БИОФИЗИКА, 2018, том 63, вып. 3, с. 421–429

МОЛЕКУЛЯРНАЯ БИОФИЗИКА

УДК 577.3

ЭФФЕКТИВНЫЙ АЛГОРИТМ КАРТИРОВАНИЯ ПРОЧТЕНИЙ НА ГЕНОМНЫЙ ГРАФ С ИСПОЛЬЗОВАНИЕМ ИНДЕКСА, ОСНОВАННОГО НА ХЭШ-ТАБЛИЦАХ, И ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

© 2018 г. С.Н. Петров* **, Л.А. Урошлев* **, А.С. Касьянов* **, В.Ю. Макеев* ** * ******

**Институт общей генетики им. Н.И. Вавилова РАН, 119991, Москва, ул. Губкина, 3*

***Московский физико-технический институт (Государственный университет), 141701, Долгопрудный Московской области, Институтский пер., 9*

****Институт молекулярной биологии РАН им. В.А. Энгельгардта, 119991, Москва, ул. Вавилова, 32*

*****Государственный научно-исследовательский институт генетики и селекции промышленных микроорганизмов Национального исследовательского центра «Курчатовский институт», 117545, Москва, 1-й Дорожный проезд, 1*

E-mail: vsevolod.makeev@gmail.com

Поступила в редакцию 03.04.18 г.

Решается задача хранения последовательностей ряда близкородственных геномов и анализа геномных вариаций. Для хранения совпадающих участков последовательностей и известных вариантов используется геномный граф, имеющий структуру ациклического направленного графа. Для выравнивания на геномный граф индивидуальных фрагментов нуклеотидных последовательностей разработан алгоритм быстрого картирования прочтений на геномный граф. Алгоритм сочетает быстрый поиск с помощью хэш-таблиц с алгоритмом динамического программирования и решает проблему экспоненциального роста количества путей на графе. Разработана реализация геномного графа и алгоритма выравнивания прочтений. Проведено сравнение с наиболее известными программами, имеющими сходный функционал.

Ключевые слова: геномные графы, графы вариаций, картирование.

Экспериментальные установки, осуществляющие чтение генетических текстов, производят данные в виде так называемых «прочтений», т.е. фрагментов длиной в несколько десятков или сотен нуклеотидов. При использовании этих методов для анализа внутривидовой или межвидовой изменчивости генетический вариант считается подтвержденным, если достаточно большое количество прочтений совпадает с известной последовательностью генома везде за исключением самого варианта. Известная последовательность, на которую картируются прочтения, обычно называется «референсным геномом». Традиционно в качестве референсного генома используется набор гаплоидных последовательностей, например по одной нуклеотидной последовательности для каждой хромосомы. Иногда это представление дополняется набором фрагментов-скаффолдов, представляющих собой характерные гаплотипы [1]. Так, например, в 2011 г. в референсный геном человека были добавлены нуклеотидные последовательности; они присутствуют, начиная с версии GRCh37. В 2017 г. в работе [1] было описано представление референсного генома как «референсной группы» – набора нуклеотидных последовательностей, который бы включал все известные вариации, а также рассмотрен ряд сложностей, вызванных использованием этой концепции. В качестве одного из варианта программной реализации «референсной группы» предлагалось использовать геномные графы.

Использование графов для хранения нуклеотидных последовательностей – давно известный прием в биоинформатике. Очень часто графы используются для промежуточного представления сборки генома, из которой затем извлекается конечная последовательность. Существует много программ, использующих этот подход для сборки *de novo*, в их число входят velvet [2], Minia [3] и SOAP de novo [4], использующие графы де Брейна [5]. Традиционная последовательность гаплоидного референсного

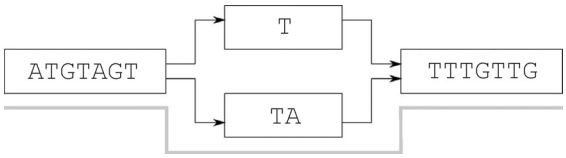

Рис. 1. Фрагмент графа вариаций, содержащий одну делецию.

генома соответствует гамильтонову пути на таком графе. Также некоторые программы-сборщики, например SPADES [6], предоставляют возможность хранить контиги и скаффолды не в виде отдельных строк, а в виде графов, сохраняя их в формате fastg [7].

Хранение генома в виде графа имеет как преимущества, так и недостатки. Из преимуществ стоит отметить возможность одновременно работать с несколькими альтернативными аллелями, которые может хранить один граф. При извлечении одного пути из всех возможных часть информации теряется. Очевидно, что в референсном геноме присутствует всего один вариант последовательности, поэтому если исследуемая особь содержит другой вариант, прочтения, содержащие эту позицию, будут картироваться на референсный геном с заменой. Для того чтобы сохранить достаточное покрытие генома прочтениями, приходится повышать разрешенное число замен при картировании, что приводит к тому, что некоторые ошибочные прочтения тоже будут успешно картированы на геном, а это в свою очередь может привести к некорректным биологическим результатам.

Наиболее существенным недостатком графов, как основной структуры данных для хранения генома, является то, что выбор системы координат – системы индексирования, позволяющей определить расстояние между двумя позициями на пути в графе, перестает быть тривиальной задачей, так как в зависимости от выбранного пути в графе координаты генетических элементов (экзонов, интронов, белоккодирующих элементов и т.д.) могут меняться.

Графы как структуры для хранения генома могут использоваться не только для хранения результата геномной сборки. Например, в 2016 г. вышла работа [8], в которой предлагалось использовать двунаправленные графы для хранения гаплотипов, используя аналог преобразования Барроуза–Уилера для сжатого хранения путей в графе. Преимуществом этого подхода является возможность работы с комбинациями имеющихся гаплотипов.

Одной из наиболее важных операций применения референсной последовательности является использование ее как шаблона для картирования прочтений, полученных при секвенировании новой особи. За несколько десятилетий развития вычислительных методов было разработано множество алгоритмов, использующих различные подходы к решению данной задачи: динамическое программирование (алгоритмы Нидлмана–Вунша и Смита–Ватермана), методы расширения якорей (BLAST [9]), выравнивание с помощью хешей (алгоритм Рабина–Карпа), алгоритмы, основанные на преобразовании исходной строки (BWT [10]), и множество других. Однако большинство из них ориентированы на классическое представление референсной последовательности как нуклеотидной строки, и только сравнительно недавно был предпринят ряд попыток создать алгоритм, предназначенный для выравнивания прочтений с шаблоном, представляющим собой более сложную структуру данных.

Основой для этого семейства алгоритмов является преобразование Барроуза–Уилера, обычно применяемое для индексирования деревьев (т.н. XBW) [11]. В алгоритме, названном GSCA [12], этот подход использовался для индексирования ациклических направленных графов. Несколько позднее алгоритм был модифицирован для работы с графами вариаций, эта модификация получила название GSCA2 [13] и была использована для написания программы variance graph [14] (далее – vg).

В данной статье излагается альтернативный подход к индексированию графа вариаций, основанный на использовании кольцевого полиномиального хеша. Построенный алгоритм имеет линейное время работы для гаплоидной референсной последовательности. Мы также рассмотрим сложности, возникающие при использовании хеш-таблицы для графов, и получим оценку сложности алгоритмов индексирования и выравнивания.

МАТЕРИАЛЫ И МЕТОДЫ

Рассмотрим направленный ациклический граф $G = \{N , E\}$ состоящий из наборов вершин $\{N\}$ и ребер $\{E\}$. Вершины графа содержат в себе фрагменты нуклеотидной последовательности. Ребро $E = (N1, N2)$ соединяет вершины $N1$ и $N2$, если в каком-то из гаплотипов, записанных в графе, существует нуклеотидная последовательность $\mathrm{str}(N1) + \mathrm{str}(N2)$, где $\mathrm{str}(N)$ – функция, возвращающая последовательность, хранящуюся в узле N.

Принципиальная схема графа вариаций, используемая в программе, приведена на рис. 1. Данный граф представляет собой фрагмент ге-

нома, содержащий один вариант – делецию. Сплошной серой линией подчеркнута референсная последовательность, по отношению к которой вычисляются вариации.

Алгоритм построения графа, содержащего два гаплотипа, состоит в следующем:

1. На вход программе подается референсная последовательность, относительно которой были посчитаны вариации, и отсортированный по координатам вектор, содержащий кортежи, состоящие из альтернативной нуклеотидной последовательности, координат начала вхождения альтернативы в референсную последовательность и координат конца вхождения. Например, для схемы, приведенной на рисунке выше, такой вектор хранил бы структуру данных вида [”T”, 7, 9].

2. Создается граф из двух узлов. Первый узел содержит в себе референсную последовательность, второй узел служит признаком конца пути в графе, не хранит никакой нуклеотидной последовательности и имеет уникальный идентификатор, позволяющий с его помощью определять конец пути.

3. Для записи, соответствующей геномной альтернативе, программа находит узлы в графе, соответствующие координатам, указанным в описании варианта, после чего каждый найденный узел делится пополам в месте вхождения варианта.

4. Создается новый узел, хранящий альтернативную нуклеотидную последовательность, а также два ребра, соединяющие созданный узел с остальным графом.

Глобальным индексом на графе вариаций является хеш-таблица, содержащая в качестве ключей значения полиномиальной хеш-фукнции, вычисленные для каждого трека – короткой последовательности нуклеотидов, полученной прохождением по графу с заданной позиции на известное заранее число букв, а в качестве значений – координаты начала и конца трека, соответствующего значению хеш-функции.

Хеш-таблица строится с помощью модифицированного кольцевого хеша, обходящего все возможные пути заданной заранее длиной (далее – *hash_length*) от заданной позиции в графе. Существует возможность эффективно распараллелить данный алгоритм, так как вычисление хеша для разных узлов не приводит к одновременной записи каких-либо данных. Средняя сложность алгоритма построения индекса – O(n + 2^(*average_SNP_density***hash_length*)), где n – количество нуклеотидов, содержащихся в данном узле графа.

Базовый алгоритм локального выравнивания прочтения на граф работает следующим образом:

1. Для каждой позиции данного прочтения S вычисляется вектор, содержащий значения хеш-функции, высчитанные для каждой подстроки прочтения.

2. Каждой позиции i данного прочтения S ставится в соответствие вектор, хранящий координаты начала и конца тех треков, которым соответствует подстрока $S[i : i + hash\_length]$.

3. Начиная с позиции (i + *hash_length*), алгоритм проходит строку слева направо с шагом *hash_length*, на каждой итерации проверяя совпадение координат конца трека (одного или нескольких), соответствующего i-й позиции, и координат начала трека, соответствующего позиции (i + *hash_length*).

4. В случае если программа не находит совпадения, выполняется проход по графу от последней выровненной позиции на расстояние в две длины хеша и сохранение полученных координат в отдельный вектор. Затем выполняется проверка на совпадение координат в данном векторе с координатами начала трека, соответствующего позиции $i + 2 \cdot hash\_length$. Подобный ход позволяет «перепрыгнуть» фрагменты чтения, содержащие ошибки секвенирования. В дальнейшем пропущенный фрагмент будет подвергнут глобальному выравниванию с помощью алгоритмов динамического программирования для получения оценки качества выравнивания и отсеивания прочтения, если качество выравнивания прочтения не сможет превысить заранее определенный порог.

5. В случае если алгоритм дошел до конца строки, прочтение считается выровненным.

Как можно видеть из описания алгоритма, его сложность можно оценить как O((*match_counts*^2)·n/*hash_length*), где *match_counts* –количество найденных возможных позиций для подстроки прочтения, а n – длина прочтения.

Общеизвестным свойством алгоритмов локального выравнивания с использованием хешей является их низкая эффективность для обработки несовпадений. Чтобы иметь возможность выполнить выравнивание фрагментов прочтения, содержащее замены и вставки-делеции, используется модифицированный алгоритм Нидлмана–Вунша для выравнивания фрагментов прочтения, пропущенных картировщиком, использующим хеш-таблицу (раздел (4) базового алгоритма локального выравнивания прочтения на граф). Аналогичный подход используется в программе glia [15], предназначен-

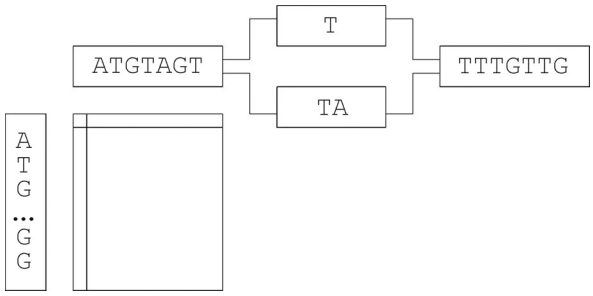

Рис. 2. Заполнение матрицы выравнивания прочтения на граф вариаций.

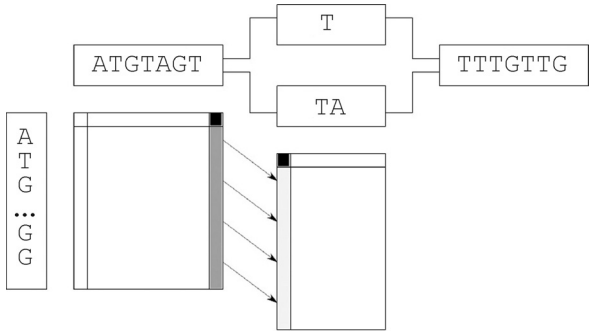

Рис. 3. Вычисление матрицы выравнивания прочтения на граф вариаций в случае узла с единственным предком. Светло-серым цветом выделен вычисляемый столбец, темно-серым – уже вычисленный последний столбец матрицы-предка.

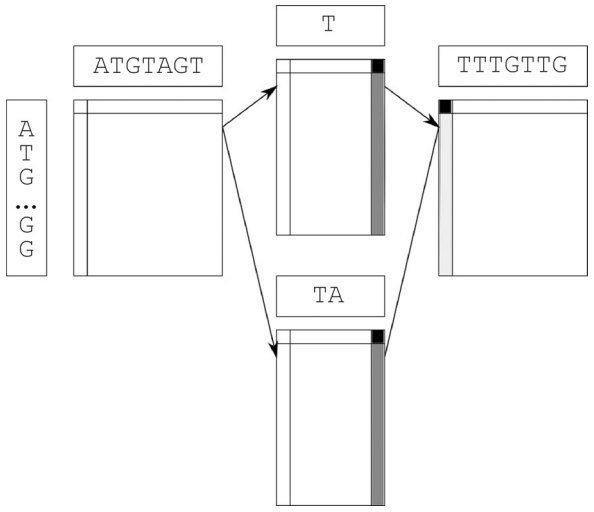

Рис. 4. Построение матрицы выравнивания, для узла, имеющего более одного предка. Ребра графа не показаны для наглядности иллюстрации. Светло-серым цветом выделен вычисляемый столбец, темно-серым – уже вычисленный последний столбец матрицы-предка.

ной для выравнивания коротких прочтений на геномные графы.

Предлагается модификация метода Нидлмана–Вунша, которая заключается в следующем: создается таблица выравнивания для каждого узла графа на участке, содержащем в себе интересующий нас трек, начало и конец которого определены из прохода по хеш-таблице глобального индекса.

При вычислении первого столбца таблицы возможны три случая:

(i). У узла нет «предков» (рис. 2). Этот случай тривиален, матрица заполняется тем же способом, что и в обычном алгоритме динамического программирования Нидлмана–Вунша.

(ii). У узла имеется единственный «предок» (рис. 3). В этом случае для инициализации построения матрицы выравнивания используется последний столбец уже вычисленной таблицы узла-«предка».

(iii). У узла имеется два (или более) «предка» (рис. 4). В этом случае происходит построение матрицы по следующему правилу: для вычисления первого столбца матрицы используются матричные элементы, соответствующие наибольшим весам выравнивания, взятые из соответствующих строк последних столбцов всех матриц выравнивания узлов-«предков», с учетом возможного штрафа за добавленные на границе матриц делеции-вставки или несовпадения. После определения узла, соответствующего наибольшему весу выравнивания, указатель на него добавляется в вектор, соответствующий первому столбцу вычисляемой матрицы выравнивания.

Выравнивание фрагмента прочтения начинается с последней заполненной матрицы, начиная с конца фрагмента геномного графа, содержащего интересующий нас трек. Когда алгоритм доходит до первого столбца матрицы, соответствующему узлу, имеющему двух и более предшественников, то выбирается тот узел, для которого соответствующая матрица-предшественник обеспечивает лучшую оценку выравнивания. Сложность данного блока алгоритма – $O(n \cdot m)$, где n – это длина фрагмента прочтения, а m – это средняя длина пути во фрагменте графа.

РЕЗУЛЬТАТЫ

В качестве тестовых последовательностей референсных геномов были выбраны сборки *E. coli* A39 [16] и K12 [17]. Они были обработаны с помощью пакетов *mauve* [18] и *mafft* [19], которые позволили построить структуру, со-

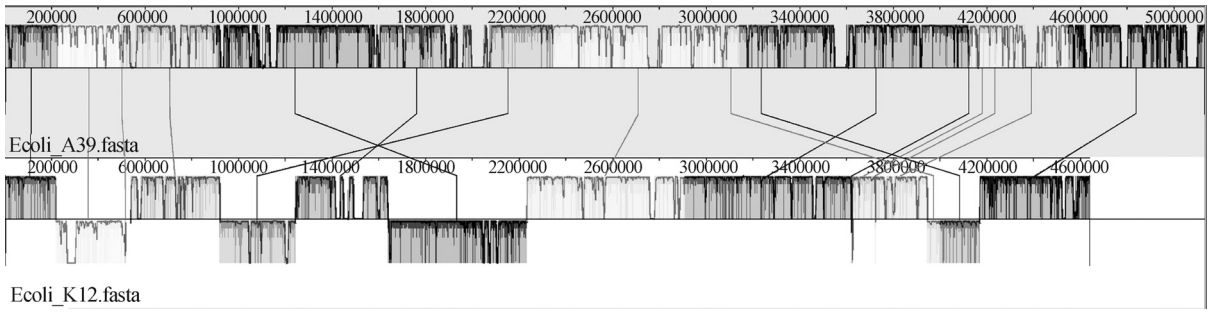

Рис. 5. Результаты работы программы mauve.

держащую референсную последовательность и набор вариантов.

Обработка происходила в следующем порядке:

1. С помощью программы mauve референсные последовательности были разделены на фрагменты, относящиеся к одной из двух категорий: (*i*) подпоследовательности, сходные друг с другом и отличающиеся не более чем несколькими однонуклеотидными альтернативами, и (*ii*) сильно отличающиеся подпоследовательности (в их число также попали последовательности, являющиеся результатом транслокации или инверсии последовательности генома) (рис. 5).

2. С помощью программы mafft для наборов сходных фрагментов последовательностей были выделены наборы однонуклеотидных альтернатив, отличающих их друг от друга.

3. Полученные результаты были преобразованы в комбинацию референсной последовательности и набора отличительных альтернатив. В качестве референсной последовательности была выбрана последовательность генома штамма A39.

Принципиально важно отметить, что на данном этапе обработке подвергаются все отличия одного генома от другого, включая как относительно мелкие альтернативы в несколько нуклеотидов, так и варианты длиной порядка сотни нуклеотидов. Таким образом, был построен геномный граф, на который производилось картирование прочтений. В качестве объектов тестирования были выбраны результаты секвенирования нескольких образцов *E. coli*, полученные по технологии Illumina [20]. Результаты их выравнивания на геномный граф приведены в таблицах ниже. Эти таблицы также содержат сравнение с результатами картирования, проведенного с помощью программ bowtie2 [21] и hisat2 [22] на последовательности геномов обоих штаммов, взятых по отдельности.

Как можно видеть из табл. 2, при выравнивании с одной разрешенной заменой и использовании дефолтных значений программы bowtie2 разработанный нами картировщик выравнивает меньшее количество прочтений, чем bowtie2. Тем не менее принципиально важным аспектом полученных результатов является наличие в данных секвенирования прочтений, являющихся комбинациями двух различных гаплотипов, что может быть важно для обработки данных по видам с консервативным геномом, например, для анализа индивидуальных геномов человека.

Таблица 1. Результаты точного выравнивания прочтений на геномный граф, построенного с помощью разработанного картировщика

Образец	Общее число уникальных прочтений	Число прочтений, выравненных на A39 (bowtie2)	Число прочтений, выравненных на K12 (bowtie2)	Число прочтений, выравненных на граф вариаций	Число новых выравненных прочтений
SRR5762429	7648422	1163234	4240086	4411742	169607
SRR5762418	7430793	1161318	4230660	4395274	162746
SRR5762419	8896595	1267501	4610771	4833116	219811

Примечание. Приведено сравнение с результатами выравнивания прочтений на графы исходных штаммов, построенных с помощью программы bowtie2, запущенной в режиме точного выравнивания.

Таблица 2. Результаты точного выравнивания прочтений на геномный граф, построенного с помощью разработанного картировщика, работающего в режиме не более чем одного несовпадения на прочтение

Образец	Общее число уникальных прочтений	Число прочтений, выравненных на A39 (bowtie2)	Число прочтений, выравненных на K12 (bowtie2)	Число прочтений, выравненных на граф вариаций	Число новых выравненных прочтений
SRR5762429	7648422	5808625	7383762	6042056	37677
SRR5762418	5676405	5676405	7197742	5937133	36824
SRR5762419	8896595	6720319	8558430	6860026	49816

Примечание. Приведено сравнение с результатами выравнивания прочтений на графы исходных штаммов, построенных с помощью программы bowtie2, запущенной с настройками по умолчанию.

Таблица 3. Результаты точного выравнивания прочтений на геномный граф, построенного с помощью разработанного картировщика, работающего в режиме не более чем одного несовпадения на прочтение

Образец	Общее число уникальных прочтений	Число прочтений, выравненных на A39 (hisat2)	Число прочтений, выравненных на K12 (hisat2)	Число прочтений, выравненных на граф вариаций	Число новых выравненных прочтений
SRR5762429	7648422	2652294	4178714	6042056	1921393
SRR5762418	5676405	2643025	4161995	5937133	1830054
SRR5762419	8896595	2849994	4477984	6860026	2431683

Примечание. Приведено сравнение с результатами выравнивания прочтений на графы исходных штаммов, построенных с помощью программы hisat2, запущенной с настройками по умолчанию.

Следует отметить, что программа hisat2 имеет возможность работы с полиморфизмами, однако сравнить функционал hisat2 с функционалом разработанной нами программы не представляется возможным ввиду требований к оперативной памяти – даже для короткого генома *E. coli*, содержащего менее 5 млн нуклеотидов, для работы hisat2 потребовалось больше одного терабайта оперативной памяти.

Опубликованная ранее программа vg [14] имеет функционал, сходный с программой, представленной в настоящей работе. Картировщик vg принимает на вход референсную последовательность и набор малых вариаций, содержащийся в файле в формате vcf [23]. По сравнению с нашим картировщиком vg имеет преимущество по скорости построения графа и индекса выравнивания. Тем не менее vg не позволяет загружать в граф вариации, длиной больше десятка нуклеотидов, в связи с чем многие прочтения, точно входящие в один из гаплотипов, не имеют возможности быть выравненными.

Для адекватного сравнения разработанного нами инструмента с программой vg было проведено два теста. Первый из них заключался в сравнении количества прочтений из тестовых наборов данных, выравненных на последовательность генома A39 с помощью каждой из программ, при рассмотрении вариаций, имеющих длину не более 10 нуклеотидов. Это было сделано для того, чтобы сравнить эффективность алгоритмов выравнивания, без учета ограничений, налагаемых особенностями хранения вариантов в программе vg. Результаты представлены в табл. 4.

Второй тест заключался в сравнении результатов выравнивания прочтений на последовательность генома штамма A39, выполненных с помощью программы vg и с помощью нового картировщика, осуществляющего выравнивание с графом вариаций, включающий полные геномы штаммов A39 и K12. Результаты приведены в табл. 5.

Таким образом, в результате работы была разработана и реализована структура данных, позволяющая обрабатывать последовательности геномов, представлять и хранить их в форме графа вариаций. Построен алгоритм, позволяющий картировать индивидуальные геномные

Таблица 4. Сравнение результатов выравнивания прочтений на последовательность генома штамма A39, для коротких вариаций для разработанной программы и программы vg

Образец	Число прочтений, выровненных с помощью vg	Число прочтений, выровненных с помощью разработанного картировщика	Улучшение результатов
SRR5762429	4053779	5027171	24%
SRR5762418	4036648	4948131	22,5%
SRR5762419	4403686	5706108	29,5%

Таблица 5. Сравнение результатов работы vg и разработанной программы при использовании полных геномов штаммов

Образец	Прочтений, выровненных с помощью vg	Прочтений, выровненных с помощью разработанного картировщика	Улучшение результатов
SRR5762429	4053779	6042056	49%
SRR5762418	4036648	5937133	47%
SRR5762419	4403686	6860026	55.7%

прочтения на разработанный геномный граф, сочетающий в себе выравнивание с помощью хеш-таблиц с выравниванием с помощью динамического программирования.

ОБСУЖДЕНИЕ РЕЗУЛЬТАТОВ

Выравнивание прочтений на геномный граф дает возможность учитывать наличие известных вариантов в геномных последовательностях. Учет возможных вариаций позволяет поднять требования к строгости выравнивания, снижая количество допустимых замен. Таким образом может быть повышена точность выравнивания каждого отдельного прочтения при сохранении достаточного большого количества выровненных прочтений, позволяющего обеспечить необходимое покрытие.

Разработанная программа демонстрирует эффективность картирования на граф вариаций, построенный из двух референсных геномов, по сравнению с картированием на каждую из последовательностей референсных геномов, взятую по отдельности. Из недостатков программы стоит отметить высокие требования к памяти: граф и глобальный индекс двух геномов штаммов *E. coli* занимают 5 гигабайт. Преимуществом программы является возможность работы с полными геномами штаммов, а не только с короткими вариациями на заданных участках.

Для использования программы в конкретных геномных исследованиях необходима ее доработка по следующим направлениям:

1. Оптимизация по времени работы и в особенности по объему используемой памяти. В настоящее время программа использует не самый оптимальный контейнер для хранения глобального индекса генома, поэтому затраты памяти могут достигать нескольких гигабайт на пять мегабайт исходного генома. Также имеет смысл ускорить программу путем распараллеливания алгоритмов на несколько потоков и оптимизации под использование графических ускорителей.

2. Оптимизация по чувствительности поиска. Специфика данных NGS, получаемых с помощью технологии Illumina, заключается в увеличении частоты ошибок секвенирования на концах прочтений. К сожалению, это может вызывать ошибки при проходе через хеш-таблицу прочтения, что приводит к снижению точности картирования в режиме неточного выравнивания. Достигнуть повышения производительности можно триммингом прочтений, а также использованием более коротких длин хеша.

3. Оптимизация основного алгоритма выравнивания для работы с данными секвенирования третьего поколения (Oxford Nanopore [24], PacBio [25], Ion Torrent [25]). Прочтения, полученные с использованием технологий SMRT (Single Molecule Real Time), отличаются принципиально иным профилем ошибок, характеризуемым более гладкой функцией распределения.

4. Увеличение количества исходных геномов. Сейчас граф может эффективно строиться только по двум геномам. В случае изменения принципа добавления новых тредов появится возможность эффективно хранить большее количество гаплотипов. Открытым вопросом остается эффективность разработанной структуры данных при значительном росте вариаций на длину хеша.

5. Поддержка стандартных форматов данных, таких как sam, bam, vcf, fastq, fastg, etc. Разработка эффективного и удобного формата данных, предназначенного как для хранения самого графа вариаций, не ограниченного длиной хранимых в узлах последовательностей, так и для хранения результатов выравнивания на такой граф.

6. Разработка программы-аннотатора, оптимизированного для работы с графом вариаций, хранящим наборы полных гаплотипов. Как следует из специфики базовой структуры данных, решение данной задачи тесно связано с разработкой эффективного индекса, позволяющего ориентироваться на геномном графе сколь угодно высокого уровня сложности.

В заключение можно сделать вывод о перспективности использования геномных графов вариаций в качестве основного формата представления референсных последовательностей. Однако для практического использования этих структур следует пройти еще достаточно долгий путь, в частности разработать эффективную схему хранения аннотаций для данных, не подразделенных на сегменты и поэтому неадекватных используемой сейчас концепции, реализованной в формате аннотации bed.

Авторы обязаны безвременно ушедшему от нас М.А. Ройтбергу, за предложение использовать хеш-таблицы для картирования прочтений на геномный граф, а также благодарны проф. Paul Flicek за ценные замечания по поводу существующих ограничений хранения геномной информации в виде геномных графов.

Исследование выполнено при финансовой поддержке Российского научного фонда (проект № 17-74-10013).

СПИСОК ЛИТЕРАТУРЫ

1. B. Paten, A. M. Novak, J. M. Eizenga, and E. Garrison, Genome Res. **27** (5), 665 (2017)
2. D. R. Zerbino and E. Birney, Genome Res. **18** (5), 821 (2008)
3. R. Chikhi and G. Rizk, Algorithms Mol. Biol. **8** (1), (2013)
4. R. Luo, B. Liu, Y. Xie, et al., GigaScience **1**, (2012)
5. P. Compeau, P. A. Pevzner, and G. Tesler, Nature Biotechnol. **29** (11), 987 (2011)
6. A. Bankevich, S. Nurk, D. Antipov, et al., J. Comput. Biol. **19** (5), 455 (2012)
7. K. Bradnam, J. Fass, A. Alexandrov, et al., GigaScience **2** (1), (2013)
8. A. M. Novak, E. Garrison, and B. Paten, Algorithms Mol. Biol. **12** (18), (2017)
9. S. Altschul, W. Gish, W. Miller, et al., J. Mol. Biol. **215** (3), 403 (1990)
10. H. Li and R. Durbin, Bioinformatics **25** (14), 1754 (2009)
11. P. Ferragina, F. Luccio, G. Manzini, et al., Journal of the ACM **57** (1), 1 (2009)
12. J. Siren, N. Valimaki, and V. Makinen, IEEE/ACM Trans. Comput. Biol. Bioinf. **11** (2), 375 (2014)
13. J. Sirén, arXiv, **1604.06605**, (2017)
14. GitHub [Электронный ресурс]: vg, 2014. URL: https://github.com/vgteam/vg (дата обращения: 10.03.2018).
15. GitHub [Электронный ресурс]: a Graph/Smith-Waterman (partial order) aligner/realigner, 2013. URL: https://github.com/ekg/glia (дата обращения: 10.03.2018).
16. NCBI [Электронный ресурс]: Escherichia coli IAI39 (E. coli) assembly, 2008. URL: https://www.ncbi.nlm.nih.gov/assembly/GCF_000026345.1 (дата обращения: 8.02.2018).
17. NCBI [Электронный ресурс]: Escherichia coli str. K-12 substr. MG1655 (E. coli) assembly, 2013. URL: https://www.ncbi.nlm.nih.gov/assembly/GCF_000005845.2 (дата обращения: 8.02.2018).
18. A. Darling, B. Mau, F. Blattner, et al., Genome Res. **14** (7), 1394 (2004)
19. K. Katoh and D. Standley, Mol. Biol. and Evolution **30** (4), 772 (2013)
20. J. Ju, D. Kim, L. Bi, et al., Proc. Natl Acad. Sci. USA **103** (52), 19635 (2006)
21. B. Langmead and S. Salzberg, Nature Methods **9** (4), 357 (2012)
22. D. Kim, B. Langmead, and S. Salzberg, Nature Methods **12** (4), 357 (2015)
23. Samtools organisation and repositories [Электронный ресурс]: The Variant Call Format (VCF) Version 4.2 Specification, 2017, URL: https://samtools.github.io/hts-specs/VCFv4.2.pdf (дата обращения: 20.03.2018).
24. M. View, H. Olsen, B. Paten, et al., Genome Biol. **17** (1), 239 (2016)
25. H. Buermans and J. den Dunnen, Biochim. Biophys. Acta **1842** (10), 1932 (2014).

An Efficient Algorithm for Read Mapping Against Graph-Based Reference Using Hash-Based Index and Dynamic Programming

S.N. Petrov* **, L.A. Uroshlev* **, A.S. Kasyanov* **, and V.Yu. Makeev* ** * ****

**Vavilov Institute of General Genetics, Russian Academy of Sciences, ul. Gubkina 3, Moscow, 119991 Russia*

***Moscow Institute of Physics and Technology (State University), Institutskiy per. 9, Dolgoprudny, Moscow Region, 141701 Russia*

****Engelhardt Institute of Molecular Biology, Russian Academy of Sciences, ul. Vavilova 32, Moscow, 119991 Russia*

*****Research Institute for Genetics and Selection of Industrial Microorganisms, Scientific Center "Kurchatov Institute", Pervyj Dorozhniy proezd 1, Moscow, 117545 Russia*

We discuss the issue concerning the storage of the sequences of a series of closely related genomes and the analysis of genome variations. A genome graph with a structure of a directed acyclic graph is used to store the matching segments of several genome sequences and the known variants. We have proposed an efficient read mapping algorithm for a genome graph to align the identical fragments of nucleotide sequences at the genome graph. The algorithm combines the fast genome graph scanning via hash tables and the dynamic programming alignment procedure and avoids the exponential increase of paths during graph searching. The program for realization of the genome graph and read alignment algorithm procedures is developed. A comparison between this program and a number of existing tools having similar functionality is performed.

Keywords: genome graph, variance graph, alignment, mapping