

===== БИОИНФОРМАТИКА =====

Применение суффиксных массивов для обнаружения повторов в геномных последовательностях

Назипова Н.Н.

Институт математических проблем биологии РАН – филиал Института прикладной математики Российской академии наук, Пущино, Россия

Аннотация. В последние годы исследователям становятся доступными геномы высших эукариот без пятен неопределенности. Это так называемые T2T (telomere-to-telomere) сборки. По сравнению с референсными геномами, изучавшимися до эры T2T, они состоят из минимального количества несобранных участков или не содержат их вовсе. Теперь стала доступной исчерпывающая информация о теломерах, центромерах, генах рибосомной РНК, сложных хромосомных участках и транспозонных элементах. Для нахождения всех без исключения повторов в геномных последовательностях нужны новые вычислительные подходы. В этой статье мы описали часть алгоритмических проблем, которые были решены нами в ходе работы над технологией обнаружения *de novo* длинных размытых повторяющихся участков в огромных символьных массивах, которыми представляются геномные последовательности. Многообещающими являются подходы, которые используют k -мерный состав анализируемой последовательности. В статье обсуждается важность подбора адекватного значения k , поскольку результат работы программы поиска зависит от этого ключевого параметра алгоритма.

Ключевые слова: повторы в ДНК, суффиксные и префиксные деревья, суффиксный массив, преобразование Барроуза-Уиллера, k -мерный словарь

ВВЕДЕНИЕ

Быстрое совершенствование технологий секвенирования позволяет проводить гораздо более глубокие и сложные исследования генома. Уже обнаружено, что повторяющиеся участки ДНК, или повторы ДНК, составляют значительную долю всего генома эукариот. Например, около 85% генома кукурузы состоит из повторов ДНК [1]. Более новое исследование предполагает, что повторы могут покрывать более 66% генома человека [2], хотя, согласно результатам проекта «Геном человека» [3], их доля оценивалась в 50%. ДНК-повторы изначально назывались мусорной или «эгоистичной» ДНК [4], «темной материей», поскольку большинство из них находится в не кодирующих областях. Для обозначения этой части генома даже придумали термин *репитом* (repeatome) [5] по аналогии с другими «омами» биоинформатики. Надо признать, что большая его часть остается недостаточно изученной, плохо аннотированной и функционально загадочной. Для больших участков повторяющихся последовательностей внутри репитома мы по-прежнему не уверены, являются ли они «мусорной ДНК», или их эволюционно приобретенные функции просто остаются непрозрачными на данный момент времени.

Однако большое количество современных исследований показало, что эти неизученные последовательности могут вносить вклад в эволюцию и регуляцию генома

[6, 7]. Многие заболевания человека также связаны с нарушениями повторов ДНК [8]. Ключевым компонентом репита являются тандемные повторы, которые по оценкам составляют более 3% генома человека. В геноме человека насчитывается более 1,5 миллионов коротких тандемных повторов (STR), которые состоят из повторяющихся единиц мотивов ДНК длиной от 1 до 6 пар оснований, и последние данные подтверждают их роль в ряде молекулярных и клеточных процессов [9,10].

С развитием технологий секвенирования, особенно секвенирования третьего поколения, можно получать более полные геномы, используя гораздо большую длину прочтения. Это позволяет выявить не определенные в ранних сборках темные области генома [11], где некоторые новые повторы могли остаться необнаруженными. Между тем, растущее число секвенированных геномов различных видов, включая T2T сборки [12], делает насущной необходимость комплексного обнаружения и анализа повторов ДНК в геномах.

Среди существующих инструментов для идентификации повторов в ДНК есть методы, основанные на библиотеках, на обходе суффиксного дерева, на алгоритмах сравнения участков и методы, основанные на анализе k -меров [13].

RepeatMasker [14] — наиболее широко используемый инструмент, основанный на библиотеке, который идентифицирует повторы путем поиска гомологии с помощью вручную составляемой базы данных повторяющихся элементов Repbase [15]. Для аннотации обнаруженных новых повторов такой инструмент очень полезен. Однако инструмент, основанный на депозитории, не обеспечивает возможности идентифицировать неаннотированные повторы.

Repeat Pattern Toolkit [16], RECON [17] и PILER [18] — это три инструмента, которые обнаруживают семейства повторов путём парного локального выравнивания. Эти инструменты относительно более точны, но требуют больше времени.

REPuter [19, 20] — это один из первых программных пакетов, работающих на использовании суффиксных деревьев для обнаружения всех точных повторов в геноме. Пакет программ REPuter состоит из двух частей. Первая, REPfind представляет собой эффективную реализацию суффиксного дерева для поиска точных повторов в линейном пространстве и времени. Найденные участки используются в качестве затравок для поиска неточных повторов. Помимо вырожденных прямых повторов, REPfind способен обнаруживать вырожденные палиндромные повторы. Полученные результаты упорядочиваются по значимости. Вывод поисковой системы REPfind отображается в виде графика повторений с помощью интерактивной программы визуализации REPvis. Это вторая программа, входящая в пакет.

Инструменты, использующие обход суффиксного дерева REPuter и RepeatMatch (создан на основе MUMmer [21]) – высокоэффективные вычислительные инструменты, способные находить точные повторы в последовательностях длиной до 10–100 Мб в полных эукариотических хромосомах. Подход RepeatFinder [22] реализует аккуратную кластеризацию данных о повторах, собранных в суффиксных деревьях. Этот метод встроен в систему, которая может находить повторы в отдельных последовательностях генома или наборах последовательностей и группировать эти повторы в классы. Ограниченность методов, основанных на обходе суффиксных деревьев, состоит в том, что они не могут находить неточные повторы.

Для выявления неточных повторов некоторые инструменты, такие как RepeatScout [23] и Repseek [24], используют высокочастотные k -меры в качестве затравок и расширяют их до консенсусных повторов для определения семейств повторов. Однако RepeatScout не обнаруживает все повторяющиеся последовательности напрямую. Консенсусные повторы следует использовать в качестве библиотеки для RepeatMasker для поиска по всему геному с целью определения положения повторов.

Spectral Repeat Finder [25] и RepeatGluer [26] являются двумя инструментами, которые генерируют консенсусные последовательности на основе преобразования

Фурье и графа де Брёйна соответственно. Программа FORRepeats [27] использует структуру данных, называемую *factor oracle*, для первоначального обнаружения точных повторов, а затем вычисляет приблизительные повторы. Некоторые более современные инструменты, такие как RAIDER [28] и его более новая версия phRAIDER [29], используют новый алгоритм, включающий модель PatternHunter с интервальным расположением затравок для улучшения идентификации элементарных повторов (определенных Чжэном и Лонарди [30]), что похоже на мозаичные узоры, используемые RepeatGluer.

Generic Repeat Finder (GRF) [31] предназначен для выявления терминальных инвертированных повторов и терминальных прямых повторов, характерных для ретротранспозонов, а также вкрапленных повторов на основе быстрых алгоритмов численных вычислений и стратегий динамического программирования.

Другие инструменты, включая Merengine [32], WindowMasker [33], Tallymer [34], Red [35] и RepLoc [13] генерируют оценки повторяемости для фрагментов генома, преобразованные из частот k -меров, чтобы напрямую определять повторяющиеся последовательности. Merengine основан на подсчете k -меров, позволяющих построить карту повторяемости последовательности (Rmap) и «геномный ландшафт». Упомянутые инструменты состоят из трех ключевых этапов: подсчет k -меров, оценка повторяемости и идентификация повторов.

Специально созданной для поиска tandemных повторов является TRF [36]. Программа ищет tandemные повторы с длиной паттерна периодичности до 2000 нуклеотидов. Она для каждого из 4^k слова длины k ($k = 4 \div 7$) строит вектор с координатами их точных вхождений. При добавлении новой позиции в строящийся вектор сканируются все более ранние вхождения с использованием списка расстояний между ранними k -мерами для нахождения тех из них, которые отстоят друг от друга примерно на расстояние d . Для массива выбранных k -мер и промежуточных участков между ними затем строится множественное выравнивание методом обертывающего динамического программирования (wraparound dynamic programming) [37] с длиной образца d . Если оно имеет приемлемое качество, то массив назначается неточным tandemным повтором с длиной паттерна $\approx d+k$.

Мы разработали свой метод поиска повторов, который находит длинные повторы (разнесенные и tandemные) в длинных последовательностях [38]. Также, как и TRF, мы сначала ищем вхождения слов длины k и потом пытаемся расширить эти затравки. Отличие нашего подхода состоит в том, что мы ищем все неточные вхождения длинных 21-мер, с минимальным уровнем совпадения между образцом и его копиями в геноме в 81 %. Мы тоже расширяем их потом в разные стороны. Если удлиненные участки, содержащие внутри себя k -мер, смыкаются, то получается tandemный повтор, если нет – разнесенный. Детали алгоритма здесь мы опускаем, потому что темой этой статьи является работа с суффиксным массивом геномной последовательности, использование которого позволяет решать различные задачи. Мы используем в своей технологии поиска повторов суффиксный массив по меньшей мере трижды. В первый раз – для построения k -мерного словаря геномной последовательности. Второй раз – при определении координат неточных вхождений копий для каждого слова из k -мерного словаря. В третий раз мы используем суффиксный массив для определения минимальной длины паттерна периодичности для участков, имеющих признаки tandemных повторов. Большинство алгоритмов, разрабатываемых в последнее время по изучению повторяемости в геноме, используют так или иначе k -меры.

Впервые обратный поиск в суффиксном массиве с использованием преобразования Барроуза-Уиллера, был реализован в программе BWA (Burrows – Wheeler Aligner) [39] выравнивания последовательности прочтений генома (reads) на референсную геномную последовательность при сборке геномов по данным секвенирования. Алгоритм BWA основан на нахождении всех копий слова в символьной последовательности с

использованием преобразования Барроуза–Уилера (BWT, Burrows – Wheeler Transform) [40] и может эффективно имитировать нисходящий обход префиксного дерева генома с использованием относительно небольшого объёма памяти и определять координаты точных вхождений строки длины m за время $O(m)$ независимо от размера генома. При неточном поиске BWA выбирает из префиксного дерева отдельные подстроки, находящиеся друг от друга на расстоянии, не превышающем k несовпадений от искомого образца.

ПРЕОБРАЗОВАНИЕ БАРРОУЗА-УИЛЛЕРА И СУФФИКСНЫЕ МАССИВЫ

Наш метод поиска размытых длинных повторов в геномной последовательности использует словарь k -меров для исследуемой текстовой последовательности X с исключением из него слов, которые встречаются в тексте меньше двух раз. Словарь – это лексикографически упорядоченный массив k -мер, которые встречаются в X не менее 2 раз. Затем для каждого k -мера вычисляется массив координат всех неточных вхождений в геномную последовательность. Порог точности совпадения вхождения и образца (k -мера) определяется заданием числа разрешенных несовпадений z .

Алгоритмы поиска вхождений слова в символьную последовательность издавна создавались математиками и специалистами компьютерных наук. Со временем они нашли себе применение в биоинформатических приложениях. Задача локального выравнивания двух геномных последовательностей подразумевает нахождение в двух последовательностях общего участка с заданной степенью совпадения. Наиболее популярными являются Z-алгоритм, алгоритм Ахо-Корасика, алгоритмы обхода суффиксных деревьев. Z-алгоритм [41] работает с одним образцом. Для поиска множества фрагментов в тексте используются деревья ключевых слов. Метод поиска называется алгоритмом Ахо-Корасика [42]

Суффиксные деревья – это структура данных, которая в свое время была средством для решения более сложных задач поиска строк. Для символьной строки X $|X| = n$ это дерево, в котором каждое ребро помечено символом из алфавита, на котором задана строка. Внутренние узлы располагаются на горизонтальных уровнях. Каждый внутренний узел помечается числом начал суффиксов в строке X , которые равны конкатенации меток ребер, ведущих в этот узел из корневого узла, и имеют длину, равную номеру уровня. Конкатенация символов, соответствующих меткам вершин на пути от корня до листа, определяет уникальный суффикс строки X . Все листовые узлы имеют метку 1.

Несмотря на то, что со временем наивный способ построения, существовавший со времен Д. Кнута, который ввел в обиход понятие суффиксного массива [43], имевший квадратичную временную сложность построения, был вытеснен тремя основными алгоритмами [44–46] с линейным временем работы, этот подход не выигрывал по сравнению Z-алгоритмом. Но, в его пользу был тот факт, что, как только суффиксное дерево было построено, в нем можно было искать не только суффиксы, но и много всего.

Префиксное дерево для символьной строки X – это дерево, в котором каждое ребро помечено символом, а конкатенация символов, соответствующих ребрам на пути от листа к корню дает уникальный префикс строки X . В префиксном дереве конкатенация меток ребер от любого узла к корню дает уникальную подстроку X . Префиксное дерево для X идентично суффиксному дереву строки X' , являющейся палиндромом X , поэтому теория суффиксных деревьев [47] применима и к префиксным деревьям.

При использовании префиксного дерева проверка того, является ли слово W точной подстрокой X , эквивалентна поиску узла, представляющего W , что можно сделать за время $O(|W|)$ путём сопоставления каждого символа в W с ребром, начиная с корня. Чтобы разрешить несовпадения, мы можем полностью обойти префиксное дерево и сопоставить W с каждым возможным путём.

Задача поиска координаты вхождения фрагмента в длинную текстовую последовательность, даже с учетом размытости целевого фрагмента, не представляла бы большой трудности, если бы не проблема больших геномных данных. Построение и работа с деревьями – это алгоритмически не самая простая задача, поэтому появление концепции суффиксного массива, более эффективной в указанных аспектах структуры данных, было прорывным шагом вперед. Суффиксный массив [48] представляет собой классическую структуру полнотекстового индекса, причем, будучи массивом целых чисел, определяющим лексикографический порядок суффиксов X , является эффективной с точки зрения использования памяти альтернативой деревьям суффиксов.

Преобразование Барроуза-Уиллера

Для объяснения идеи BWT введем обозначения. Пусть $A = \{a_0, \dots, a_{m-1}\}$ – алфавит длины m , на котором задана строка $X = x_0, \dots, x_{n-1}$ длины $|X| = n$. Символ $\$$ отсутствует в A , он считается лексикографически меньшим всех символов a_i , и строка X всегда заканчивается символом $\$$ который может появиться только в конце строки X ($a_{n-1} = \$$).

Обозначим i -й символ строки X $X[i] = x_i$, тогда $X[i, j] = a_i \dots a_j$ – это подстрока, а $X_i = X[i, n-1]$ – суффикс X . Массив суффиксов S из X – это перестановка целых чисел $\{0, \dots, n-1\}$, такая, что $S(i)$ – начальная позиция суффикса X_i .

Преобразование Барроуза-Уиллера текстовой строки X осуществляется последовательностью шагов.

Сначала конструируется матрица Барроуза-Уиллера, строки которой содержат все циклические перестановки строки X :

$$\begin{array}{l} X_0 X_1 X_2 \dots X_{n-3} X_{n-2} \$, \\ X_1 X_2 \dots X_{n-3} X_{n-2} \$ X_0, \\ \dots \\ X_{n-2} \$ X_0 X_1 X_2 \dots X_{n-3}, \\ \$ X_0 X_1 X_2 \dots X_{n-2}. \end{array}$$

Из строк матрицы Барроуза-Уиллера получается массив суффиксов:

$$\begin{array}{l} X_0 X_1 X_2 \dots X_{n-3} X_{n-2} \$, \\ X_1 X_2 \dots X_{n-3} X_{n-2} \$, \\ \dots \\ X_{n-2} \$, \\ \$, \end{array}$$

которые упорядочиваются лексикографически. Элементом вектора B с номером j является символ, предшествующий в исходной строке X началу j -го суффикса в упорядоченном лексикографически массиве суффиксов. Тогда массив суффиксов – это вектор, j -ый элемент которого является номером позиции, с которой в строке X начинается j -ый суффикс.

В таблице 1 показано, как строится BWT-строка B и массив суффиксов S для строки $X = \text{cattattagga}\$$, заданной над нуклеотидным алфавитом $A = \{a, c, g, t\}$. В колонке 2 последовательно выписаны все возможные циклические перестановки строки X , а в колонке 3 – все суффиксы строки X . В колонке 4 суффиксы из колонки 3 расположены в лексикографическом порядке. В колонке 5 – элементы BWT-строки B . В колонке 6 приведены элементы суффиксного массива, это номера позиций, с которых в исходной позиции начинается соответствующий суффикс.

Таблица 1. Пример построения суффиксного массива S и BWT-строки B

Поз. i	Матрица Барроуза- Уиллера	Суффиксы	Лексикографи- чески упорядоченные суффиксы	B	S
1	2	3	4	5	6
0	cattattagga\$	cattattagga\$	\$	a	0
1	attattagga\$c	attattagga\$	a\$	g	11
2	ttattagga\$ca	ttattagga\$	agga\$	t	8
3	tattagga\$cat	tattagga\$	attagga\$	t	5
4	attagga\$catt	attagga\$	attattagga\$	c	2
5	ttagga\$catta	ttagga\$	cattattagga\$	\$	1
6	tagga\$cattat	tagga\$	ga\$	g	10
7	agga\$cattatt	agga\$	gga\$	a	9
8	gga\$cattatta	gga\$	tagga\$	t	7
9	ga\$cattattag	ga\$	tattagga\$	t	4
10	a\$cattattagg	a\$	ttagga\$	a	6
11	\$cattattagga	\$	ttattagga\$	a	3

На практике можно сначала построить суффиксный массив строки X , а затем построить с его помощью элементы BWT-строки B , $|B|=|X|=n$, воспользовавшись формулой:

$$B(i) = \begin{cases} \$, & \text{если } S(i) = 0, \\ X[S(i) - 1], & \text{если } S(i) \neq 0. \end{cases} \quad (1)$$

Использование суффиксного массива и BWT-строки для поиска координат вхождений слова W

Основное свойство суффиксного массива состоит в том, что, если строка W является подстрокой X , номера позиций, в которых начинаются вхождения подстроки, будут входить в некий интервал суффиксного массива. Это потому, что все суффиксы, начинающиеся с W , в отсортированном массиве лежат вместе. Основываясь на этом наблюдении, определяются переменные [39]:

$$\begin{aligned} \underline{R}(W) &= \min(k : W \text{ есть префикс } X_{S(k)}), \\ \bar{R}(W) &= \max(k : W \text{ есть префикс } X_{S(k)}). \end{aligned}$$

В частности, если W есть пустая строка, то $\underline{R}(W) = 1$ и $\bar{R}(W) = n - 1$. Интервал $[\underline{R}(W), \bar{R}(W)]$ называется интервалом суффиксного массива (SA-интервалом) для подстроки W , и все вхождения подстроки W в X есть интервал элементов суффиксного массива $\{S(k) : \underline{R}(W) \leq k \leq \bar{R}(W)\}$. Например, SA-интервал для подстроки 'att' из примера, показанного на рисунке 2, будет [3,4], а номера позиций X , в которых встречается 'att' – это 4 и 1. Таким образом, задача поиска координат точного вхождения слова в текстовую последовательность состоит в определении одного SA-интервала. А для неточных вхождений нужно искать конкатенацию нескольких интервалов суффиксного массива.

В работе [49] предложен элегантный способ поиска SA-интервалов. Если $C(a)$ – число символов в X $[0, n-2]$, которые лексикографически меньше символа $a \in A$ и $O(a, i)$ – число вхождений a в B $[0, i]$, $i=1, \dots, n-2$, то для подстроки W авторы доказали, что верны соотношения:

$$\underline{R}(aW) = C(a) + O(a, \underline{R}(W) - 1) + 1,$$

$$\bar{R}(aW) = C(a) + O(a, \bar{R}(W)),$$

и что $\underline{R}(aW) \leq \bar{R}(aW)$ тогда и только тогда, когда aW является подстрокой X . Это означает, что можно считать координаты вхождений подпоследовательности W любой длины в X , рассчитывая значения границ интервала суффиксного массива, начиная с последнего символа W . Этот подход получил название *обратного поиска* (backward search) [39].

Оригинальный алгоритм обратного поиска [39] ищет все вхождения образца в текстовую строку с различными видами несовпадений, включая вставки и выпадения символов. Мы использовали его для поиска вхождений слов с z заменами без вставок/выпадений и адаптировали его для этой цели. Кроме того, мы не проверяем предварительно, входит ли вообще слово W в X , потому что мы ищем в геноме слова из словаря этой же последовательности. Поэтому в оригинальный алгоритм были внесены необходимые правки.

Алгоритм обратного поиска слова W в строку X с не более, чем z ошибками требует предварительного вычисления суффиксного массива для исходной строки и массива $C(a)$ со значениями количеств всех букв $a \in A$ в строке. Далее нужно построить массивы B и $O(a, i)$, $\forall a \in A, i=1, \dots, n-2$. Строка X заканчивается символом $\$$, а слово W – одним из символов алфавита A . Поиск начинается с конца слова W . Рекуррентный алгоритм для поиска в длинной строке X всех вхождений слова W с не более, чем z заменами выглядит так:

```

Algorithm InexactSearch ( $W, X, z$ );
begin
    return InexRecur ( $W, |W|-1, z, 1, |X|-1$ );
end

procedure InexRecur ( $W, i, z, k, l$ )
begin
    if  $z < 0$  then
        return  $\emptyset$ ;
    if  $i < 0$  then
        return  $[k, l]$ 
     $I := \emptyset$ ;
    for each  $a \in \{a, c, g, t\}$  do
        begin
             $k := C(a) + O(a, k-1) + 1$ ;
             $l := C(a) + O(a, l)$ ;
            if  $k \leq l$  then
                if  $a = W[i]$  then
                     $I := I \cup \text{InexRecur}(W, i-1, z, k, l)$ ;
                else
                     $I := I \cup \text{InexRecur}(W, i-1, z-1, k, l)$ ;
            end
        end
    return  $I$ ;
end

```

Построение суффиксного массива

Построение суффиксных массивов для очень длинных символьных строк «наивным способом» затратно по времени при использовании любого линейного алгоритма сортировки строк. Суффиксный массив S для геномной последовательности, длина которой исчисляется десятками и сотнями миллионов н.п., можно строить путем разбиения его на разделы, чтобы делать для ускорения сортировку по разделам.

Пусть $A = \{a_0, a_1, a_2, a_3\} = \{a, c, g, t\}$ – алфавит ДНК, на котором задана геномная последовательность $X = x_0, \dots, x_{n-1}$ длины $|X| = n$. Массив S – это целочисленный вектор длины n , упорядоченный так, что суффикс X_{S_j} лексикографически меньше или равен $X_{S_{j+1}}$. В каждом суффиксе выделяется его начало, подпоследовательность длины l , и продолжение, которое начинается с $(l+1)$ -й позиции суффикса. Всего разделов $p = 4^l$.

При загрузке геномной последовательности в память создается вспомогательный целочисленный массив C длины p . При его построении происходит перекодировка символьного l -мера в число. Любой l -мер $X[i, i+l-1]$ можно представить в виде числа num в позиционной системе счисления с основанием 4 следующим образом:

$$num_i = \text{ind}(x_i) \cdot 4^{l-1} + \text{ind}(x_{i+1}) \cdot 4^{l-2} + \dots + \text{ind}(x_{i+l-2}) \cdot 4^1 + \text{ind}(x_{i+l-1}),$$

где $\text{ind}(x_i)$ – функция, которая выдает порядковый номер буквы x_i в алфавите A . При этом оказывается, что число num_i есть номер i -го l -мера в лексикографическом списке l -мер с нумерацией от 0.

Продemonстрируем это на примере для случая $l = 3$. Сначала запишем все $4^3 = 64$ триплета в лексикографическом порядке:

```
aaa aac aag aat aca acc acg act aga agc agg agt ata atc atg att
caa cac cag cat cca ccc ccg cct cga cgc cgg cgt cta ctc ctg ctt
gaa gac gag gat gca gcc gcg gct gga ggc ggg ggt gta gtc gtg gtt
taa tac tag tat tca tcc tcg tct tga tgc tgg tgt tta ttc ttg ttt.
```

Тогда для триплета **gct** $num = 2 \cdot 4^2 + 1 \cdot 4 + 3 = 37$.

Если число num_i для i -го l -мера определяет его порядковый номер в массиве $C = c_0 c_1 \dots c_{p-1}$, то в самом элементе можно хранить количество таких l -мер в геноме, то есть размер раздела.

Мы группируем все суффиксы, начинающиеся с одного и того же l -мера, в один раздел суффиксного массива, всего разделов будет p . Короткими суффиксами длиной меньше l , которые получаются на конце геномной последовательности, можно в нашем случае пренебречь. Использование разделенного суффиксного массива поможет сэкономить время, потому что сортировка строк разделе будет производиться, начиная с l -го символа суффикса.

Для разметки в суффиксном массиве начал разделов понадобится еще один целочисленный массив $IC = ic_0, ic_1, \dots, ic_{p-1}$, элементы которого $ic_i = \sum_{j=0}^{i-1} c_j$ считаются

после заполнения массива C . Элемент массива IC с номером i является указателем в массиве S на местоположение первого суффикса, который начинается l -мера $X[i, i+l-1]$, то есть началом i -го раздела в суффиксном массиве S . Массивы прямого доступа IC и C придется хранить в оперативной памяти, поэтому выбор значения l должен быть небольшим (например, $l = 13$ для 4-буквенного алфавита).

Этот подход может хорошо ускорить построение суффиксного массива.

Проблема выбора значения k

В геномном контексте k -меры представляют собой подстроки нуклеотидов длиной k , содержащиеся в последовательности ДНК или РНК. Любую геномную последовательность можно разложить на ряд пересекающихся последовательных k -меров, так, что следующий k -мер начинается во второй позиции предыдущего. Количество k -меров в последовательности X длиной $n = |X|$ равно $n - k + 1$.

В настоящее время наблюдается тенденция к расширению применения k -меров в приложениях разных «омик». Концепция использования k -мер получила широкое распространение в 2000-х годах с выпуском хороших геномных сборщиков,

выравнивателей прочтений и специализированного программного обеспечения для подсчета k -меров, такого как Jellyfish [50].

Впервые анализ размера генома на основе k -меров был предложен Ли и Уотерманом в 2003 году [51], которые использовали данные секвенирования полного генома методом дробовика (WGS) для определения его общего размера и доли повторов. Подход включает в себя: 1) создание достаточно большого набора данных коротких подпоследовательностей (k -меров) из случайных участков генома, чтобы геном был представлен несколько раз; 2) определение среднего количества появлений этих подпоследовательностей в наборе данных с помощью анализа статистического распределения (например, гистограммы k -меров); и 3) деление общего количества подпоследовательностей на это среднее значение.

Выбор размера k -мера является сложной задачей для большинства инструментов обнаружения повторов, основанных на подсчёте k -меров [13, 52, 53]. Для последовательности длиной L в некоторых методах часто используется «универсальная» длина k -мера, равная $\log_4(L)$ [35, 54, 54], в этом случае для генома человека длина k -мера будет 15–16 пар нуклеотидов (п.н.).

«Наилучшее» значение k для анализа геномных последовательностей в конечном итоге определяется компромиссом между длиной k и ошибкой секвенирования: более длинные k -меры с большей вероятностью представляют уникальные геномные последовательности, но также с большей вероятностью содержат ошибку секвенирования [56]. На практике во многих исследованиях используют k из диапазона 20–40 п.н. [57], поскольку k -меры в этом диапазоне можно надёжно секвенировать с помощью коротких прочтений, и они часто однозначно выравниваются с исходным геномом [58, 59]. Например, $k = 32$ охватывает 85,7% уникальных последовательностей в геноме человека [60], тогда как $k = 21$ различает многие виды эукариот, бактерий и архей [61]. Существует множество исследований, предлагающих разные критерии выбора конкретных значений k [53].

Один из методов «грубой силы» для проверки чувствительности результатов к k заключается в простом повторении анализа несколько раз для разных значений k . Этот подход особенно распространён среди алгоритмов сборки генома [62], но теоретически может быть применён практически к любому анализу. Однако его главный недостаток — высокая вычислительная нагрузка, связанная с многократным выполнением одного и того же анализа. Кроме того, без дополнительной информации сложно определить, дают ли анализы, выполненные для определённых значений k , более точные результаты, чем анализы для других значений k , что обуславливает необходимость минимизации или максимизации критериев отбора k .

Авторы подхода Repeat Locator (RepLoc), который строит покрытие последовательности взвешенными k -мерами для количественной оценки повторяемости генома и локализации повторяющихся последовательностей [13], провели методом «грубой силы» исследование влияния выбора значения k на результат. Покрытие генома в геноме человека оказалось чрезвычайно высоко в случае «универсальной» длины k -мера (~95% для $k = 16$), а большинство участков определяются как повторяющиеся последовательности, что означает высокое количество ложноположительных результатов. Когда длина k -мера достигает 20, покрытие уменьшается до реалистичного значения ~56%. Покрытие генома кукурузы получилось стабильно высоким для всех k -меров в диапазоне 10–50, поскольку геном кукурузы известен своей повторяемостью. Авторы [13] пришли к выводу, что длина $k = \lfloor \log_4 L \rfloor + 5$ будет подходящей для обработки большинства геномов с использованием RepLoc. В этом исследовании значение $k = 18$ выбрано для геномов *A. thaliana* и *D. melanogaster*, 19 — для *O. sativa* и 20 — для *D. rerio*, *M. musculus* и *H. sapiens*. Для кукурузы *Z. mays* размер k -мера также был

установлен равным 20, производительность RepLoc была удовлетворительной, хотя покрытие 20-мерами составляет ~88 %.

ЗАКЛЮЧЕНИЕ

Платформы NGS (секвенирование следующего поколения) предлагают возможность генерировать большие объемы данных ДНК-секвенирования за короткое время и с низкими затратами. Однако анализ колоссальных объемов данных ДНК-секвенирования по-прежнему является вычислительной проблемой. Исследователи предложили множество методов повышения производительности инструментов и приложений анализа данных ДНК-секвенирования.

Один из методов заключается в запуске этих инструментов и приложений на высокопроизводительных вычислительных архитектурах, таких как суперкомпьютеры, кластеры и даже облачные среды. Другой метод заключается в использовании ускорителей, таких как графические процессоры и программируемых логических интегральных схем (ПЛИС), для ускорения трудоемких ядер этих инструментов и приложений с целью повышения их производительности. Выбор инструмента в этом случае зависит от задачи. Если решается задача быстрого поиска ответа в реальном времени, то выбирается второй путь реализации алгоритмов. А если производится одноразовый анализ геномной последовательности, то удобно и целесообразно из соображений экономии пользоваться компьютерным кластером средней мощности. В любом случае нужно использовать алгоритмы и структуры данных, которые обеспечивают приемлемые времена расчетов. При этом нужно еще правильно осуществлять выбор значений параметров алгоритма, которые обеспечивают получение правдоподобных результатов, как было показано выше на примерах подбора длины слова k при оценке степени повторяемости генома.

Появление геномов высших эукариот без пятен неопределенности (T2T-сборки) позволяет находить все без исключения повторы в геномных последовательностях. Нами создана вычислительная процедура полногеномного сканирования для выявления протяженных tandemных и разнесенных повторов. В этой статье мы описали часть алгоритмических проблем, которые были решены в ходе работы. Наша процедура также, как популярный алгоритм TRF [36] поиска tandemных повторов, находит ядра повторов в виде вхождений 21-грамм с разрешенным количеством несовпадений, которые максимально расширяются право и влево. При этом не применяется никакой эвристики, рассматриваются буквально все вхождения неуникальных слов геномного словаря. По результатам расширения происходит разделение участков на два пула – tandemные и разнесенные повторы.

Роль повторов для организма может быть значительной. Возможно, что это резерв генома для оперативного формирования механизмов защиты организма от угроз путем внутригеномного переноса генетического материала. Они могут служить для горизонтального переноса генетического материала между организмами. Есть свидетельства, что периодические участки обеспечивают специфические формы укладки хроматина, что является еще одним механизмом регуляции экспрессии генов, когда одни гены экспонируются для транскрипции, а другие, наоборот, прячутся внутрь. Известно, что места локализации tandemных повторов в геномах одного вида консервативны, непостоянно лишь число копий, этот вид изменчивости индивидуумов одного вида называется VCNP (Variable Copy Number Polymorphism). Индивидуумы мало отличаются по нуклеотидному составу, они отличаются структурными вариациями геномов. Возможно, tandemные повторы – это временно молчащие блоки для накопления мутаций, эволюционные резервуары. В них могут созреть новые для эволюции генома элементы.

Разнесенные повторы могут приводить к мутациям, изменению экспрессии генов, перестройкам хромосом и т.д., что связано со многими заболеваниями, такими как рак, генетические заболевания, аутоиммунные заболевания и нарушения обмена веществ. Тандемные повторы, особенно STR, обладают высокой вариабельностью, что может ускорять эволюцию экспрессии генов и генерировать достаточную изменчивость, обеспечивающую быструю эволюцию и адаптацию организмов, а также играть жизненно важную роль в структурной стабильности генетического материала и регулировать экспрессию генов, вызывая различные заболевания. Из-за отсутствия достаточно развитых технологий обнаружения, роль и влияние повторов в геномах, особенно в геноме человека, недооцениваются.

Автор выражает благодарность своей коллеге Е.И. Теплухиной за самоотверженную помощь в работе.

СПИСОК ЛИТЕРАТУРЫ

1. Schnable P.S., Ware D., Fulton R.S., Stein J.C., Wei F., Pasternak S., Liang C., Zhang J., Fulton L., Graves T.A., et al. The B73 maize genome: complexity, diversity, and dynamics. *Science*. 2009. V. 326. P. 1112–1115. doi: [10.1126/science.1178534](https://doi.org/10.1126/science.1178534)
2. de Koning A.P., Gu W., Castoe T.A., Batzer M.A., Pollock D.D. Repetitive elements may comprise over two-thirds of the human genome. *PLoS Genet*. 2011. V. 7. Article No. e1002384. doi: [10.1371/journal.pgen](https://doi.org/10.1371/journal.pgen.1002384)
3. Lander E.S., Linton L.M., Birren B., Nusbaum C., Zody M.C., Baldwin J., Devon K., Dewar K., Doyle M., FitzHugh W. et al. Initial sequencing and analysis of the human genome. *Nature*. 2001. V. 409. P. 860–921. doi: [10.1038/35057062](https://doi.org/10.1038/35057062)
4. Orgel L.E., Crick F.H. Selfish DNA: the ultimate parasite. *Nature*. 1980. V. 284. P. 604–607. doi: [10.1038/284604a0](https://doi.org/10.1038/284604a0)
5. Hannan A.J. Tandem repeat polymorphisms: mediators of genetic plasticity, modulators of biological diversity and dynamic sources of disease susceptibility. *Adv. Exp. Med. Biol*. 2012. V. 769. P. 1–9.
6. Charlesworth B., Sniegowski P., Stephan W. The evolutionary dynamics of repetitive DNA in eukaryotes. *Nature*. 1994. V. 371. P. 215–220. doi: [10.1038/371215a0](https://doi.org/10.1038/371215a0)
7. Davidson E.H., Britten R.J. Regulation of gene expression: possible role of repetitive sequences. *Science*. 1979. V. 204. P. 1052–1059. doi: [10.1126/science.45154](https://doi.org/10.1126/science.45154)
8. Mirkin S.M. Expandable DNA repeats and human disease. *Nature*. 2007. V. 447. P. 932–940. doi: [10.1038/nature05977](https://doi.org/10.1038/nature05977)
9. Gymrek M., Willems T., Guilmatre A. Abundant contribution of short tandem repeats to gene expression variation in humans. *Nat. Genet*. 2016. V. 48. P. 22–29. doi: [10.1038/ng.3461](https://doi.org/10.1038/ng.3461)
10. Hannan A.J. Tandem repeats mediating genetic plasticity in health and disease. *Nat. Rev. Genet*. 2018. V.19. P. 286–298. doi: [10.1038/nrg.2017.115](https://doi.org/10.1038/nrg.2017.115)
11. Vollger M.R., Dishuck P.C., Sorensen M., Welch A.E., Dang V., Dougherty M.L., Graves-Lindsay T.A., Wilson R.K., Chaisson M.J.P., Eichler E.E. Long-read sequence and assembly of segmental duplications. *Nat. Methods*. 2019. V. 16. P. 88–94. doi: [10.1038/s41592-018-0236-3](https://doi.org/10.1038/s41592-018-0236-3)
12. Nurk S., Koren S., Rhie A., Rautiainen M., Bzikadze A.V., Mikheenko A., Vollger M.R., Altemose N., Uralsky L., Gershman A., et al. 2022. The complete sequence of a human genome. *Science*. V. 376. P. 44–53. doi: [10.1126/science.abj6987](https://doi.org/10.1126/science.abj6987)

13. Feng C., Dai M., Liu Y., Chen M. Sequence repetitiveness quantification and *de novo* repeat detection by weighted k-mer coverage. *Brief Bioinform.* 2021. V.22. Article No.:bbaa086. doi: [10.1093/bib/bbaa086](https://doi.org/10.1093/bib/bbaa086)
14. Smit A., Hubley R., Green P. *RepeatMasker*. URL: <https://www.repeatmasker.org/> (accessed 15.06.2025).
15. Bao W., Kojima K.K., Kohany O. Repbase Update, a database of repetitive elements in eukaryotic genomes. *Mob. DNA*. 2015. V. 6. Article No. 11. doi: [10.1186/s13100-015-0041-9](https://doi.org/10.1186/s13100-015-0041-9)
16. Agarwal P., States D.J. The Repeat Pattern Toolkit (RPT): analyzing the structure and evolution of the *C. elegans* genome. *Proc. Int. Conf. Intell. Syst. Mol. Biol.* 1994. V. 2. P. 1–9.
17. Bao Z., Eddy S.R. Automated *de novo* identification of repeat sequence families in sequenced genomes. *Genome Res.* 2002. V. 12. P. 1269–1276. doi: [10.1101/gr.88502](https://doi.org/10.1101/gr.88502)
18. Edgar R.C., Myers E.W. PILER: identification and classification of genomic repeats. *Bioinformatics*. 2005. V. 21. No. Suppl 1. P. i152–i158. doi: [10.1093/bioinformatics/bti1003](https://doi.org/10.1093/bioinformatics/bti1003)
19. Kurtz S., Schleiermacher C. REPuter: fast computation of maximal repeats in complete genomes. *Bioinformatics*. 1999. V. 15. P. 426–427. doi: [10.1093/bioinformatics/15.5.426](https://doi.org/10.1093/bioinformatics/15.5.426)
20. Kurtz S., Choudhuri J.V., Ohlebusch E., Schleiermacher C., Stoye J., Giegerich. REPuter: The manifold applications of repeat analysis on a genomic scale. *Nucleic Acids Res.* 2001. V. 29. P. 4633–4642. doi: [10.1093/nar/29.22.4633](https://doi.org/10.1093/nar/29.22.4633)
21. Delcher A.L., Kasif S., Fleischmann R.D., Peterson J., White O., Salzberg S.L. Alignment of whole genomes. *Nucleic Acids Res.* 1999. V.27. P. 2369–2376. doi: [10.1093/nar/27.11.2369](https://doi.org/10.1093/nar/27.11.2369)
22. Volfovsky N., Haas B.J., Salzberg S.L. A clustering method for repeat analysis in DNA sequences. *Genome Biol.* 2001. V.2. Article No. RESEARCH0027. doi: [10.1186/gb-2001-2-8-research0027](https://doi.org/10.1186/gb-2001-2-8-research0027)
23. Price A.L., Jones N.C., Pevzner P.A. *De novo* identification of repeat families in large genomes. *Bioinformatics*. 2005. V. 21. No. Suppl 1. P. i351– i358. doi: [10.1093/bioinformatics/bti1018](https://doi.org/10.1093/bioinformatics/bti1018)
24. Achaz G., Boyer F., Rocha E.P., Viari A., Coissac E. Repseek, a tool to retrieve approximate repeats from large DNA sequences. *Bioinformatics*. 2007. V. 23. P. 19–21. doi: [10.1093/bioinformatics/btl519](https://doi.org/10.1093/bioinformatics/btl519)
25. Sharma D., Issac B., Raghava G., Ramaswamy R. Spectral Repeat Finder (SRF): identification of repetitive sequences using Fourier transformation. *Bioinformatics*. 2004. V. 20. P. 1405–1412. doi: [10.1093/bioinformatics/bth103](https://doi.org/10.1093/bioinformatics/bth103)
26. Pevzner P.A., Tang H., Tesler G. *De novo* repeat classification and fragment assembly. *Genome Res.* 2004. V. 14. P. 1786–1796. doi: [10.1101/gr.2395204](https://doi.org/10.1101/gr.2395204)
27. Lefebvre A., Lecroq T., Dauchel H., Alexandre J. FORRepeats: detects repeats on entire chromosomes and between genomes. *Bioinformatics*. 2003. V. 19. P. 319–326. doi: [10.1093/bioinformatics/btf843](https://doi.org/10.1093/bioinformatics/btf843)
28. Figueroa N., Liu X., Wang J., Karro J. RAIDER: rapid ab initio detection of elementary repeats. In: *Advances in Bioinformatics and Computational Biology. BSB 2013*. Eds. Setubal J.C., Almeida N.F. Recife, Pernambuco, Brazil: Springer, 2013. P. 170–180. (Lecture Notes in Computer Science, V. 8213). doi: [10.1007/978-3-319-02624-4_16](https://doi.org/10.1007/978-3-319-02624-4_16)
29. Schaeffer C.E., Figueroa N.D., Liu X., Karro J.E. phRAIDER: pattern-hunter based rapid ab initio detection of elementary repeats. *Bioinformatics*. 2016. V. 32. P. i209–i215. doi: [10.1093/bioinformatics/btw258](https://doi.org/10.1093/bioinformatics/btw258)

30. Zheng J., Lonardi S. Discovery of repetitive patterns in DNA with accurate boundaries. In: *Fifth IEEE Symposium on Bioinformatics and Bioengineering (BIBE'05)*. Minneapolis, Minnesota, USA: IEEE, 2005. P. 105–112. doi: [10.1109/BIBE.2005.23](https://doi.org/10.1109/BIBE.2005.23)
31. Shi J., Liang C. Generic Repeat Finder: a high-sensitivity tool for genome-wide *de novo* repeat detection. *Plant Physiol.* 2019. V. 180. P. 1803–1815. doi: [10.1104/pp.19.00386](https://doi.org/10.1104/pp.19.00386)
32. Healy J., Thomas E.E., Schwartz J.T., Wigler M. Annotating large genomes with exact word matches. *Genome Res.* 2003. V. 13. P. 2306–2315. doi: [10.1101/gr.1350803](https://doi.org/10.1101/gr.1350803)
33. Morgulis A., Gertz E.M., Schäffer A.A., Agarwala R. WindowMasker: window-based masker for sequenced genomes. *Bioinformatics.* 2006. V. 22. P. 134–141. doi: [10.1093/bioinformatics/bti774](https://doi.org/10.1093/bioinformatics/bti774)
34. Kurtz S., Narechania A., Stein J.C., Ware D. A new method to compute *K*-mer frequencies and its application to annotate large repetitive plant genomes. *BMC Genomics.* 2008. V. 9. Article No. 517. doi: [10.1186/1471-2164-9-517](https://doi.org/10.1186/1471-2164-9-517)
35. Girgis H.Z. Red: an intelligent, rapid, accurate tool for detecting repeats *de-novo* on the genomic scale. *BMC Bioinformatics.* 2015. V. 16. Article No. 227. doi: [10.1186/s12859-015-0654-5](https://doi.org/10.1186/s12859-015-0654-5)
36. Benson G. Tandem repeats finder: a program to analyze DNA sequences. *Nucleic Acids Res.* 1999. V. 27. P. 573–580. doi: [10.1093/nar/27.2.573](https://doi.org/10.1093/nar/27.2.573)
37. Benson G. Sequence Alignment with Tandem Duplication. *J. Comput. Biol.* 1997. V. 4. P. 351–367. doi: [10.1089/cmb.1997.4.351](https://doi.org/10.1089/cmb.1997.4.351)
38. Назипова Н.Н., Теплухина Е.И. Технология полногеномного поиска размытых мегасателлитов и протяженных разнесенных повторов. В: *Доклады Международной конференции “Математическая биология и биоинформатика”*. Под ред. В.Д. Лахно. Том 10. Пушино: ИМПБ РАН, 2024. Статья № е44. doi: [10.17537/icmbb24.56](https://doi.org/10.17537/icmbb24.56)
39. Li H., Durbin R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics.* 2009. V. 25. P. 1754–1760. doi: [10.1093/bioinformatics/btp324](https://doi.org/10.1093/bioinformatics/btp324)
40. Burrows M., Wheeler D.J. *A Blocksorting Lossless Data Compression Algorithm*. Palo Alto, Calif: Digital Equipment Corporation, 1994. (SRC research reports, V. 124)
41. Gusfield D. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge: Cambridge University Press, 1997. P. 8ff
42. Aho A., Corasick M. Efficient string matching: an aid to bibliographic search. *Communications of the ACM.* 1975. V. 18. P. 333–340. doi: [10.1145/360825.360855](https://doi.org/10.1145/360825.360855)
43. Knuth D.E. *The Art of Computer Programming*. Boston: Addison-Wesley, 1997. V. 1: Fundamental Algorithms.
44. McCreight E.M. A space-economic suffix tree construction algorithm. *Journal of the ACM.* 1976. V. 23. P. 262–272.
45. Ukkonen E. On-line construction of suffix-trees. *Algorithmica.* 1995. V. 14. P. 249–260. doi: [10.1007/BF01206331](https://doi.org/10.1007/BF01206331)
46. Weiner P. Linear pattern matching algorithms. Proceedings of the 14th IEE Annual Symposium on Switching and Automata Theory. 1973. P. 1–11. doi: [10.1109/SWAT.1973.13](https://doi.org/10.1109/SWAT.1973.13)
47. Apostolico A. The Myriad Virtues of Subword Trees. In: *Combinatorial Algorithms on Words*. Eds. Apostolico A., Galil Z. Berlin, Heidelberg: Springer, 1985. P. 85–96. (NATO ASI Series, V. 12). doi: [10.1007/978-3-642-82456-2_6](https://doi.org/10.1007/978-3-642-82456-2_6)
48. Manber U., Myers E.W. Suffix Arrays: A New Method for On-Line String Searches. *SIAM J. Comput.* 1993. V. 22. P. 935–948. doi: [10.1137/0222058](https://doi.org/10.1137/0222058)

49. Ferragina P., Manzini G. Opportunistic data structures with applications. In: *Proceedings of the 41st Symposium on Foundations of Computer Science (FOCS 2000)*. Redondo Beach, CA, USA: IEEE, 2000. P. 390–398. doi: [10.1109/SFCS.2000.892127](https://doi.org/10.1109/SFCS.2000.892127)
50. Marçais G, Kingsford C. A fast, lock-free approach for efficient parallel counting of occurrences of k -mers. *Bioinformatics*. 2011. V. 27. P. 764–770. doi: [10.1093/bioinformatics/btr011](https://doi.org/10.1093/bioinformatics/btr011)
51. Li X., Waterman M.S. Estimating the repeat structure and length of DNA sequences using l -tuples. *Genome Research*. 2003. V. 13. P. 1916–1922. doi: [10.1101/gr.1251803](https://doi.org/10.1101/gr.1251803)
52. Jenike K.M., Campos-Domínguez L., Boddé M., Cerca J., Hodson C.N., Schatz M.C., Jaron K.S. K -mer approaches for biodiversity genomics. *Genome Res*. 2025. V. 35. P. 219–230. doi: [10.1101/gr.279452.124](https://doi.org/10.1101/gr.279452.124)
53. Roberts M.D., Davis O., Josephs E.B., Williamson R.J. K -mer-based Approaches to Bridging Pangenomics and Population Genetics. *Molecular Biology and Evolution*. 2025. V. 42. Article No. msaf047. doi: [10.1093/molbev/msaf047](https://doi.org/10.1093/molbev/msaf047)
54. Campagna D., Romualdi C., Vitulo N., Del Favero M., Lexa M., Cannata N., Valle G. RAP: a new computer program for *de novo* identification of repeated sequences in whole genomes. *Bioinformatics*. 2004. V. 21. P. 582–588. doi: [10.1093/bioinformatics/bti03](https://doi.org/10.1093/bioinformatics/bti03)
55. Gu W., Castoe T.A., Hedges D.J., Batzer M.A., Pollock D.D. Identification of repeat structure in large genomes using repeat probability clouds. *Anal. Biochem*. 2008. V. 380. P. 77–83. doi: [10.1016/j.ab.2008.05.01](https://doi.org/10.1016/j.ab.2008.05.01)
56. Rahman A., Hallgrímsdóttir I., Eisen M., Pachter L. Association mapping from sequencing reads using k -mers. *Elife*. 2018. V. 7. Article No. e32920. doi: [10.7554/eLife.32920](https://doi.org/10.7554/eLife.32920)
57. Ponsero A.J., Miller M., Hurwitz B.L. Comparison of k -mer-based *de novo* comparative metagenomic tools and approaches. *Microbiome Res. Rep*. 2023. V. 2. Article No. 27. doi: [10.20517/mrr.2023.26](https://doi.org/10.20517/mrr.2023.26)
58. Wu D.Y., Ugozzoli L., Pal B.K., Qian J., Wallace R.B. The effect of temperature and oligonucleotide primer length on the specificity and efficiency of amplification by the polymerase chain reaction. *DNA Cell Biol*. 1991. V. 10. P. 233–238. doi: [10.1089/dna.1991.10.233](https://doi.org/10.1089/dna.1991.10.233)
59. Becher H., Sampson J., Twyford A.D. Measuring the invisible: the sequences causal of genome size differences in eyebrights (Euphrasia) revealed by k -mers. *Front. Plant Sci*. 2022. V. 13. Article No. 818410. doi: [10.3389/fpls.2022.818410](https://doi.org/10.3389/fpls.2022.818410)
60. Shajii A., Yorukoglu D., William Yu.Y., Berger B. Fast genotyping of known SNPs through approximate k -mer matching. *Bioinformatics*. 2016. V. 32. P. i538–i544. doi: [10.1093/bioinformatics/btw460](https://doi.org/10.1093/bioinformatics/btw460)
61. Bussi Y., Kapon R., Reich Z. Large-scale k -mer-based analysis of the informational properties of genomes, comparative genomics and taxonomy. *PLoS One*. 2021. V. 16. Article No. e0258693. doi: [10.1371/journal.pone.0258693](https://doi.org/10.1371/journal.pone.0258693)
62. Chikhi R., Medvedev P. Informed and automated k -mer size selection for genome assembly. *Bioinformatics*. 2014. V. 30. P. 31–37. doi: <https://doi.org/10.1093/bioinformatics/btt310>

Рукопись поступила в редакцию 11.06.2025, переработанный вариант поступил 27.07.2025.
Дата опубликования 16.08.2025.

Application of Suffix Arrays to Detect Repeats in Genomic Sequences

Nazipova N.N.

Institute of Mathematical Problems of Biology, Keldysh Institute of Applied Mathematics of Russian Academy of Sciences, Pushchino, Russia

Abstract. In recent years, genomes of higher eukaryotes without gaps have become available to researchers. These are the so-called T2T (telomere-to-telomere) assemblies. Compared to the reference genomes studied before the T2T era, they consist of a minimal number of undefined regions or do not contain them at all. Comprehensive information on telomeres, centromeres, ribosomal RNA, complex chromosomal regions and transposon elements has now become available. New computational approaches are needed to find all repeats in genomic sequences without exception. In this article, we described some of the algorithmic problems that we solved in the course of our work on the technology of de novo detection of long fuzzy repeat fragments in huge symbolic arrays that represent genomic sequences. Approaches that use the k-mer composition of the analyzed sequence are promising. The article discusses the importance of choosing an adequate value of k, since the result of the analysis depends on this key parameter of the algorithm.

Key words: DNA repeats, suffix and prefix trees, suffix array, Burrows – Wheeler Transform, k-mer dictionary