ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ АКАДЕМИЧЕСКИЙ УНИВЕРСИТЕТ РОССИЙСКОЙ АКАДЕМИИ НАУК

Диссертация допущена к защите

Зав. кафедрой

_____ А.В. Омельченко

«____» _____ 2015 г.

ДИССЕРТАЦИЯ
НА СОИСКАНИЕ УЧЕНОЙ СТЕПЕНИ
МАГИСТРА

Тема: **Разработка алгоритмов реконструкции и линеаризации предковых геномов с учетом вставок и удалений генов.**

Направление: 03.04.01 – Прикладные математика и физика

Выполнил студент П. В. Авдеев

(подпись)

Руководитель:

PhD, доцент М. А. Алексеев

(подпись)

Рецензент:

м.н.с. А. В. Банкевич

(подпись)

Санкт-Петербург

2015

Содержание

Введение

Геномные перестройки были впервые открыты Дображанским и Стуртевант в 1938 [1], но их систематическое изучение началось с распорстранением и удешевлением технологий секвенирования. В 1987 Дэй и Санкофф [2], предложили две основные проблемы в области перестроек:

Проблема 0.1. *Расстояние редактирования.*
Для двух геномов и модели, которая описывает набор допустимых перестроек, найти кратчайшую последовательность операций, трансформирующих один геном в другой.

Проблема 0.2. *Медиана.*
Для трех геномов, построить четвертый геном, который минимизирует сумму парных расстояний редактирования между ним и оставшимися тремя геномами.

Если первая проблема может быть решена за линейное время для большиства моделей, то проблема медианы является NP-трудной для почти всех оценок расстояния редактирования [3]. В тоже время, благодаря инициативе 10000 тысяч геномов(`https://genome10k.soe.ucsc.edu/`), количество отсеквенированных полных геномов растет, и перед исследователями встает новая проблема, заключающаяся в нахождении предковых геномов, которая является обобщением проблемы медианы.

Нахождение таких геномов является важной задачей при анализе собранных геномов [4, 5]. Полученная информация позволяет прояснить механизмы эволюции, уточнить существующие филогенетические деревья и улучшить референсную сборку геномов [6, 7].

На данный момент существует огромное количество программных средств, решающих данную задачу, и каждое из них имеет ограничения. Из всего этого множества методов стоит выделить алгоритмы, основанные на модели анализа

перестроек. Данные программные продукты, в отличие от конкурентов, позволяют получать трансформации, которые говорят, как именно были получены предковые геномы. Отметим, что в последние пять лет такие методы показали более точные результаты на реальных данных [8, 9]. Основной проблемой существующих алгоритмов, основанных на модели анализа перестроек, является отсутствие возможности работы с событиями вставок и удалений генов, хотя данные операции широко распространены в ходе эволюции. Другая проблема состоит в том, что предковые геномы могут содержать циклические хромосомы, хотя их появление невозможно. Проблему линеаризации (отсутствие циклических хромосом в предковых геномах) пытались решать разными способами [10–12], но остается неясным, насколько решение этой задачи влияет на предковые геномы.

В данной работе представлено два алгоритма: первый восстанавливает предковые геномы с учетом операции вставок и удалений генов, второй решает проблему линеаризации предковых геномов. Разработанный метод использует модель, основанную на множественном брейкпоинт графе, впервые предложенную в [13]. Алгоритм реализован в виде программного пакета MGRA2 на языке C++ и доступен под лицензией GNU GPL v2.0. Анализ на симулированных и реальных данных при помощи MGRA2 показывает высокую точность результатов полученного метода, а также демонстрирует возможность анализа геномов млекопитающих с учетом событий вставок и удалений генов.

Диссертация состоит из четырех глав. В первой главе описывается постановка задачи, дается обзор существующих методов и строится модель, в рамках которой будет реализован алгоритм. Вторая глава содержит подробное описание разработанного алгоритма: сначала приводится исследование построенной модели, а затем исчерпывающее формальное описание алгоритма. Третья глава полностью посвящена задачи линеаризации медианного генома. Четвертая глава содержит информацию о сравнении известными методами на симуляционных и реально существующих наборах данных.

Глава 1

Обзор предметной области и постановка задачи

1.1. Постановка задачи

1.1.1. Общие сведения

Известно, что молекула ДНК, открытая в 1868 году Иоганном Фридрихом Мишером, в ходе своей жизни может подвергаться изменениям. Все такие изменения делятся на два класса: точечные мутации (замена, вставка, удаление на уровне нуклеотидов) и структурные изменения, также известные как геномные перестройки. На данный момент известны следующие структурные изменения:

- *Инверсия (реверсия).* Сегмент генома разворачивается, и направление меняется.
- *Разделение.* Одна хромосома разделяется на две.
- *Слияние.* Две хромосомы объединяются в одну.
- *Взаимная транслокация.* Сегмент хромосомы, содержащий теломеру, меняется с сегментом другой хромосомы, также содержащей теломеру.
- *Транспозиция.* Сегмент из генома перемещается в другое место этого же генома.
- *Удаление.* Сегемент генома удаляется.
- *Вставка.* Сегемент генома вставляется.
- *Дупликация.* Сегмент ДНК копируется и вставляется в геном.
  - *Тандемная дупликация.* Копия сегмента вставляется в геном сразу за оригинальным сегментом.

  - *Ретротранспозиция.* Копия сегмента вставляется в произвольное место генома.
  - *Полная геномная дупликация.* Происходит копирование всего генома или одной из хромосом.

- *Горизонтальное перемещение.* Сегмент из одного генома копируется в другой геном.

В ходе последнего двадцатилетия [14–16] стало известно, что геномные перестройки не могут наблюдаться в произвольных участках генома, поэтому была предложена следующая модель [17]:

Модель 1.1.1. *Хрупкая модель.*
Существуют «хрупкие» геномные регионы, которые с большей вероятностью подвержены геномным перестройкам, чем остальные части генома. В данных «горячих» точках наблюдается высокий уровень повторного использования структурных изменений.

Далее в данной работе геном будет представляться как набор хромосом, где каждая хромосома представлена в виде набора консервативных сегментов (генов, синтени блоков [18, 19]), и молекула ДНК подвержена только структурным изменениям. Хромосомы в геноме могут быть циклическими и линейными. Другими словами, геном имеет вид, напоминающий хорошо известный математический объект, - перестановку. Для получения такого представления генома из последовательности нуктлеотидов существует целый ряд программных средств, таких как Sibelia [20], DRIMM-Synteny [21], Cytenator [22], i-ADHoRe3.0 [23].

Существует две модели перестановок: знаковые и беззнаковые. Описание проблем и задач для беззнаковых перестроек приведены не будут, так как данные знания не важны для понимания дальнейшей работы [3]. Стоит отметить, что использование знаковых перестановок биологически оправдано, так как

ДНК обладает двумя комплиментарными цепями. Скажем, что если направление гена совпадает с направлением 5'-3', то он имеет положительный знак, в противном случае, отрицательный.

Приведем примеры оценки расстояния редактирования для двух геномов (для двух знаковых перестановок).

Определение 1.1.1. ***Брейкпоинт расстояние*** *[24] - это расстояние, вычисляемое по формуле* $d_{BP} = n - a - \frac{t}{2}$, *где* n *– количество генов в каждом геноме,* a *- количество общих связностей для двух геномов,* t *– количество общих связностей, где один конец связности является теломерой (концом линейной хромосомы).*

Приведенное выше расстояние можно отнести к классу «наблюдаемых» расстояний, так как оно никак не связано с видами операций, которые были приведены в начале параграфа. Брейкпоинт расстояние требует отсутствия вставок, удалений и дуплицированных блоков, что делает его использование очень ограниченным. Далее будет описано другое расстояния между геномами, но перед этим введем понятие DCJ.

Определение 1.1.2. ***Double Cut and Join (DCJ)*** *[25, 26] (так же известное, как* ***2-брейк*** *[27, 28]) заменяет связности* pq, rs *в геноме на связности* pr, qs *или* qr, ps.

Другими словами DCJ *разрывает* две связности (pq, rs) в геноме и *соединяет* полученные концы в другом порядке (pr, qs или qr, ps). Данная операция унифицирует набор перестроечных операций: транслокацию, разделение, слияние, инверсию [25]. Для того чтобы ввести формулу для DCJ-расстояния, необходимо дать определение графовой конструкции, часто использующейся при исследовании расстояний между двумя геномами.

Определение 1.1.3. ***Брейкпоинт граф*** *[29]*

Пусть даны циклические однохромосомные геномы P *и* Q *(циклические знаковые перестановки) над одним и тем же множеством генов* Γ.

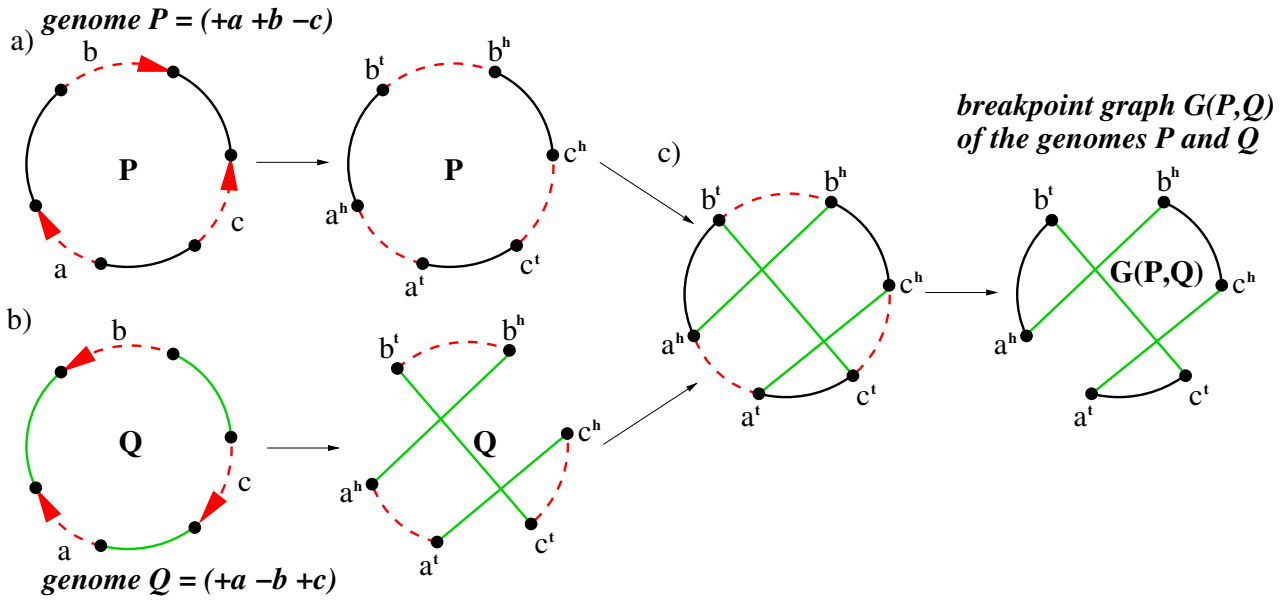

Рис. 1.1. **a)** Однохромосомный геном $P = (+a\ +b\ -c)$, представленный как черно-красный цикл **b)** Однохромосомный геном $Q = (+a\ -b\ +c)$, представленный как зелено-красный цикл **c)** Брейкпоинт граф $G(P,\ Q)$ с и без наблюдаемых ребер

Брейкпоинт граф $G(P, Q)$ *– это граф, определенный над множеством вершин* $V = \{x^t, x^h | x \in \Gamma\}$ *с ребрами трех цветов: наблюдаемые (ребра, связывающие веришны* x^t *и* x^h*), черные (связности генов в* P*) и зеленые (связности генов в* Q*).*

Пример брейкпоинт графа приведен на рисунке 1.1. Отметим, что черно-зеленые альтернирующие циклы играют важную роль в изучении перестроек. В дальнейшей работе для простоты изложения вместо понятия черно-зеленых альтернирующих циклов/путей будет использоваться понятие альтернирующих циклов/путей. Для того чтобы представлять линейные хромосомы с помощью брейкпоинт графа, введем одну нерегулярную вершину ∞, соединив с ней все концы линейных хромосом. Такая модификация брейкпоинт графа для линейных хромосом позволяет проводить исследование DCJ операций, аналогично случаю анализа циклических хромосом. Известно, что сценарий для линейных хромосом хорошо приближается трансформацией для циклических хромосом [28]. На рисунке 1.2 приведено представление всех стандартных перестроек в модели брейкпоинт графа.

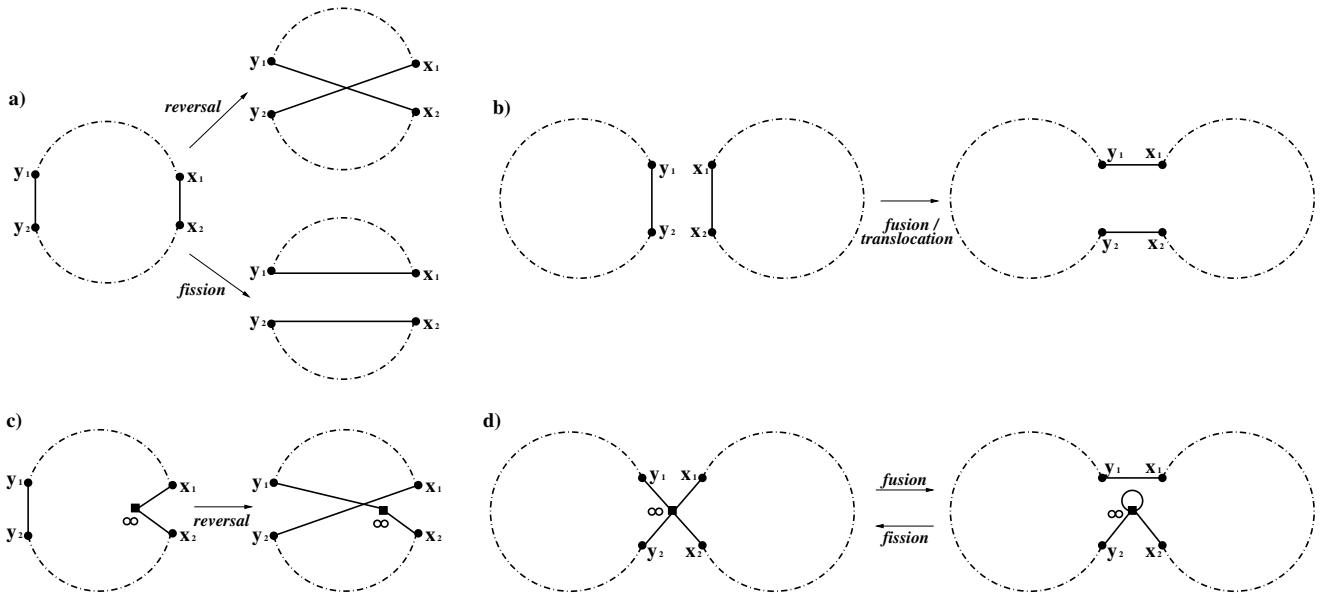

Рис. 1.2. **a)** DCJ на ребрах $(x_1,\ x_2)$ и $(y_1,\ y_2)$, лежащих на одной хромосоме, соответствует либо инверсии, либо разделению. **b)** DCJ на ребрах $(x_1,\ x_2)$ и $(y_1,\ y_2)$, лежащих на разных хромосомах, соответствует транслокации/слиянию. **c)** DCJ на ребрах $(x_1,\ \infty)$ и $(y_1,\ y_2)$, лежащих на линейной хромосоме, соответствует инверсии, происходящей на хромосомном конце x_1 и создающей новый хромосомный конец y_1. **d)** DCJ на ребрах $(x_1,\ \infty)$ и $(y_1,\ \infty)$, лежащих на разных линейных хромосомах, моделирует слияние. Событие разделения можно представить как DCJ, оперирующая нерегулярным ребром $(\infty,\ \infty)$ и произвольным регулярным ребром в геноме.

Теперь можно дать формулу для DCJ-расстояния [26]

$$d_{DCJ} = n - c - \frac{p}{2}$$

где n – количество генов в каждом геноме, c – количество альтернирующих циклов в брейкпоинт графе и p – количество альтернирующих путей четной длины.

Данное расстояние имеет целый ряд расширений. Например, к данному расстоянию можно добавить учет событий вставок и удалений [30–33]. Другое улучшение позволяет находить расстояние для геномов, содержащих дупликации [31]. Также существует обобщение операции DCJ на произвольное количество разрывов, известное как *k-брейк* [27, 28], позволяющее учитывать транспозиции. Кроме того существует работы по оценке расстояния с условием, что операции имеют различные веса [34, 35].

Таким образом, введеное DCJ-расстояние дает хорошую аппроксимацию для реального расстояния и упрощает анализ операций перестроек. Далее в работе будет использоваться именно эта метрика для оценки расстояния. Переформулируем проблему медианы, упомянутую во введении, с условием, что расстояние между геномами оценивается как DCJ-расстояние.

Проблема 1.1.1. *Проблема медианы для модели DCJ-операций.*
Для трех данных геномов A, B, C найти такой геном M, чтобы сумма

$$d_{DCJ}(A, M) + d_{DCJ}(B, M) + d_{DCJ}(C, M)$$

была минимальна.

Теперь можно дать описание проблем, решаемых в данной работе.

1.1.2. Проблема восстановления предковых геномов

Следующая задача является обобщением проблемы медианы и называется *малой филогенетической проблемой*. Прежде чем она будет сформулирована, стоит отметить, что филогенетическое дерево обычно представляется как полное бинарное дерево, где внутренние узлы – это предковые геномы, которые необходимо восстановить, а листья – это существующие геномы.

Проблема 1.1.2. *Малая филогенетическая проблема.*
Для данного филогенетического дерева и существующих геномов найти предковые геномы так, чтобы сумма из расстояний для каждого ребра филогенетического дерева была минимальна.

Стоит отметить, что если взять внутренний узел и три его соседних узла, соответствующих каким-то геномам, то получается проблема медианы, упомянутая выше. Также существует *большая филогенетическая проблема*, которая возникает, когда филогенетическое дерево не известно, но данная проблема не затрагивается в дальнейшей работе.

1.1.3. Задача линеаризации медианного генома

Существует целый ряд продвинутых алгоритмов для решения проблемы медианы в модели оценки расстояния с помощью DCJ [36–39], допускающих наличие циклических хромосом в медианном геноме. Одной из целей дальнейшей работы является создание метода, который позволит на основе решения проблемы медианы получить медианный геном, содержащий линейные хромосомы по оптимальному сценарию.

Проблема 1.1.3. *Линейная проблема медианы для модели DCJ-операций. Для линейных геномов G_1, G_2, G_3 и построить медианный линейный геном M' (не содержащий циклические хромосомы), такой что*

$$\sum_{i=1}^{3} d_{DCJ}(M', G_i)$$

была минимальна.

Постановка проблемы и ее аппроксимационное решение, с помощью построения линейного медианного генома из медианного генома, впервые была предложена в [40]. В данной статье при доказательстве основной теоремы был пропущен распространенный случай комбинации перестроечных операций, который может повлиять на результат алгоритма. Стоит отметить, что описанный способ применим только на данных, не содержащих события вставок и удалений. Поэтому задачей данной работы является исправить вышеописанные проблемы, реализовать данный алгоритм на практике, а также продемонстрировать, как этот алгоритм может быть применен в случае малой филогентической проблемы.

1.2. Существующие методы

На данный момент существует невероятное количество программных средств, решающих малую филогенетическую проблему. Все их условно можно разде-

лить на два больших семейства: алгоритмы, базирующиеся на перестроечной модели, и алгоритмы, базирующиеся на цитогенетической модели.

Первое семейство алгоритмов благодаря модели является более общим и позволяет учитывать различные наборы операций при анализе. В основе данных алгоритмов обычно лежат разные техники, обобщающие решение проблемы медианы или анализ перестроек между двумя геномами.

Вторая модель вдохновлена экспериментальной техникой при «раскраске хромосомы» из гибридизации. Основной принцип, лежащий в основе данных алгоритмов, заключается в том, что если два блока имеют общую связь в двух существующих геномах, то, возможно, данная связь существует в их общем предке.

Также стоит упомянуть об алгоритмах, которые ищут предковые геномы на основе выравнивания генов. Данные методы используют информацию, получаемую в процессе выравнивания для вывода предковых геномов.

Задача линеаризации медианных (предковых) геномов может возникнуть только в результате полученном с помощью алгоритмов из первого семейства. Но ни один из алгоритмов в данном семействе не умеет бороться с изложенной проблемой линеаризации предковых геномов.

Далее будет приведено подмножество алгоритмов с описанием их возможностей.

1.2.1. Алгоритмы, основанные на цитогенетической модели

inferCARs и inferCARsPro

InferCARs исторически первый алгоритм, появившийся в 2006 году [41], который восстанавливает связные предковые регионы. В 2010 [42] вышло улучшение данного алгоритма, которое получило название inferCARsPro. Обе версии работают с геномами, которые подвергались четырем перестроечным операциям: инверсии, транслокации, слиянию, разделению, но не работают с событиями

вставок и удалений. Алгоритм требует информации о филогенетическом дереве и знания о расстоянии на каждой ветке дерева. Стоит отметить, что качество реконструкции данного алгоритма для небольшого количества перестановочных операций является очень высоким и не уступает новым аналогам. Программа работает на геномах, сопоставимых по размеру с геномами млекопитающих. К сожалению, найти вторую версию данного алгоритма в свободном доступе не представляется возмножным.

dupCARs

Программа, которая расширяет алгоритм inferCARs, выпущенная теми же авторами в 2008 [43]. Данное программное средство восстанавливает связные предковые регионы. Главным добавлением данного алгоритма является наличие дуплицированных генов в существующих геномах при восстановлении предковых регионов. Наличие дуплицированных генов сказывается на качестве реконструкции. Например, в случае отсутствия дуплицированных генов в данных, алгоритм получает результаты хуже, чем inferCARsPro, поэтому и рассмотрен как отдельное программное средство.

PMAG, PMAG$^+$ и MLWD

Данные алгоритмы объединены в одну большую программу MLGO [44], которая позволяет решать большую филогенетическую проблему. PMAG [45], восстанавливает предковые геномы, которые подвергались стандартным перестроечным операциям. PMAG$^+$ [46] расширяет алгоритм PMAG на события вставок и удалений. Алгоритм MLWD [47] восстанавливает филогенетические деревья с учетом стандартных перестроечных операций, событий вставок, удалений и дупликаций генов. Данные алгоритмы имеют достаточно высокую точность результатов. Кроме того MLGO имеет удобный интерфейс работы. Данная программа является одним из лучших решений, так как охватывает наибольшее

количество событий и имеет хорошее качество получаемых результатов.

GapAdj

Программа, выпущенная в 2012 [48], позволяет восстанавливать предковые геномы, которые подверглись событиям вставок и удалений, полной геномной дупликации и стандартным перестроечным операциям. Требует заданного филогенетического дерева без указания расстояний на ветках. Имеет ограничение по длине на входные геномы и не применима к геномам млекопитающих. Как показывает анализ из других статей, качество реконструкции предковых геномов невысоко.

1.2.2. Алгоритмы, основанные на перестроечной модели

BPAAnalysis

Одна из исторически первых программ, базирующихся на оценки расстояния, появившаяся в 1998 году [49]. Данный алгоритм базируется на расширении брейкпоинт расстояния и восстанавливает только филогенетические деревья. Качество получаемых деревьев невысоко, отсутствует вывод предковых геномов и информации о перестройках.

MGR

Экспоненциальный переборный алгоритм, разработанный в 2002 году [50], позволяющий минимизировать расстояния вдоль каждой ветки дерева и восстанавливать филогенетические деревья. Данная программа активно использовалась в первом десятелитии двадцать первого века для вывода перестроек. Алгоритм работает с геномами, содержащими несколько хромосом. Программа не восстанавливает предковые геномы. Требует, чтобы геномы были небольшого размера, отсутствовали события вставок, удалений и дупликаций.

GRAPPA

Программное средство, включающее в себя целую серию различных алгоритмов. Основным вкладом является алгоритм, вычисляющий количество инверсий [51] между однохромосомными геномами с заданным филогенетическим деревом. Программа не восстанавливает предковые геномы.

EMRAE

Данный алгоритм выпущен в 2008 году [52, 53], он восстанавливает предковые геномы, а также выводит надежные перестройки, позволяющие частично понять, как эти геномы были получены. Программа не работает с событиями вставок, удалений, дупликаций, требует на вход однохромосомные геномы и не выводит филогенетические деревья.

GASTS

Один из самых сложных алгоритмов из когда-либо созданных в этой области [10]. Данная программа решает большую филогенетическую проблему, то есть может восстанавливать предковые геномы, филогенетические деревья и выводит количество операций, необходимых для получения данных предковых геномов. Алгоритм является результатом продолжительного исследования [36–38] и содержит в себе точный алгоритм решения проблемы медианы для небольших размеров входных геномов, а также разнообразные эвристики для обхода ограничения на длину входных геномов. Качество реконструкции предковых геномов является самым точным среди существующих программных средств, работает на геномах, сопоставимых по размеру с геномами млекопитающих, однако, алгоритм ограничен только стандартными перестроечными операциями: инверсиями, транслокациями, слияниями, разделениями.

MGRA

Алгоритм, предложенный в [13], базирующийся на модели множественного брейкпоинт графа, которая обобщает проблему медианы и проблему расстояния на случай множества геномов. В статье было показано, как с помощью данной модели реконструировать предковые геномы, трансформацию между ними, а также продемонстрирована возможность восстановления филогенетического дерева на основе анализа графа. Авторами был разработан алгоритм восстановления предковых геномов, который давал превосходные результаты, однако учитывал только стандартные перестроечные операции и требовал не слишком большое количество операций на каждой ветке дерева.

TIBA

Программа [54], целью которой является восстановление филогенетического дерева на основе информации о перестройках. Не восстанавливает предковые геномы, учитывает только стандартные перестроечные операции, не ищет перестроечные сценарии.

EGCHEL

Данная программа [55] не восстанавливает предковые геномы, филогенетические деревья или перестройки. Основной задачей данного программного средства является конвертация входных геномов, состоящих из неравного генного контента, в геномы с равным генным контентом. Данный алгоритм вставляет «протезные» хромосомы для минимизации суммарного попарного расстояния между геномами.

SCJ

Программа [56], позволяющая решать большую филогенетическую проблему, то есть восстанавливающая предковые геномы и деревья. Алгоритм вводит

новую оценку расстояния между геномами – SCJ-расстояние, которое базируется на обобщении брейкпоинт расстояния для мультихромосомных геномов. Программа требует, чтобы генный контент во всех геномах был равный. Качество получаемых результатов невысоко.

1.2.3. Другие подходы

multiOrthoAlign

Программа [57], решающая малую филогенетическую проблему, основанная на выравнивании последовательности генов. Алгоритм решает проблему медианы итеративно для каждого узла дерева. Работает с событиями удалений, дупликаций, инверсий и транспозиций, но не работает с событиями вставок. Почти нигде не сравнивался с другими программами, поэтому качество результатов неизвестно.

Глава 2

Восстановление предковых геномов

2.1. Построение модели на основе множественного брейкпоинт графа

Основываясь на постановке задачи, описанной в 1.1.2, предположим, что геномы $P_1, P_2, \cdots, P_k$ состоят из линейных или/и циклических хромосом над множеством генов B. Поставим в соответствие каждому геному P_i уникальный цвет (обозначим, как и геном, P_i). Тогда назовем P_i-ребром ребро с цветом P_i. Брейкпоинт граф $G(P_i)$ состоит из P_i-ребер, представленных в геноме P_i.

Определение 2.1.1. ***Множественный брейкпоинт граф*** $G(P_1, P_2, \cdots, P_k)$ – *это суперпозиция брейкпоинт графов* $G(P_1), G(P_2), \cdots, G(P_k)$.

$G(P_1, P_2, \cdots, P_k)$ состоит из ребер k-цветов $P_1, P_2, \cdots P_k$, представляющих связности соответствующих геномов (пример множественного брейкпоинт графа для трех геномов приведен на рис. 2.1).

Продолжая построение модели, обозначим, что C – это множество всех цветов $P_1, P_2, \cdots, P_k$. Тогда назовем *мультицветом* любое непустое подмножество множества C. Все ребра, соединяющие вершины x и y в множественном брейкпоинт графе $G(P_1, P_2, \cdots, P_k)$, формируют *мультиребро* (x, y), имеющее мультицвет, собранный из цветов входящих ребер. *Мультистепенью* вершины x является число мультиребер, инцидентных вершине x. Вершина является брейкпоинтом, если её мультистепень больше одного. Множественный брейкпоинт граф без брейкпоинтов называется *идентичным множественным брейкпоинт графом* $G(R, R, \cdots, R)$. Также *идентичный множественный брейкпоинт граф* можно охарактеризовать как множественный брейкпоинт граф, состоящий из полных мультиребер цвета C, которые соответствуют связности в геноме R.

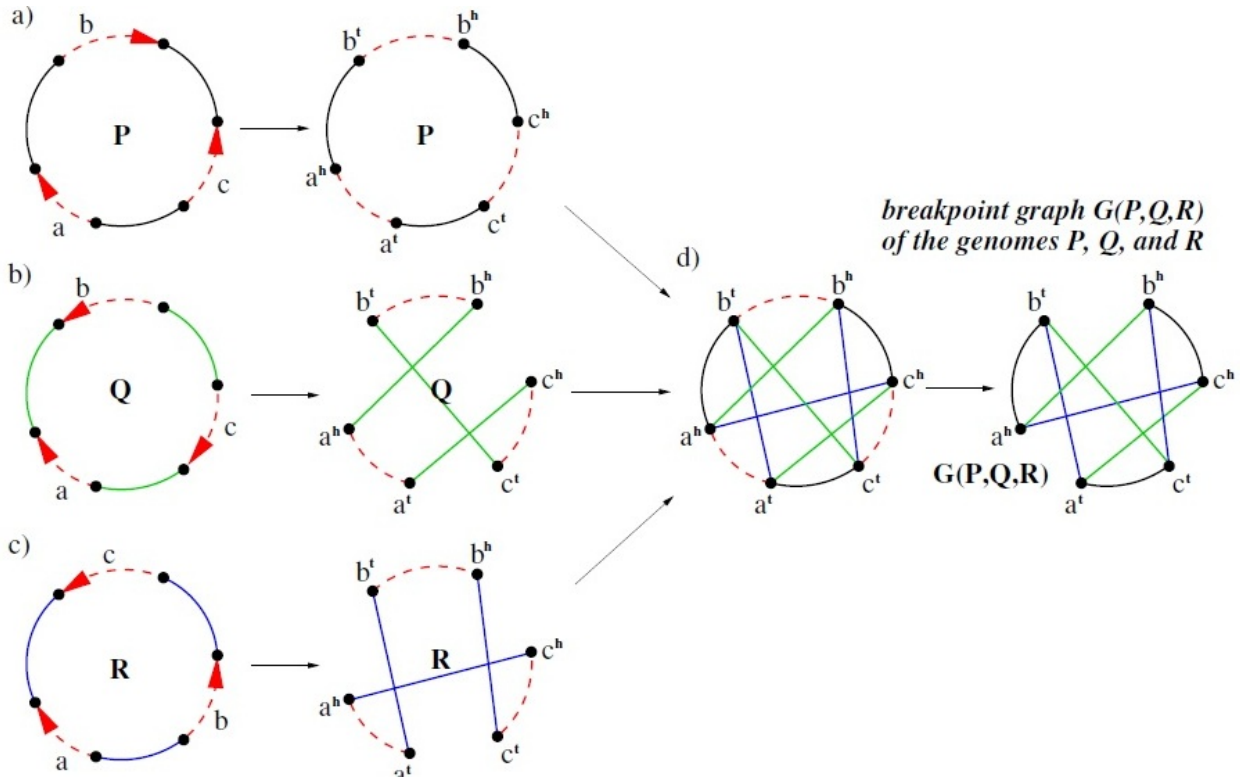

Рис. 2.1. **a)** Однохромосомный геном $P = (+a\ +b\ -c)$, представленный как черно-красный цикл. **b)** Однохромосомный геном $Q = (+a\ -b\ +c)$, представленный как зелено-красный цикл. **c)** Однохромосомный геном $R = (+a\ -c\ -b)$, представленный как сине-красный цикл. **d)** Множественный брейкпоинт граф $G(P,\ Q,\ R)$ с и без наблюдаемых ребер.

Пусть X – общий предковый геном для геномов $P_1, P_2, \cdots, P_k$, где каждый геном P_i получен из X с помощью серии перестроек. Следовательно, существует трансформация из k-кортежей из геномов и их брейкпоинт графов

$$t : G(X, X, \cdots, X) \to \cdots \to G(P_1, P_2, \cdots, P_k)$$

где каждый шаг представляет собой DCJ (2-брейк), происходящий либо на одном из геномов $P_1, \cdots, P_k$ (в этом случае один геном из кортежа изменяется), либо в одном из предковых геномов (в этом случае множество геномов, общих для предкового генома, изменяются одновременно). Другими словами, каждый DCJ в трансформации происходит над мультицветом, состоящим из одного цвета, или над мультицветом, который соответствует предковому геному. Такие мультицвета назовем *T-консистентными*. Они сформированы из цветов листьевых геномов, лежащих в поддереве, которое полученно путем удаления

соответвующего ребра из дерева. Кроме того, направленные (ненаправленные) ветки из дерева имеют биективное соответствие с направленными (ненаправленными) парами комплиметарных T-консистентных мультицветов (см рис. 2.2). Если мультицвет Q является T-консистентным, тогда его дополнение $C \setminus Q$ также T-консистентно, то есть T-консистентные мультицвета формируют пары комплиментарных цветов.

Отметим, что трансформация t может быть представлена как коллекция из путей в дереве, начинающихся из корня X и заканчивающихся в каждом листьевом геноме. Каждый внутренний узел в филогенетическом дереве достижим одним из этих путей, и он определяет геном в данном узле 2.2. Таким образом, для реконструкции геномов во внутренних узлах дерева достаточно найти трансформацию t. MGRA пытается восстановить реверс-трансформацию для t:

$$t^{-1} : G(P_1, \ \cdots \ P_k) \to \cdots \to G(X, \ \cdots, \ X)$$

Отметим, что свойство X быть общим предковым геномом для $P_1, \cdots, P_k$ не существенно. X может быть любым геномом из курса эволюции (то есть X находится в любом месте дерева), включая сами геномы $P_1, \cdots, P_k$. Поэтому MGRA не фиксирует геном X, но определяет его *пост фактум*. Другими словами, MGRA пытается найти трансформацию из $G(P_1, \ \cdots \ P_k)$ в любой идентичный множественный брейкпоинт граф с минимальным числом операций. Также эта трансформация может быть рассмотрена как избавление от брейкпоинтов в графе $G(P_1, \ \cdots \ P_k)$.

В трансформации t все DCJ «направлены» из X в направлении листьевых геномов, следовательно, в любой паре комплиментарных T-консистентных мультицветов только один мультицвет может быть использован для DCJ в трансформации t. Единственное исключение – это мультицвета, соответствующие ветке дерева, где находится X. Для избавления этого ограничения за-

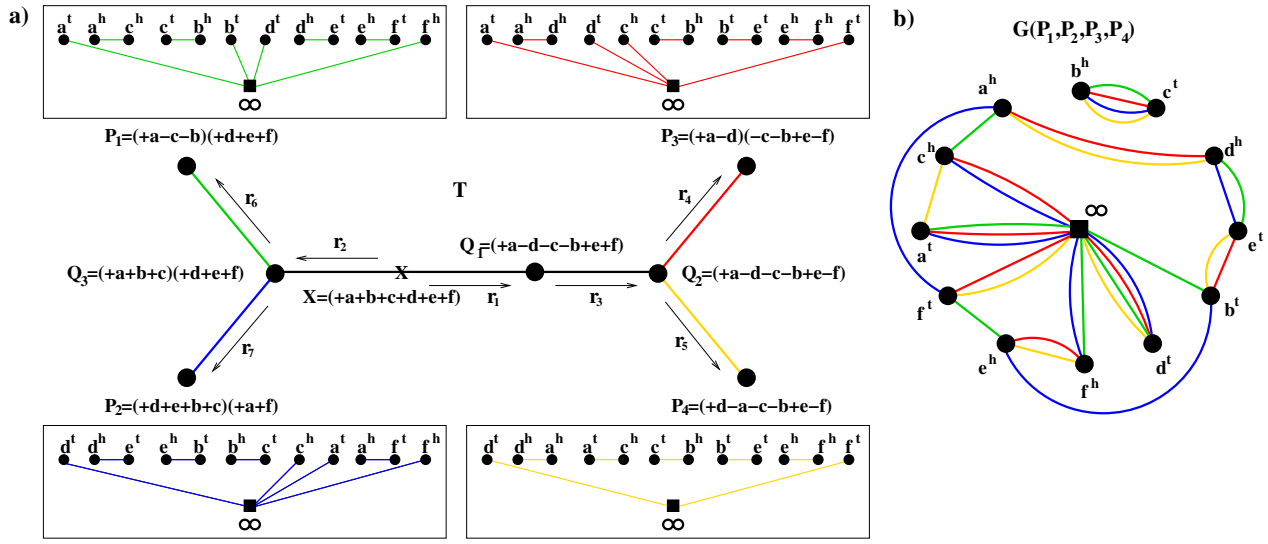

Рис. 2.2. Пример с четырьмя геномами с филогенетическим деревом и их множественным брейкпоинт графом. **a)** Филогенетическое дерево с четырьмя геномами P_1 (зеленый), P_2 (синий), P_3 (красный), P_4 (желтый), расположенными в листьях, и с заданными промежуточными геномами. Геномы $P_1,\ P_2,\ P_3,\ P_4$ представлены как геномные графы (наблюдаемые ребра не показаны). **b)** Множественный брейкпоинт граф $G(P_1,\ P_2,\ P_3,\ P_4)$, полученный как суперпозиция геномных графов.

фиксируем ветку из филогенетического дерева. Предположим, что X лежит на данной ветке; разорвем данную ветку на две подветки и зададим направление на дереве в направлении к корню X (см рис. 2.2). Мультицвета, которые соответствуют начальным узлам из каждой направленной ветки, называются $\vec{T}$-консистентными, и только они участвуют в операциях DCJ в трансформации t^{-1}. Например, в дереве T на рисунке 2.3, мультицвет MR $\vec{T}$-консистентный, в то время как $RDQHC$ T-консистентный, но не $\vec{T}$-консистентный.

Важным свойством трансформации t^{-1} (и t) является, то, что это *строгая* трансформация, то есть для каждой пары $\vec{T}$-консистентных мультицветов $(Q_1,\ Q_2)$ все DCJ с мультицветами Q_1 должны предшествовать всем DCJ с Q_2 в t^{-1}. Соответственно в t все DCJ с Q_2 должны предшествовать всем DCJ с мультицветами Q_1. Имея трансформацию t' из $G(P_1,\ \cdots\ P_k)$ в $G(X,\ \cdots\ X)$ с DCJ из $\vec{T}$-консистентных мультицветов в любом порядке, возможно переупорядочить DCJ в этой трансформации для получения строгой трансформации. Таким образом, MGRA не имеет заявленных ограничений и строит трансфор-

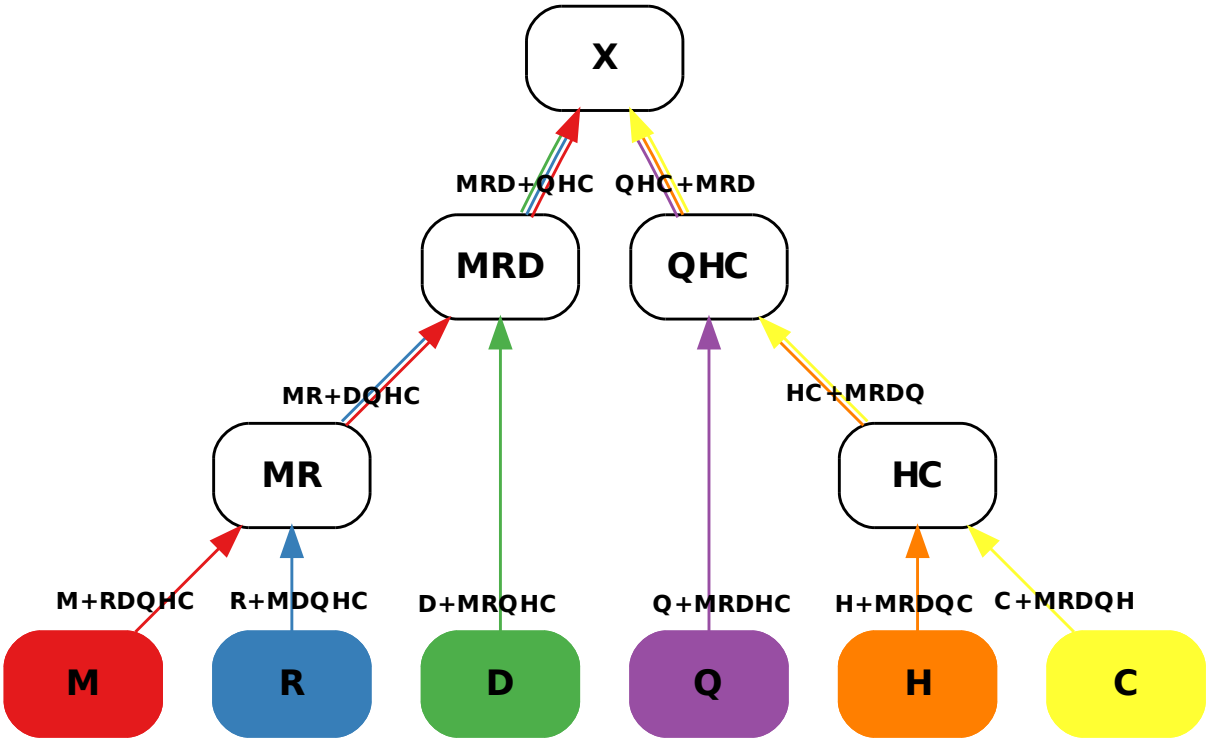

Рис. 2.3. Филогенетическое дерево T для шести геномов млекопитающих: мышь (красный), крыса (синий), собака (зеленый), макака (фиолетовый), человек (оранжевый), шимпанзе (желтый). Корень X расположен на ветке $MRD + QHC$. Ветки имеют направление к X и помечены соответствующей парой комплиментарных T-консистентных мультицветов. $\vec{T}$-консистентным мультицветом помечен каждый узел, соответствующий исходящей направленной ветке. Ориентация дерева не обязана коррелировать с временной шкалой, и корневому геному X нет необходимости быть общим предком для всех листьевых геномов. Рисунок воспроизведен из [13].

мацию t', затем переупорядочивает t' в t^{-1}, и, в конце концов, восстанавливает t и предковые геномы.

В дальнейшей работе описываются важные свойства построенной модели, которые используются для разработки шагов, а также приводится исчерпывающее описание разработанных шагов для получения трансформации t'.

2.2. Свойства построенной модели

Прежде чем перейти к описанию основных принципов работы MGRA, на которых базируются придуманные алгоритмы, позволяющие получить транс-

формацию t', опишем базовые свойства T-консистентных мультицветов.

Для дальнейшего изложения предполагаем, что филогенетическое дерево T и его корень X фиксированы, и они определяют T-консистентные и $\vec{T}$-консистентные мультицвета как композицию листьев дерева T. Следующая лемма напрямую следует из определения T-консистентных мультицветов:

Лемма 1. *Для любых T-консистентных мультицветов Q_1 и Q_2, таких что $Q_1 \cap Q_2 \neq \emptyset$ имеем, что $Q_1 \subseteq Q_2$ или $Q_2 \subseteq Q_1$.*

Теорема 2. *Пусть Q – любой мультицвет, тогда существует уникальная коллекция из непересекающихся T-консистентных мультицветов $\mathcal{Q} = \{Q_1, \cdots, Q_m\}$, такая что $Q = Q_1 \cup \cdots \cup Q_m$ и m минимальна.*

Доказательство. Пусть R – это T-консистентный мультицвет, такой что $R \subseteq Q$, и не существует другого T-консистентного мультицвета R', такого что $R \subsetneq R' \subseteq Q$. Тогда $R \in \mathcal{Q}$ и, более того, Q содержит все такие мультицвета.

Действительно, рассмотрим все элементы из $\mathcal{Q}$, имеющие непустое пересечение с R, и без потери общности обозначим $Q_1, Q_2, \ldots, Q_r$. Поскольку $R \subseteq Q$ имеем, что $R \subseteq Q_1 \cup \cdots \cup Q_r$. С другой стороны, по Лемме 1 каждый из Q_i, $1 \leq i \leq r$ должен быть подмножеством R и, таким образом, получаем $Q_1 \cup \cdots \cup Q_r \subseteq R$. Из этого следует, что $R = Q_1 \cup \cdots \cup Q_r$. Минимальность m следует из условия $r = 1$ (другими словами, заменяя $Q_1, \ldots, Q_r$ из $\mathcal{Q}$ на R, уменьшаем m). Получаем, что $R \in \mathcal{Q}$.

Поскольку для каждого мультицвета $Q_i \in \mathcal{Q}$ можно найти R (как определено выше) с условием, что $Q_i \subseteq R$, тогда имеем $Q_i = R$. Получаем, что $\mathcal{Q}$ полностью состоит из таких мультицветов R. □

Следующая лемма также следует из определения T-консистентных и $\vec{T}$-консистентных мультицветов:

Лемма 3. *Пусть Q – T-консистентный мультицвет и пусть T_Q – поддерево дерева T, индуцированное по листьям из мультицвета Q. Тогда Q – это*

$\vec{T}$-консистентный мультицвет тогда и только тогда, когда T_Q не содержит корень дерева T.

Теорема 4. *Пусть Q_1 – T-консистентный мультицвет и Q_2 – $\vec{T}$-консистентный мультицвет, такие что $Q_1 \subseteq Q_2$. Тогда Q_1 также $\vec{T}$-консистентен.*

Доказательство. Если $Q_1 = Q_2$, то условие теоремы выполнено. Для доказательства предположим, что $Q_1 \subsetneq Q_2$. Пусть T' – поддерево дерева T, индуцированное с помощью Q_2. Поскольку Q_2 – $\vec{T}$-консистентный мультицвет, T' не содержит корня дерева T.

Пусть T'' – поддерево дерева T, индуцированное с помощью Q_1. Поскольку $Q_1 \subsetneq Q_2$, то T'' – поддерево T'. Тогда T'' не содержит корня T, следовательно, Q_1 – также $\vec{T}$-консистентный мультицвет. □

Теорема 5. *Пусть Q – любой цвет, такой что $Q = Q_1 \cup Q_2 \cup \cdots \cup Q_m$, где Q_i – непересекающиеся T-консистентные мультицвета. Тогда все Q_i, за возможным исключением одного, являются $\vec{T}$-консистентными.*

Доказательство. Поскольку Q не T-консистентный мультицвет, имеем $m \geq 2$. Если все Q_i – $\vec{T}$-консистентные мультицвета, тогда теорема доказана.

Если не все Q_i $\vec{T}$-консистентны, тогда без потери общности предположим, что Q_1 не $\vec{T}$-консистентный мультицвет. Следовательно, $\overline{Q_1}$ является $\vec{T}$-консистентным. Для каждого $i > 1$, поскольку Q_1 и Q_i не пересекаются, имеем $Q_i \subseteq \overline{Q_1}$. Тогда по Лемме 4 Q_i должно быть $\vec{T}$-консистентным мультицветом. □

Теперь перейдем к описанию принципов работы MGRA2, на которых базируются нижеописанные алгоритмы. Первый принцип работы следует из описания модели:

(P1) MGRA2 выполняет DCJ только на $\vec{T}$-консистентных мультицветах.

Второй принцип вдохновлен свойством кратчайшего сценария между двумя геномами. Брейкпоинт граф для двух циклических геномов состоит из циклов и путей, представляющих собой нетривиальные компоненты связности. С другой стороны, любой идентичный брейкпоинт граф содержит только полные мультиребра, которые формируют тривиальные компоненты связности. Таким образом, трансформация между брейкпоинт графом и идентичным брейкпоинт графом может быть представлена как процесс увеличения (но никогда как процесс уменьшения) числа компонент связности. Для множества геномов ослабим данный принцип:

(P2) MGRA2 никогда не уменьшает число компонент связностей в брейкпоинт графе.

Третий принцип необходим для борьбы с *брейкпоинт-переиспользованием* [28, 58]. Оно происходит, когда различные DCJ оперируют над одними и теми же вершинами. Одним из результатов таких событий является наличие мультиребер с не T-консистентными мультицветами в множественном брейкпоинт графе. Для борьбы с такими мультиребрами на последних шагах MGRA2 разделяет каждое такое мультиребро на множество параллельных мультиребер. Так как MGRA2 считает, что T-консистентные мультицвета соответствуют связности в соответствующем предковом геноме, алгоритм не разделяет мультиребра с T-консистентными мультицветами. Этот принцип также может быть рассмотрен как обобщение так называемого *идеального перестроечного* сценария [59]. Таким образом, можно сформулировать третий принцип:

(P3) MGRA2 никогда не разделяет T-консистентные мультицвета на подмультицвета.

Последний принцип также вдохновлен обработкой циклов в брейкпоинт графе между двумя геномами. В случае выбора ребер для DCJ, в брейкпоинт графе они всегда соединены третьим ребром, отсюда вытекает следующий принцип для множественного брейкпоинт графа:

(P4) MGRA2 выполняет DCJ на двух мультиребрах, только если они соединены другим мультиребром.

Сформулировав и доказав свойства модели и основные принципы, которые не должны нарушаться, перейдем к описанию алгоритмов, которые позволяют находить кратчайший сценарий между геномами.

2.3. Обработка событий вставок и удалений

Когда ген b представлен во всех геномах $P_1, \cdots, P_k$, соответствующие вершины b^t и b^h в $G(P_1, \cdots, P_k)$ инцидентны ребрам всех k геномных цветов. С другой стороны, когда ген b отсутствует в одном из геномов P_i, вершины b^t и b^h не инцидентны ребрам цвета P_i . Вершины в брейкпоинт графе $G(P_1, \cdots, P_k)$ назовем *несбалансированными*, если число инцидентных ребер меньше, чем k. Стоит отдельно отметить, что для каждой несбалансированной вершины её противоположная вершина также несбалансирована. Другими словами, в обоих вершинах отсутствуют одинаковые цвета среди их инцидентных ребер.

Результатом событий вставок и удалений является неравный генный контент среди геномов, что приводит к несбалансированным вершинам в множественном брейкпоинт графе. Для реализации обработки событий вставок и удалений в MGRA2 была обобщена идея «протезных» хромосом [33, 55], использованных в исследованиях между двумя геномами. Для пары несбалансированных вершин b^t и b^h, у которых цвета потерянных инцидентных ребер формируют множество Q (другими словами, в геномах множества Q отсутствует ген b), проведем процедуру балансировки путем добавления «протезного» мультиребра (b^h, b^t) мультицвета Q в множественный брейкпоинт граф (рис. 2.4а). На данной стадии такое мультиребро может быть представлено как циклическая хромосома, состоящая из единственного гена b, в каждом геноме из множества Q. Исходя из этого наблюдения, предполагаем, что данное мультиребро не имеет

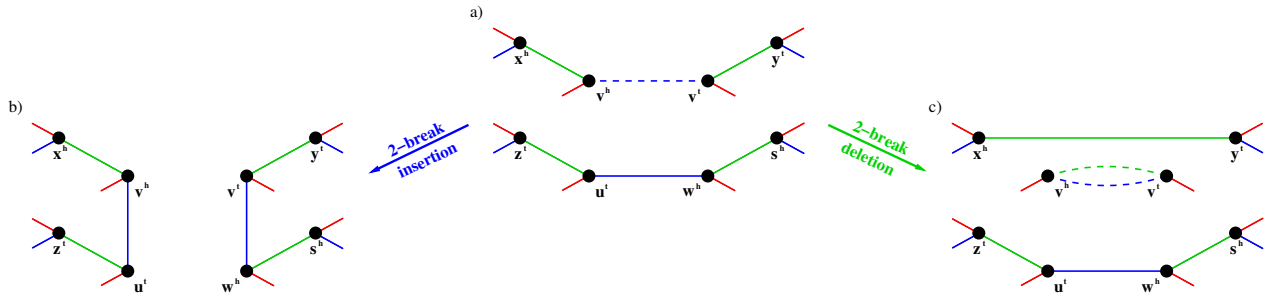

Рис. 2.4. **a)** Подграф сбалансированного брейкпоинт графа для трех геномов, покрашенных в синий, красный, зеленый цвет. Синее ребро (w^h, u^t) кодирует связность генов wu в синем геноме, с другой стороны, синее «протезное» ребро (v^t, v^h) представляет отсутствующий ген v. **b)** Подграф, полученный после DCJ над синим геномом, который заменяет «протезное» ребро (v^t, v^h) и регулярное ребро (w^h, u^t) на регулярные ребра (w^h, v^t) и (v^h, u^t), кодирующие связность последовательности генов wvu. Другими словами, отсутствующий блок v вставляется между блоками w и u. **c)** Подграф, полученный после DCJ над зеленым геномом, который заменяет регулярные ребра (x^h, v^h) и (v^t, y^t), кодирующие связность последовательности генов $x - vy$, на ребра (v^h, v^t) и (x^h, y^t). Параллельные зеленое и синее ребра на вершинах (v^h, v^t) формируют «протезное» ребро (x^h, y^t). Полученный подграф кодирует связность последовательности генов xy, то есть ген v был удален из зеленого генома.

никакой геномной интерпретации. Вместо этого, можно рассматривать данное ребро, как «черную дыру», из которой может возникнуть либо событие вставки, либо событие удаления.

После балансировки полученный множественный брейкпоинт граф подвергается перестроечному анализу, который описан в следующем параграфе, без разделения на обычные или «протезные» ребра. Однако, в получившемся сценарии каждое DCJ, включающее «протезное» мультиребро, можно интерпретировать как событие вставки или удаления. Будем рассматривать DCJ как событие вставки в Q, если оно перемещает «протезное» мультиребро мультицвета Q на другую позицию (рис. 2.4a,b). С другой стороны, DCJ, которое перемещает обычное ребро мультицвета Q в «протезную» позицию, будем интерпретировать как событие удаления в Q (рис. 2.4a,c). В результирующем идентичном брейкпоинт графе удалим все полные ребра на «протезных» позициях. Получим трансформацию из оригинального несбалансированного брейкпоинт графа (без

«протезных» мультиребер), которая включает стандартные перестроечные операции, события вставок и удалений. Данная трансформация будет приводить к множественному идентичному брейкпоинт графу, где некоторые вершины отсутствуют (такие вершины соответствуют удаленным генам в геномах). Стоит отметить, что итоговое число перестроек и событий вставок и удалений равно числу перестроек в сценарии, включающем «протезные» хромосомы.

2.4. Анализ перестроек

2.4.1. T-консистентные адекватные подграфы

Данный шаг алгоритма MGRA2 является комбинацией и обобщением двух идей: хорошие циклы/пути, описанные в [13], и адекватные подграфы, описанные в [38]. Далее кратко описано, в чем заключаются данные решения.

MGRA использует циклы (пути) в брейкпоинт графе как руководство по нахождению надежных перестроек. Вершина (брейкпоинт) называется *простой*, если ее мультистепень равна 2. Путь (цикл) называется *хорошим*, если все внутренние вершины простые, и мультиребра, соединяющие эти вершины альтернируют между $\vec{T}$-консистентным мультицветом Q и его комплиментарным $\overline{Q}$ (на рис. 2.2b мультиребра (a^h, d^h), (d^h, e^t) и (e^t, b^t) формируют хороший путь с мультицветами альтернирующих между красным+желтым и зеленым+синим). Хороший цикл на $2m$ мультиребрах, альтернирущий между $\vec{T}$-консистентным мультицветом Q и его комплиментарным мультицветом $\overline{Q}$, трансформируется в m полных мультиребер с $m-1$ DCJ на мультицвете Q. Аналогично, хороший путь с m мультиребрами из $\vec{T}$-консистентных мультицветов Q сначала трансформируется с помощью DCJ на первом и последнем мультиребре мультицвета Q в хороший цикл. Затем он трансформируется в $m-1$ полные мультиребра с $m-2$ DCJ (см. рис 2.5).

ASMedian использует адекватные подграфы для декомпозиции брейкпоинт графа в методе ветвей и границ для решения проблемы медианы. Связный

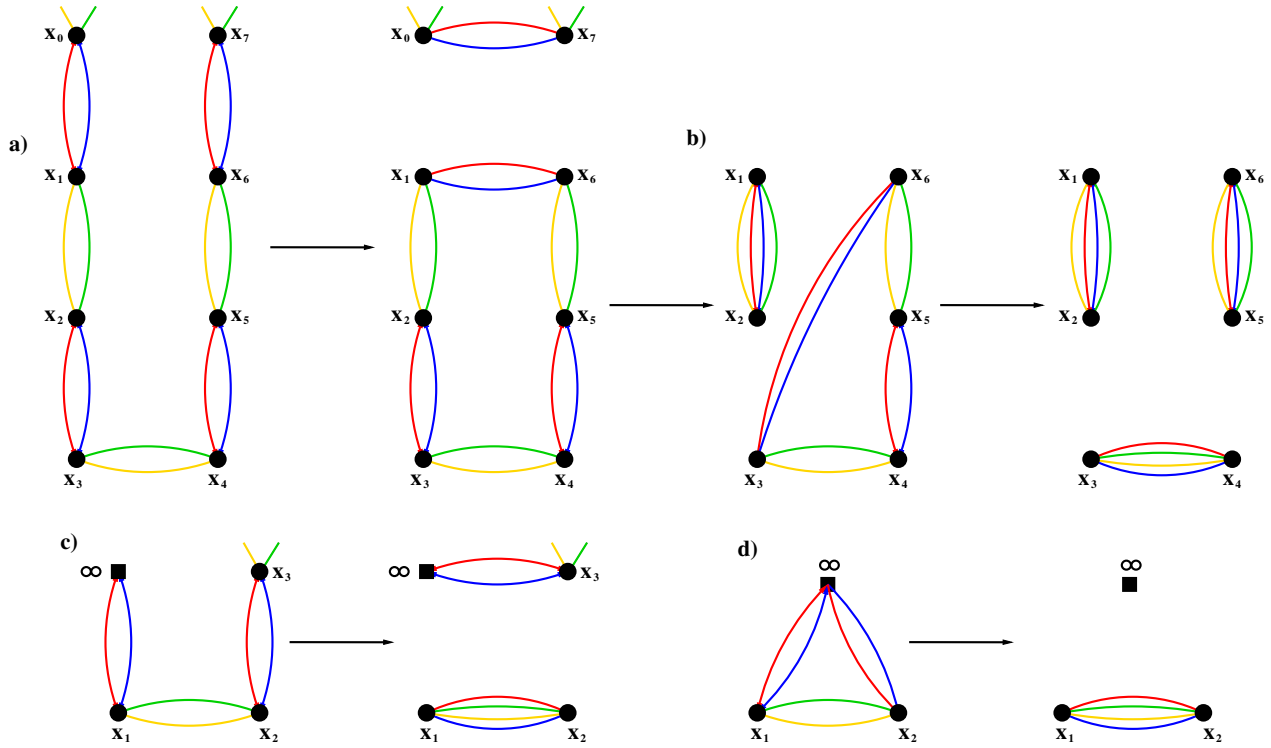

Рис. 2.5. Обработка хороших путей при использовании красно-синего $\vec{T}$-консистентного мультицвета. **a)** Хороший путь на вершинах $x_1,\ x_2,\ \cdots,\ x_6$ трансформируется в цикл над тем же множеством вершин с помощью DCJ над ребрами (x_0, x_1) и (x_6, x_7). **b)** Трансформация хорошего цикла на шести вершинах в полные мультиребра с помощью операции DCJ на мультиребрах (x_3, x_6) и (x_4, x_5), которая следует за DCJ на мультиребрах (x_1, x_6) и (x_3, x_4). **c)** DCJ на нерегуляном ребре, которая соответствует инверсии, включающей конец хромосомы. **d)** DCJ на двух нерегулярных ребрах, которая соответствует операции слияния.

подграф H размера m называется *адекватным (строго адекватным)* подграфом, если $c_{max}(H) > \frac{3m}{2}$ ($c_{max}(H) \geq \frac{3m}{2}$), где c_{max} – максимальное число альтернирующих циклов между ребрами медианного генома M и ребрами существующих геномов $G_1,\ G_2,\ G_3$. Одним из основных свойств адекватных подграфов является то, что ребра медианного генома лежат внутри данного подграфа. Список *простых* адекватных подграфов (не содержащих других адекватных подграфов) приведен на рис. 2.6. В большинстве приведенных подграфов ребра медианного генома изначально известны или существует небольшое множество вариантов, которое можно перебрать за небольшое время.

Как уже упоминалось ранее, каждая внутренняя вершина дерева с тремя ближайшими соседями представляет собой проблему медианы. Если в брейкпо-

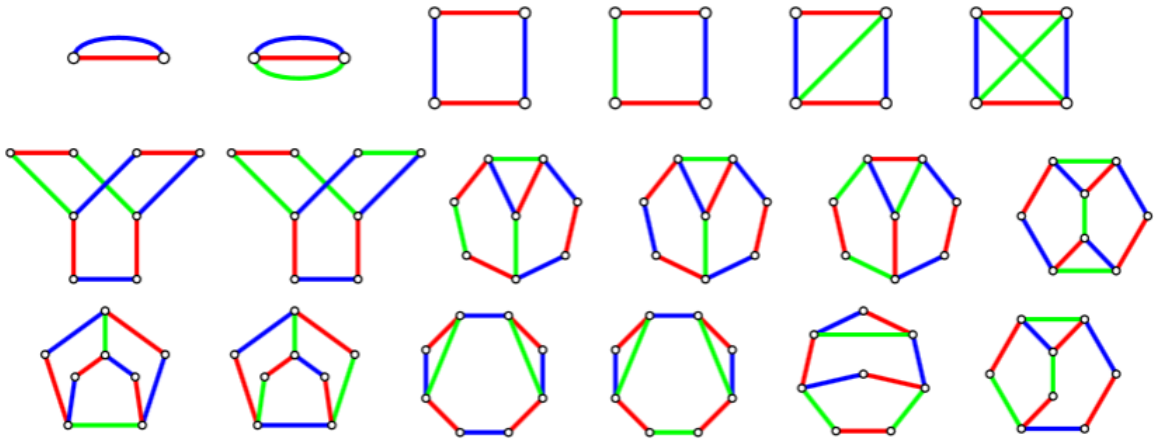

Рис. 2.6. Простые адекватные подграфы размера 1, 2 и 4 множественного брейкпоинт графа для трех геномов. Рисунок воспроизведен из [38]

инт графе встречается последовательность связанных адекватных подграфов размера один, то получается путь (цикл), который можно разрешать способом, реализованным в MGRA. Поэтому тройку T-консистентных мультицветов назовем *медианными*, если корни индуцированных поддеревьев на листьях данных мультицветов являются тремя ближайшими соседями какой-то внутренней вершины. Стоит отметить, что два мультицвета будут $\vec{T}$-консистентными мультицветами, а третий T-консистентным, но не $\vec{T}$-консистентным мультицветом. Назовем T-консистентными адекватными подграфами адекватные подграфы, состоящие из мультиребер, которые имеют мультицвета из медианной тройки T-консистентных мультицветов (на рисунке 2.6, пусть красный цвет соответствует какому-то T-консистентному мультицвету Q_1, синий – Q_2, зеленый – Q_3). Будем искать такие медианные адекватные подграфы и обрабатывать их аналогично тому, как это сделано в ASMedian и MGRA.

2.4.2. Обработка немобильных ребер, шаг 1

Рассмотрим трансформацию брейкпоинт графа в идентичный брейкпоинт граф с DCJ на $\vec{T}$-консистентных мультицветах как изменение мультицветов у мультиребер, инцидентных фиксированной вершине. Каждый DCJ в трансфор-

мации либо не влияет на такие мультицвета, либо объединяет два из них в один мультицвет. В последнем случае один из объединяемых мультицветов будет $\vec{T}$-консистентным, поскольку участвует в операции DCJ. Следующее определение характеризует, какие мультицвета могут потенциально включаться в такой сценарий. Множество непересекающихся мультицветов $\{Q_1, Q_2, \ldots, Q_m\}$ называется *соединяемым*, если они могут быть соединены в единственный мультицвет $Q_1 \cup Q_2 \cup \cdots \cup Q_m$ с помощью $m-1$ шага, где каждый шаг объединяет два мультицвета в один, причем хотя бы один из них является $\vec{T}$-консистентным.

Согласно принципу (P4) для поиска надежных перестроек, включающих мультиребро e, достаточно исследовать небольшую окрестность графа вокруг ребра e (в пределах путей длины 3 начинающихся на ребре e). Также по принципу (P1) и (P4) следует, что мультиребро (x, y) мультицвета Q двигается только в том случае, если Q – это $\vec{T}$-консистентный мультицвет, и существует мультиребро (z, t) мультицвета Q, и существует одно из мультиребер (x, z) или (y, z). Мультиребро (z, t) может быть не представлено непосредственно в текущем состоянии графа, но существует возможность его появления позже. Данное наблюдение приводит к следующему определению.

Мультиребро (x, y) мультицвета Q называется *мобильным*, если Q – $\vec{T}$-консистентный мультицвет, и существует вершина z смежная с x или y ($z \neq x,\ z \neq y$), с инцидентными мультиребрами мультицветов $S_1, S_2, \cdots, S_m$, таких что данное множество является соединяемым. Число таких вершин z назовем *мобильностью* мультиребра (x, y) (например, на рис. 2.2b мультиребро (d^h, e^t) зелено-синего мультицвета имеет мобильность два, поскольку у смежных вершин a^h и b^t могут сформироваться зелено-синие мультиребра). Другими словами, мобильность ребра (x, y) равна количеству мультиребер (существуюших или возможно появляюшихся позже), которые вместе с (x, y) формируют возможные операции DCJ. Мультиребро может иметь только не отрицательную мобильность. Ребро, имеющее нулевую мобильность, является *немобильным*.

Для дальнейшего исследования интерес представляют немобильные муль-

тиребра, поскольку они могут не двигаться с помощью DCJ, пока их мультицвет не будет обогащен до другого мультицвета (что может сделать ребро мобильным). В большинстве случаев немобильные мультиребра остаются такими через всю трансформацию и определяют позицию полного мультиребра в идентичном брейкпоит графе. Ясно, что каждое не $\vec{T}$-консистентное мультиребро немобильно, но некоторые $\vec{T}$-консистентные мультиребра также могут быть немобильными. Это расширение применимо в множестве алгоритмов MGRA2, которые опираются на факт, что такие мультиребра не могут двигаться.

Немобильное мультиребро (x, y) мультицвета Q называется *фиксированным*, если все мультиребра, инцидентные x и y, $\vec{T}$-консистентны, их набор мультицветов не является соединяемым, и существует взаимооднозначное соответствие между мультиребрами, инцидентными x и y, по инварианту соответствующего мультицвета. Такое фиксированное ребро может быть трансформировано в полное мультиребро с помощью нескольких DCJ на соответствующих $\vec{T}$-консистентных мультиребрах, инцидентных x и y.

Обработка фиксированного мультиребра может быть обобщена на случай, когда немобильное мультиребро (x, y) имеет подмножество инцидентных мультиребер x и y, являющихся $\vec{T}$-консистентыми и имеющих взаимооднозначное соответствие. В данном случае MGRA2 для определения DCJ руководствуется либо мобильностью мультиребер из этого подмножества, либо возможностью создания T-консистентного мультицвета на мультиребре (x, y). Например, пусть вышеописанное подмножество содержит мультиребра с мультицветами $Q_1, Q_2, \ldots, Q_t$. MGRA2 выполняет DCJ на мультиребрах мультицвета Q_i, только если мобильность обоих этих мультиребер равна одному (то есть, они представляют единственный вариант взаимодействия друг для друга). Если таких мультиребер не существует, MGRA2 пытается найти подмножество S из $\{Q_1, Q_2, \ldots, Q_t\}$, такое что объединение мультицветов их $S \cup Q$ T-консистентно. Если подмножество S не является соединяемым, алгоритм выполнит все DCJ на мультицветах S и превратит (x, y) в T-консистентное мультиребро.

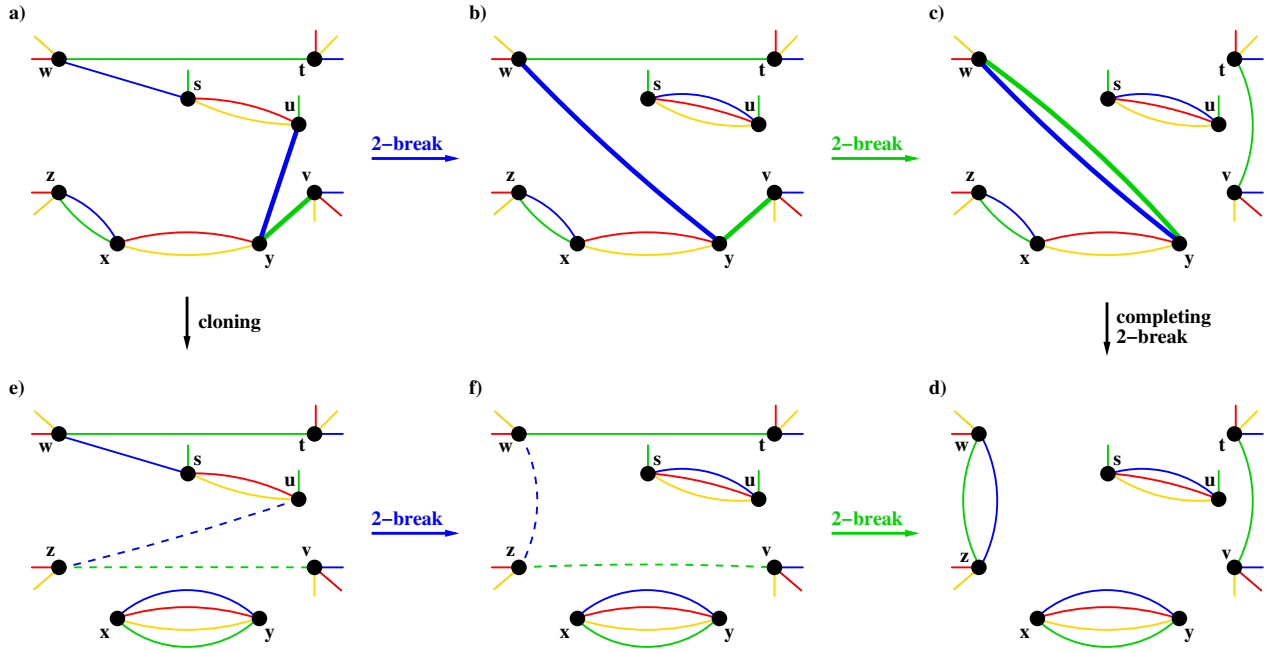

Рис. 2.7. Две эквивалентные трансформации на брейкпоинт графе, построенном на красном, зеленом, синем и желтом геноме. Каждый одиночный цвет и сине-зеленый мультицвет являются $\vec{T}$-консистентными, в то время как красно-желтый цвет не является. **a)** Мультиребра (z, x), (x, y), (y, u) и (y, v) формируют вилку, чьи зубчики нарисованы как жирные ребра. **b-c)** Графы, появляющиеся в сценарии из двух DCJ, которые делают зубчики вилки параллельными, создавая простой путь на вершинах z, x, y, w. **d)** Граф после завершающего DCJ, который из пути создает полное мультиребро (x, y). **e)** Граф после клонинга, в котором зубчики вилки переходят в клонированные ребра (z, u) и (z, v) и создается полное мультиребро (x, y), **f-d)** Графы, появляющиеся в сценарии из двух DCJ, включающие в себя клонированные ребра, эквивалентные серии DCJ на регулярных ребрах в сценарии b-c. Клонированные ребра становятся параллельными и формируют регулярное мультиребро (z, w).

2.4.3. Обработка немобильных ребер, шаг 2

Существует еще один общий объект, наблюдаемый в брейкпоинт графе, который получил название *вилка*. Данный паттерн не был представлен в алгоритме MGRA.

Вилка формируется с помощью *центрального* немобильного мультиребра (x, y), где x имеет мультистепень 2 и y имеет мультистепень 3, вместе с тремя смежными мультиребрами: *зубчиками* (y, u), (y, v) и *ручкой* (x, z) из $\vec{T}$-консистентных мультицветов Q_1, Q_2 и $Q = Q_1 \cup Q_2$ соответственно (см. рисунок 2.7a).

Поскольку центральное ребро не мобильно, в любой трансформации оно

остается на том же месте, в то время как смежные мультиребра в конечном итоге становятся параллельны ему, и, таким образом, все вместе формируют полное мультиребро. Другими словами, вилка всегда является субъектом в следующем сценарии: после некоторого числа DCJ зубчики сходятся (то есть становятся параллельными) и формируют мультиребро (y, w) из мультицвета Q (см. рис. 2.7c), которое получается после *завершающего DCJ* на этом мультиребре и ручке (x, z), формирующих полное мультиребро (x, y) (см. рис. 2.7d). К сожалению, существует множество сценариев для того, чтобы зубчики вилки соединились, и не возможно непосредственно определить самый надежный из набора сценариев.

В MGRA2 предложен новый подход, названный *клонингом*, который создает полное мультиребро (x, y) заранее и позволяет зубчикам (y, u) и (y, v) соединяться после данного события. Точнее, клонинг заменяет зубчики на *клонированные* мультиребра (z, u) и (z, v) с мультицветами Q_1 и Q_2 соответственно (см. рис. 2.7e). Любые DCJ, включающие клонированные мультиребра, соответствуют DCJ с оригинальными мультиребрами, и наоборот (например, DCJ в рисунке 2.7a,b эквивалентны DCJ на рисунке 2.7d,e). Поэтому клонинг и следующая за ним серия DCJ на клонированных мультиребрах до момента их соединения в результирующем графе H, соответствует серии DCJ на оригинальном брейкпоинт графе (перед клонингом) и результирующему DCJ, после которого получаем итоговый граф H (например, сценарий на рис. 2.7a,e,f,d с клонингом и сценарий на рис. 2.7a,b,c,d без клонинга эквивалентны).

MGRA2 также обрабатывает обобщенные вилки, имеющие больше чем два зубчика, а также поддерживает события телоскопического клонинга (события клонинга идут один за другим). Стоит заметить, что операция клонинга – это обычный отложенный результирующий DCJ, следовательно, можно применить большинство правил, изложенных в предыдущей секции про обработку немобильных ребер. Приведем применяемые правила.

Если мультиребро (x, y) немобильное с мультицветом Q, то:

1. Если все смежные ребра мобильны и все мультицвета «ручек» не могут скомбинироваться в $\vec{T}$-консистентный мультицвет, тогда выполняем все операции и создаем полное мультиребро.

2. Если некоторые смежные ребра не мобильны, пусть Z - минимальный T-консистентный мультицвет, который включает Q. Если смежные мобильные ребра могут сформировать мультицвет Z и не могут соединиться друг с другом, то выполняем соответствующие операции.

2.4.4. Увеличение компонент связности

Данный шаг является обобщением жадного алгоритма поиска перестроечного сценария между двумя геномами. Напомним, что для случая двух геномов жадный алгоритм ищет такую операцию DCJ, которая увеличивает число циклов на один. В случае, когда все хромосомы циклические, увеличение числа циклов соответствует увелечению числа компонент связности графа.

Другими словами, ясно, что каждый DCJ может увеличивать число компонент связности не более чем на один. Тогда кратчайший сценарий трансформации должен полностью состоять из операций, каждая из которых увеличивает количество компонент связности [27]. Кроме того, кратчайший сценарий может быть получен повторяющимся нанесением произвольных разрывов такого вида.

Поскольку идентичный множественный брейкпоинт граф обладает максимальным числом компонент связности, то трансформацию из множественного брейкпоинт графа в идентичный брейкпоинт граф можно рассматривать как процесс увеличения числа компонент связности. Тогда в ходе поиска кратчайшего расстояния логично искать DCJ, которые увеличивают количество копонент связности.

Для нахождения таких DCJ, обладающих $\vec{T}$-консистентным мультицветом Q, строим компоненты связности на мультиребрах графа всех мультицветов, исключая Q, и смотрим на мосты мультицвета Q между этими компонентами.

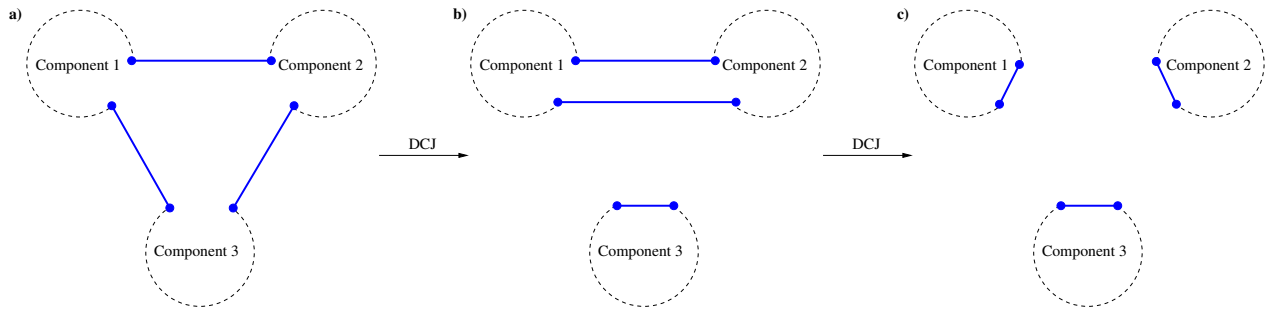

Рис. 2.8. Одна из трех возможных трансформаций брейкпоинт графа над синими ребрами, которые являются мостами графа. Все три трансформации приводят к одному и тому же результату. **a)** Брейкпоинт граф, содержащий три компоненты связности, которые соединены мостами синего цвета. **b)** Промежуточный граф, полученный с помощью DCJ на синих ребрах. **c)** Граф, получаемый после обработки всех мостов.

Если одна из компонент соединена двумя мостами с остальным графом, то выполняем DCJ над этими мостами, увеличивая число компонент связности в исходном графе на всех мультицветах (включая Q). Простейший пример работы данного шага показан на рисунке 2.8.

Также была придумана эвристика, расширяющая применение данного шага на случай, если одна из компонент соединена одним мостом мультицвета Q с остальным графом. Основная идея данной эвристики: обработка нерегулярных ребер (соединяющих вершину ∞ с вершиной графа) мультицвета Q, исходящих из данной компоненты. Если имеется только одно нерегулярное ребро мультицвета Q, то выполним DCJ над мостом и данным ребром. В том случае, когда нерегулярных ребер мультицвета Q несколько, то в соответствие с принципом работы (P4), описанным выше, выбираем те нерегулярные ребра, которые связаны с мостом каким-либо ребром. Далее, для каждого такого связующего ребра, обладающего некоторым мультицветом Z, проверяем, что объединение мультицветов $Q \cup Z$ является T-консистентным мультицветом. Если данное условие выполняется, то совершаем DCJ над мостом и нерегулярным ребром, которые связаны данным связующим ребром. Условие Теоремы 5 гарантирует, что такое связующее ребро всего одно.

В случае, когда компонента связана одним мостом мультицвета Q и не

существует исходящих нерегулярных ребер этого мультицвета, то выполняем операцию разделения над данным мостом.

Во всех трех случаях, описанных в эвристическом расширении данного шага, после операции DCJ происходит увеличение числа компонент связности в исходном множественном брейкпоинт графе на всех мультицветах (включая Q).

2.5. Работа с брейкпоинт-переиспользованием

В ходе эволюции на геномах может возникать так называемое брейкпоинт-переиспользование, которое означает, что один и тот же участок генома подвергался структурным изменениям нсколько раз. Результатом брейкпоинт-переиспользования во множественном брейкпоинт графе является появление сложных брейкпоинтов (мультистепень вершины > 2) и мультиребр не T-консистентного мультицвета. Все предыдущие шаги, базирующиеся на определение мобильности мультиребер и клонинге, позволяют обрабатывать сложные брейкпоинты, но оставшиеся ребра не T-консистентного цвета, которые остались после выполнения всех шагов, требуют специального подхода.

Введем правило, что на последних шагах MGRA2 рассматривает кажое мультиребро не T-консистентного мультицвета Q не как единственное мультиребро, а как коллекцию параллельных подмультиребер, имеющих не пересекающиеся T-консистентные мультицвета $Q_1, Q_2, \ldots, Q_m$, такие что $Q = Q_1 \cup \cdots \cup Q_m$ и m минимально. Теорема 2 гарантирует, что такое представление единственно.

Когда каждое не T-консистентное мультиребро представлено как набор параллельных T-консистентных подмультиребер (некоторые из них являются $\vec{T}$-консистентными согласно Теореме 5), тогда брейкпоинт граф может быть рассмотрен как граф без не T-консистентных мультиребер. MGRA2 выполняет все вышеописанные алгоритмы по поиску надежных DCJ, что позволяет значительно приблизить граф к идентичному брейкпоинт графу (в большинстве

случаев можем получить идентичный брейпоинт граф). При этом на данной стадии DCJ могут оперировать на T-консистентных подмультиребрах.

В результате, в MGRA2, помимо вышеописанных шагов, были введены три раунда, которые позволяют пользователю контролировать обработку брейкпоинт-переипсользования.

(Round 1) Не T-консистентные мультиребра не разделяются на парралельные мультиребра.

(Round 2) Не T-консистентные мультиребра разделяются на коллекцию параллельных мультиребер только в случае. если непересекающееся разделение имеет значение $m = 2$.

(Round 3) Разделяются все не T-консистентные мультиребра на коллекцию параллельных мультиребер независимо от размера данной коллекции.

2.6. Метод грубой силы

Метод грубой силы представляет собой альтернативный подход для нахождения трансформации в идентичный брейкпоинт граф. Хотя он и был разработан для компонент множественного брейкпоинт графа, которые оставались после всех алгоритмов, изложенных выше, данный метод можно применить к исходному брейкпоинт графу. Как показали эксперименты, метод грубой силы, примененный к брейкпоинт графу без других алгоритмов, дает очень низкие результаты. С другой стороны, применение данного подхода на оставшемся брейкпоинт графе после вышеизложенных алгоритмов дает высокие результаты.

Идея метода базируется на использовании алгоритма Blossom V [60], который ищет парасочетание минимального веса в произвольном графе. Напомним, что цель MGRA2– найти трансформацию кратчайшей длины до идентичного брейкпоинт графа. Данную цель можно переформулировать следующим образом: найти такое паросочетание в исходном графе, чтобы трансформация до

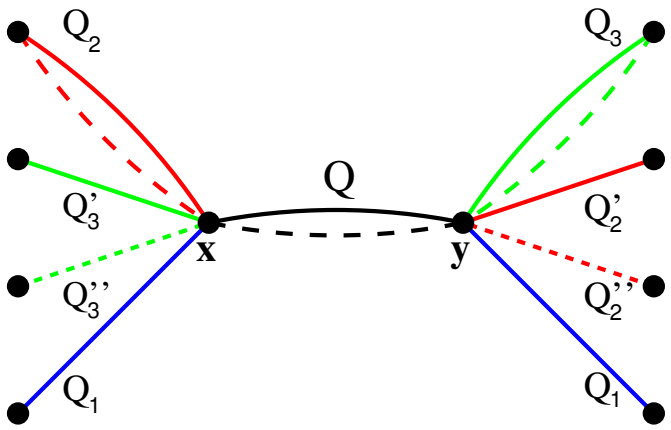

Рис. 2.9. Мультиребро (x, y) с мультицветом Q, имеющее перестроечную оценку, равную 5. Все цвета, за исключением Q, являются $\vec{T}$-консистентными. Ребра формируют три класса эквивалентности: $Q_1 = Q_1$, $Q_2 = Q'_2 \cup Q''_2$ и $Q_3 = Q'_3 \cup Q''_3$. Подсчитывая количество классов эквивалентности (три класса) и количество объединений в этих классах (два объединения), получаем перестроечную оценку равную пяти (сумма классов и объединений).

его идентичного графа имела наименьшую длину. Если бы на каждом ребре было написано точное число необходимых операций для конвертации его в полное мультиребро, то нахождение паросочетания минимального веса давало бы ответ на поставленную цель.

Пользуясь вышеизложенным наблюдением, припишем каждому мультиребру брейкпоинт графа *перестроечную* оценку, которая будет давать информацию о количестве операций, необходимых для превращения ребра в полное мультиребро, и запустим Blossom V. Получив паросочетание минимального веса, начнем выполнять любую трансформацию, которая приведет к полным мультиребрам в брейкпоинт графе. Стоит отметить, что чем ближе к теоретическому результату перестроечная оценка, тем точнее будут получаться итоговые результаты.

В MGRA2 была предложена следующая перестроечная оценка. Для мультиребра (x, y) мультицвета Q обозначим все инцидентные к x мультиребра как $\{L_1,\ L_2,\ \cdots\ L_k\}$ и все инцидентные к y мультирбера как $\{R_1,\ R_2,\ \cdots,\ R_t\}$. Все инцидентные мультиребра имеют $\vec{T}$-консистентный мультицвет. В случае, если инцидентное мультиребро не T-консистентно, то по Теореме 2 разбиваем

его на параллельные мультиребра $\vec{T}$-консистентных мультицветов. Для обоих множеств построим классы эквивалентности по инварианту пересечения мультицветов. Так как все мультицвета $\vec{T}$-консистентны, то по Лемме 1 получаем либо $L_i \subseteq R_j$, либо $R_j \subseteq L_i$. Значит, в каждом классе эквивалентности будет представитель – один мультицвет, который включает в себя все остальные. Другим свойством представителя класса эквивалентности является то, что все другие мультицвета в данном классе инцидентны другой вершине. Другими словами, два множества мультицветов разбиваются на классы двух видов: либо на $L_i = R_{j_1} \cup \cdots \cup R_{j_n}$, либо на $R_j = L_{i_1} \cup \cdots \cup L_{i_m}$. Значением оценки будет количество таких классов эквивалентности и суммарное количество знаков объединения для каждого класса. На рисунке 2.9 приведен пример вычисления данной оценки.

Предложенная оценка имеет простую интерпретацию. Для создания полного мультиребра необходимо совершить DCJ над каждым инцидентным $\vec{T}$-консистентным мультицветом (количество классов эквивалентности в оценке). Если у одной из вершин такой мультицвет отсутствует, то перед тем, как выполнить операцию над таким мультицветом, необходимо собрать отсутствующий мультицвет у вершины (количество операций объединения в классе эквивалентности). Эта оценка имеет недостатки, так как не учитывает изменения брейкпоинт графа при выполнении операций, и не точно учитывает количество DCJ для создания отсутствующих мультицветов у вершины.

Глава 3

Линеаризация медианного генома

3.1. Построение решения

Предположим, что дано три линейных генома G_1, G_2, G_3 и их медианный геном M, а также кратчайшая трансформация $M \xrightarrow{T_i} G_i$ $(i = 1, 2, 3)$. Алгоритм, описанный ниже, модифицирует одну из таких трансформаций, скажем, T_1, и производит трансформацию $T_1{}'$, сформированную следующим образом:

$$M \xrightarrow{t_0} M' \xrightarrow{t_1} G_1,$$

где M' – линейный геном, $|t_0| = c(M)$ (то есть каждый DCJ t_0 уменьшает число циклических хромосом) и $|t_0| + |t_1| \geq |T_1|$ (см. рис. 3.1).

В оставшейся части главы события вставок и удалений будут представлены аналогично описанию в параграфе 2.3. В брейкпоинт графе такие события выглядят как DCJ над ребрами, где хотя бы одно из них является «протезным». В геномном графе, если DCJ имеет только одно «протезное» ребро, данное событие выглядит как удаление или вставка одного гена из генома. В случае, если DCJ имеет два «протезных» ребра, то в геномном графе данное событие выглядит как удаление или вставка двух генов из генома.

Легко заметить, что события удаления генов могут уменьшать количество циклических хромосом, а события вставки генов могут увеличивать количество циклических хромосом. Поэтому первым шагом является разделение трансформации на серию идущих подряд событий удалений, обычных перестроечных операций и серию событий вставок. Существует следующая теорема:

Теорема 6 ([61]). *Пусть P и Q геномы без дуплицированных генов и $P \xrightarrow{z} Q$ трансформация между ними. Тогда существует эквивалентная трансформация следующего вида $P \xrightarrow{z_{ins}} \bar{P} \xrightarrow{z'} \bar{\bar{P}} \xrightarrow{z_{del}} Q$, такая что z_{ins} состоит*

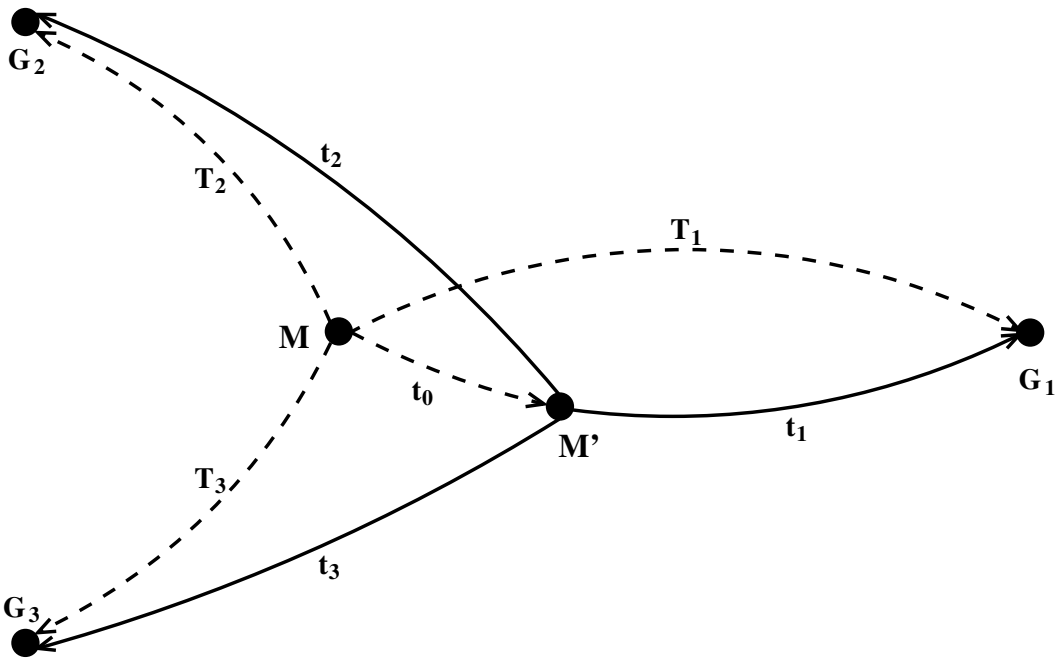

Рис. 3.1. Линейные геномы G_1, G_2, G_3 и их медианный геном M, представленные как вершины, и соответствующие кратчайшие трансформации T_1, T_2, T_3, нарисованные как направленые пунктирные ребра. При условии наличия в геноме M циклических хромосом строим другую кратчайшую трансформацию из M в G_1 (почти эквивалентную T_1), состоящую из трансформаций t_0 и t_1, таких что результатом t_0 является геном M' и $|t_0| = c(M)$. Соответствующие кратчайшие трансформации из M' в G_2 и G_3 представлены как жирные направленные ребра и обозначены как t_2 и t_3.

только из событий вставок, z_{del} состоит только из событий удалений и $|z| = |z_{ins}| + |z'| + |z_{del}|$.

По аналогии с выше изложенной Теоремой 6 можем получить следующую трансформацию:

$$P \xrightarrow{z_{del}} \bar{P} \xrightarrow{z'} \bar{\bar{P}} \xrightarrow{z_{ins}} Q$$

где z_{del} состоит только из событий удалений, z_{ins} состоит только из событий вставок и $|z| = |z_{del}| + |z'| + |z_{ins}|$. Применяя алгоритм, который описан в параграфе 3.2, из трансформации z_{del} получим трансформацию $P \xrightarrow{z^0_{del}} P' \xrightarrow{z_{del}} \bar{P}$, где $c(P') = c(\bar{P})$ и $|z^0_{del}| = c(P) - c(\bar{P})$. Воспользовавшись алгоритмом, который изложен в параграфе 3.3, из трансформации z' получим трансформацию $\bar{P} \xrightarrow{z'_0} \bar{P}' \xrightarrow{z'_1} \bar{\bar{P}}$, где $c(\bar{P}') = c(\bar{\bar{P}})$ и $|z'_0| = c(\bar{P}) - c(\bar{\bar{P}})$. В результате этих

действий имеем:

$$P \xrightarrow{z^0_{del}} P' \xrightarrow{z_{del}} \bar{P} \xrightarrow{z'_0} \bar{P}' \xrightarrow{z'_1} \bar{\bar{P}} \xrightarrow{z_{ins}} Q$$

где $c(P') = c(\bar{P})$ и $c(\bar{P}') = c(\bar{\bar{P}})$. Так как события удаления могут быть представлены как события вставки в противоположном направлении, модифицируем геном $\bar{P}'$ вставкой соответствующих генов для получения итоговой трансформации:

$$P \xrightarrow{z^0_{del}} P' \xrightarrow{z'_0} \bar{P}'' \xrightarrow{z_{del}} \bar{P}''' \xrightarrow{z'_1} \bar{\bar{P}} \xrightarrow{z_{ins}} Q$$

где $c(\bar{P}'') = c(Q) = 0$, другими словами $\bar{P}''$ эквивалентен геному M' и является линейным геномом.

В оставшейся части главы показывается, как получить линеаризующие трансформации для каждой серии событий.

3.2. Линеаризация истории, состоящей из событий удаления

Как было сказано в предыдущем параграфе, события удаления, чье представление описано в секции 2.3, могут удалять один или два гена из генома. Стоит отметить, что данная модель налагает ограничение на появление событий удаления двух генов. Данная операция может произойти только на циклической хромосоме, имеющей длину, равную двум генам. Таким образом, циклическая хромосома длины n генов будет удалятся за $n - 1$ событие удаления генов.

Над циклической хромосомой, которая удаляется в трансформации, состоящей из событий удалений, выполним операцию разделения, превратив хромосому в линейную. Данное событие разделения поместим в новую трансформацию z^0_{del}. Выполнив операции разделения над каждой удаляемой циклической хромосомой, получим трансформацию z^0_{del}

3.3. Линеаризация истории, состоящей из перестроечных событий

Алгоритм линеаризации для истории, состоящей из классических перестроек, следует из теоремы:

Теорема 7. *Пусть* $P \xrightarrow{z} Q$ *трансформация между геномами* P *и* Q*, при условии* $c(P) > c(Q)$*. Тогда существует трансформация* $P \xrightarrow{r} P' \xrightarrow{z'} Q$*, такая что* r *– единственная DCJ,* $c(P') = c(P) - 1$ *и* $|z'| = |z| - 1$*.*

Доказательство. Пусть $P \xrightarrow{z} Q$ трансформация между геномами P и Q, при условии $c(P) > c(Q)$. Предположим, что $z = (q_1, q_2, \ldots, q_n)$, где q_i DCJ, то есть z имеет форму: $P = P_0 \xrightarrow{q_1} P_1 \xrightarrow{q_2} \cdots \xrightarrow{q_n} P_n = Q$. Для доказательства теоремы воспользуемся индукцией по $n = |z|$.

Если $|z| = 1$, тогда теорема тривиальным образом выполнена.

Предположим, что теорема выполнена для всех $|z| \leq n$.

Пусть $z = (q_1, q_2, \ldots, q_n, q_{n+1})$ трансформация между геномами P и Q при условии $c(P) > c(Q)$. Рассмотрим следующие случаи:

1. Если $c(P_0) > c(P_1)$, тогда скажем, что $r = q_1$, $P' = P_1$ и $z' = (q_2, q_3, \ldots, q_n, q_{n+1})$.

2. Если $c(P_0) = c(P_1)$, тогда $c(P_1) > c(P_{n+1})$. По индукции, получим трансформацию z_1, сформированную таким образом, что $P_0 \xrightarrow{q_1} P_1 \xrightarrow{q_2'} P_2' \xrightarrow{y} Q$, где y – трансформация с длиной $|y| = n-1$ и $c(P_1) > c(P_2')$. Используя Теорему 11 или 12, которые описаны ниже, получаем новую трансформацию z_2 из z_1 с помощью замены пары из DCJ $P_0 \xrightarrow{q_1} P_1 \xrightarrow{q_2'} P_2'$ на эквивалентную пару из DCJ $P_0 \xrightarrow{q_1'} P_1' \xrightarrow{q_2''} P_2''$, таких что $c(P_0) > c(P_1')$ в z_2. Теперь обозначим за $r = q_1'$, $P' = P_1'$ и $z' = (q_2'', y)$.

3. Если $c(P_0) < c(P_1)$, то, каждая DCJ может изменять число циклических хромосом не более чем на 1, имеем $n > 2$ и $c(P_1) - 1 > c(Q)$.

По индукции получим трансформацию z_1, сформированную таким образом, что $P_0 \xrightarrow{q_1} P_1 \xrightarrow{q_2'} P_2' \xrightarrow{y_1} Q$, где y_1 – трансформация с длиной $|y_1| = n - 1$ и $c(P_1) > c(P_2') = c(P_1) - 1 > c(Q)$. Аналогично, по индукции относительно y_1 получим трансформацию z_2, сформированную как $P_0 \xrightarrow{q_1} P_1 \xrightarrow{q_2'} P_2' \xrightarrow{q_3'} P_3' \xrightarrow{y_2} Q$, где y_2 – трансформация с $|y_2| = n - 2$ и $c(P_1) > c(P_2') > c(P_3')$. Используя Теорему 16 или 17, которые описаны ниже, получаем новую трансформацию z_3, сформированную следующим образом $P_0 \xrightarrow{q_1''} P_1'' \xrightarrow{q_2''} P_2'' \xrightarrow{q_3''} P_3' \xrightarrow{y_2} Q$, такую что $c(P_0) > c(P_1')$ в z_3. Теперь предположим, что $r = q_1'$, $P' = P_1'$ и $z' = (q_2'', q_3'', y_2)$.

□

Применяя Теорему 7 $c(P) - c(Q)$ раз, получаем следующее следствие.

Следствие 8. *Пусть $P \xrightarrow{z} Q$ – трансформация между геномами P и Q с $c(P) > c(Q)$. Тогда существует трансформация $P \xrightarrow{r_1} P' \xrightarrow{r_2} P'' \xrightarrow{r_3} \dots \xrightarrow{r_k} P^{(k)} \to Q$ той же длины $|z|$, где $k = c(P) - c(Q)$, $r_1, r_2, \dots, r_k$ DCJ, и $c(P^{(k)}) = c(Q)$.*

Применяя данное следствие для $z = T_1$ (с $P = M$ и $Q = G_1$), получим $M' = P^{(k)}$ с $c(M') = c(G_1) = 0$ хромосомами (то есть M' – линейный геном).

Прежде чем перейти ко всем теоремам, упомянутых в предыдущем доказательстве, приведем ряд важнейших определений, которыми в дальнейшем будем пользоваться.

В оставшейся части данной главы рассматриваем DCJ как операцию, удаляющую и создающую ребра в геномном графе (представление генома в виде графа). Две последовательные DCJ α и β в трансформации называются *независимыми*, если β удаляет ребра, которые не были созданы DCJ α. С другой стороны, когда β удаляет ребро(а), созданные с помощью α, скажем, что β *зависит* от α.

Предположим, что DCJ β зависит от DCJ α. Пусть $k \in \{1, 2\}$ – это число ребер, создаваемых с помощью α и удаляемых β. Тогда β *строго зависит* от

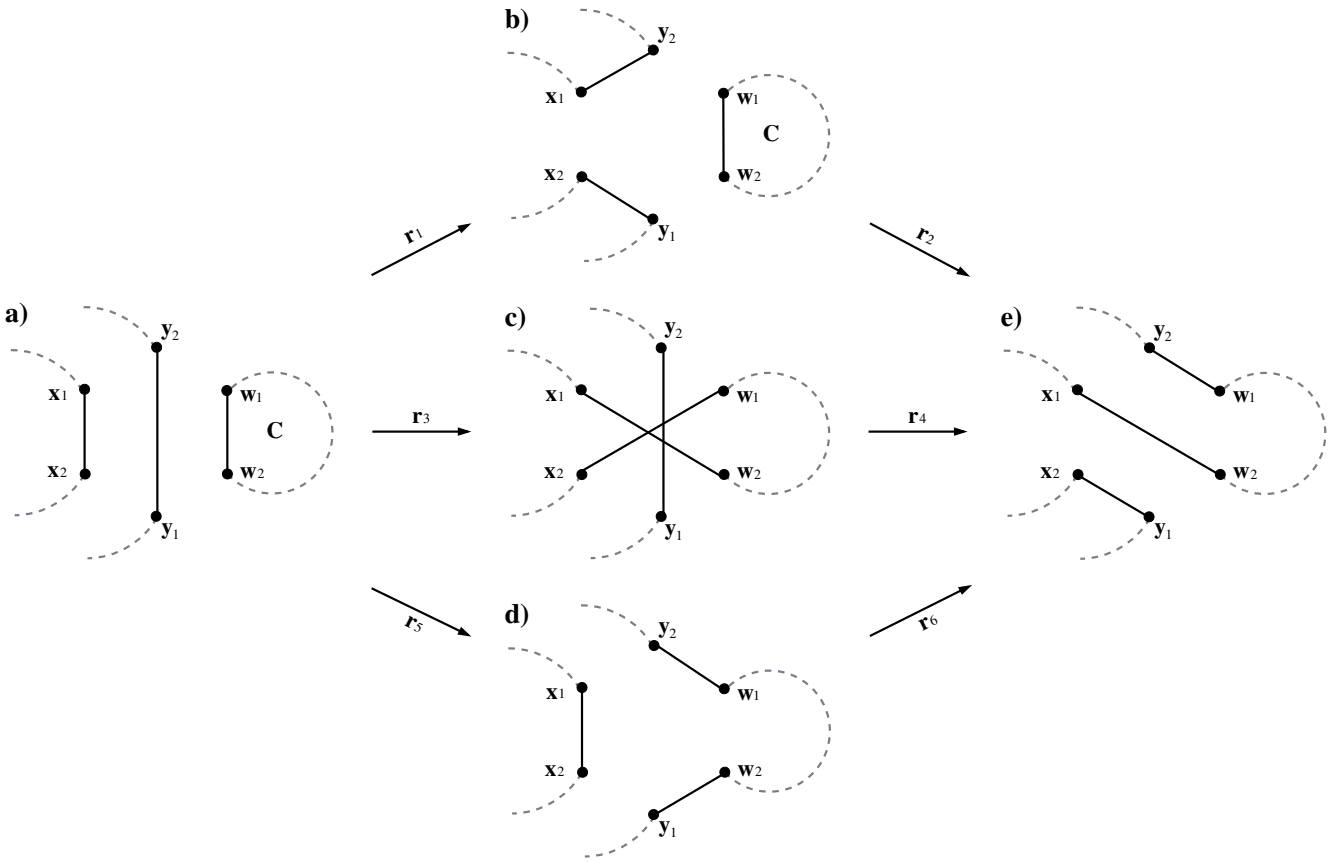

Рис. 3.2. Иллюстрация к Лемме 9. **a)** Начальный геномный граф, где пунктирные ребра представляют генную последовательность. Пунктирные ребра и ненаправленные черные ребра между w_1 и w_2 формируют циклическую хромосому C. **b-d)** Промежуточный геномный граф после первого DCJ из трех эквивалентных пар слабо зависимых DCJ. **e)** Результирующий геномный граф после эквивалентной пары DCJ, где хромосома C разрушена. Во всех случаях C разрушена с помощью операций r_2, r_3, и r_5.

α, если $k = 2$, и *слабо зависит* от α, если $k = 1$. Отметим, что пара последовательных строго зависимых DCJ не может возникнуть в кратчайшем сценарии между геномами, поскольку такая пара может быть заменена на эквивалентную единственную DCJ, которая уменьшает длину сценария.

В геномном графе пара зависимых DCJ заменяет три ребра на другие три над одним и тем же множеством из шести вершин (как упоминалось ранее, данная операция также известна как 3-break [27]). Легко увидеть, что для каждой пары слабо зависимых DCJ существует ровно две других эквивалентных пары из слабо зависимых DCJ (см. рис. 3.2).

Лемма 9. *Тройка ребер* $\{(x_1, x_2),\ (y_1, y_2),\ (w_1, w_2)\}$ *может быть трансфор-*

мирована в тройку ребер $\{(x_1, w_2),\ (y_1, x_2),\ (w_1, y_2)\}$ *с помощью двух слабо зависимых DCJ тремя разными путями:*

1. $\{(x_1, x_2), (y_1, y_2), (w_1, w_2)\} \xrightarrow{r_1} \{(x_1, y_2), (y_1, x_2), (w_1, w_2)\} \xrightarrow{r_2}$ $\{(x_1, w_2), (y_1, x_2), (w_1, y_2)\}$;

2. $\{(x_1, x_2), (y_1, y_2), (w_1, w_2)\} \xrightarrow{r_3} \{(x_1, w_2), (y_1, y_2), (w_1, x_2)\} \xrightarrow{r_4}$ $\{(x_1, w_2), (y_1, x_2), (w_1, y_2)\}$;

3. $\{(x_1, x_2), (y_1, y_2), (w_1, w_2)\} \xrightarrow{r_5} \{(x_1, x_2), (y_1, w_2), (w_1, y_2)\} \xrightarrow{r_6}$ $\{(x_1, w_2), (y_1, x_2), (w_1, y_2)\}$.

Для дальнейших доказательств потребуется следующая важная лемма.

Лемма 10. *Пусть* ϑ *– DCJ, которая трансформирует геном* P *в геном* Q. $c(P) > c(Q)$ *тогда и только тогда, когда два ребра, удаляемых с помощью* ϑ *в* P*, принадлежат различным хромосомам, и хотя бы одна из них циклическая.*

Доказательство. Если $c(P) > c(Q)$, то ϑ должна либо разрушать одну циклическую хромосому в геномном графе P, либо соединять две циклические хромосомы в одну новую. В обоих случаях два ребра, удаляемые ϑ, должны принадлежать различным хромосомам в P, и хотя бы одна из них циклическая.

Если два ребра, удаляемые с помощью ϑ в P, принадлежат различным хромосомам, одна из которых циклическая, тогда ϑ разрушает эту циклическую хромосому. В другом случае, если $c(Q) < c(P)$ и ϑ создает новую циклическую хромосому в Q, то ϑ также должна разрушить другую циклическую хромосому в P (то есть ϑ – это операция слияния на циклических хромосомах) и тогда следует, что $c(Q) < c(P)$. □

3.3.1. Случаи, которые были рассмотрены в статье

Теорема 11. *Пусть* $P \xrightarrow{\vartheta_1} Q \xrightarrow{\vartheta_2} R$ *– трансформация между геномами* P, Q, R*, такая что* ϑ_1, ϑ_2 *– независимые DCJ, и* $c(P) \geq c(Q) > c(R)$. *Тогда*

в трансформации $P \xrightarrow{\vartheta_2} Q' \xrightarrow{\vartheta_1} R$ *имеем* $c(P) > c(Q')$.

Доказательство. Пусть a и b – ребра, удаляемые с помощью ϑ_2. Поскольку ϑ_1 и ϑ_2 независимы, ребра a и b представлены в обоих геномах P и Q. Из Лемме 10 следует, что если $c(Q) > c(R)$, то в Q одно ребро, скажем, a, принадлежит циклической хромосоме, которая не содержит b.

Предположим, что a принадлежит циклической хромосоме C в Q. Геном P может быть получен из генома Q с помощью DCJ ϑ_1^{-1}, которая является обратной к ϑ_1 (то есть ϑ_1^{-1} удаляет ребра, создаваемые в ϑ_1, и создает ребра, удаляемые в ϑ_1). Рассмотрим два случая в зависимости от выполнения условия $c(P) > c(Q)$ или $c(P) = c(Q)$.

Если $c(P) > c(Q)$, то ϑ_1^{-1} должно либо разделить одну циклическую хромосому в Q на две, либо создать новую циклическую хромосому из линейной хромосомы. В первом случае ребро a принадлежит циклической хромосоме C' в P, даже если ϑ_1^{-1} разделяет хромосому C. Множество вершин хромосомы C' является подмножеством вершин хромосомы C и, таким образом, не содержит b. Во втором случае C не изменяется, и ребро b лежит вне хромосомы.

Если $c(P) = c(Q)$, то ϑ_1^{-1} либо не влияет на C, либо оставляет хромосому C циклической после разворота сегмента.

В обоих случаях получаем, что a принадлежит циклической хромосоме в P, и эта хромосома не содержит b. По Лемме 10 ϑ_2 будет уменьшать число циклических хромосом в P, то есть $c(P) > c(Q')$. □

Теорема 12. *Пусть* $P \xrightarrow{\vartheta_1} Q \xrightarrow{\vartheta_2} R$ *– трансформация между геномами* P, Q, R, *такая что* ϑ_2 *зависит от* ϑ_1 *и* $c(P) \geq c(Q) > c(R)$. *Тогда существует трансформация:* $P \xrightarrow{\vartheta_3} Q' \xrightarrow{\vartheta_4} R$, *где* ϑ_3 *и* ϑ_4 *– DCJ и* $c(P) > c(Q')$.

Доказательство. Если DCJ ϑ_2 строго зависит от ϑ_1, тогда $(\vartheta_1, \vartheta_2)$ эквивалентно одной DCJ ϑ' между геномами P и R. Для завершения доказательства в данном случае скажем, что $\vartheta_3 = \vartheta'$ и ϑ_4 – любые идентичные DCJ (которые удаляют и добавляют одинаковые ребра, таким образом не меняя геном) и $Q' = R$.

Чтобы закончить доказательство данной теоремы предположим, что ϑ_2 слабо зависима от ϑ_1.

По Лемме 9 для пары DCJ $(\vartheta_1, \vartheta_2)$ существует другая эквивалентная пара $(\vartheta_3, \vartheta_4)$, которая также трансформирует P в R. Пусть Q' – полученный геном после применения ϑ_3 к P. Используя Лемму 10 для доказательства $c(P) > c(Q')$, покажем что два ребра, удаляемые ϑ_3, принадлежат различным хромосомам, одна из которых циклическая.

Пусть a, b – ребра, удаляемые с помощью ϑ_1 в P, и c, d – ребра, удаляемые с помощью ϑ_2 в Q. Поскольку $c(Q) > c(R)$, то по Лемме 10 одно из ребер удаляемых ϑ_2, скажем, c, принадлежит циклической хромосоме C в Q, и C не содержит ребра d. Так как ϑ_2 слабо зависит от ϑ_1, или c, или d созданы с помощью ϑ_1. Рассмотрим эти случаи ниже.

Если ребро d создано с помощью ϑ_1, тогда ребро c существует в P и удаляется ϑ_3. Пусть e – другое ребро, создаваемое ϑ_1 в Q. Если e принадлежит циклической хромосоме C, тогда по Лемме 10 ϑ_1^{-1} (трансформирующая Q в P) уменьшает число циклических хромосом, то есть $c(P) < c(Q)$, получено противоречие. Таким образом, ни одно из ребер d, e не принадлежит C и следует, что C также существует в P и не содержит ребер a, b. Тогда ребра, удаляемые с помощью ϑ_3 (то есть c и одно из a, b), принадлежат циклической хромосоме C и какой-то другой хромосоме, следовательно, по Лемме 10 $c(P) > c(Q')$.

Если ребро c создано ϑ_1, тогда d существует в P и удаляется с помощью ϑ_3. Другое ребро, созданное ϑ_1, должно также принадлежать C, потому что в противном случае по Лемме 10 ϑ_1^{-1} (трансформирующая Q в P) могла уменьшить число циклических хромосом, то есть $c(P) < c(Q)$, получено противоречие. Таким образом, ϑ_1^{-1} заменяет два ребра в C на ребра a, b, получая в результате одну или две циклические хромосомы в P. В обоих случаях ребра a и b в P принадлежат одной или двум циклическим хромосомам, которые не содержат ребра d. Поскольку ϑ_3 оперирует ребром d и одним из a, b, по Лемме 10 $c(P) > c(Q')$. □

3.3.2. Случай, который не был рассмотрен в статье

Лемма 13. *Пусть* $P \xrightarrow{\vartheta_1} Q \xrightarrow{\vartheta_2} R$ *– трансформация между геномами* P, Q *и* R, *такая что* $c(Q) > c(P) = c(R)$, *и геномы* P *и* R *различаются ровно в одной хромосоме. Тогда эта хромосома имеет один и тот же тип в геномах* P *и* R.

Лемма 14. *Пусть* $P \xrightarrow{\vartheta_1} Q \xrightarrow{\vartheta_2} Q' \xrightarrow{\vartheta_3} R$ *– трансформация между геномами* P, Q, Q' *и* R, *такая что (i)* $c(Q) > c(P) = c(Q') > c(R)$; *(ii) геномы* P *и* Q' *различаются ровно в одной хромосоме; (iii) DCJ* ϑ_3 *независимо от* ϑ_2 *и* ϑ_1. *Тогда для трансформации* $P \xrightarrow{\vartheta_3} T \xrightarrow{\vartheta_1} T' \xrightarrow{\vartheta_2} R$ *имеем* $c(P) > c(T)$.

Доказательство. Пусть C_P – хромосома, существующая в P, но не в Q', и $C_{Q'}$ – хромосома, существующая в Q', но не в P. Пусть a и b – ребра, удаляемые с помощью DCJ ϑ_3. Из Леммы 10 следует, что если $c(Q') > c(R)$, то ребра a и b принадлежат разным хромосомам в Q', одна из которых циклическая. Поскольку ϑ_3 независима от ϑ_1 и ϑ_2, ребра a и b также существуют в геноме P. По Лемме 13 C_P и $C_{Q'}$ имеют один и тот же тип, и это единственное различие в хромосомах между геномами P и Q'. Следовательно, ребра a и b принадлежат различным хромосомам в P, одна из которых циклическая. По Лемме 10 получаем, что $c(P) > c(T)$. □

Лемма 15. *Пусть* $P \xrightarrow{\vartheta_1} Q \xrightarrow{\vartheta_2} Q' \xrightarrow{\vartheta_3} R$ *– трансформация между геномами* P, Q, Q' *и* R, *такая что (i)* $c(Q) > c(P) = c(Q') > c(R)$; *(ii) геномы* P *и* Q' *отличаются ровно в одной хромосоме; (iii) DCJ* ϑ_3 *слабо зависима от* ϑ_1 *или* ϑ_2. *Тогда существует трансформация* $P \xrightarrow{\vartheta_4} T \xrightarrow{\vartheta_5} T' \xrightarrow{\vartheta_6} R$, *где* $\vartheta_4, \vartheta_5, \vartheta_6$ *– DCJ и* $c(P) > c(T)$.

Доказательство. Пусть C_P – хромосома, существующая в P, но не в Q', и $C_{Q'}$ – хромосома, существующая в Q', но не в P. Пусть a и b – ребра, удаляемые с помощью DCJ ϑ_1, и c и d – ребра, удаляемые с помощью DCJ ϑ_3. Поскольку

$c(R) < c(Q')$, тогда по Лемме 10 ребра c и d принадлежат двум различным хромосомам в Q'. Из того, что ϑ_3 слабо зависимо от ϑ_1 или ϑ_2, следует, что одно из двух ребер, скажем, c, принадлежит $C_{Q'}$, в то время как другое ребро d не принадлежит. По Лемме 13 C_P и $C_{Q'}$ имеют один и тот же тип, и это единственное различие в хромосомах между геномами P и Q'. Следовательно, хромосома C_d содержит ребро d, существующее в двух геномах P и Q'. С помощью перестановки независимых DCJ и замены слабо зависимых DCJ на эквивалентные слабо зависимые DCJ, введеные в Лемме 9, получаем эквивалентную трансформацию $P \xrightarrow{\vartheta_4} T \xrightarrow{\vartheta_5} T' \xrightarrow{\vartheta_2} R$, где DCJ ϑ_4 удаляет ребра d и одно из a или b. Поскольку одна из хромосом C_d и C_P является циклической, получаем $c(P) > c(T)$. □

Теорема 16. *Пусть $P \xrightarrow{\vartheta_1} Q \xrightarrow{\vartheta_2} Q' \xrightarrow{\vartheta_3} R$ – трансформация между геномами P, Q, Q' и R, такая что $c(Q) > c(P) = c(Q') > c(R)$, и ϑ_1, ϑ_2 независимы. Тогда существует трансформация $P \xrightarrow{\vartheta_4} T \xrightarrow{\vartheta_5} T' \xrightarrow{\vartheta_6} R$, где $\vartheta_4, \vartheta_5, \vartheta_6$ – DCJ и $c(P) > c(T)$.*

Доказательство. Пусть a и b – ребра, удаляемые с помощью ϑ_2. По Лемме 10, если $c(Q) > c(Q')$, то ребра a и b принадлежат двум разным хромосомам в Q, одна из которых циклическая. Поскольку ϑ_1 и ϑ_2 независимы, ребра a и b также присутствуют в P. Ниже рассмотрим два случая в зависимости от принадлежности ребер a и b разным хромосомам в P, и покажем, что существует трансформация $P \xrightarrow{\vartheta_4} T \xrightarrow{\vartheta_5} T' \xrightarrow{\vartheta_6} R$, где $c(P) > c(T)$.

Если ребра a и b принадлежат разным хромосомам в P, то, аналогично доказательству в Теореме 11, существует трансформация $P \xrightarrow{\vartheta_2} T \xrightarrow{\vartheta_1} T' \xrightarrow{\vartheta_3} R$, где $c(P) > c(T)$.

Если ребра a и b принадлежат одной и той же хромосоме в P, тогда эта хромосома должна быть разделена с помощью DCJ ϑ_1 на две хромосомы, каждая из которых содержит по одному ребру из a и b. Далее две полученные хромосомы снова сливаются с помощью DCJ ϑ_2 в одну хромосому. Такие сценарии из DCJ уже были описаны в Леммах 14 и 15, где оба генома P и Q' различаются

в одной хромосоме, имеющей один и тот же тип. Как было доказано, в таком случае существует трансформация $P \xrightarrow{\vartheta_4} T \xrightarrow{\vartheta_5} T' \xrightarrow{\vartheta_6} R$, где $c(P) > c(T)$. $\square$

Теорема 17. *Пусть $P \xrightarrow{\vartheta_1} Q \xrightarrow{\vartheta_2} Q' \xrightarrow{\vartheta_3} R$ – трансформация между геномами P, Q, Q' и R, такая что $c(Q) > c(P) = c(Q') > c(R)$, и ϑ_2 зависит от ϑ_1. Тогда существует трансформация $P \xrightarrow{\vartheta_4} T \xrightarrow{\vartheta_5} T' \xrightarrow{\vartheta_6} R$, где $\vartheta_4, \vartheta_5, \vartheta_6$ – DCJ и $c(P) > c(T)$.*

Доказательство. Аналогично доказательству Теоремы 12, если DCJ ϑ_2 строго зависит от ϑ_1, тогда сразу получаем трансформацию, где $c(P) > c(T)$. Для завершения доказательства осталось рассмотреть случай, что ϑ_2 слабо зависит от ϑ_1.

Пусть a и b – ребра, удаляемые с помощью ϑ_1 в P, c и d – ребра, удаляемые с помощью ϑ_2 в Q и e и f – ребра, удаляемые ϑ_3 в Q'. Предположим, что одно из ребер, удаляемых ϑ_3, скажем, f, принадлежит циклической хромосоме, а другое ребро e принадлежит любой другой хромосоме. Поскольку ϑ_2 слабо зависимо от ϑ_1, одно из ребер, скажем, c, создано с помощью ϑ_1, в то время как другое ребро d существует в P. Обозначим хромосому, содержащую ребра $a,\ b$, как C_1, и хромосому, содержащую ребро d, как C_2. Ниже будут рассмотрены различные случаи в зависимости от принадлежности ребра d хромосоме C_1 и линейности хромосом C_1 и C_2 в геноме P. Для всех трех случаев покажем существование такой трансформации $P \xrightarrow{\vartheta_4} T \xrightarrow{\vartheta_5} T' \xrightarrow{\vartheta_6} R$, где $c(P) > c(T)$.

Если C_1 содержит ребро d, тогда ϑ_1 и ϑ_2 соответствуют транспозиции, происходящей на одной хромосоме. С помощью Леммы 14 или 15 получаем требуемую трансформацию, где $c(P) > c(T)$.

Если C_1 не содержит ребра d, и хотя бы одна из хромосом C_1 или C_2 циклическая, тогда ребра $a,\ b$ и d принадлежат различным хромосомам в P, одна из которых циклическая. По Лемме 9 получаем пару из DCJ $(\vartheta_4, \vartheta_5)$ эквивалентных $(\vartheta_1, \vartheta_2)$, где DCJ ϑ_4 удаляет ребра d и одно из ребер $a,\ b$. По Лемме 10 получаем, что $c(P) > c(T)$.

Если C_1 не содержит ребра d и обе хромосомы C_1 и C_2 линейны, тогда в геноме P обе C_1 и C_2 не содержат ребра f. Таким образом, циклическая хромосома содержит ребро f в Q' и ребра в геноме P. Переставляя независимые DCJ и заменяя слабо зависимые DCJ на эквивалентные слабо зависимые DCJ, по Лемме 10 получаем эквивалентную трансформацию $P \xrightarrow{\vartheta_4} T \xrightarrow{\vartheta_5} T' \xrightarrow{\vartheta_2} R$, где DCJ ϑ_4 удаляет ребра f и одно из $a,\ b$. Поскольку ребра a и b принадлежат C_1 и d принадлежит другой циклической хромосоме в P, то по Лемме 10 имеем, что $c(P) > c(T)$. □

3.4. Оценка точности решения

Отметим, что $d_{DCJ}(G_1, M') = |t_1| = |T_1| - c(M)$. Ясно, что t_1 представляет кратчайшую трансформацию из M' в G_1. Пусть t_2 и t_3 – кратчайшие трансформации из M' в G_2 и G_3 соответственно (Рисунок 3.1). По неравенству треугольника $|t_i| = d_{DCJ}(G_i, M') \leq d_{DCJ}(G_i, M) + d_{DCJ}(M, M') = |T_i| + c(M)$ для $i = 2, 3$. Таким образом, имеем

$$\begin{aligned} & \sum_{i=1}^{3} d_{DCJ}(M', G_i) - \sum_{i=1}^{3} d_{DCJ}(M, G_i) = \sum_{i=1}^{3} |t_i| - \sum_{i=1}^{3} |T_i| \\ \leq\ & (|T_1| + |T_2| + |T_3| + c(M)) - (|T_1| + |T_2| + |T_3|) = c(M). \end{aligned}$$

3.5. Применение алгоритма в контексте малой филогенетической проблемы

В малой филогенетической проблеме дано полное бинарное дерево. Предположим, что данная проблема была решена каким-то алгоритмом (например, описанным в предыдущей главе), и были получены все предковые геномы во внутренних узлах. Данные геномы могут содержать циклические хромосомы, в то время как все существующие листьевые геномы содержат линейные хромосомы. В таком случае появление циклических хромосом в предковых геномах

маловероятно, и будет лучше их линеаризовать.

Как упоминалось в параграфе 1.1.2, если взять внутренний узел и рассмотреть его три непосредственных соседних узла, соответствующие каким-то геномам, тогда будет получана проблема медианы. Данное наблюдение позволяет применить алгоритм, описанный выше, для каждого внутреннего узла филогенетического дерева. Однако стоит отметить, что циклические хромосомы могли появиться в более ранних предковых геномах и повлиять на более старые предковые геномы, поэтому определим очередь запуска данного алгоритма в обратном порядке обхода дерева: запуск алгоритма для левого сына, правого сына, затем для текузего узла. Такая очередь обработки внутренних узлов гарантирует, что все внутренние геномы будут линеаризованы за полиномиальное время.

Глава 4

Экспериментальные результаты

Для оценки качества реконструкции предковых геномов, полученных с помощью MGRA2 и других программных средств, было проведено сравнение на реальных и симуляционных данных. Из всего множества продуктов, описанных в параграфе 1.2, было выбрано два программных средства: PMAG$^+$ и GapAdj, по следующим причинам:

- поддерживают работу с событиями вставок, удалений и со стандартными перестроечными операциями
- найдены в свободном доступе
- работают на всех входных данных
- применимы на реальных биологических данных

Оба программных средства имеют входные параметры, которые настраивают качество работы соответствующих алгоритмов. Для PMAG$^+$ были использованы стандартные параметры, а для GapAdj параметры были установлены как в статье [46]: $delta = 25$, $threshold = 0.6$.

4.1. Симуляционные данные

Для сравнения были сгенерированы три множества данных с $N = 6$, $N = 10$ и $N = 12$ симуляционных геномов. Для упрощения данные геномы являлись однохромосомными геномами, которые подвергались только инверсиям и событиям вставок и удалений.

Для симуляции листьевых геномов («существующих видов»), первоначально выбиралось корневое бинарное дерево с N листьями из равномерного рас-

пределения корневых полных бинарных деревьев. Начальный однохромосомный геном с $G = 1000$, $G = 1500$ различных «генов» помещался в корень бинарного дерева. Случайные инверсии, события удаления и события вставки гена применялись к начальному однохромосомному геному вдоль каждой ветки дерева. Максимальное число событий вдоль каждой ветки случайно выбиралось из диапозона $[\frac{E}{2}, E]$ для $E = 100$ и $E = 200$. Среди всех этих операций $R \in \{0\%, 20\%, 40\%, 60\%\}$ были вставками и удалениями (по половине каждого вида событий), а оставшаяся часть – реверсиями. Равное число вставок и удалений вдоль каждой ветки дерева гарантирует, что размер для всех геномов в узлах дерева будет равным.

Для каждого набора параметров N, G, E, R было создано десять экземпляров данных с листьевыми геномами и предковыми геномами во внутренних узлах. При запуске MGRA2, PMAG$^+$ и GapAdj на листьевых геномах и входном дереве были получены предковые геномы во внутренних узлах. Рисунок 4.1 показывает среднюю точность восставления предковых геномов в терминах корректности (истинно-положительное), потеряности (ложно-отрицательное) и некорректности (ложно-положительное) среди связностей в полученных геномах при сравнении с известными симуляционными геномами.

Аналогично, рисунок 4.2 показывает среднее DCJ-indel расстояние, вычисленное с помощью алгоритма, описанного в [30], для каждой соответствующей пары симуляционного предкового генома и восстановленного генома.

GapAdj демонстрирует худшие результаты на всех экспериментах. Стоит отметить, что данное программное средство чуствительно к проценту событий вставок и удалений, так как наблюдается значительная деградация качества для $R > 0$. С другой стороны, PMAG$^+$ более чуствителен к количеству перестроек, чем к проценту событий вставок и удалений, и результаты хуже для $E = 200$, чем для $E = 100$. В отличие от них, MGRA2 менее чуствителен к R и E, и, очевидно, выигрывает по качеству у GapAdj и PMAG$^+$ в каждой из метрик.

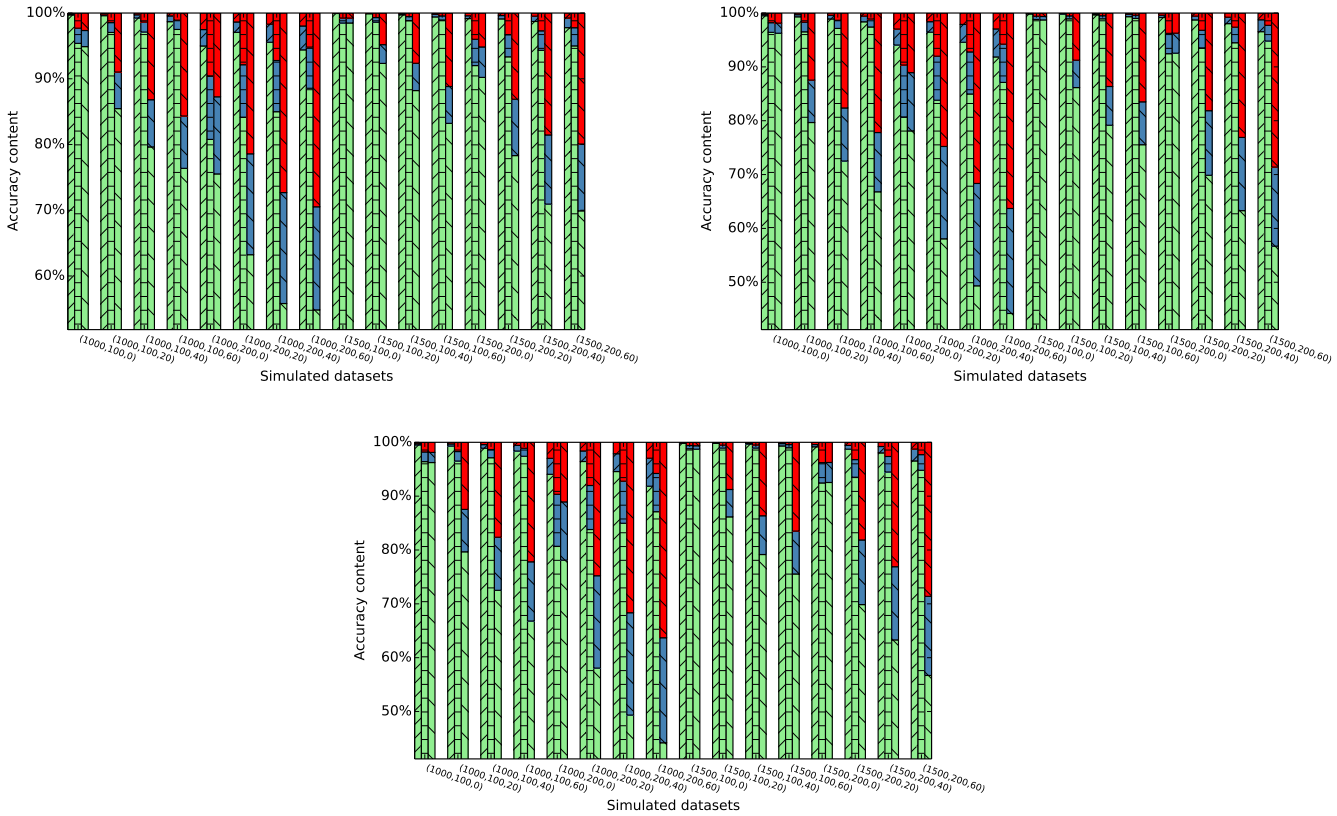

Рис. 4.1. Нормализованные истинно-положительные (зеленые столбцы), ложно-отрицательные (синие столбцы) и ложно-положительные (красные столбцы) значения для востановленных связностей генов для каждого набора параметров (G, E, R), $N = 6$ геномов (верхняя панель), $N = 10$ геномов (средняя панель), $N = 12$ геномов (нижняя панель). Для каждого множества параметров три столбца соответствуют программам MGRA2 (левый столбец), PMAG$^+$ (средний столбец) и GapAdj (правый столбец).

4.2. Реальные данные

Для получения последовательности генов на реальных данных были выбраны шесть геномов млекопитающих и получены их парные ортологичные отношения с помощью программы Ensembl BioMart [62]:

- *Homo sapiens* (GRCh37.p12)
- *Mus musculus* (GRCm38.p1)
- *Rattus norvegicus* (Rnor_5.0)

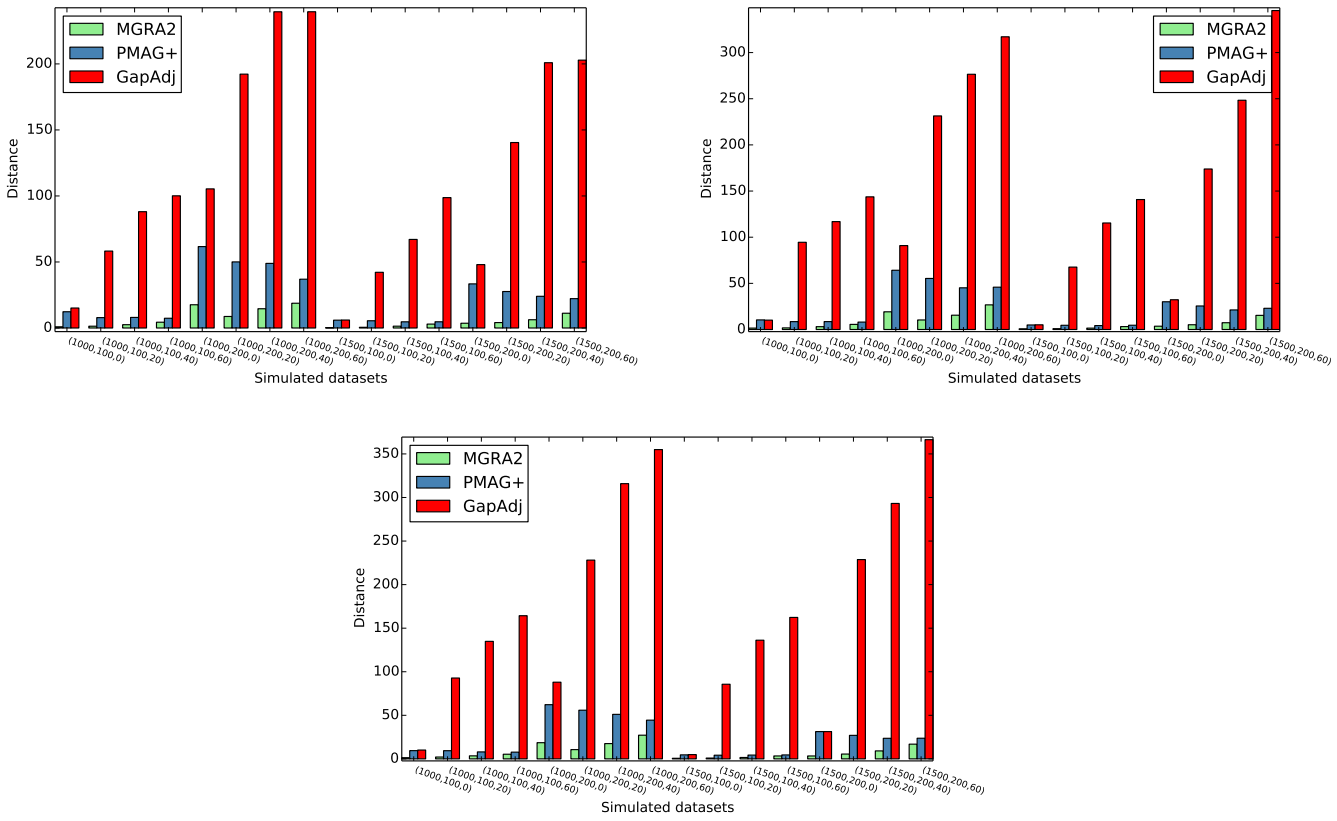

Рис. 4.2. Усредненное DCJ-indel расстояние по десять датасетам между соответствующими восстановленными и симулированными предковыми геномами для набора параметров (G, E, R), аналогичных рисунку 4.1 при $N = 6$ геномов (левая панель), $N = 10$ геномов (средняя панель), $N = 12$ геномов (нижняя панель).

- *Canis familiaris* (CanFam3.1)
- *Macaca mulatta* (MMUL_1.0)
- *Pan troglodytes* (CHIMP2.1.4)

Их филогенетическое дерево показано на рисунке 2.3.

Так как размер геномов лежит в диапазоне от $14,597$ генов для собаки до $17,959$ генов для человека, то применение GapAdj невозможно. В то же время PMAG$^+$ предоставляет бессмысленные результаты, поэтому анализ на реальных данных ограничен только X хромосомой (размер генов лежит в интервале от 591 генов для собаки до 770 генов для человека).

Число парных ортологов между X хромосомами варьируется от $17,654$ (собака - шимпанзе) до $28,732$ (мышь - крыса). Из этих парных ортологов было построено $1,009$ генных семейств, из которых потом были удалены все паралоги и получено 879 генных семейств.

Для каждого из четырех предковых геномов MR (мышь - крыса), MRD (мышь - крыса - собака), HC (человек - шимпанзе), QHC (макака - человек - шимпанзе) диаграмма 4.3 показывает уникальные и общие связности генов среди X хромосом, восстановленных различными программами.

Результаты показывают, что число общих связностей генов между MGRA2 и GapAdj очень мало (от 2 до 20), но известно, что GapAdj вряд ли правильно реконструировал все связности правильно, которые не смог восстановить PMAG$^+$. Для относительно недавних предков MR и HC число уникальных связностей между генами достаточно мало для всех алгоритмов, но на более древних предках (MRD и QHC) данное число значительно вырастает. В случае результатов GapAdj данная ситуация, вероятно, может означать, что большинство соответствующих связностей восстановлено GapAdj некорректно. Кроме того, для геномов MRD и QHC, полученных с помощью MGRA2, подтверждается значительное число связностей (53 и 93 соответственно), найденных PMAG$^+$, но не GapAdj. Такие результаты показывают некоторую надежность MGRA2 и PMAG$^+$ для более древних восстановленных геномов.

Стоит также отметить, что для всех геномов, полученных с помощью PMAG$^+$ и GapAdj, наблюдается значительное число общих связностей (в диапазоне от 51 до 74), но они не общие для геномов, полученных MGRA2. Данное смещение, возможно, вызвано тем, что программы PMAG$^+$ и GapAdj базируются на гомологичной вероятностной модели.

Таблица 4.1 содержит информацию о длине каждой ветки филогенетического дерева, изображенного на рисунке 2.3, между существующими и реконструированными предковыми X хромосомами. Общая длина веток между геномами, полученными с помощью PMAG$^+$, больше на 9% относительно геномов,

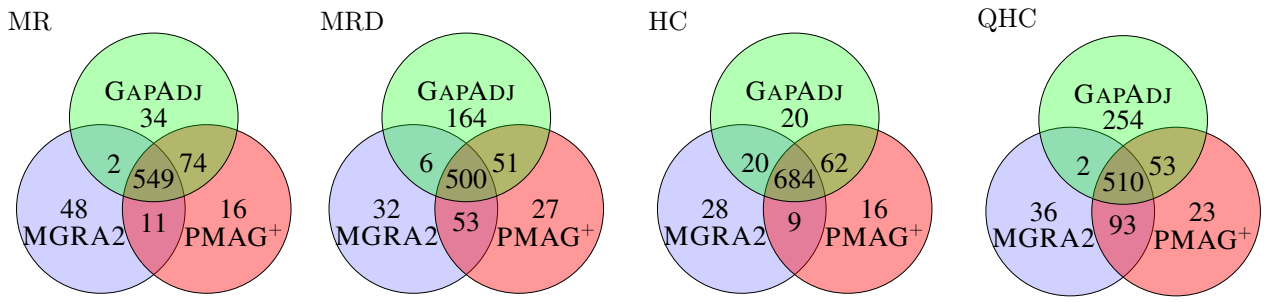

Рис. 4.3. Уникальное и общее число связностей генов между восстановленными X хромосомами предковых геномов MR (мышь - крыса), MRD (мышь - крыса - собака), HC (человек - шимпанзе), и QHC (макака - человек - шимпанзе).

полученных с помощью MGRA2. Также общая длина веток для геномов, полученных GapAdj, больше на 43%. Это доказывает, что MGRA2 дает лучшие результаты с точки зрения минимизации суммарного расстояния для всех веток филогенетического дерева.

Таблица 4.1. Число DCJ, событий вставок и удалений для каждой ветки филогенетического дерева (вычисленное с помощью [30] алгоритма) между X хромосомами млекопитающих, реконструированных различными программами.

	MGRA2			GapAdj			PMAG+		
Ветка	DCJ	инделы	длина	DCJ	инделы	длина	DCJ	инделы	длина
$MRD+$	0	47	47	38	156	194	24	65	89
$MR+$	17	62	79	55	29	84	44	70	114
$D+$	15	46	61	40	79	119	16	32	48
$M+$	17	34	51	28	25	53	14	20	34
$R+$	30	46	76	47	36	83	46	42	88
$HC+$	2	97	99	17	24	41	9	121	130
$Q+$	1	46	47	23	110	133	2	14	16
$H+$	0	38	38	9	14	23	0	0	0
$C+$	7	83	90	11	94	105	18	100	118
Всего	89	499	**588**	268	567	**835**	173	464	**637**

В завершение стоит отметить, что, в отличие от других программ, MGRA2 также обрабатывает полные геномы млекопитающих и восстанавливает предковые геномы, которые хорошо согласуются с восстановленными предковыми X хромосомами этих же геномов.

Заключение

В данной работе представлено улучшение программы MGRA. В ходе процесса модификации данной программы была исследована модель множественного брейкпоинт графа, были доказаны необходимые свойства модели, были сформулированы основные принципы, на которых основывается алгоритм. В модель и алгоритм была добавлена возможность обработки событий вставок и удалений. Для возможности работы с большим значением брейкпоинт-переиспользования были обобщены, изменены или добавлены новые шаги в существующий алгоритм. Также в MGRA был реализован алгоритм линеаризации медианного генома. Для его реализации исходный алгоритм [40] был расширен на события вставки и удаления и был доказан пропущенный случай для обычных перестроек.

В результате было создано новое программное средство, которое получило название MGRA2, доступное по адресу `https://github.com/ablab/mgra`. Также у данной программы есть веб-сайт с дружественным интерфейсом, который можно найти по адресу `http://mgra.cblab.org/`.

Полученный алгоритм обладает целым рядом преимуществ по сравнению с другими программными средствами. Программа позволяет работать с геномами млекопитающих, учитывать события вставок и удалений, получать высокое качество реконструкции предковых геномов. Из недостатков стоит отметить, что программа предполагает отсутствие хромосом, состоящих из одного гена, а также существует ограничение на длину событий вставок и удалений в один ген.

Основной целью дальнейшей работы является добавление событий полного дуплицирования генома, дальнейшее улучшение качества реконструкции предковых геномов, добавление возможности реконструировать филогенетические деревья и проводить различный анализ на получаемых сценариях между геномами (например, предсказание количества транспозиций).

По результатам работы готовятся две статьи для публикации в журналах.

Литература

1. Dobzhansky T., Sturtevant A. Inversions in the chromosomes of Drosophila pseudoobscura // Genetics. 1938. Vol. 23, no. 1. P. 28.

2. Day W. H., Johnson D. S., Sankoff D. The computational complexity of inferring rooted phylogenies by parsimony // Mathematical biosciences. 1986. Vol. 81, no. 1. P. 33–42.

3. Fertin G. Combinatorics of genome rearrangements. MIT press, 2009.

4. Gordon J. L., Byrne K. P., Wolfe K. H. Additions, losses, and rearrangements on the evolutionary route from a reconstructed ancestor to the modern Saccharomyces cerevisiae genome // PLoS genetics. 2009. Vol. 5, no. 5. P. e1000485.

5. Jung S., Cestaro A., Troggio M. et al. Whole genome comparisons of Fragaria, Prunus and Malus reveal different modes of evolution between Rosaceous subfamilies // BMC genomics. 2012. Vol. 13, no. 1. P. 129.

6. Kolmogorov M., Raney B., Paten B., Pham S. Ragout—a reference-assisted assembly tool for bacterial genomes // Bioinformatics. 2014. Vol. 30, no. 12. P. i302–i309.

7. Aganezov S., Sitdykova N., Alekseyev M. A. et al. Scaffold assembly based on genome rearrangement analysis // Computational biology and chemistry. 2015. Vol. 57. P. 46–53.

8. Tannier E. Yeast ancestral genome reconstructions: The possibilities of computational methods // Comparative Genomics. Springer, 2009. P. 1–12.

9. Chauve C., Gavranovic H., Ouangraoua A., Tannier E. Yeast ancestral genome reconstructions: the possibilities of computational methods II // Journal of Computational Biology. 2010. Vol. 17, no. 9. P. 1097–1112.

10. Xu A. W., Moret B. M. GASTS: parsimony scoring under rearrangements // Algorithms in Bioinformatics. Springer, 2011. P. 351–363.

11. da Silva P. H., Machado R., Dantas S., Braga M. D. Restricted DCJ-indel model: sorting linear genomes with DCJ and indels // BMC bioinformatics. 2012. Vol. 13, no. Suppl 19. P. S14.

12. Maňuch J., Patterson M., Wittler R. et al. Linearization of ancestral multichromosomal genomes // BMC Bioinformatics. 2012. Vol. 13, no. Suppl 19. P. S11.

13. Alekseyev M. A., Pevzner P. A. Breakpoint graphs and ancestral genome reconstructions // Genome Research. 2009. Vol. 19, no. 5. P. 943–957.

14. Peng Q., Pevzner P. A., Tesler G. The fragile breakage versus random breakage models of chromosome evolution // PLoS computational biology. 2006. Vol. 2, no. 2. P. e14.

15. Sankoff D., Trinh P. Chromosomal breakpoint reuse in genome sequence rearrangement // Journal of Computational Biology. 2005. Vol. 12, no. 6. P. 812–821.

16. Sankoff D., Trinh P. Chromosomal breakpoint re-use in the inference of genome sequence rearrangement // Proceedings of the eighth annual international conference on Resaerch in computational molecular biology / ACM. 2004. P. 30–35.

17. Pevzner P., Tesler G. Human and mouse genomic sequences reveal extensive breakpoint reuse in mammalian evolution // Proceedings of the National Academy of Sciences. 2003. Vol. 100, no. 13. P. 7672–7677.

18. Renwick J. The mapping of human chromosomes // Annual review of genetics. 1971. Vol. 5, no. 1. P. 81–120.

19. Hardison R. C. Comparative genomics // PLoS biology. 2003. Vol. 1, no. 2. P. e58.

20. Minkin I., Patel A., Kolmogorov M. et al. Sibelia: a scalable and comprehensive synteny block generation tool for closely related microbial genomes // Algorithms in Bioinformatics. Springer, 2013. P. 215–229.

21. Pham S. K., Pevzner P. A. DRIMM-Synteny: decomposing genomes into evolutionary conserved segments // Bioinformatics. 2010. Vol. 26, no. 20. P. 2509–2516.

22. Rodelsperger C., Dieterich C. CYNTENATOR: progressive gene order alignment of 17 vertebrate genomes // PloS one. 2010. Vol. 5, no. 1. P. e8861.

23. Proost S., Fostier J., De Witte D. et al. i-ADHoRe 3.0—fast and sensitive detection of genomic homology in extremely large data sets // Nucleic acids research. 2012. Vol. 40, no. 2. P. e11–e11.

24. Sankoff D., Blanchette M. The median problem for breakpoints in comparative genomics // Computing and combinatorics. Springer, 1997. P. 251–263.

25. Bergeron A., Mixtacki J., Stoye J. A unifying view of genome rearrangements // Algorithms in Bioinformatics. Springer, 2006. P. 163–173.

26. Yancopoulos S., Attie O., Friedberg R. Efficient sorting of genomic permutations by translocation, inversion and block interchange // Bioinformatics. 2005. Vol. 21, no. 16. P. 3340–3346.

27. Alekseyev M. A., Pevzner P. A. Multi-Break Rearrangements and Chromosomal Evolution // Theoretical Computer Science. 2008. Vol. 395, no. 2-3. P. 193–202.

28. Alekseyev M. A. Multi-break rearrangements and breakpoint re-uses: from circular to linear genomes // Journal of Computational Biology. 2008. Vol. 15, no. 8. P. 1117–1131.

29. Hannenhalli S., Pevzner P. A. Transforming men into mice (polynomial algo-

rithm for genomic distance problem) // Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on / IEEE. 1995. P. 581–592.

30. Braga M. D. V., Willing E., Stoye J. Double Cut and Join with Insertions and Deletions // Journal of Computational Biology. 2011. Vol. 18, no 9. P. 1167–1184.

31. Yancopoulos S., Friedberg R. DCJ Path Formulation for Genome Transformations which Include Insertions, Deletions, and Duplications // Journal of Computational Biology. 2009. Vol. 16, no. 10. P. 1311–1338.

32. Compeau P. E. A simplified view of DCJ-Indel distance // Algorithms in Bioinformatics. Springer, 2012. P. 365–377.

33. Compeau P. E. DCJ-Indel sorting revisited. // Algorithms for Molecular Biology. 2013. Vol. 8, no. 6.

34. Compeau P. E. A Generalized Cost Model for DCJ-Indel Sorting // Algorithms in Bioinformatics. Springer, 2014. P. 38–51.

35. Braga M. D., Machado R., Ribeiro L. C., Stoye J. On the weight of indels in genomic distances // BMC bioinformatics. 2011. Vol. 12, no. Suppl 9. P. S13.

36. Xu A. W. A fast and exact algorithm for the median of three problem: A graph decomposition approach // Journal of Computational Biology. 2009. Vol. 16, no. 10. P. 1369–1381.

37. Xu A. W. DCJ Median Problems on Linear Multichromosomal Genomes: Graph Representation and Fast Exact Solutions // Comparative Genomics / Ed. by F. D. Ciccarelli, I. Miklós. 2009. Vol. 5817 of Lecture Notes in Computer Science. P. 70–83.

38. Xu A. W., Sankoff D. Decompositions of multiple breakpoint graphs and

rapid exact solutions to the median problem // Algorithms in Bioinformatics. Springer, 2008. P. 25–37.

39. Zhang M., Arndt W., Tang J. An exact solver for the DCJ median problem // Pacific Symposium on Biocomputing. Vol. 14. 2009. P. 138–149.

40. Jiang S., Alekseyev M. A. Linearization of Median Genomes under DCJ // Algorithms in Bioinformatics. Springer, 2014. P. 97–106.

41. Ma J., Zhang L., Suh B. B. et al. Reconstructing contiguous regions of an ancestral genome // Genome Research. 2006. Vol. 16, no. 12. P. 1557–1565.

42. Ma J. A probabilistic framework for inferring ancestral genomic orders // Proceedings of 2010 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). 2010. P. 179–184.

43. Ma J., Ratan A., Raney B. J. et al. Dupcar: reconstructing contiguous ancestral regions with duplications // Journal of Computational Biology. 2008. Vol. 15, no. 8. P. 1007–1027.

44. Hu F., Lin Y., Tang J. MLGO: phylogeny reconstruction and ancestral inference from gene-order data // BMC bioinformatics. 2014. Vol. 15, no. 1. P. 354.

45. Hu F., Zhou L., Tang J. Reconstructing Ancestral Genomic Orders Using Binary Encoding and Probabilistic Models // Proceedings of the 9th International Symposium on Bioinformatics Research and Applications (ISBRA) / Ed. by Z. Cai, O. Eulenstein, D. Janies, D. Schwartz. Vol. 7875 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013. P. 17–27.

46. Hu F., Zhou J., Zhou L., Tang J. Probabilistic Reconstruction of Ancestral Gene Orders with Insertions and Deletions // IEEE/ACM Transactions on Computational Biology and Bioinformatics. 2014. Vol. 11, no. 4. P. 667–672.

47. Lin Y., Hu F., Tang J., Moret B. Maximum likelihood phylogenetic reconstruction from high-resolution whole-genome data and a tree of 68 eukaryotes // Pacific Symposium on Biocomputing / World Scientific. 2013. P. 357–366.

48. Gagnon Y., Blanchette M., El-Mabrouk N. A flexible ancestral genome reconstruction method based on gapped adjacencies // BMC bioinformatics. 2012. Vol. 13, no. Suppl 19. P. S4.

49. Sankoff D., Blanchette M. Multiple genome rearrangement and breakpoint phylogeny // Journal of Computational Biology. 1998. Vol. 5, no. 3. P. 555–570.

50. Bourque G., Pevzner P. A. Genome-scale evolution: reconstructing gene orders in the ancestral species // Genome research. 2002. Vol. 12, no. 1. P. 26–36.

51. Moret B., Wyman S., Bader D. et al. A new implementation and detailed study of breakpoint analysis // Proc. 6th Pacific Symp. Biocomputing (PSB 2001). 2001. P. 583–594.

52. Zhao H., Bourque G. Recovering True Rearrangement Events on Phylogenetic Trees // Proceedings of the 5th Annual RECOMB Satellite Workshop on Comparative Genomics (RECOMB-CG) / Ed. by G. Tesler, D. Durand. Vol. 4751 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007. P. 149–161.

53. Zhao H., Bourque G. Recovering genome rearrangements in the mammalian phylogeny // Genome research. 2009. Vol. 19, no. 5. P. 934–942.

54. Lin Y., Rajan V., Moret B. M. E. TIBA: a tool for phylogeny inference from rearrangement data with bootstrap analysis // Bioinformatics. 2012. Vol. 28, no. 24. P. 3324–3325.

55. Arndt W., Tang J. Emulating Insertion and Deletion Events in Genome Rear-

rangement Analysis // IEEE International Conference on Bioinformatics and Biomedicine (BIBM). 2011. P. 105–108.

56. Biller P., Feijao P., Meidanis J. Rearrangement-based phylogeny using the single-cut-or-join operation // IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB). 2013. Vol. 10, no. 1. P. 122–134.

57. Benzaid B., El-Mabrouk N. Gene order alignment on trees with multi-OrthoAlign // BMC genomics. 2014. Vol. 15, no. Suppl 6. P. S5.

58. Alekseyev M., Pevzner P. Comparative genomics reveals birth and death of fragile regions in mammalian evolution // Genome Biology. 2010. Vol. 11, no. 11. P. R117.

59. Berard S., Chateau A., Chauve C. et al. Computation of perfect DCJ rearrangement scenarios with linear and circular chromosomes // Journal of Computational Biology. 2009. Vol. 16, no. 10. P. 1287–1309.

60. Kolmogorov V. Blossom V: a new implementation of a minimum cost perfect matching algorithm // Mathematical Programming Computation. 2009. Vol. 1, no 1. P. 43–67.

61. da Silva P. H., Machado R., Dantas S., Braga M. D. Restricted DCJ-indel model: sorting linear genomes with DCJ and indels // BMC bioinformatics. 2012. Vol. 13, no. Suppl 19. P. S14.

62. Kasprzyk A. BioMart: driving a paradigm change in biological data management // Database. 2011. Vol. 2011.