ребрами, одно из них или оба могут быть висячими. Оно зацикливается особой двойной переклейкой в 2-цикл (см. рис 1,*в*).

Если сегмент s крайний, то его единственное ребро образует цепь длины 1 (оно зацикливается обычной склейкой (см. [12, правая часть рис. 3,*в*]) или имеет ровно одно соседнее особое ребро, возможно, висячее. Оно зацикливается обычной полуторной переклейкой в 2-цикл (см. рис. 1,*г*).

Каждый циклический сегмент приводится к финальному виду операциями двойной переклейки, выделяющими из него 2-циклы (см. [12, рис. 3,*г*]).

Очевидно, что в результате получается граф особого вида.

Длиной сегмента назовем число ребер в нем. Для каждого сегмента число обычных операций в алгоритме равно половине длины сегмента, если она четная; на 0,5 меньше, если она нечетная и сегмент не крайний; на 0,5 больше, если она нечетная и сегмент крайний. Для каждого циклического сегмента число обычных операций на 1 меньше половины его длины.

3.3. Алгоритм приведения общего графа к финальному виду. Дан граф $a + b$. Выполняются следующие шаги.

Шаг 1. Удаление всех особых петель (т.е. ни с чем не склеенных блоков). Преобразование обычных ребер, которые не принадлежат 2-циклам, в 2-циклы. После этого остаются только особые ребра, 2-циклы и изолированные вершины. Результат этого шага – граф $B(a+b)$, построенный в п. 3.2. Теперь наша задача – разбить все циклы длины строго большей 2 и цепи длины строго большей 0 на 2-циклы, и из них удалить все особые ребра. После этого получится граф финального вида.

Шаг 2. Каждому циклу и цепи в графе $B(a + b)$ приписывается его (ее) тип.

a-цепью называется нечетная цепь, у которой крайние невисячие ребра помечены a, или цепь, являющаяся изолированной вершиной, помеченной b. Аналогично определяется *b-цепь*.

Циклу приписываем тип "цикл". Для a-цепи ее тип определяется следующим образом: $1a$, если в ней одно висячее ребро; $2a$, если в ней два таких ребра или это изолированная особая вершина, помеченная именем b; $3a$, если у нее нет висячих ребер. Для b-цепей определение аналогично. Для четной цепи ее тип определяется следующим образом: 1, если в ней одно висячее ребро; 2, если в ней два таких ребра; 3, если в ней нет висячих ребер; 0, если она – изолированная обычная вершина. Для четных цепей нет нужды различать a- и b-цепи.

Шаг 3. Опишем совокупность множеств, в которой каждому множеству приписан вес, равный сумме весов всех его элементов. Каждый элемент – множество цепей в графе $B(a + b)$. Множество должно состоять из попарно непересекающихся элементов.

Вес элемента задает выигрыш в числе обычных операций в результате обработки элемента алгоритмом. Алгоритм использует любое *множество* M максимального веса из этой совокупности; оно будет максимальным и по включению. Осталось определить, какие типы цепей могут встретиться в элементе, и приписать ему вес.

Итак, допустимы элементы, которые составлены из цепей следующих типов, по одной цепи от каждого типа; вес элемента указан в круглых скобках: $\{1a, 1b\}$ (2), $\{2a, 3b\}$ (1), $\{2b, 3a\}$ (1), $\{2, 3\}$ (1), $\{1a, 2b, 3\}$ (2), $\{1b, 2a, 3\}$ (2), $\{1a, 3b, 2\}$ (2), $\{1b, 3a, 2\}$ (2), $\{1a, 2\}$ (1), $\{1b, 2\}$ (1), $\{1a, 3\}$ (1), $\{1b, 3\}$ (1), $\{1a, 1a, 2b, 3b\}$ (3),

$\{1b, 1b, 2a, 3a\}$ (3), $\{1a, 1a, 2b\}$ (2), $\{1b, 1b, 2a\}$ (2) $\{1a, 1a, 3b\}$ (2), $\{1b, 1b, 3a\}$ (2), $\{1a, 1a\}$ (1), $\{1b, 1b\}$ (1), $\{1a, 2b\}$ (1), $\{1b, 2a\}$ (1), $\{1a, 3b\}$ (1), $\{1b, 3a\}$ (1), $\{2a, 2b, 3, 3\}$ (2), $\{3a, 3b, 2, 2\}$ (2), $\{2, 2, 3a\}$ (1), $\{2, 2, 3b\}$ (1), $\{3, 3, 2a\}$ (1), $\{3, 3, 2b\}$ (1), $\{2a, 2b, 3\}$ (1), $\{3a, 3b, 2\}$ (1). Мощность элемента принимает значения от 2 до 4.

Шаг 4. Циклы и не вошедшие в M цепи приводятся к финальному виду по отдельности, независимо друг от друга. А именно, цепи длины -1 удаляются. В 2-циклах и в цепях длины 0 удаляются особые ребра. Циклы длины строго больше 2 разбиваются на 2-циклы, из которых удаляются особые ребра. Цепи длины строго больше 0 замыкаются в циклы, которые приводятся к финальному виду, как описано выше. При этом нечетные цепи замыкаются в цикл склейкой их концов. Четные цепи замыкаются в цикл полуторной переклейкой с отрезанием крайней обычной вершины (возможно, с висячим ребром, которое затем удаляется) и склейкой образовавшегося края с противоположным краем цепи.

Шаг 5. Для каждого элемента e из M определим граф $G(e)$, состоящий из цепей, входящих в e, и приведем его к финальному виду следующим образом. Ниже рассматривается один, первый из возможных случаев, второй случай аналогичен.

Особой называем цепь, состоящую целиком из особых ребер. Шаг 4 применяем для результатов следующих пунктов. Полезно заметить, что во всех пунктах операция является особой.

5.1. Элемент типа $\{1a, 1b\}$. Полуторная переклейка. В $1a$-цепи делаем внешнюю расклейку (т.е. расклейку крайнего не висячего ребра), и особую вершину склеиваем с концом висячего ребра $1b$-цепи. В результате получается цепь типа 0 и особая четная цепь типа 1 (применяя к ней шаг 4, приводим ее к финальному виду без взаимодействий с другими цепями).

5.2. Элемент типа $\{1a, 1a\}$ или $\{1b, 1b\}$. Склеиваем концы висячих ребер двух $1a$-цепей. Получаем особую нечетную $3a$-цепь.

5.3. Элемент типа $\{1a, 2b\}$ или $\{1b, 2a\}$. В $1a$-цепи делаем внешнюю расклейку и особую вершину склеиваем с концом висячего ребра $2b$-цепи или с изолированной особой вершиной (если $2b$-цепь ею является) – полуторная переклейка. Получаем цепь типа 0 и особую четную цепь типа 2. Здесь и вплоть до п. 5.8 используем полуторную переклейку.

5.4. Элемент типа $\{1a, 3b\}$ или $\{1b, 3a\}$. В $3b$-цепи делаем внешнюю расклейку и особую вершину склеиваем с концом висячего ребра $1a$-цепи. Получаем цепь типа 0 и особую четную цепь типа 3.

5.5. Элемент типа $\{1a, 2\}$ или $\{1b, 2\}$. В $1a$-цепи делаем внешнюю расклейку и особую вершину склеиваем с концом висячего ребра 2-цепи. Получаем цепь типа 0 и особую нечетную цепь типа $2a$.

5.6. Элемент типа $\{1a, 3\}$ или $\{1b, 3\}$. В 3-цепи делаем расклейку крайнего b-ребра и особую вершину склеиваем с концом висячего ребра $1a$-цепи. Получаем цепь типа 0 и особую нечетную цепь типа $3a$.

5.7. Элемент типа $\{2a, 3b\}$ или $\{2b, 3a\}$. В $3b$-цепи делаем внешнюю расклейку и особую вершину склеиваем с концом висячего ребра $2a$-цепи или с изолированной особой вершиной (если $2a$-цепь ею является). Получаем цепь типа 0 и особую четную цепь типа 1.

5.8. Элемент типа $\{2, 3\}$. В 3-цепи делаем внешнюю расклейку и особую вершину склеиваем с концом висячего ребра 2-цепи – полуторная переклейка. Получаем цепь типа 0 и особую четную цепь типа 1.

5.9. Элемент типа $\{1a, 1a, 2b\}$ или $\{1b, 1b, 2a\}$. С двумя $1a$-цепями выполняем шаг 5.2. Получаем особую нечетную цепь типа $3a$. С ней и исходной $2b$-цепью выполняем п. 5.7.

5.10. Элемент типа $\{1a, 1a, 3b\}$ или $\{1b, 1b, 3a\}$. С цепями $1a$ и $3b$ выполняем п. 5.4. Получаем особую четную цепь типа 3 и цепь типа 0. С первой из них и второй исходной $1a$-цепью выполняем п. 5.6.

5.11. Элемент типа $\{1a, 2b, 3\}$ или $\{1b, 2a, 3\}$. С цепями $1a$ и $2b$ выполняем 5.3. Получаем особую четную цепь типа 2 и цепь типа 0. С первой из них и исходной 3-цепью выполняем п. 5.8.

5.12. Элемент типа $\{1a, 3b, 2\}$ или $\{1b, 3a, 2\}$. С цепями $1a$ и $3b$ выполняем 5.4. Получаем особую четную цепь типа 3 и цепь типа 0. С первой из них и исходной 2-цепью выполняем п. 5.8.

5.13. Элемент типа $\{2a, 2b, 3\}$. В 3-цепи делаем расклейку крайнего b-ребра и особую вершину склеиваем с концом висячего ребра $2a$-цепи или с изолированной особой вершиной (если $2a$-цепь ею является) – полуторная переклейка. Получаем особую нечетную цепь типа $1a$ и цепь типа 0. Затем с парой цепей $\{1a, 2b\}$ выполняем п. 5.3.

5.14. Элемент типа $\{3a, 3b, 2\}$. В $3a$-цепи делаем внешнюю расклейку и особую вершину склеиваем с концом висячего ребра 2-цепи – полуторная переклейка. Получаем особую нечетную цепь типа $1a$ и цепь типа 0. Затем с парой цепей $\{1a, 3b\}$ выполняем п. 5.4.

5.15. Элемент типа $\{2a, 3, 3\}$ или $\{2b, 3, 3\}$. С парой цепей $\{2a, 3\}$ выполняем первое действие из 5.13. Получаем особую нечетную цепь типа $1a$ и цепь типа 0. С парой цепей $\{1a, 3\}$ выполняем п. 5.6.

5.16. Элемент типа $\{3a, 2, 2\}$ или $\{3b, 2, 2\}$. С парой цепей $\{3a, 2\}$ выполняем первое действие из 5.14. Получаем особую нечетную цепь типа $1a$ и цепь типа 0. С парой цепей $\{1a, 2\}$ выполняем п. 5.5.

5.17. Элемент типа $\{2a, 2b, 3, 3\}$. С тройкой цепей $\{2a, 2b, 3\}$ выполняем 5.13. Получаем особую четную цепь типа 2 и цепь типа 0. С полученной 2-цепью и второй исходной 3-цепью выполняем п. 5.8.

5.18. Элемент типа $\{3a, 3b, 2, 2\}$. С тройкой цепей $\{3a, 3b, 2\}$ выполняем 5.14. Получаем особую четную цепь типа 3 и цепь типа 0. С полученной 3-цепью и второй исходной 2-цепью выполняем п. 5.8.

5.19. Элемент типа $\{1a, 1a, 2b, 3b\}$ или $\{1b, 1b, 2a, 3a\}$. С парой цепей $\{1a, 2b\}$ выполняем 5.3. Получаем особую четную цепь типа 2 и цепь типа 0. С другой парой исходных цепей $\{1a, 3b\}$ выполняем 5.4. Получаем особую четную цепь типа 3 и цепь типа 0. С полученными 2-цепью и 3-цепью выполняем п. 5.8.

3.4. Точность и линейная сложность алгоритма приведения общего графа к финальному виду.

1) Из описания алгоритма видно, что он приводит граф $G = a + b$ к финальному виду. Оценим время работы алгоритма. Пусть число вершин в графе G равно n. Тогда удаление особых петель требует не более n операций; вырезание обычных ребер – не более n операций. При построении множества M добавляется по одному не более n наборов цепей мощности от 2 до 4; при финализации компонент не из M для каждой компоненты размера m требуется не более одной операции замыкания цепи в цикл, не более m операций вырезания из цикла 2-циклов, не более m операций удаления блоков; при финализации наборов цепей для каждого набора размера m требуется не более трех операций межкомпонентных взаимодействий, не более одной операции замыкания цепи в цикл, не более m операций вырезания из цикла 2-циклов, не более m операций удаления блоков. Таким образом, время алгоритма линейно зависит от n.

По определению любой последовательности, приводящей граф $a + b$ к финальному виду, число особых операций равно B, где B – число особых вершин (блоков) в графе G.

2) *Дефектом* компоненты графа $B(a+b)$ назовем

- 0, если это цикл или цепь типа $2a$, $2b$, 0, 1, 2;
- 1, если это цепь типа $1a$, $1b$, $3a$, $3b$, 3.

Интуитивно дефект – это число обычных операций в минимальной последовательности для этой компоненты.

Пусть D – сумма дефектов всех компонент графа $B(a+b)$, P – вес множества M, S – сумма целых частей половин длин сегментов в графе $a+b$ плюс число нечетных крайних сегментов минус число циклических сегментов. В следующем пункте указан алгоритм построения множества M.

Число обычных операций в последовательности, которую строит алгоритм, равно $S+D-P$ (лемма 5).

3) Обозначим $t(G)=B+S+D-P$. Любая операция меняет величину $t(G)$ не более чем на 1 (теорема 5).

4) Последовательность операций длины строго меньшей $t(G)$ не приводит граф G к финальному виду (теорема 6).

Из пп. 2)–4) следует точность алгоритма, т.е. он выдает минимальную последовательность. Указанные в них утверждения доказаны в п. 3.6.

3.5. Алгоритм построения множества элементов максимального веса. В общем случае нахождение в данной совокупности множеств максимального по мощности множества – NP-трудная задача. В нашем случае это преодолевается за счет специфического порядка в нашей совокупности. В результате получается алгоритм линейной сложности.

Четными назовем типы – 1, 2, 3. Нечетными – типы $1a$, $1b$, $2a$, $2b$, $3a$, $3b$. Типы 0 и "цикл" не входят в элементы. В доказательстве существенно используются следующие две биекции на множестве всех типов цепей. Первая из них переводит члены пар $(1a,1b)$, $(2a,2b)$, $(3a,3b)$ друг в друга, оставляя на месте четные типы. Назовем ее *горизонтальным автоморфизмом*. Вторая биекция переводит члены пар $(2a,3a)$, $(2b,3b)$, $(2,3)$ друг в друга, оставляя на месте другие типы. Назовем ее *вертикальным автоморфизмом*. Обе биекции "сохраняют допустимость типа" элемента в том смысле, что после автоморфизма допустимый тип переходит в допустимый.

Множество M элементов максимального веса строится последовательным добавлением в него элементов. Сначала множество M пустое. Если T – тип элемента, то назовем *приращением* M по T добавление к M максимального количества попарно дизъюнктных между собой и с M элементов типа T; распределение цепей по этим элементам может быть произвольным. Это количество равно минимальной (по типам t из T) мощности множества цепей типа t, не содержащихся в текущем M. Цепь, не содержащуюся в текущем M, назовем *свободной*.

Алгоритм состоит из следующих последовательных приращений.

По типу $\{1a,1b\}$, после чего не останется либо свободных цепей типа $1a$, либо свободных цепей типа $1b$. Рассмотрим первый случай, иначе нужно во всех последующих шагах заменить a на b.

Далее, по типам $\{2a,3b\}$ и $\{2b,3a\}$; по типу $\{2,3\}$; по типам $\{1a,2b,3\}$ и $\{1a,3b,2\}$; по типам $\{1a,2\}$ и $\{1a,3\}$; по типу $\{1a,1a,2b,3b\}$; по типам $\{1a,1a,2b\}$ и $\{1a,1a,3b\}$; по типу $\{1a,1a\}$; по типам $\{1a,2b\}$ и $\{1a,3b\}$; по типам $\{2a,2b,3,3\}$ и $\{3a,3b,2,2\}$; по типам $\{2,2,3a\}$ и $\{3,3,2a\}$; по типам $\{2,2,3b\}$ и $\{3,3,2b\}$; по типам $\{2a,2b,3\}$ и $\{3a,3b,2\}$.

Алгоритм не требует перечисления всех допустимых элементов, достаточно группировать цепи в $B(a+b)$ по типам.

Теорема 2. *Алгоритм строит множество M максимального веса.*

Схема доказательства. Будем считать, что алгоритм строит множество M, добавляя в него элементы по одному, в соответствии с описанным порядком. Достаточно показать, что на каждом шаге текущее множество M является подмножеством некоторого множества максимального веса (назовем его максимальным множеством). Рассуждаем индукцией по построению M.

Пусть на очередном шаге в M добавляется элемент e. Пусть имеется некоторое максимальное множество M' и элемент e не входит в него, а все элементы из M входят. Покажем, что можно так перестроить множество M', чтобы оно осталось максимальным и содержало элемент e. Рассмотрим множество K элементов из M', содержащих хотя бы одну цепь из e. Множество цепей, содержащихся в элементах из K, обозначим через $I(K)$, а в элементе e – через $I(e)$. Перебирая возможные сочетания типов элементов из K, убеждаемся, что во всех случаях из цепей множества $I(K) \setminus I(e)$ можно составить множество K' попарно непересекающихся элементов суммарного веса не меньше $w(K) - w(e)$, где $w(K)$ – суммарный вес элементов из K, $w(e)$ – вес элемента e. Заменяя в M' множество K на объединение K' и e, получаем требуемое максимальное множество. Отметим, что определенный порядок добавления элементов в M существенен: перебирая возможные типы множества K, мы используем, что из цепей в объединении $I(K)$ и $I(e)$ нельзя составить никакой элемент типа, который был добавлен в M ранее (в частности, в K нет элементов, имеющих такие типы). ▲

Технические детали доказательства теоремы 2 вынесены в Приложение на странице `http://lab6.iitp.ru/ru/pr_chromo/`.

3.6. Формулировки вспомогательных утверждений из п. 3.4 и их доказательства.

Лемма 5. *Алгоритм приведения графа G к финальному виду строит последовательность из $t(G)$ операций.*

Доказательство. Очевидно, что количество особых операций в алгоритме равно B, осталось подсчитать количество обычных. Из описания приведения общего графа $a + b$ к особому виду видно, что количество обычных операций в нем равно S. Если приводить к финальному виду все компоненты графа $B(a + b)$ по отдельности, потребуется ровно D обычных операций (следует из описания в п. 3.3, шаг 4, см. также теорему 3). Использование "сокращений" с использованием множества максимального веса снижает эту оценку на P (следует из описания в п. 3.3, шаг 5, см. также теорему 4). ▲

Теорема 3. *Если особый общий граф имеет одну компоненту C, то $D(C) \geqslant p$, где p – число обычных операций в минимальной последовательности.*

Доказательство. Неравенство $D(C) \leqslant p$ также верно (см. следствие 6 ниже), но не используется. Рассмотрим все типы графа C. Если это цикл, рассуждаем индукцией по его длине. Для длины 2 обычных операций не требуется. Для произвольной длины возьмем два (особых) одинаково помеченных ребра, разделенных ровно одним ребром. Двойной переклейкой с удалением этих ребер, разобьем цикл на два цикла так, что один цикл имеет длину 2, а другой – особый (см. рис 2,*а*).

Эта операция особая, сливает два блока в один. По предположению индукции особый цикл меньшей длины можно привести к финальному виду последовательностью особых операций. Тогда и цикл C приводится к финальному виду такой последовательностью, что и требовалось. Рассмотрим случаи линейных компонент.

Случай нечетных цепей (см. рис 2,*б*).

Тип 1. Склеивая конец висячего ребра с противоположным краем цепи, получим особый цикл (обычная операция) – верхняя стрелка на рис. 2,*б*. По доказанному ранее, особый цикл можно привести к финальному виду последовательностью особых

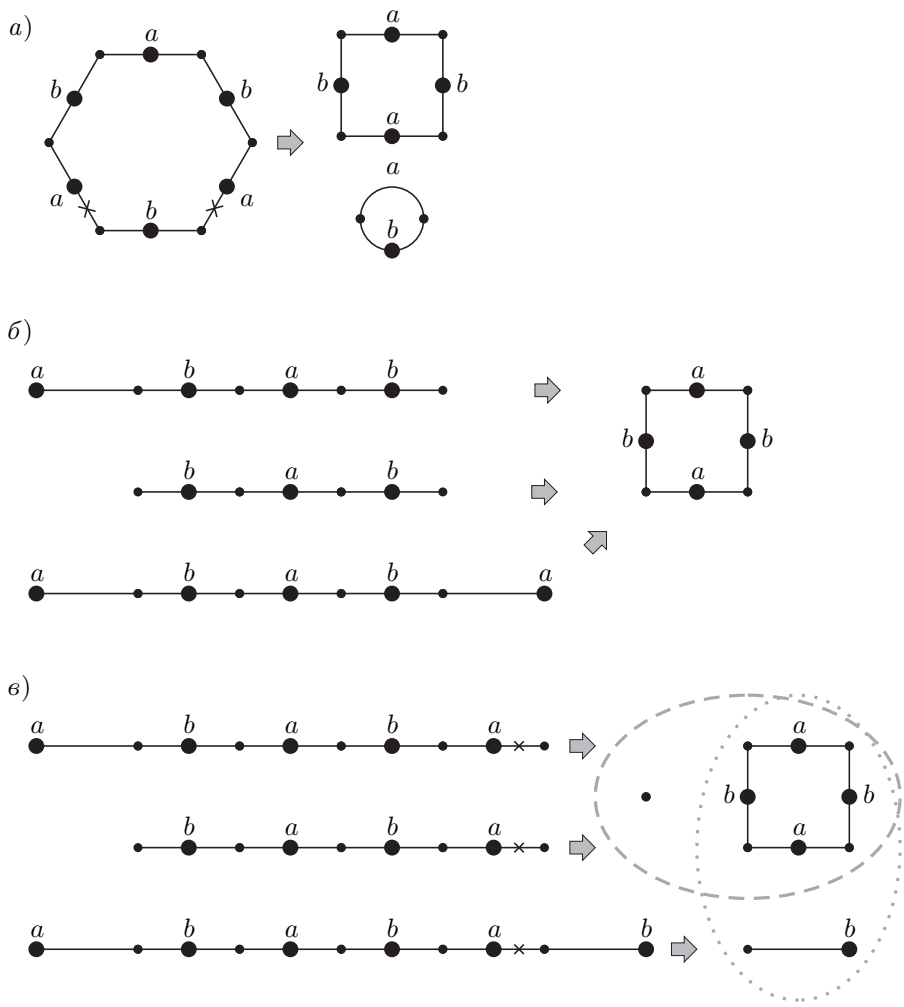

Рис. 2. К доказательству теоремы 3: *а* – особая операция разбиения цикла на два цикла; *б* – зацикливание нечетных цепей (верхнее и среднее – обычные операции, нижнее – особая); *в* – зацикливание четных цепей (верхнее и нижнее – особые операции, среднее – обычная)

операций. Тогда граф C приводится к финальному виду последовательностью из особых и одной обычной операции, что и требовалось.

Тип 2. Если C – цепь длины -1, то обычных операций не требуется. В противном случае, склеив концы висячих ребер, получим особый цикл (особая операция) – нижняя стрелка на рис. 2,*б*. Дальнейшее рассуждение аналогично предыдущему.

Тип 3. Склеивая края цепи ребром, получим цикл с одним обычным ребром (назовем такой цикл *полуособым*) – средняя стрелка на рис. 2,*б*, верхний блок цикла считаем отсутствующим. Это обычная операция. Если цикл имеет длину 2, доказывать нечего. В противном случае, удаляя из цикла обычное ребро (как в алгоритме приведения графа к особому виду), получаем, что полуособый цикл можно привести к финальному виду последовательностью особых операций. Тогда граф C приводится к финальному виду последовательностью из особых и одной обычной операции, что и требовалось.

Случай четных цепей (см. рис 2,*в*).

Тип 0. Утверждение очевидно.

Тип 1. Если длина цепи равна 0, утверждение очевидно. В противном случае сделаем особую полуторную переклейку, расклеив крайнее невисячее ребро (такую расклейку назовем *внешней*) и склеив конец висячего ребра с крайней особой вершиной – верхняя стрелка на рис. 2,*в*. Получим особый цикл и цепь типа 0. Таким образом, граф C приводится к финальному виду последовательностью особых операций, что и требовалось.

Тип 2. Если длина цепи равна 0, утверждение очевидно. В противном случае сделаем особую полуторную переклейку с внешней расклейкой крайнего невисячего ребра и склеиванием крайней особой вершины с концом противоположного висячего ребра – нижняя стрелка на рис. 2,*в*. В результате образуется особый цикл и цепь длины 0 с одним висячим ребром. Дальнейшее рассуждение очевидно.

Тип 3. Сделаем полуторную переклейку с внешней расклейкой и склеиванием крайней особой вершины с противоположным краем цепи – средняя стрелка на рис. 2,*в*. Это обычная операция, в результате которой образуется особый цикл и цепь типа 0. ▲

Для любого элемента e обозначим через $G(e)$ граф, состоящий из его цепей, $D(e)$ – сумма дефектов всех цепей из e, $c(e)$ – вес элемента e.

Теорема 4. *Число p обычных операций в минимальной последовательности для $G(e)$ не превышает $D(e) - c(e)$.*

Доказательство. Легко проверить, что в каждом из описанных в п. 3.3 случаев 5.1–5.19 для слияния цепей из элемента e в одну цепь применяются лишь особые операции, и дефект цепи, получившейся в результате, равен $D(e) - c(e)$. Эта цепь по теореме 3 приводится к финальному виду с применением не более $D(e) - c(e)$ обычных операций. Таким образом, приведение к финальному виду графа $G(e)$ требует не более $D(e) - c(e)$ обычных операций. Для облегчения проверки представим результат каждого пункта в виде равенств: слева сумма исходных типов цепей, справа – тип итоговой цепи, в квадратных скобках – разность суммы дефектов цепей слева и дефекта цепи справа, равная $c(e)$. Тогда

5.1. $1a + 1b = 1$ [2];
5.2. $1a + 1a = 3a$ [1];
5.3. $1a + 2b = 2$ [1];
5.4. $1a + 3b = 3$ [1];
5.5. $1a + 2 = 2a$ [1];
5.6. $1a + 3 = 3a$ [1];
5.7. $2a + 3b = 1$ [1];
5.8. $2 + 3 = 1$ [1];
5.9. $1a + 1a + 2b = 3a + 2b = 1$ [2];
5.10. $1a + 1a + 3b = 1a + 3 = 3a$ [2];
5.11. $1a + 2b + 3 = 2 + 3 = 1$ [2];
5.12. $1a + 3b + 2 = 3 + 2 = 1$ [2];
5.13. $2a + 2b + 3 = 2a + 1b = 2$ [2];
5.14. $3a + 3b + 2 = 3a + 1b = 3$ [1];
5.15. $2a + 3 + 3 = 1a + 3 = 3a$ [1];
5.16. $3a + 2 + 2 = 1a + 2 = 2a$ [1];
5.17. $2a + 2b + 3 + 3 = 1a + 1b = 1$ [2];
5.18. $3a + 3b + 2 + 2 = 1a + 1b = 1$ [2];
5.19. $1a + 1a + 2b + 3b = 2 + 3 = 1$ [3]. ▲

Неравенство $D(e) - c(e) \leqslant p$ также выполняется (см. следствие 6 ниже), и поэтому в теореме 4 выполняется равенство.

Теорема 5. *Для любой операции o и любого общего графа G выполняется неравенство $t(G) - 1 \leqslant t(o(G)) \leqslant t(G) + 1$.*

Схема доказательства. Назовем компоненту общего графа *стандартной*, если в ней нет особых ребер. Для связной нестандартной компоненты C графа $a + b$ обозначим через $B'(C)$ нефинальную компоненту графа $B(C)$ (очевидно, она единственная). Следующая лемма позволяет быстро определять по каждой компоненте C исходного графа $a + b$ тип соответствующей компоненты в графе $B(a + b)$. Таким образом, классификация компонент особого общего графа по типам распространяется на компоненты произвольного общего графа.

Лемма 6. *Имеют место следующие утверждения.*

1) *Граф $B(C)$ имеет финальный вид тогда и только тогда, когда компонента C стандартная.*
2) *$B'(C)$ – особый цикл тогда и только тогда, когда C является нестандартным циклом.*
3) *$B'(C)$ – цепь типа 1a (аналогично для 1b) тогда и только тогда, когда C является нестандартной нечетной a-цепью, у которой с одного края расположен крайний четный сегмент (либо сегмент отсутствует – тогда считаем его четным сегментом длины 0), а с другого края – либо висячее ребро, либо крайний нечетный сегмент.*
4) *$B'(C)$ – цепь типа 2a (аналогично для 2b) тогда и только тогда, когда C является нестандартной нечетной a-цепью, у которой с каждого края расположено либо висячее ребро, либо крайний нечетный сегмент (к этому типу относится также изолированная особая вершина с пометкой b).*
5) *$B'(C)$ – цепь типа 3a (аналогично для 3b) тогда и только тогда, когда C является нестандартной нечетной a-цепью, у которой с каждого края расположен крайний четный сегмент.*
6) *$B'(C)$ – цепь типа 1 тогда и только тогда, когда C является нестандартной четной цепью, у которой с одного края расположен крайний четный сегмент, а с другого края – либо висячее ребро, либо крайний нечетный сегмент.*
7) *$B'(C)$ – цепь типа 2 тогда и только тогда, когда C является нестандартной четной цепью, у которой с каждого края расположено либо висячее ребро, либо крайний нечетный сегмент.*
8) *$B'(C)$ – цепь типа 3 тогда и только тогда, когда C является нестандартной четной цепью, у которой с каждого края расположен крайний четный сегмент.*

Доказательство. Скажем, что компонента C имеет вид i, $i = 1, 2, \ldots, 8$, если она соответствует описанию в правой части доказываемой эквивалентности в пункте i формулировки леммы 6. Очевидно, каждая компонента имеет ровно один вид. Поэтому достаточно доказать все эквивалентности лишь справа налево. Для доказательства достаточно проверить, что каждая операция алгоритма приведения графа к особому виду не меняет вид компоненты, к которой она применяется, что очевидно. ▲

Дальнейшее доказательство теоремы 5 состоит, по сути, в переборе возможных типов операции o и ее аргументов. Для всех операций, кроме двойной переклейки, перебор осуществляется непосредственно. Например, пусть o – операция удаления изолированной особой вершины (напомним, она считается цепью типа 2a или 2b). Тогда при переходе от графа G к графу $o(G)$ величина B уменьшается на 1, величины S и D не меняются. Величина P либо не меняется, либо уменьшается на 1. Действительно, уменьшаться больше чем на 1 она не может, так как любой элемент веса 2 (соответственно, веса 3) после удаления из него цепи типа 2a или 2b превращается в элемент веса 1 (соответственно, веса 2). Таким образом, величина $t(G)$ либо не меняется, либо уменьшается на 1. Отсюда следует справедливость доказываемого утверждения.

Аналогично рассматривается случай двойной переклейки, когда она применяется к одной компоненте, либо к двум: цепи и циклу. В случае, когда двойная переклейка o применяется к двум цепям, непосредственный перебор вариантов затруднен в силу их многочисленности. Точнее, затруднение вызывает тот случай, когда с одной стороны от некоторой расклейки из o нет блоков. Тогда задача сводится к случаю полуторной переклейки или склейки, как описано в п. 5.3.2 доказательства теоремы 5 в Приложении на странице `http://lab6.iitp.ru/ru/pr_chromo/`, где это доказательство приведено подробно. ▲

Теорема 6. *Алгоритм приведения графа $a+b$ к финальному виду строит минимальную последовательность операций. Время алгоритма линейно от размера общего графа двух исходных структур.*

Доказательство. Для финального графа $t(G) = 0$. Уменьшить начальное значение $t(G)$ до 0 за меньше чем $t(G)$ операций можно лишь, если какая-то операция уменьшает значение $t(G)$ больше чем на 1. Это противоречит утверждению теоремы 5. Линейность времени алгоритма следует из его описания. ▲

Следствие 6. *В обозначениях теорем* 3, 4 *выполняются оценки $D(C) \leqslant p$ и $D(e) - c(e) \leqslant p$.*

Доказательство. Очевидно, что $t(C) = B + D(C)$, где B – количество блоков в C. Если компоненту C можно было бы привести к финальному виду с использованием менее чем $D(C)$ обычных операций, то всего было бы затрачено менее $t(C)$ операций. Это противоречит теореме 6 в сочетании с леммой 5. Доказательство второго неравенства аналогично с учетом очевидного равенства $t(G(e)) = B + D(e) - c(e)$, где B – количество блоков в графе $G(e)$. ▲

Авторы благодарят рецензента за ценные замечания.

СПИСОК ЛИТЕРАТУРЫ

1. *Blanchette M., Kunisawa T., Sankoff D.* Gene Order Breakpoint Evidence in Animal Mitochondrial Phylogeny // J. Mol. Evol. 1999. V. 49. № 2. P. 193–203.
2. *Lyubetsky V.A., Gershgorin R.A., Seliverstov A.V., Gorbunov K.Yu.* Algorithms for Reconstruction of Chromosomal Structures // BMC Bioinformatics. 2016. V. 17. Research Article 40.
3. *Yancopoulos S., Attie O., Friedberg R.* Efficient Sorting of Genomic Permutations by Translocation, Inversion and Block Interchange // Bioinformatics. 2005. V. 21. № 16. P. 3340–3346.
4. *Hannenhalli S., Pevzner P.A.* Transforming Men into Mice (Polynomial Algorithm for Genomic Distance Problem) // Proc. 36th Annual Sympos. on Foundations of Computer Science (FOCS'95). Milwaukee, WI, USA. October 23–25, 1995. P. 581–592.
5. *Bergeron A., Mixtacki J., Stoye J.* A Unifying View of Genome Rearrangements // Algorithms in Bioinformatics (Proc. 6th Int. Workshop, WABI'2006. Zurich, Switzerland. September 11–13, 2006). Lect. Notes Comp. Sci. V. 4175. Berlin: Springer, 2006. P. 163–173.
6. *Braga M.D.V., Willing E., Stoye J.* Genomic Distance with DCJ and Indels // Algorithms in Bioinformatics (Proc. 10th Int. Workshop, WABI'2010. Liverpool, UK. September 6–8, 2010). Lect. Notes Bioinformat. V. 6293. Berlin: Springer, 2010. P. 90–101.
7. *Braga M.D.V., Willing E., Stoye J.* Double Cut and Join with Insertions and Deletions // J. Comput. Biol. 2011. V. 18. № 9. P. 1167–1184.
8. *da Silva P.H., Braga M.D.V., Machado R., Dantas S.* DCJ-indel Distance with Distinct Operation Costs // Algorithms in Bioinformatics (Proc. 12th Int. Workshop, WABI'2012. Ljubljana, Slovenia. September 10–12, 2012). Lect. Notes Bioinformat. V. 7534. Berlin: Springer, 2012. P. 378–390.
9. *Compeau P.E.C.* A Simplified View of DCJ-Indel Distance // Algorithms in Bioinformatics (Proc. 12th Int. Workshop, WABI'2012. Ljubljana, Slovenia. September 10–12, 2012). Lect. Notes Bioinformat. V. 7534. Berlin: Springer, 2012. P. 365–377.

10. *Compeau P.E.C.* DCJ-Indel Sorting Revisited // Algorithms Mol. Biol. 2013. V. 8. Research Article 6.
11. *Compeau P.E.C.* A Generalized Cost Model for DCJ-Indel Sorting // Algorithms in Bioinformatics (Proc. 14th Int. Workshop, WABI'2014. Wrocław, Poland. September 8–10, 2014). Lect. Notes Bioinformat. V. 8701. Berlin: Springer, 2014. P. 38–51.
12. *Горбунов К.Ю, Гершгорин Р.А., Любецкий В.А.* Перестройка и реконструкция хромосомных структур // Молекулярная биология. 2015. Т. 49. № 3. С. 372–383.
13. *Alekseyev M.A., Pevzner P.A.* Multi-Break Rearrangements and Chromosomal Evolution // Theor. Comput. Sci. 2008. V. 395. № 2–3. P. 193–202.
14. *Alekseyev M.A.* Multi-Break Rearrangements and Breakpoint Re-uses: From Circular to Linear Genomes // J. Comput. Biol. 2008. V. 15. № 8. P. 1117–1131.
15. *Gershgorin R.A., Gorbunov K.Yu., Seliverstov A.V., Lyubetsky V.A.* Evolution of Chromosome Structures // Информационные технологии и системы 2015. 39-я междисциплинарная школа-конференция ИППИ РАН (ИТиС'2015). Олимпийская деревня, Сочи, Россия. 7–11 сентября, 2015. С. 105–120.
16. *Любецкий В.А., Горбунов К.Ю.* Задачи и алгоритмы, связанные с хромосомными перестройками // Современные информационные технологии и ИТ-образование. 2013. Т. 9. С. 764–768.
17. *Lyubetsky V.A., Gorbunov K.Yu.* Chromosome Structures Reconstruction // Molecular Phylogenetics: Contributions to the 4th Moscow Int. Conf. on Molecular Phylogenetics (MolPhy-4). Moscow, Russia. September 23–26, 2014. M.: Торус-Пресс, 2014. С. 42.
18. *Lyubetsky V.A., Seliverstov A.V., Gorbunov K.Yu.* Rearrangement of Chromosomes: Problems, Algorithms, Databases, and Gene Expression Regulations // The 9th Int. Conf. on Bioinformatics of Genome Regulation and Structure\Systems Biology (BGRS\SB'2014). Novosibirsk, Russia. June 23–28, 2014. Oral presentation.

Горбунов Константин Юрьевич
Любецкий Василий Александрович
Институт проблем передачи информации
им. А.А. Харкевича РАН
gorbunov@iitp.ru
lyubetsk@iitp.ru

Поступила в редакцию
29.12.2014
После переработки
25.04.2016