УДК 519.23

ПРИМЕНЕНИЕ АЛГОРИТМОВ ПОСТРОЕНИЯ ПАРОСОЧЕТАНИЙ В ГРАФЕ К ЗАДАЧАМ ГЕНОМИКИ

THE USE OF ALGORITHMS FOR CONSTRUCTING MATCHINGS IN A GRAPH TO GENOMICS PROBLEMS

Горбунов К.Ю.
Gorbunov K.Yu.

РГУ им. А.Н. Косыгина (Технологии. Дизайн. Искусство), Москва
The Kosygin State University of Russia, Moscow

ВВЕДЕНИЕ

Задачи преобразований графов имеют непосредственное приложение к геномике. Можно рассматривать геном на макроуровне, как состоящий из хромосом, а каждая хромосома – линейная или кольцевая последовательность генов. Таким образом, геном представляется ориентированным графом. Возникающие при этом задачи – это прежде всего построение кратчайшего преобразования одного генома в другой и реконструкция эволюции геномов на филогенетическом дереве. Этим задачам посвящены работы [1-2] и раздел «Задача реконструкции» настоящей статьи.

Но можно рассматривать геном и на микроуровне, как множество циклических и линейных последовательностей нуклеотидов – букв в алфавите A, C, T, G. Тогда встаёт задача сборки генома – его восстановления по известным сравнительно небольшим кускам – ридам, как правило, неточным. Эта задача рассматривается в разделе «Задача сборки», где описывается эффективный алгоритм сборки генома из ридов при наличии ошибок в них, а также показывается, что собранный геном близок (в некотором смысле) к исходному.

В решении обеих задач основную роль играет построение в некотором нагруженном графе оптимального взвешенного паросочетания, т.е. множества попарно неинцидентных друг другу рёбер с минимальным или максимальным суммарным весом. В теории графов известны достаточно эффективные алгоритмы построения такого множества.

ЗАДАЧА СБОРКИ

Постановка задачи

При секвенировании генома с хромосомы копируются сравнительно короткие последовательности (риды), и возникает задача восстановления по ним генома. Как правило, в качестве начальных данных берётся множество k-меров, т. е. последовательностей фиксированной длины k (параметр алгоритма), обычно состоящее из всех участков длины k данных ридов.

Алгоритм сборки должен расположить их в циклическом или линейном порядке так, чтобы суффикс каждого предыдущего *k*-мера совпадал с префиксом последующего. Как правило, это осуществляется с помощью построения графа де Брёйна и поиска в нём Эйлерова цикла или пути [3] (в этой работе приведены многочисленные ссылки по истории вопроса). Такой метод предполагает довольно жёсткое условие: отсутствие ошибок секвенирования, что обеспечивает точное совпадение упомянутых суффикса и префикса. Мы описываем другой метод, который допускает наличие ошибок секвенирования, и, соответственно, неточное их совпадение. Метод основан на построении двудольного графа между *k*-мерами и поиске в нём максимального паросочетания.

Описание алгоритма

Дано множество *k*-меров (пусть всего их *n*), каждый *k*-мер имеет начало и конец (два разных *k*-мера могут совпадать как последовательности). *Качеством* упорядоченной пары (*u*,*v*) различных *k*-меров назовём качество выравнивания (*k*–1)-суффикса (т.е. суффикса из *k*–1 буквы) *k*-мера *u* и (*k*–1)-префикса *k*-мера *v*, при условии, что делеции разрешаются лишь справа от суффикса и слева от префикса (такое выравнивание назовём *суффиксно-префиксным*). Занумеруем данные *k*-меры, отождествив *k*-мер с его номером. Строим двудольный граф *G*, его доли будем именовать как левую и правую. Каждая доля состоит из всех *k*-меров, и *k*-меры из разных долей с различными номерами соединены ребром. Помечаем каждое ребро качеством упорядоченной пары *k*-меров: *k*-мер на левом конце, *k*-мер на правом конце. Строим в *G* максимальное взвешенное паросочетание. Оно определяет множество циклических и линейных последовательностей *k*-меров, где переход от *k*-мера к следующему происходит по ребру паросочетания. Для каждой последовательности отождествим суффикс каждого предыдущего *k*-мера с префиксом следующего за ним, где суффикс и префикс – те, которые при выравнивании, определяющем качество этой пары *k*-меров, сопоставились буквам (не делециям). Получим множество циклических и линейных последовательностей позиций – множеств отождествлённых позиций *k*-меров. Каждой позиции приписываем наиболее частую букву в соответствующих позициях *k*-меров (если таких букв несколько – любую из них). Получаем множество циклических и линейных последовательностей букв – это множество кольцевых хромосом собранного генома.

Замечание 1. В общем случае не все данные *k*-меры участвуют в собранном геноме, т. е. построенное паросочетание в *G* не является полным. Обоснованием этого служит, во-первых, то, что при значительном числе ошибок секвенирования *k*-мер может стать настолько непохожим на соответствующий участок генома, что его будет невыгодно включать в процесс сборки. А во-вторых, с одного и того же участка хромосомы может читаться несколько ридов, так что возможны *k*-меры, соответствующие одному и тому же участку в исходном геноме. А поскольку в итоговом

геноме одному участку длины k ставится в соответствие не более одного k-мера, некоторые k-меры окажутся «лишними».

Замечание 2. Мы принимаем естественное предположение, что длина хромосомы не меньше k. Поэтому запрещаем ребро в G, соединяющее k-мер с самим собой и линейную последовательность из одного k-мера.

Замечание 3. Веса делеций и соответствий букв при выравнивании являются параметрами алгоритма. Запрет делеций внутри перекрытия суффикса и префикса означает перебор k возможных выравниваний и выбор из них наилучшего. Однако если при секвенировании допускаются вставки и удаления участка хромосомы, то такие делеции допускаются, и выравнивание производится обычным алгоритмом динамического программирования. В остальном алгоритм остаётся прежним.

Собранный геном может содержать большее количество хромосом, чем ожидалось. Для их уменьшения к нему применяется *шлифовка,* включающая следующие возможные этапы.

1) Удаление «лишних» хромосом (обычно небольшого размера), состоящих, как правило, из k-меров, однотипных тем, которые входят в другие хромосомы, имеющие больший размер.

2) Объединение двух хромосом, одна из которых кольцевая, в одну двойной или полуторной переклейкой [Архив-2020]. Иначе говоря, кольцевая хромосома разрезается и вставляется в другую.

3) Объединение двух линейных хромосом в одну склейкой. Иначе говоря, одна хромосома подклеивается к краю другой.

Можно выполнять эти операции на полученном паросочетании в G. Двойная переклейка: замена рёбер (u,v) и (w,t) на рёбра (u,t) и (v,w). Полуторная переклейка: замена рёбра (u,v) на ребро (u,t) или (w,v), где t, w – свободные вершины. Склейка: соединение ребром двух свободных вершин.

Точность алгоритма и пример его работы

Алгоритм выдаёт итоговый геном вместе с множеством k-меров, наложенных на него и упорядоченных на каждой хромосоме в линейном или циклическом порядке. Назовём такой геном *размеченным*. Его *качеством* назовём сумму качеств всех пар (k-мер, следующий k-мер). *Однотипной выборкой* назовём выборку одного k-мера из каждого множества k-меров, соответствующих одному и тому же участку в исходном геноме. Каждая однотипная выборка естественным образом превращает исходный геном в размеченный. Точность алгоритма выражается следующим простым утверждением.

Утверждение. *Пусть для исходного генома и секвенированного с него множества M k-меров существует однотипная выборка, для которой качество соответствующего размеченного генома равно h. Тогда алгоритм построит по M размеченный геном качества не меньше h.*

Доказательство. Из условия вытекает существование в графе G паросочетания веса h. Алгоритм построит паросочетание не меньшего веса и такого же качества размеченный геном. Утверждение доказано.

Пример. Пусть исходный геном – одна кольцевая хромосома (ACTGTCAACTGTCAACTGTCA). Как видно, она состоит из трёх одинаковых участков, а наличие большого количества повторов представляет значительную сложность для сборки.

Пусть с каждого участка длины k=7 секвенировался один k-мер. В каждом из них произошла одна ошибка в средней (четвёртой) позиции – буква в ней заменилась на одну из трёх других (для каждого из трёх одинаковых участков – на разную). Таким образом, получены двадцать один 7-мер, которые для удобства приведём по тройкам с соответствующими сдвигами и номерами:

```
1.1. ACTATCA
1.1. ACTCTCA
1.1. ACTTTCA
2.1. CTGACAA
2.2. CTGCCAA
2.2. CTGGCAA
3.1. TGTAAAC
3.2. TGTGAAC
3.3. TGTTAAC
4.1. GTCCACT
4.2. GTCGACT
4.3. GTCTACT
5.1.  TCACCTG
5.2.  TCAGCTG
5.3.  TCATCTG
6.1.  CAAATGT
6.2.  CAAGTGT
6.3.  CAATTGT
7.1.  AACAGTC
7.2.  AACCGTC
7.3.  AACGGTC
```

Для каждого из этих 7-меров максимальное качество суффиксно-префиксного выравнивания достигается на любом из k-меров следующей тройки (следующей за седьмой тройкой является первая). Действительно, это выравнивание содержит 4 совпадения и 2 несовпадения и не содержит делеций.

Максимальное паросочетание определяет либо одну кольцевую хромосому – исходную, либо две кольцевые хромосомы, из которых двойной

переклейкой легко получить исходную, либо три кольцевых хромосомы, из которых двумя двойными переклейками нетрудно получить исходную.

Пусть, например, получены три хромосомы (ACTGTCA), соответствующие случаю, когда каждый 7-мер соединился в паросочетании с тем же по номеру 7-мером следующей тройки. Тогда двойной переклейкой, заменяющей рёбра (1.1,2.1) и (1.2,2.2) на (1.1,2.2) и (1.2,2.1), получаем две хромосомы – (ACTGTCAACTGTCA) и (ACTGTCA), после чего двойной переклейкой, заменяющей рёбра (1.1,2.2) и (1.3,2.3) на (1.1,2.3) и (1.3,2.2), получаем исходную хромосому.

ЗАДАЧА РЕКОНСТРУКЦИИ

Постановка задачи

Геномная структура (далее просто *структура*) – ориентированный граф, состоящий из цепей и циклов, без изолированных вершин, но, возможно, с петлями. Его рёбра занумерованы, мы рассматриваем случай, когда номера не повторяются.

Рассматриваются четыре SCJ-операции над структурой: расклейка её вершины степени 2, склейка двух вершин степени 1, удаление изолированного ребра и добавление изолированного ребра. Даны также рациональные цены этих операций. SCJ-*расстояние* от структуры a до структуры b равно минимальной суммарной цене операций, которыми a преобразуется в b.

Если рассматривать каждое ребро структуры, как имеющее два края – начало и конец, то очевидно, что это расстояние равно взвешенной сумме числа пар краёв, склеенных в a, но не в b, склеенных в b, но не в a, числа рёбер, присутствующих в a, но не в b и числа рёбер, присутствующих в b, но не в a.

Дано корневое дерево, которое может быть небинарным. В каждом его листе задана структура без петель. Требуется расставить структуры по внутренним вершинам дерева так, чтобы сумма по всем его рёбрам SCJ-расстояний между структурами на концах ребра была минимальной. Описанную постановку будем называть *общим случаем* задачи. Выделим т. н. *циклический случай* задачи, когда все компоненты данных структур в листьях – циклы и это же требование накладывается на все структуры в расстановке.

В [5] описан точный эффективный алгоритм реконструкции структур при равных ценах SCJ-операций и равном составе структур. В [6] такой алгоритм приведён для неравного состава.

В [2] описан эффективный точный алгоритм реконструкции структур на простейшем дереве типа «звезда» при произвольных ценах SCJ-операций и неравном составе структур при условии, что структуры не содержат петель.

Дальнейшие ссылки по рассматриваемой проблеме можно найти в [2], а также в [7], где рассмотрено некоторое обобщение задачи реконструкции, доказана его NP-трудность и описан FTP-алгоритм для него.

В настоящей работе мы расширяем результат из [2] на двузвёздное полубинарное дерево, т. е. дерево с двумя внутренними вершинами, нижней из которых инцидентно ровно два выходящих ребра, рис. 1.

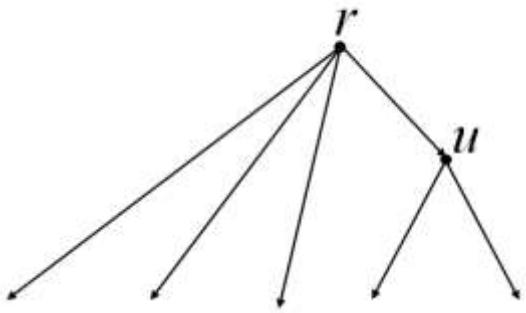


Рис. 1. Двузвёздное полубинарное дерево

Описание алгоритма

Как и в [2], рассматриваем структуры без петель. Если n-е ребро в структуре отсутствует, добавим его туда в качестве петли – этот приём сводит задачу к случаю равного состава всех структур. При этом цена склейки изолированного ребра в петлю равна цене его удаления, цена расклейки петли – цене добавления ребра. Имена краёв n-го ребра структуры обозначим как n_1 (начало ребра) и n_2 (его конец). Возникает взаимно-однозначное соответствие между структурой и паросочетанием на множестве M имён краёв всех рёбер, присутствующих хотя бы в одном листе. Дано двузвёздное полубинарное дерево (рис. 1), т. е. дерево с двумя внутренними вершинами (которые обозначены как r и u), любым количеством листьев, с которыми соединена вершина r, и двумя листьями при вершине u.

Назовём *признаком* пары p краёв два её возможных состояния: склеены (говорим: p положительна) или не склеены (говорим: p отрицательна). Для каждой пары p независимо от других пар вычислим по индукции (динамическим программированием) сначала для u, затем для r величины p_{yes} – минимальную цену событий с этой парой ниже u (соответственно, ниже r), при условии, что в u (соответственно, в r) пара p положительна и p_{no} – аналогичные величины при условии, что в u (соответственно, в r) p отрицательна. Придадим каждой паре p в r вес $p_c=p_{yes}-p_{no}$ и построим минимальное взвешенное паросочетание P_r.

В качестве итоговой структуры в r возьмём структуру, определяемую паросочетанием P_r. Для каждой пары p краёв в r обратный ход определяет в u признак пары p по её признаку в r. Ниже (лемма 1) будет показано, что при этом любому краю в u инцидентно не более одной положительной пары, так что набор признаков определяет в u некоторую структуру. Её и возьмём в качестве итоговой.

Точность алгоритма

Назовём *p-сценарием* набор признаков пары p во всех вершинах дерева. *Цена p-сценария* – суммарная цена событий с p согласно этим признакам. Для любого паросочетания P в r обозначим как $w(P)$ его вес и как $c(P)$ – минимальную суммарную (по всем p) цену таких p-сценариев, что в вершине r положительны лишь пары из P. Саму совокупность этих сценариев обозначим $s(P)$. Пусть P_0 – пустое паросочетание.

Лемма 1. *Минимум величины $c(P)$ достигается на $P=P_r$ и равен $c(P_0)+w(P_r)$. В совокупности $s(P_r)$ в вершине и нет инцидентных друг другу положительных пар.*

Доказательство. Из определения величин p_{yes} и p_{no} следует, что для любого паросочетания P в r выполняется $w(P)=c(P)–c(P_0)$. Полагая здесь $P=P_r$, получаем первое утверждение леммы. Для доказательства второго утверждения предположим (от противного), что в u есть две инцидентные друг другу положительные пары p_1 и p_2. Поскольку в каждом из трёх соседей вершины u хотя бы одна из p_1, p_2 отрицательна, хотя бы одна из p_1, p_2 (пусть p_1) отрицательна по крайней мере в двух соседях. Тогда, сделав p_1 отрицательной в u, мы строго уменьшим цену p-сценария, что противоречит минимальности $c(P_r)$. Лемма доказана.

Теорема. *Описанный алгоритм точен. Время его работы имеет порядок не более, чем $n^2l^2\log(nl)$, где n – число рёбер в структуре, l – число листьев в дереве.*

Доказательство. Предположим (от противного), что имеется расстановка строго меньшей цены, чем построенная. Рассмотрим определяемое ей паросочетание P в r. По лемме 1 имеем, что $c(P)=w(P)+c(P_0)$, а по нашему предположению $c(P)<c(P_r)$. Это противоречит минимальности паросочетания P_r. Время работы алгоритма – сумма времени вычисления величин p_{yes} и p_{no} и времени построения минимального взвешенного паросочетания в графе. Поскольку нетривиально вычисляются лишь p_{yes} и p_{no} для p, положительных хотя бы в одном листе, первое время имеет порядок nl. Минимальное паросочетание строится за время порядка $n'm+(n')^2\log(n')$, где n' – число вершин в графе, m – число рёбер в нём [8, Chapter 26]. Поскольку паросочетание P_r содержит лишь пары, положительные хотя бы в одном листе, у нас это время имеет порядок $n^2l+(nl)^2\log(nl)$. Теорема доказана.

Пример работы алгоритма

Дано двузвёздное полубинарное дерево и структуры в его листьях, рис.2.

Пусть сначала цены всех операций равны 1.

В вершине u имеем: $p_{yes}=0$, $p_{no}=2$ при $p=(1_2,2_2)$; $p_{yes}=1$, $p_{no}=1$ при $p=(3_i,1_1)$ и $p=(3_i,2_1)$ (i любое); $p_{yes}=2$, $p_{no}=0$ при других p.

В вершине r имеем: $p_{yes}=3\rightarrow yes$, $p_{no}=1\rightarrow yes$ при $p=(1_2,2_2)$; $p_{yes}=3\rightarrow yes$, $p_{no}=2\rightarrow no$ при $p=(3_2,1_1)$ и $p=(3_2,2_1)$; $p_{yes}=4\rightarrow yes$, $p_{no}=1\rightarrow no$ при $p=(3_1,1_1)$ и

p=(3_1,2_1); p_{yes}=3→*no*, p_{no}=1→*no* при тех семи p, который склеены в r-листе, но не в u-листе, и p_{yes}=4→*no*, p_{no}=0→*no* при других p. Здесь после стрелки указан соответствующий признак в u.

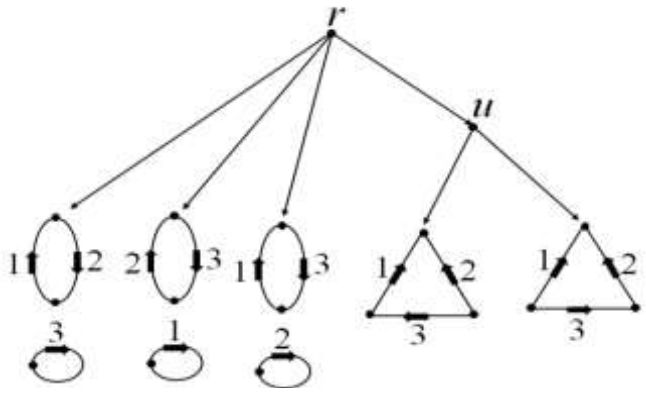


Рис. 2. Исходные данные к примеру работы алгоритма реконструкции структур

Веса всех пар положительны, поэтому минимальное паросочетание – пустое. Следовательно, в r склеек нет, по ссылкам получаем итоговую структуру в u, итоговая расстановка показана на рис. 3. Её цена равна 14.

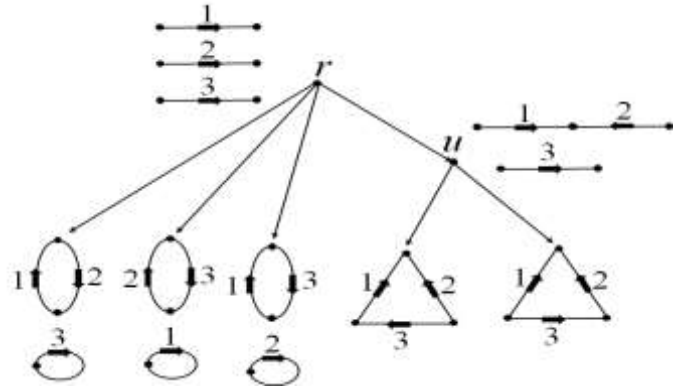


Рис. 3. Итоговая расстановка к примеру работы алгоритма: единичные цены

Теперь пусть цена расклейки всех пар равна 1, а цена их склейки равна 4. В вершине u имеем: p_{yes}=0, p_{no}=8 при p=(1_2,2_2); p_{yes}=1, p_{no}=4 при p=(3_i,1_1) и p=(3_i,2_1) (i любое); p_{yes}=8, p_{no}=0 при других p.

В вершине r имеем: p_{yes}=3→*yes*, p_{no}=4→*yes* при p=(1_2,2_2); p_{yes}=3→*yes*, p_{no}=8→*no* при p=(3_2,1_1) и p=(3_2,2_1); p_{yes}=4→*yes*, p_{no}=4→*no* при p=(3_1,1_1) и p=(3_1,2_1); p_{yes}=3→*no*, p_{no}=4→*no* при тех семи p, который склеены в r-листе, но не в u-листе, и p_{yes}=13→*no*, p_{no}=0→*no* при других p. Таким образом, для весов пар получаем: p_c=–1 при p=(1_2,2_2); p_c=–5 при p=(3_2,1_1) и p=(3_2,2_1); p_c=0 при p=(3_1,1_1) и p=(3_1,2_1); p_c=–1 при тех семи p, который склеены в r-листе, но не в u-листе, и p_c=13 при других p. Минимальное паросочетание имеет вес –7 и состоит, например, из пар (3_2,1_1), (1_2,2_1) и (2_2,3_1). Итоговая расстановка показана на рис. 4. Её цена равна 49.

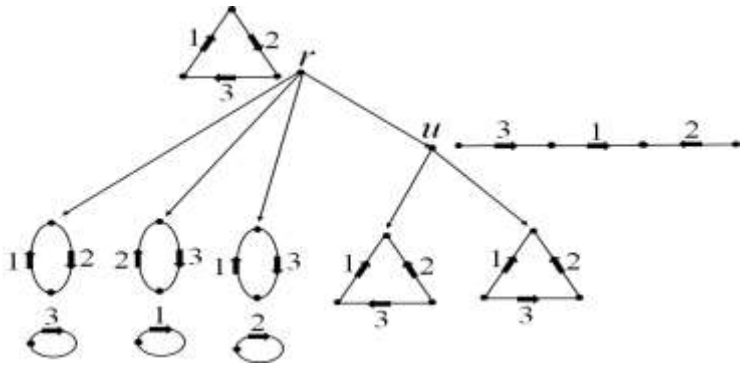


Рис. 4. Итоговая расстановка к примеру работы алгоритма:
цена расклейки равна 1, цена склейки равна 4

Циклический случай задачи реконструкции

Автору не известно, возможен ли точный полиномиальный по времени алгоритм реконструкции структур на двузвёздном полубинарном дереве. Эвристический алгоритм легко получить модификацией описанного алгоритма. А именно, в качестве P_r берём минимальное полное взвешенное паросочетание. После этого, как описано в [2] выполняем спуск в точку локального минимума, чередуя замены полных паросочетаний в u и r (начиная с u). Напомним, что расстановка является локально минимальной, если в любой внутренней вершине дерева замена структуры на любую другую не уменьшает цену расстановки. В очередной вершине v паросочетание заменяется на минимальное полное относительно весов $p_c=p_{yes}-p_{no}$ пар p, где p_{yes} – суммарная цена событий с парой p на инцидентных v рёбрах при условии, что в v она положительна p_{no} – аналогичная цена при условии, что в v пара p отрицательна.

В вышеупомянутом примере (рис. 2, пусть цены единичные) минимальное полное паросочетание P_r, например, такое же, как в рассмотренном случае неравных цен (рис. 4). Легко проверить, что спуск в точку локального минимума выполняется за один шаг; итоговая расстановка показана на рис. 5, её цена равна 20. В данном случае локальный минимум является и глобальным.

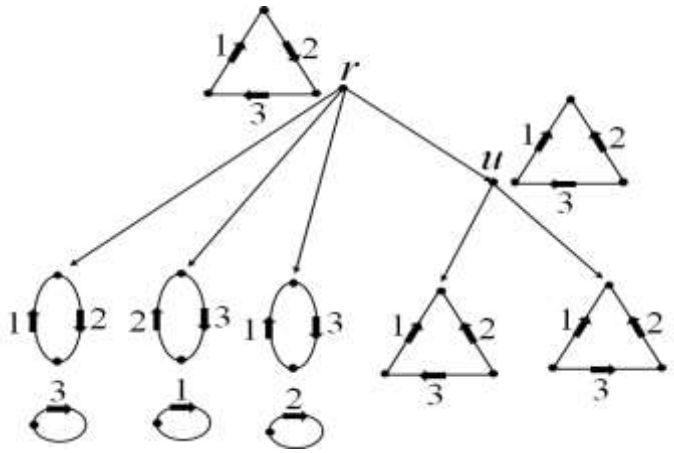


Рис. 5. Итоговая расстановка к примеру работы эвристического алгоритма:
цены единичные.

ЗАКЛЮЧЕНИЕ

Мы описали применение алгоритмов построения оптимального паросочетания в графе для решения задач геномики. В задаче сборки дальнейшие исследования могут быть направлены на разработку эффективных алгоритмов сборки с учётом топологических требований на итоговый геном. В задаче реконструкции остаётся открытым вопрос о возможности точного эффективного алгоритма реконструкции геномных структур на произвольном дереве при произвольных ценах SCJ-операций, а также алгоритма их реконструкции в циклическом случае.

Л И Т Е Р А Т У Р А

1. Gorbunov K.Yu., Lyubetsky V.A. Linear time additively exact algorithm for transformation of chain-cycle graphs for arbitrary costs of deletions and insertions // Mathematics. – 2020. – V.8(11). – Art.2001.

2. Gorbunov K.Yu., Lyubetsky V.A. Multiplicatively exact algorithms for transformation and reconstruction of directed path-cycle graphs with repeated edges // Mathematics. – 2021. – V.9(20). – Art.2576.

3. Medvedev P. Modeling biological problems in computer science: a case study in genome assembly // Briefings in Bioinformatics. – 2019. – V.20(4). – P.1376-1383.

4. Gorbunov K.Yu., Lyubetsky V.A. An Almost Exact Linear Complexity Algorithm of the Shortest Transformation of Chain-Cycle Graphs // Eprint, arXiv:2004.14351 [math.CO] 2020.

5. Feijao P., Meidanis J. SCJ: A Breakpoint-Like Distance that Simplifies Several Rearrangement Problems // IEEE/ACM Transactions on Computational Biology and Bioinformatics. – 2011. – V.8(5).– P.1318-1429.

6. Горбунов К.Ю., Гершгорин Р.А., Любецкий В.А. Перестройка и реконструкция хромосомных структур // Молекулярная биология. – 2015. – Т.49(3). – С.372-383.

7. Luhmann N., Lafond M., Thevenin A., Ouangraoua A., Wittler R. and Chauve C. The SCJ Small Parsimony Problem for Weighted Gene Adjacencies // IEEE/ACM Transactions on Computational Biology and Bioinformatics. – 2019. – V.16(4). – P.1364-1373.

8. Schrijver A. Combinatorial Optimization: Polyhedra and Efficiency. – Berlin; Heidelberg: Springer-Verlag, 2004. – 1881p.