Введение

С появлением в начале XXI века новых методов секвенирования [2] (расшифровки биологических последовательностей) резко увеличился объем генетической информации. Рост банков данных носит экспоненциальный характер и есть все основания предполагать, что в ближайшее время этот процесс только усилится. Увеличивающийся объем данных открывает возможности для проведения полномасштабных исследований на уровне целых геномов, однако это требует новых подходов. Дело в том, что почти все существующие алгоритмы обработки биологических последовательностей - это адаптированные алгоритмы обработки текстовых строк, т.к. биологические данные в расшифрованном виде представляют собой последовательности символов. Однако данные алгоритмы изначально не учитывают мутационных процессов, таких как вставка, замена, делеция. Учет же таких точечных мутаций является дорогой с вычислительной точки зрения операцией, вносящей существенную нелинейность в подобные алгоритмы, что значительно увеличивает время их выполнения и делает непригодными для практических целей.

Все это привело к тому, что существующие вычислительные мощности не удовлетворяют потребностям биологов. Становится актуальной разработка программных средств, позволяющих быстро обрабатывать большие биологические данные. Данная дипломная работа посвящена параллельным методам анализа и обработки биологических последовательностей на суперкомпьютерах с целью сокращения времени обработки и увеличения объёма обрабатываемых данных.

Одной из частных задач молекулярной генетики является поиск повторяющихся элементов, изучение их структуры и распределения в биологических последовательностях. Повторы играют важную роль в функционировании организма, т.к. составляют значительную часть генома, возможно участвуют в его реорганизации и при попадании в кодирующие области повторы могут быть причиной нарушения функций этих геномов, что ведет к развитию заболеваний. Таким образом повторы могут использоваться для диагностики генетических заболеваний и определения родства организмов.

Существует масса алгоритмов, ориентированных на поиск относительно точных или коротких (до 500 н.п.) повторов. Однако именно протяженные (от 1000 н.п.) повторы могут служить источником для фундаментальных исследований при решении эволюционных и филогенетических задач - определении родства групп организмов на генном уровне. Для полногеномного сравнения на практике часто применяется ДНК-ДНК гибридизация - биохимическая реакция, протекающая с участием ДНК двух организмов, проводимая для получения количественной оценки их схожести. Однако данный метод очень дорогостоящий, долгий и недостаточно точный.

В данной работе рассматривается метод поиска протяженных неточных повторов, разработанный коллективом кафедры математических методов прогнозирования и института математических проблем в биологии Российской академии наук (Пущино). Данный метод позволяет эффективно решать задачу поиска неточных протяженных повторяющихся структур в генетических текстах. Основное отличие данного метода состоит в переходе от дискретного анализа к непрерывному анализу. Таким образом для задачи дискретной по своей природе, применяется континуальный подход, основанный на приближении непрерывных функций с помощью

ортогональных многочленов. Благодаря этому метод решает недостатки дискретного подхода и обладает следующими свойствами:

- Линейная сложность алгоритма
- Высокая степень устойчивости к мутациям
- Возможность распараллеливания алгоритма на кластерных системах

Для выполнения дипломной работы необходимо было решить следующие задачи:

- Разработать и реализовать параллельный алгоритм поиска повторов, ориентированного на суперкомпьютерную реализацию: большой объем обрабатываемых данных (размер данных порядка 1 ГБ), время обработки в пределах 1 часа.
- Исследовать эффективность использования графических процессоров для решения поставленной задачи.
- Анализировать масштабируемость разработанного алгоритма на примере задачи сравнения конкретных геномов
- Разработать графический интерфейс для работы пользователя на локальной и удаленных системах.

Глава 1

Исследование и построение решения

1.1 Математическая модель поиска повторов в биологических последовательностях

Формально **биологическая последовательность**:

$$X = (x_n)_{n=1}^{N},\ N \in \mathbb{N},\ x_n \in \{A, T, G, C\},$$

где N - длина последовательности, A, T, G, C - озбоначения нуклеотидов.

Введем обозначение. **Подпоследовательностью** $X|_i^k$ последовательности X называется часть последовательности X с элемента с номером i длиной k при условии $i + k < |X|$:

$$X|_i^k = \{x_i, x_{i+1}, ..., x_{i+k-1}\}$$

Под **повтором** будем понимать пару последовательностей (X_1,X_2) для которых справедливо неравенство:

$$\rho(X_1, X_2) \leq \varepsilon,\ |X_2| = |X_2| = K$$

где:

- $\rho(X_1, X_2)$ - расстояние редактирования, оценка близости последовательностей, Конкретный вид функции расстояния редактирования $\rho(X_1, X_2)$ **определяется алгоритмом**
- ε - задаваемая точность поиска, значение которой будет **зависеть от задаваемой функции растояния редактирования**
- K - длина последовательностей

Пусть есть две последовательности $X = (x_n)_{n=1}^{N_x}$ и $Y = (y_n)_{n=1}^{Nk_y}$.
Под **задачей поиска повторов** будем понимать нахождение всех троек $\{i_x, i_y, k\},\ i_x, i_y, k \in \mathbb{N}$, таких что:

$$i_x + k \leq N_x$$
$$i_y + k \leq N_y$$
$$(X|_{i_x}^k, Y|_{i_y}^k) \text{ - повтор длины k}$$

То есть задача найти все такие подпоследовательности последовательностей X и Y, что эти подпоследовательности будут повторами.

1.2 Спектрально-аналитический метод поиска повторов

В работе исследуется спектральный метод поиска повторов в биологических последовательностях предложенный и разработанный коллективом кафедры математических методов прогнозирования ВМК МГУ и института математических проблем в биологии Российской академии наук (Пущино).

Метод разбивается на четыре основных этапа:

1. Получение профилей биологических последовательностей
2. Спектральное индексирование полученных профилей
3. Сравнение коэффициентов спектрального индексирования и построение гомологической матрицы
4. Анализ гомологической матрицы

1.2.1 Профили биологических последовательностей

Одной из главных особенностей рассматриваемого метода поиска повторов является то, что на первом этапе последовательность преобразуется из дискретной в непрерывную область. Это достигается построением т.н. профилей.

Под GC-профилем последовательности $X = (x_n)_{n=1}^{N_x}$ с окном w в дальнейшем будем понимать такую последовательность $P_{GC}(X, w) = (p_n^{GC})_{n=1}^{N_p}, N_p = N_x - w + 1$, что

$$p_i^{GC} = \sum_{k=i}^{i+w} I^{GC}(x_k), i = \overline{1, N_p}$$

где

$$I^{GC} = \begin{cases} 1, \ x_n \in \{G, C\} \\ 0, \ \text{иначе} \end{cases}, n = \overline{1, N_x}$$

Понятие GA-профиля определяется аналогично:

$$P_{GA}(X, w) = (p_n^{GA})_{n=1}^{N_p} : p_i^{GA} = \sum_{k=i}^{i+w} I^{GA}(x_k), i = \overline{1, N_p},$$

где

$$I^{GA} = \begin{cases} 1, \ x_n \in \{G, C\} \\ 0, \ \text{иначе} \end{cases}, n = \overline{1, N_x}$$

1.2.2 Спектральное индексирование профилей

На этом этапе профили переводятся в спектральное представление с использованием в качестве базиса полиномов Чебышева дискретного аргумента.

Под спектральным представлением сигнала $P = (p_n)_{n=1}^{N_p}$ будем понимать вектор $\overline{C} = C_m(P) = (c_0, ..., c_{m-1})$, где $c_0, ..., c_{m-1}$ - первые m коэффициентов разложения сигнала P по некоторой системе ортогональных функций $u_0(x), ..., u_{m-1}(x), ...$

Полиномы Чебышева определяются при помощи рекуррентного соотношения:

$$u_0(x) = 1$$
$$u_1(x) = x$$
$$...$$
$$u_{n+1}(x) = 2xu_n(x) - u_{n-1}(x)$$

Вычисление коэффициентов разложения выполняется по рекуррентным соотношениям.

1.2.3 Спектральное сравнение профилей

На этой стадии спектральное представление профилей используется для производства сравнения на основе некоторого специально разработанного критерия.

Теперь можно ввести понятие расстояния редактирования для рассматриваемого метода поиска повторов.

Пусть X_1, X_2 - две биологические последовательности, такие что $|X_1| = |X_2| = K \geq w$. Будем понимать под GC- и GA-расстоянием редактирования для X_1 и X_2:

$$\rho^{GC}(X_1, X_2) = \left\| \overline{C_1^{GC}} - \overline{C_2^{GC}} \right\|,$$
$$\rho^{GA}(X_1, X_2) = \left\| \overline{C_1^{GA}} - \overline{C_2^{GA}} \right\|,$$
$$\left\| \overline{C} \right\| = \sum_{i=0}^{m-1} c_i^2$$

Под повтором будем понимать пару последовательностей X_1, X_2, удовлетворяющих системе:

$$\begin{cases} \rho^{GC}(X_1, X_2) < \varepsilon \\ \rho^{GA}(X_1, X_2) < \varepsilon \end{cases}$$

Под матрицей гомологии окном профилей w, окном аппроксимации a и шагом аппроксимации s для последовательностей X_1, X_2 будем понимать матрицу

$$M(X, Y, w, s, a) = (m_{ij})^{L_x \times L_y},$$

$$L_x = [\tfrac{N_x - a - w + 1}{s}], L_y = [\tfrac{N_y - a - w + 1}{s}]$$

$$m_{ij} = \begin{cases} 1, \ x_n \in \{G, C\} \\ 0, \text{ иначе} \end{cases}$$

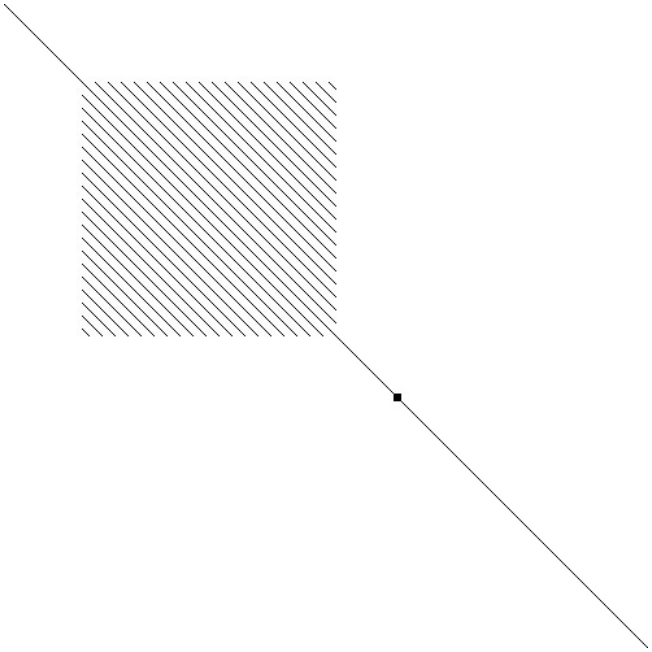

Рисунок 1.1: Пример гомологической матрицы

На рисунке 1.1 приведена гомологическая матрица, образованная в результате сравнения последовательности с самой собой. Матрица имеет симметричный вид.

1.2.4 Анализ гомологической матрицы

Рассмотрим все диагонали гомологической матрицы параллельные главной диагонали, включая ее саму. Фиксируем диагональ D_l. Назовем диагональным элементом $R(x, y, k)$ такое множество элементов m_{ij} гомологической матрицы $M^{L_x \times L_y}$, что:

$$m_{ij} \in D_l$$

$$m_{ij} = 1, \quad i = \overline{y, y+k-1}, \quad j = \overline{x, x+k-1}, k \in \mathbb{N}$$

$$m_{x-1,y-1} = 0, \text{если } x > 0, y > 0$$

$$m_{x+k,y+k} = 0, \text{если } x + k < L_x, y + k < L_y$$

На последнем этапе анализа гомологической матрицы нужно найти все диагональные элементы этой матрицы. Зная параметры алгоритма по диагональным элементам можно восстановить искомые повторы.

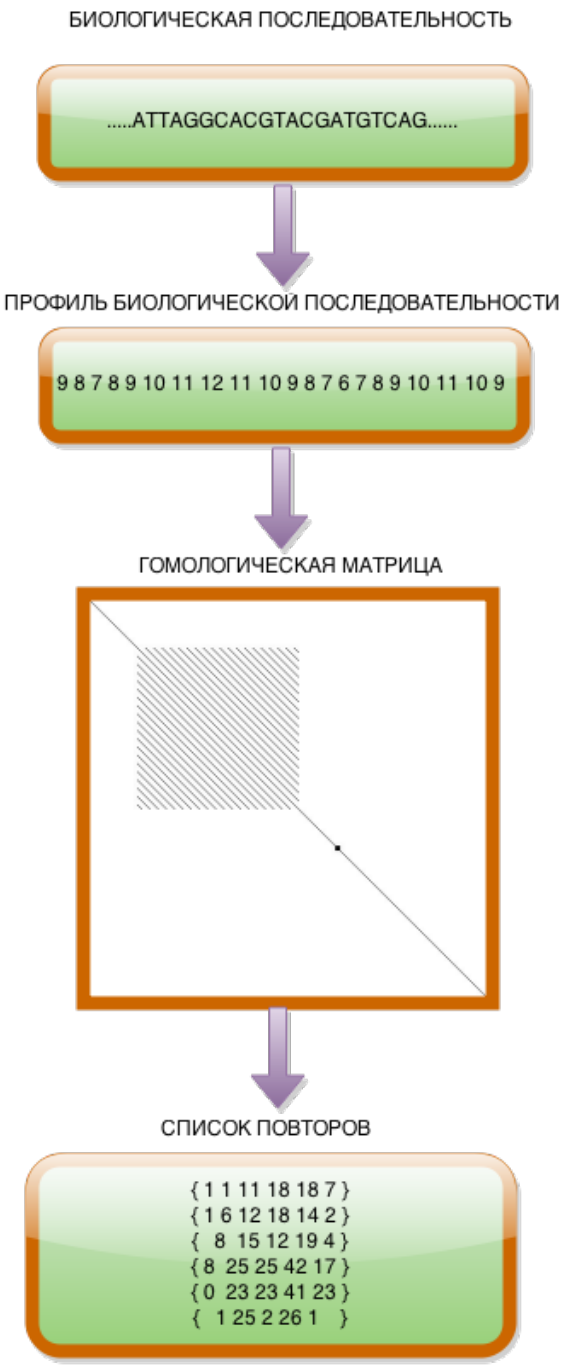

Рисунок 1.2: Схема работы спектрально-аналитического метода поиска повторов

1.3 Параллельный метод поиска повторов

Основная идея параллллельного алгоритма поиска повторов заключается в равномерном распределении входных данных по процессам и их независимой обработке, запрашивая недостающие элементы у соседних процессов.

При разработке параллельного алгоритма во главу угла была поставлена автономность каждого этапа вычислений. Такой подход позволяет достигнуть следующих результатов:

- Логическое следование алгоритму
- Прозрачность системы
- Возможность отладки каждого этапа

1.3.1 Профилирование и спектральное индексирование

На рисунке 1.3 представлена схема выполнения первых двух этапов: получения профиля биологической последовательности и спектрального индексирования.

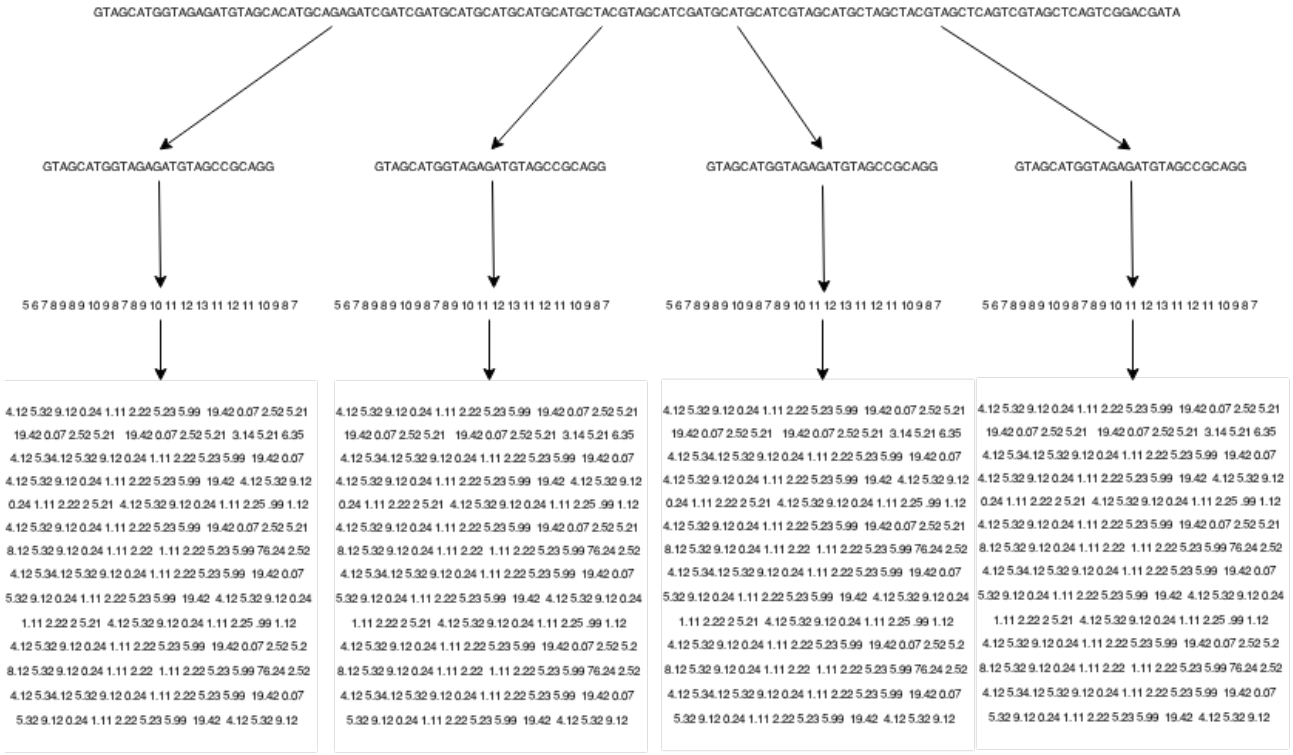

Рисунок 1.3: Параллельная схема работы этапа профилирования и спектрального индексирования

Параллельная схема двух этапов очень похожа, поэтому есть смысл рассматривать их вместе. На каждом этапе процесс получает недостающие элементы у соседнего процесса и выполняет обработку данных. Данная схема основывается на принципе независимого выполнения этапов.

Естественно, что такой алгоритм не единственно возможный. Например, можно было бы использовать такую схему, когда профилирование и спектральное индексирование объединялись бы в один этап и по параметрам алгоритма каждый процесс рассчитывал сколько ему нужно взять элементов таким образом, чтобы обойтись без пересылок между процессами. Но такой подход имеет и ряд недостатков: дублирование вычислений, сложность отладки, невозможность повторного использования данных для других параметров алгоритма.

К недостаткам предложенного алгоритма можно отнести обмены между процессами. Однако каждый процесс может начинать работу не дожидаясь завершения приема данных, т.к. они понадобятся только в конце работы. Таким образом можно скрыть передчу данных на фоне выполнения полезной работы и добиться ситуации, когда процессор не будет простаивать в ожидании данных.

Два последующих этапа алгоритма - спектральное сравнение и анализ гомологической матрицы, можно организовать двумя способами, которые будут рассмотрены ниже.

1.3.2 Спектральное сравнение и анализ матрицы гомологии. Матричный метод

Во время спектрального сравнения каждый процесс будет отвечать за сравнение своей части спектров первой последовательности со всеми спектрами второй последовательности. Таким образом процесс будет хранить часть строк гомологической матрицы, соответствующих спектрам из первого набора.

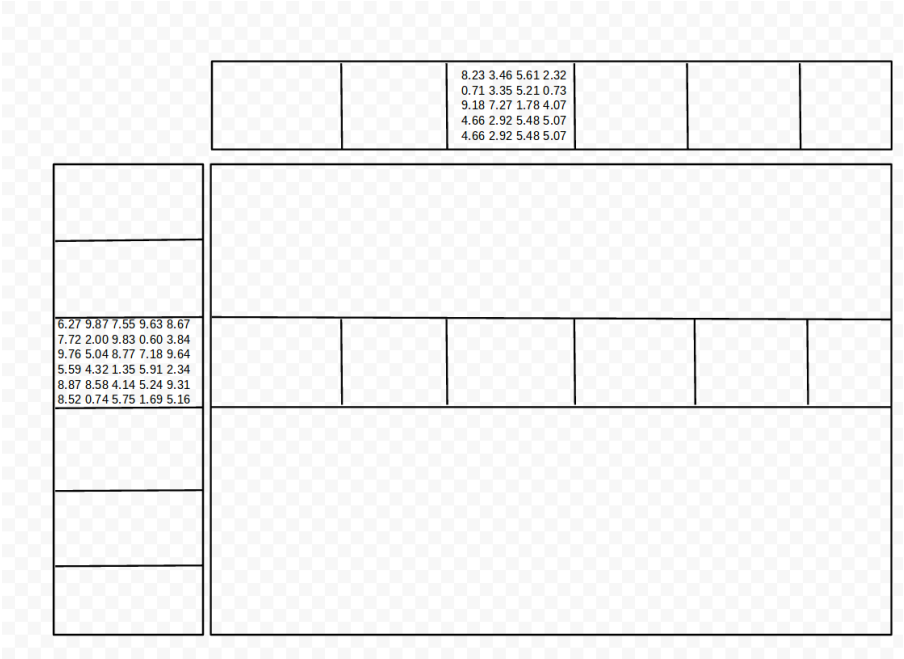

Рисунок 1.4: Этапы 1 и 2

Для выполнения данного этапа процессу требуются собрать спектры второй последовательности со всех процессов. Однако и здесь можно добиться отсутствия простоя вычислительных платформ, если не дожидаться получения всех данных, а начинать работу по возможности. То есть как только придут данные хотя бы с одного процесса - сразу строить часть матрицы.

Таким образом можно скрыть большую часть пересылок данных за полезными вычислениями и добиться хорошей масштабируемости алгоритма.

На последнем этапе каждый процесс анализирует имеющуюся часть гомологической матрицы на наличие диагональных элементов. Однако следует учитывать, что найденный диагональный элемент может иметь свое начало или окончание на другом процессе и таким образом быть разбитым по процессам, поэтому нужно "склеить"диагональные элементы с разных процессов.

Здесь будем действовать по следующему принципу: процесс "склеивает"диагональный элемент (т.е. ищет его целиком), только если диагональный элемент принадлежит нашему

процессу. Диагональный элемент $R(x, y, k)$ принадлежит процессу i, если начало этого повтора находится на процессе i, т.е.:

$m_{xy} \in M_i^{L_x \times L_y}$, где $M_i^{L_x \times L_y}$ – гомологическая матрица, вычесленная на процессе i

Таким образом алгоритм анализа будет работать следующим образом:

1. Найти все диагональные элементы
2. Удалить диагональные элементы, не принадлежащие нашему процессу
3. Найти конец всех диагональных элементов

Такой алгоритм нахождения диагональных элементов, при котором последовательно идут этапы спектрального сравнения с полным построением матрицы гомологии и ее последующим анализом будем называть матричным методом.

1.3.3 Спектральное сравнение и анализ матрицы гомологии. Блочный метод

Явным недостатком матричного метода является объем памяти, необходимый для гомологической матрицы. Размер этой матрицы N/s, где N - размер входных данных, s - сдвиг окна индексирования. Тогда, например, объем гомологической матрицы, необходимый для анализа человеческого генома (3.1 млрд. н.п.), со сдвигом s равным 100, будет составлять $3.1 * 10^7 * 3.1 * 10^7 = 9 * 10^14 = 900$. Такой размер гомологической матрицы существенно сокращает количество возможных вычислительных платформ на которых можно было бы запускать программу.

Чтобы уйти от этого недостатка нам придется пренебречь принципом автономности этапов. Оснавная идея блочного метода заключается в объединении этапов спектрального индексирования и анализа гомологической матрицы для обработки небольшого блока.

Итак, пусть пользователь задает новый параметр алгоритма - лимит оперативной памяти для гомологической матрицы. Также как и в матричном методе процесс будет отвечать за сравнение спектров первой последовательности со всеми спектрами второй последовательности. Единственное отличие блочного метода от матричного в том, что результат сравнения не будет сохраняться в матрице гомологии, а будет сразу же подвергаться анализу на наличие диагональных элементов. Таким образом гомологическая матрица становится чисто виртуальной структурой - на деле ничего в памяти не хранится.

На рис разобрана схема блочного метода. Как только процесс принял часть спектров второй последовательности начинается их сравнение со спектрами первой последовательности. Выделяется блок памяти, объем которого не превышает заранее определенной квоты пользователя. Строится часть виртуальной гомологической матрицы равная размеру выделенного блока. Блок сразу же подвергается анализу и результат анализа сохраняется в списке диагональных элементов. Блок сдвигается ниже на его высоту, идет построение и анализ. Результат анализа "склеивается"по вертикали с предыдущими результатами. Операция "склейки"диагональных элементов по вертикали не имеет ничего общего со "склейкой"между процессами, т.к. здесь процесс знает всю информацию о склеивающихся структурах.

После постоения столбца виртуальной гомологической матрицы результат сохраняется и происходит построение следующего столбца. После его постройки происходит "склейка"диагональных элементов по горизонтали. И так далее, пока не получим результат анализа с одним из процессов.

На последнем этапе будем иметь результаты анализа сравнения со всеми процессами. Остается применить ко всем операцию "склейка"по горизонтали, и получим список диагональных элементов (см. рис)

То есть будем иметь такой же список, как и при матричном способе. Остается применить операцию "склейка"по процессам и перевести диагональные элементы в искомые повторы.

Список литературы

[1] *Антонов А.С.* Технологии параллельного программирования MPI и OpenMP. — Издательство Московского университета, 2012. — С. 344.

[2] Название статьи / Автор1, Автор2, Автор3, Автор4 // *Журнал*. — 2012. — Vol. 1. — P. 100.

[3] *Сандрес Дж., Кэндрот Э.* Технология CUDA в примерах: ввведение в программирование графических процессоров / Под ред. Боресков А.В. — ДМК Пресс, 2013. — С. 232.