На правах рукописи

Ройтберг Михаил Абрамович

Алгоритмы сравнительного анализа первичных структур биополимеров

03.00.28 Биоинформатика

АВТОРЕФЕРАТ

диссертации на соискание ученой степени

доктора физико-математических наук

Москва – 2009

Работа выполнена в Учреждении Российской Академии Наук
Институт математических проблем биологии РАН

Официальные оппоненты: доктор биологических наук,
профессор В.В. Поройков

доктор физико-математических наук
О.В. Галзитская

доктор физико-математических наук
А.М. Райгородский

Ведущая организация: Учреждение Российской академии
наук Институт теоретической и
экспериментальной биофизики РАН

Защита состоится «_____» _____2009 г. в _____ч. на заседании диссертационного совета Д 002.077.02 при Учреждении Российской академии наук Институт проблем передачи информации РАН им. А.А. Харкевича по адресу 127994, Москва, ГСП-4, Большой Каретный переулок, д.19, стр.1.

С диссертацией можно ознакомиться в библиотеке Учреждения Российской академии наук Институт проблем передачи информации РАН им. А.А. Харкевича

Автореферат разослан «_____»_____2009г.

Ученый секретарь диссертационного совета Д 002.077.02
доктор биологических наук, профессор Рожкова Г.И.

Общая характеристика работы

Актуальность темы. Последовательности (первичные структуры) нуклеиновых кислот и белков – наиболее массовый и наиболее доступный в настоящее время вид молекулярно-биологических экспериментальных данных. Особенностью этого рода экспериментальных данных является то, что темпы их получения с самого начала опережали темпы обработки данных; анализ биологических последовательностей изначально был отделен от их получения. Задачи исследования последовательностей (изучение их внутреннего строения, связи с пространственной структурой, функциональной аннотации, изучения эволюции) решались различными методами, среди которых можно выделить две группы: методы, анализирующие собственно данную последовательность и методы, проводящие сравнение нескольких последовательностей. В последнем случае речь может идти как о выделении сходных (и, возможно, имеющих сходную биологическую функцию) фрагментов, так и о переносе свойств хорошо изученной последовательности на соответствующие фрагменты сходной с ней последовательности, или на всю последовательность в целом (например, при определении типа пространственной структуры белка). По мере накопления экспериментальных данных сравнительные методы играют все более важную роль. Таким образом, задача разработки алгоритмов сравнительного анализа биологических последовательностей изначально являлась одной из важнейших задач биоинформатики.

Практически одновременно с накоплением данных о биологических последовательностях (в 60-х – 70-х годах XX века) происходило развитие прикладной теории алгоритмов – разработка базовых алгоритмов анализа символьных последовательностей и связанных с ними алгоритмов сортировки и алгоритмов на графах. Начиная с 70-х годов XX века аппарат прикладной теории алгоритмов начал применяться для анализа (прежде всего – сравнения) биологических последовательностей. При этом достаточно быстро стало понятно, что постановки задач должны максимально учитывать специфику предметной области, а собственно алгоритмы поиска (построения) искомых объектов должны дополняться исследованием статистической значимости полученного результата и/или его соответствия эталонным (экспериментально подтвержденным) результатам – для тех случаев, когда эти результаты известны.

Именно с этих позиций в диссертации рассмотрена классическая задача биоинформатики – задача парного выравнивания биологических последовательностей.

Парное выравнивание является базовым методом сравнения биологических последовательностей. Говоря неформально, выровнять две последовательности - это поместить их друг над другом, возможно, вставляя в обе последовательности пробелы, так, чтобы сделать их длины равными. При этом позиции, оказавшиеся друг над другом, считаются сопоставленными друг другу, а остальные символы (расположенные напротив пробелов) – удаленными. Две данные символьные последовательности можно выровнять многими способами. Алгоритмический выбор нужного выравнивания двух данных последовательностей основан на понятии веса выравнивания – строится оптимальное выравнивание,

т.е. выравнивание, имеющее максимально возможный вес. Вес выравнивания определяется как сумма весов сопоставленных символов минус сумма штрафов за удаленные фрагменты.

Основные понятия выравнивания биологических последовательностей были сформированы в конце 70-х – начале 80-х годов XX века в работах Нидлмана, Вунша, Смита, Уотермана, Селлерса, Санкоффа, Эриксона, Туманяна и др. В частности, были предложены основанные на методе динамического программирования алгоритмы, которые строят оптимальное выравнивание последовательностей для различных классов весовых функций. Алгоритм для линейных штрафов имеет время работы O(m•n); алгоритмы построения оптимального выравнивания для выпуклых весов делеций и для произвольных весов делеций имеют временную сложность соответственно $O(m \cdot n \cdot (m+n))$ и $O(m^2 \cdot n^2)$ (m и n – длины последовательностей). Однако оставался открытым вопрос – существует ли класс весовых функций, более широкий, чем линейные функции, допускающий построение оптимального выравнивания за время O(m•n)? Кроме того, указанные алгоритмы имеют два существенных недостатка с точки зрения биологических приложений. Во-первых, оптимальное выравнивание аминокислотных последовательностей белков в ряде важных случаев по внутренним причинам не может воспроизвести выравнивание этих белков, согласованное с их пространственной структурой. Во-вторых, не разработана теория выбора значений параметров штрафов за делеции. Как правило, выбор параметров осуществлялся эмпирически. Это определяет важность поиска путей инкорпорирования содержательной биологической информации, как при собственно выравнивании, так и при подборе параметров штрафов.

Еще одним недостатком упомянутых алгоритмов динамического программирования является их относительно невысокое быстродействие. Даже на современных компьютерах невозможно за приемлемое время выровнять последовательности длиной более миллиона символов (как при сравнении геномов) или провести сотни тысяч сравнений последовательностей длиной несколько тысяч (как при сравнении протеомов). Выход, предложенный в работах Пирсона, Липпмана, Гиша, Укконена и др., состоял в построении выравнивания как оптимальной (в некотором смысле) цепочки локальных сходств. При этом аналогом штрафов за делеции становятся штрафы за «невыровненные» участки между локальными сходствами. Отметим, что проблема выбора параметров этих штрафов так же, как и выбор параметров штрафов за делеции, решалась чисто эмпирически.

Собственно задача поиска локальных сходств решалась следующим двухэтапным методом. На первом этапе с помощью построения индексных таблиц находятся «затравочные сходства» - точные совпадения заданной длины. На втором этапе происходит поиск локальных сходств (возможно, содержащих несовпадения и делеции), при этом поиск ведется лишь в окрестности затравочных сходств. Последнее приводит к тому, что некоторые *a priori* интересные локальные сходства могут быть утеряны. Таким образом, качество поиска с помощью затравок характеризуется двумя величинами – чувствительностью и избирательностью. Неформально говоря, чувствительность затравки характеризует долю представляющих интерес («целевых») сходств, которые могут быть найдены с ее

помощью, а избирательность – количество затравочных сходств при сравнении случайных последовательностей. Около 10 лет назад было показано, что качество поиска может быть существенно улучшено, если вместо классических «сплошных» затравок рассмотреть затравки более сложного вида – «разреженные». Это сначала было сделано в теоретических работах Бургхарда и Каркайнена, а затем – в ориентированных на биологические приложения работах группы Минг Ли. Впоследствии в работах Брауна, Брежовой, Кучерова и др. были предложены и более сложные виды затравок. Однако, как и в случае параметров штрафов, подбор конкретных затравок, как правило, возможен лишь путем компьютерных экспериментов. Таким образом, важен поиск новых типов затравок, а также разработка методов доказательства оптимальности конкретных затравок. Как и в случае алгоритмов динамического программирования, необходимо привлечение содержательной информации о сравниваемых последовательностях.

Построение выравниваний и поиск локальных сходств связаны с оценкой достоверности полученных результатов. Это может быть сделано двумя способами. Если доступна обучающая выборка, в которой наряду со сравниваемыми объектами есть и результат сравнения, то качество алгоритма можно проверить на этой выборке. В противном (и более распространенном) случае стандартным способом проверки достоверности полученного результата является вычисление вероятности «случайного» возникновения такого события (P-value). В терминах вычисления P-value может быть переформулирована и упоминавшаяся выше задача о вычислении чувствительности затравок. По-видимому, впервые в задачах биоинформатики понятие P-value было введено в работах Карлина и Альтшуля, где было показано, что распределение веса наилучшего безделеционного локального сходства двух независимых случайных последовательностей асимптотически описывается распределением экстремальных значений.

Однако, ситуация, когда можно вывести хотя бы асимптотическое распределение, встречается крайне редко. В большинстве случаев, значение P-value вычисляется алгоритмически, исходя из выбранной вероятностной модели. Подобные алгоритмы были предложены для расчета чувствительности затравок различных типов при бернуллиевской модели случайных биологических последовательностей. Как и следовало ожидать, такие алгоритмы сходны друг с другом. Тем не менее, это сходство не было выявлено, что затрудняло построение алгоритмов вычисления P-value для других типов задач и обобщения уже существующих алгоритмов на более сложные вероятностные распределения на множестве биологических последовательностей. Поэтому важно было выработать общий подход к решению таких задач, выявить их алгоритмическую связь с другими задачами биоинформатики, в частности с задачей построения оптимального выравнивания.

Цель исследования: разработка методов сравнительного анализа первичных структур биополимеров, в частности, их парного выравнивания, включая разработку методов оценки достоверности полученных результатов.

Задачи исследования.

1. Создание эффективных алгоритмов глобального парного выравнивания символьных последовательностей при различных видах штрафов за делеции, а также в отсутствии явно заданных штрафов.

2. Исследование соответствия между алгоритмически оптимальными и структурно подтвержденными выравниваниями аминокислотных последовательностей; разработка алгоритмов, позволяющих повысить точность и достоверность алгоритмически оптимальных выравниваний аминокислотных последовательностей белков относительно эталонных (структурно подтвержденных) выравниваний этих белков.

3. Разработка специализированных методов анализа нуклеотидных последовательностей (сравнение геномов, сравнение последовательностей РНК с известной вторичной структурой), разработка методов предсказания вторичной структуры РНК.

4. Разработка методов построения затравок для поиска локальных сходств в нуклеотидных последовательностях и анализа их чувствительности; разработка методов вычисления вероятностей событий, связанных с поиском локальных сходств и обнаружением заданных сигналов.

Методы исследования. В работе использованы методы теории алгоритмов, теории вероятностей, математической статистики, молекулярной биологии и генетики.

Научная новизна. Научная новизна работы состоит в сформулированном нами едином подходе к решению ряда алгоритмических задач анализа последовательностей, а также в разработанных оригинальных алгоритмах.

Теоретическая и практическая ценность. Теоретическая значимость работы состоит в исследовании роли штрафов за делеции в задачах парного выравнивания последовательностей и разработке единого подхода к вычислению вероятностей появления мотивов в последовательностях и их выравниваниях. Это позволило разработать ряд эффективных алгоритмов и исследовать их адекватность биологическим приложениям.

Основными из этих алгоритмов являются:

1) алгоритм построения оптимального выравнивания нуклеотидных последовательностей при штрафах за делеции, задаваемых кусочно-линейными функциями;
2) алгоритм построения множества Парето-оптимальных выравниваний относительно векторной весовой функции;
3) алгоритм выравнивания аминокислотных последовательностей белков с учетом их вторичной структуры
4) алгоритм предсказания внутренних циклов во вторичной структуре РНК;
5) алгоритм выравнивания вторичных структур РНК при заданной вторичной структуре;
6) алгоритм выравнивания геномов;

7) алгоритм вычисления чувствительности затравок для поиска локальных сходств;
8) алгоритм оценки значимости кластеров регуляторных сайтов

Для большинства предложенных алгоритмов созданы реализующие их общедоступные компьютерные программы, эти программы используются в работах как отечественных, так и зарубежных исследовательских групп.

Апробация результатов. Материалы диссертации докладывались на международных и всероссийских конференциях и семинарах, в том числе: Московском семинаре по компьютерной генетике; отчетных конференциях программы «Геном человека»; II Съезде биофизиков России (Москва, 1999), симпозиуме «The Informatics of Protein Classification» (Университет Раттерс, США, 2000), III, IV,V,VI международных конференциях по биоинформатике и регуляции структуры генома (Новосибирск, 2002, 2004, 2006, 2008), I, II и III Московских международных конференциях по вычислительной биологии (2003, 2005, 2007), международной конференции Combinatorial Pattern Matching-2004 (Стамбул, Турция), международной конференции Workshop on Algorithms in Bioinformatica (Пальма де Майорка, Испания, 2005), международной конференции «Implementation and Application of Automata» (Прага, Чехия, 2007), международном симпозиуме NETTAB (Варенна, Италия, 2008).

С использованием материалов диссертации автором сделаны доклады в NCBI USA (2000), Georgia Tech (2005), INRIA (2003, 2007), на семинарах в Институте белка РАН, Московском физико-техническом институте, факультете биоинженерии и биоинформатики МГУ и ряде других учреждений.

Публикации. Основные материалы диссертации изложены в 37 статьях в реферируемых научных изданиях (из них 33 в соавторстве).

Структура и объем работы. Диссертация состоит из введения, пяти глав, заключения и списка литературы (345 наименований). Полный объем диссертации составляет 223 страницы, количество рисунков - 29, количество таблиц - 17.

Содержание работы

Во введении дана общая характеристика работы и приведены основные определения и обозначения. Глава 1 посвящена обзору литературы, отмечена связь проанализированных работ с предметом исследования диссертации. Результаты работы представлены в главах 2 – 5.

Во второй главе дан теоретический анализ проблемы выравнивания двух символьных последовательностей. Результаты, представленные в этой главе, могут быть применены не только к биологическим последовательностям, но и к последовательностям иной природы.

Напомним основные понятия. Выравнивание символьных последовательностей u и v - это тройка $<u, v, S>$, где $S = \{<i_1, j_1>, \ldots <i_n, j_n>\}$ – набор пар пози-

ций в словах u и v соответственно, таких, что $1 \le i_1 < \ldots < i_n \le |u|$; $1 \le j_1 < \ldots < j_n \le |v|$. Имеется в виду, что i_k-я позиция слова u сопоставлена с j_k-й позицией слова v $(k = 1, \ldots, n)$, а непустые фрагменты вида $u[i_k+1, i_{k+1}-1]$ и $v[j_k+1, j_{k+1}-1]$ удалены (здесь $k = 0, \ldots, n$; $i_0 = j_0 = 0$; $i_{n+1} = |u|+1$; $j_{n+1} = |v|+1$). Пары позиций $<i_s, j_s>$ называются склейками. Вес $P(A)$ выравнивания A определяется как функция (обычно – разность) весов сопоставлений букв и штрафов за удаление фрагментов. Предполагается, что нулевой (начальной) буквой любой символьной последовательности является начальный маркер, один и тот же для всех последовательностей.

В разделе 2.1 введено понятие кусочно-линейных функций штрафов (весовых функций) за удаление фрагмента и представлен соответствующий алгоритм построения оптимального выравнивания. Предложенный класс функций является наиболее широким из известных классов весовых функций, для которых построен квадратичный алгоритм построения оптимального выравнивания (под квадратичным алгоритмом понимается алгоритм со временем работы, пропорциональным длинам сравниваемых последовательностей). Класс весовых функций, рассмотренный в разделе 2.2, наоборот, - простейший класс, при котором задача построения оптимального выравнивания двух последовательностей не сводится к хорошо изученной задаче построения максимальной общей подпоследовательности (Longest Common Subsequence, LCS). Для указанного класса функций предложен адаптивный алгоритм построения оптимального выравнивания. Хотя оценка времени алгоритма в худшем случае является квадратичной, время работы алгоритма для близких последовательностей почти линейно. В частности, если для последовательностей v_1 и v_2 длины m существует оптимальное выравнивание, содержащее t несовпадений и s удаленных символов, то время работы предлагаемого алгоритма составляет $O((t+s)m)$.

Рассмотрим подробнее каждый из упомянутых алгоритмов.

Говоря неформально, кусочно-линейная система весовых функций – это система, которая обладает следующими свойствами:

1) штрафы за удаления фрагментов на концах последовательностей могут быть произвольными;

2) штраф за удаление фрагмента может зависеть от граничных позиций удаляемого фрагмента;

3) зависимость штрафа от длины фрагмента может задаваться произвольной кусочно-линейной функцией;

4) штрафы за удаления фрагментов в каждой из сравниваемых последовательностей могут задаваться по-своему;

5) вес сопоставления символов $u[i]$ и $v[j]$ задается произвольной функцией $\eta(i, j, u, v)$.

Вес выравнивания последовательностей определяется как разность $S - D$, где S – сумма весов сопоставлений букв, D – сумма штрафов за удаление фрагментов.

Идея алгоритма построения оптимального выравнивания символьных последовательностей (слов) v_1 и v_2 при заданной кусочно-линейной системе весовых функций состоит в следующем. Последовательно (в цикле по i, а внутри него – в цикле по j) вычисляются т.н. оптимальные (i, j)-выравнивания $A[i, j]$ и их веса $P[i, j]$; здесь (i, j)-выравнивание слов u и v – это такое выравниваний пре-

фиксов *u[1, i]*, *v[1, j]*, в котором сопоставлены позиции *u[i]* и *v[j]*. Одновременно вычисляется выравнивание *M[i, j]* – оптимальное выравнивание слов *u* и *v*, в котором последнее сопоставление позиций *(x, y)* удовлетворяет условиям $x \le i$; $y \le j$. Выравнивание *M[|u|, |v|]* будет искомым. Для вычисления веса *P[i, j]* и соответствующего ему выравнивания *A[i, j]* алгоритм поддерживает $k_1 \cdot k_2$ рекурсивно вычисляемых величин *m[s, t]*, где k_i – количество интервалов линейности у функции штрафов за удаление фрагментов внутри слова v_i. При вычислении веса *P[i, j]* величина *m[s, t]* хранит (в приведенной форме, см. подробнее п.2.1.5 диссертации) максимальный вес среди весов таких *(x, y)*-выравниваний, что длина *i-x-1* попадает в *s*-й интервал линейности слова *u*, а длина *j-y-1* – в *t*-й интервал линейности слова *v*. При этом каждая из величин *m[s, t]* для очередной пары *(i, j)* перевычисляется за конечное время, что и обеспечивает указанное время работы алгоритма $O(k_1 \cdot k_2 \cdot |v_1| \cdot |v_2|)$. Память, нужная алгоритму, оценивается формулой (см. раздел 2.1.10 диссертации)

$$S \le (c_1 + c_2 \cdot s_1 + c_3 \cdot k_{s_1}^1) \cdot m_2 + c_4 \cdot m_2^2,$$

где $c_1, \ldots, c_4$ – константы. Однако большая часть этой памяти нужна лишь при восстановлении оптимального выравнивания как пути в специальном дереве (дереве опорных склеек). Поэтому для потребной оперативной памяти получаем оценку $O(k_{s_1}^1 \cdot m_2)$.

В разделе 2.2 вес выравнивания *G* последовательностей v_1 и v_2 длины соответственно m_1 и m_2 задается формулой

$$W(G) = p - f \cdot r - d \cdot (m_1 - p - r) - d \cdot (m_2 - p - r)$$

где *p* – количество совпадений в выравнивании; *r* – количество несовпадений; m_1-*p*-*r* – количество символов, удаленных в первой последовательности; m_2-*p*-*r* – количество символов, удаленных во второй последовательности; *f*, *d* – весовые коэффициенты. Таким образом, система весовых функций описывается двумя коэффициентами *f* и *d*.

Базовыми понятиями алгоритма, описанного в разделе 2.2, являются понятие опорного множества и специального веса начального выравнивания. Начальным выравниванием слов v_1 и v_2 называется выравнивание префиксов $v_1[0, x_1]$ и $v_2[0, x_2]$. Начальное выравнивание $G = < v_1[0, x_1], v_2[0, x_2], Q>$ слов v_1 и v_2 называется *граничным*, если $x_1 = |v_1|$ или $x_2 = |v_2|$.

Определение 2.2.2. Множество *S* начальных выравниваний слов v_1 и v_2 называется *опорным* множеством, если в *S* найдется выравнивание, которое является префиксом некоторого оптимального выравнивания слов v_1 и v_2.

Например, множество S_0, состоящее из единственного начального выравнивания $A_0 = < v_1[0], v_1[0], \{(0,0)\}>$, является опорным.

Определение 2.2.1. Пусть $G = < v_1[0, x_1], v_2[0, x_2], Q>$ – начальное выравнивание слов v_1 и v_2 длин m_1 и m_2. *Специальным весом* выравнивания *G* называется величина

$$SP(G) = P(G) + c \cdot min(m_1 - x_1, m_2 - x_2) - d \cdot |(m_1 - x_1) - (m_2 - x_2)|.$$

Начальное выравнивание *G* называется максимальным в множестве *S*, если в *S* нет выравнивания, которое имеет специальный вес больше, чем *SP(G)*. Опорное множество *S* называется *терминальным*, если некоторое граничное выравнивание является максимальным в нем.

Специальный вес выравнивания обладает следующими свойствами.

1. Пусть G - выравнивание слов v_1 и v_2 и начальное выравнивание B является префиксом выравнивания G. Тогда

$$P(G) \leq SP(B).$$

2. Пусть S – опорное множество для слов v_1 и v_2; $B \in S$ - граничное начальное выравнивание, которое является максимальным в S. Тогда выравнивание B' слов v_1 и v_2, имеющее тот же набор сопоставлений, что и выравнивание B – оптимальное выравнивание слов v_1 и v_2.

В силу этих свойств, для построения оптимального выравнивания достаточно построить терминальное опорное множество и выделить граничный максимальный элемент в нем.

Построение терминального опорного множества и соответствующего граничного максимального выравнивания выполняется с помощью построения *т.н. канонической последовательности* опорных множеств $\{S_i\}$. Множество S_0 состоит из одного выравнивания $<v_1[0,0], v_2[0,0], \{(0,0)\}>$. Пусть опорное множество S_i построено. Если S_i – терминальное множество, то процесс построения канонической последовательности закончен. В противном случае выбирается некоторое начальное выравнивание $L_i \in S_i$ (*лидер* множества S_i) и это выравнивание заменяется таким набором продолжений $D(L_i)$, что любое оптимальное выравнивание, продолжающее выравнивание L_i, является продолжением одного из выравниваний множества $D(L_i)$. Это гарантирует то, что полученное множество S_{i+1} будет опорным. Далее из построенного множества, возможно, удаляются некоторые избыточные выравнивания, т.е. такие выравнивания, удаление которых оставляет множество S_{i+1} опорным. Таким образом,

$$S_{i+1} \subseteq (S_i - \{ L_i \}) \cup D(L_i).$$

Центральная идея алгоритма в том, что в качестве лидера выбирается максимальное выравнивание текущего опорного множества. Это позволяет избежать рассмотрения «неперспективных» начальных выравниваний и обеспечивает экономию времени при сравнении близких последовательностей.

Время работы алгоритма *Time* и потребная память *Space* описываются соотношениями $Space = O(K + |v_1| + |v_2|)$, где K – длина канонической последовательности; $Time = O(K \log(|v_1| + |v_2|))$. При этом для величины K имеет место следующая оценка. Пусть для слов v_1 и v_2 длин соответственно m_1 и m_2 существует оптимальное выравнивание, содержащее t несовпадений и s удаленных символов. Тогда

$$K \leq (2t + s + 2.5 \cdot |m_1 - m_2|) \cdot \min(m_1, m_2).$$

Отметим, что классические алгоритмы выравнивания могут быть переформулированы в терминах построения последовательностей опорных множеств. Однако, для этих алгоритмов процедура построения опорных множеств универсальна и не зависит от сравниваемых последовательностей. В нашем случае, благодаря введению оценочной функции (специального веса) и использованию в качестве лидера элемента, имеющего максимальный специальный вес, эта процедура адаптивна, что и приводит к экономии времени.

Выбор вида весовой функции и значений ее параметров играет важную роль при выравнивании последовательностей. В разделе 2.3 описан многокритериальный подход, который позволяет построить представление всех выравнива-

ний, которые могут быть оптимальными при заданном виде весовой функции и произвольных числовых коэффициентах. Этот подход основан на использовании векторных весов выравниваний.

Определение 2.3.1. Пусть $k \geq 2$ – целое число. *Векторная весовая функция* – это функция, сопоставляющая каждому выравниванию A k–мерный вектор $V(A)$, называемый (векторным) весом выравнивания A.

Примером векторного веса является вес

$$V(A) = (NumMatch(A), -NumDel(A)),$$

где $k = 2$; $NumMatch(A)$ и $NumDel(A)$ – это соответственно число совпадений и число удаленных символов в выравнивании A. Компоненты векторного веса (в данном случае $NumMatch(A)$ и $-NumDel(A)$) называются элементарными весовыми функциями. Другими примерами элементарных весовых функций являются:

$-NumGap(A)$ – количество удаленных сегментов последовательностей («делеций», "gaps");

$WeightMatch(A)$ – суммарный вес сопоставлений символов относительно выбранной матрицы замен;

$-MisMatch(A)$ – количество несовпадений (соответствует использованию единичной матрицы замен);

Определение 2.3.4. Пусть S_1 и S_2 – последовательности; V – векторная весовая функция. Выравнивание A последовательностей S_1 и S_2 называется Парето-оптимальным относительно весовой функции V, если $V(A)$ является Парето-оптимальным вектором в множестве векторных весов всех возможных выравниваний последовательностей S_1 и S_2.

Важность множества Парето-оптимальных выравниваний определяется следующим наблюдением. Пусть $V(A) = < V_1(A), ..., V_k(A) >$ - векторная весовая функция; $g(x_1, ..., x_k)$ – функция k переменных, монотонно неубывающая по каждому из аргументов, $\varphi(A) = g(V_1(A), ..., V_k(A))$ – скалярная весовая функция и B – оптимальное выравнивание некоторых последовательностей относительно функции φ. Тогда B – оптимальное выравнивание этих последовательностей относительно вектор-функции V. В частности, если B – оптимальное выравнивание некоторых последовательностей относительно линейной комбинации функций $V_1(A), ..., V_k(A)$ с положительными коэффициентами, то B – оптимальное выравнивание этих последовательностей относительно вектор-функции $V(A)$.

В разделе 2.3 представлен алгоритм построения множества всех Парето-оптимальных выравниваний данных последовательностей S_1 и S_2 относительно данной весовой функции $V(A)$. Этот алгоритм является алгоритмом динамического программирования и основан на соотношении дистрибутивности между операцией сложения векторов и операцией взятия Парето-подмножества объединения двух Парето-оптимальных множеств. Более точно, пусть c – число, множества $T, T_1, T_2 \subseteq \boldsymbol{R}^k$, причем T, T_1, T_2 - Парето-оптимальные множества; через $Pareto(T)$ обозначается Парето-подмножество множества T;. Рассмотрим следующие операции:

$$c \otimes T = Pareto(\{< x_1+c, ..., x_k+c> \mid < x_1, ..., x_k > \in T\}),$$

$$T_1 \oplus T_2 = Pareto(T_1 \cup T_2).$$

Тогда

$$c \otimes (T_1 \oplus T_2) = (c \otimes T_1) \oplus (c \otimes T_2).$$

Доказательство непосредственно следует из определения операций $\oplus$ и $\otimes$ и определения Парето-множества.

Для последовательностей длины m_1 *и* m_2 время работы такого алгоритма оценивается как $O(c(m_1, m_2)\cdot m_1\cdot m_2)$, где коэффициент $c(m_1, m_2)$ зависит от выбранной весовой функции и определяется временем выполнения операций $\otimes$ и $\oplus$ для этой функции. В некоторых случаях приведенная оценка может быть улучшена. В частности, оценку времени для случая упомянутой выше весовой функции $VD(A) = (NumMatch(A), -NumDel(A))$ дается следующей леммой:

Утверждение 2.3.4. Пусть S_1 и S_2 – последовательности; их длины равны соответственно n и m. Пусть p – длина наибольшей общей подпоследовательности S_1 и S_2; $d = m + n - 2p$ и $r = \min(p, d)$.

Алгоритм Pareto_Align_Del_Dmax строит множество Парето-оптимальных весов и Парето-оптимальных выравниваний для последовательностей S_1 и S_2 относительно весовой функции $VD(A) = (Match(A), -NumDel(A))$ за время $O(\min(n, 2d)\cdot m\cdot n\cdot\log(r))$.

Такая же оценка верна и для весовой функции $VG(A) = (Match(A), -NumGap(A))$.

Заключительный раздел главы 2 посвящен следующей задаче. Пусть даны две гомологичные, т.е. происходящие от общего предка, биологические последовательности. Как среди множества Парето-оптимальных выравниваний этих последовательностей выбрать наиболее адекватное с биологической точки зрения? Отметим, что при традиционном подходе к задаче парного выравнивания выбор искомого выравнивания производится путем установки штрафов за удаление фрагментов. Изложенные ниже результаты могут трактоваться как подход к обоснованию выбора адекватных значений этих штрафов.

Предположим, как это обычно делается в теории эволюции, что сравниваемые последовательности получены из предковой последовательности с помощью операций замены символов, а также операций удаления и вставки фрагментов. Учитывая, что теория построения весовых матриц сопоставлений хорошо разработана, будем считать также, что выбрана адекватная матрица весов сопоставлений.

Предлагаемый подход основан на следующем наблюдении. Рассмотрим две последовательности u и v и фиксируем некоторую весовую матрицу замен. Для произвольного целого g, через $S(g)$ обозначим наибольший суммарный вес сопоставлений среди выравниваний, содержащих не более g удаленных фрагментов. Через $D(g)$ обозначим «производную» $S(g)$:

$$D(g) = S(g+1) - S(g).$$

Значения $S(g)$ могут быть найдены путем построения Парето-оптимальных выравниваний относительно векторной весовой функции

$$WG(A) = (WeightMatch(A), -NumGap(A))$$

Допустим, что удалением некоторого количества фрагментов из последовательностей u и v можно получить достаточно похожие последовательности равной длины; эти фрагменты назовем «правильными». Удаление «правильного» фрагмента восстанавливает соответствие между (обычно, протяженными) гомологичными фрагментами исходных последовательностей и, тем самым, ведет к существенному увеличению веса $S(g)$ (значение $D(g)$ велико). Когда все

«правильные» фрагменты удалены, новые удаления уже не мотивированы биологически и, следовательно, не могут привести к существенному увеличению $S(g)$ (значение $D(g)$ мало). Наибольшее значение параметра g, при котором значение $D(g)$ велико, назовем критическим значением, а соответствующее ему выравнивание - критическим выравниванием. Мы полагаем, что критическое выравнивание наиболее (среди других Парето-оптимальных выравниваний) соответствует биологически корректному выравниванию данных последовательностей. Это приводит к следующему методу выравнивания последовательностей.

1. Построить множество Парето-оптимальных выравниваний относительно векторной весовой функции $WG(A) = (WeightMatch(A), -NumGap(A))$.

2. Определить критическое значение количества удаленных фрагментов g и взять критическое выравнивание в качестве результата.

Успешность применения этого метода зависит от алгоритма, который определяет порог, отделяющий «малые» и «большие» значения $D(g)$. Компьютерные эксперименты показали, что для нуклеотидных последовательностей со степенью сходства не менее 30% и для аминокислотных последовательностей со степенью сходства не менее 20% выбор такого порога возможен. Получена верхняя оценка для величины D_{cr} «малых» значений $D(g)$:

$$D_{cr} \le \log(L)/\log(1/(1-p)) - 1, \qquad (2.4.1)$$

где L – средняя длина последовательностей, p – доля совпадающих букв. Эта оценка хорошо согласуется с данными компьютерных экспериментов при $p \ge 0.4$ (см. таблицу 2.4.8). Величина ошибки увеличивается с убыванием p и возрастанием L.

Таблица 2.4.8.

Оценки значений величины D_{cr}

L		P						
		0.3	0.4	0.5	0.6	0.7	0.8	0.9
200	FORM	13.8	9.4	6.6	4.8	3.4	2.3	1.3
	EXP	10.0	8.0	6.0	4.0	3.0	2.0	1.0
300	FORM	15.0	10.2	7.2	5.2	3.7	2.5	1.5
	EXP	10.0	8.0	6.0	5.0	3.0	2.0	1.0
700	FORM	17.3	11.8	8.5	6.1	4.4	3.1	1.8
	EXP	11.0	9.0	7.0	5.0	3.0	3.0	1.0
1000	FORM	18.3	12.5	9.0	6.6	4.7	3.3	2.0
	EXP	11.0	9.0	7.0	6.0	4.0	3.0	2.0

Верхняя граница для величины D_{cr}, вычисленная по формуле (2.4.1), строки FORM, и максимальные значения величины D_{cr}, полученные в ходе компьютерных экспериментов (см. п.2.4.2 диссертации, строки EXP. Данные приведены для различных длин L и уровней сходства p.

В свою очередь, ожидаемое значение M величины $D(g)$ при удалении «правильного» фрагмента (что соответствует «докритической» области значений $D(g)$) зависит не только от L и p, но и от других характеристик сравниваемых последовательностей. Такими характеристиками являются: длина d удаляемого фрагмента, длина D каждого из гомологичных фрагментов, которые оказались сопоставленными правильно в результате удаления; доля совпадений p_0 при сопостав-

лении негомологичных (случайных) фрагментов последовательностей. Для среднего значения M имеем формулу:

$$M = D(p - p_0) - pd.$$

Если указанное значение будет менее D_{cr}, то удаление таких «правильных» фрагментов не может быть диагностировано на фоне статистического шума. Это показывает статистические пределы применимости алгоритмических методов восстановления биологически корректных выравниваний. Другие ограничения возможностей алгоритмических методов рассмотрены в главе 3.

Таким образом, в главе 2 рассмотрена общая задача построения оптимального выравнивания символьных последовательностей, в частности, - проблема выбора штрафов за удаление фрагментов. В главе 3 рассматривается более специальная задача - выравнивание биологических последовательностей. Центральная тема этой главы – учет пространственной структуры сравниваемых молекул при сравнении их первичных структур (последовательностей). Имея это ввиду, мы ограничиваемся рассмотрением наиболее распространенного в современной биоинформатике класса весовых функций удаления фрагментов – аффинными функциями. Это позволяет существенно упростить изложение, хотя приведенные в разделах 3.2 и 3.3 алгоритмы могут быть обобщены на случай рассмотренных в главе 1 кусочно-линейных функций. В разделе 3.4. представлен оригинальный алгоритм предсказания вторичной структуры РНК.

Раздел 3.1 посвящен изучению связи между биологически корректными выравниваниями аминокислотных последовательностей и выравниваниями, полученными с помощью алгоритма Смита-Уотермана (*SW*) - наиболее распространенного в настоящее время алгоритма построения оптимального выравнивания последовательностей. В качестве «биологически корректных» (эталонных) выравниваний использованы выравнивания, основанные на наложении пространственных структур белков. Адекватность такого подхода обоснована существенно большей консервативностью пространственной структуры белков по сравнению с их первичной структурой. В качестве источника эталонных выравниваний использовалась база данных BaliBase [http://www-igbmc.u-strasbg.fr/BioInfo/BAliBASE/]. Большая часть сравниваемых пар последовательностей (всего 583 пары) имела процент совпадений %ID от 15% до 40%.

Исследована зависимость степени сходства между алгоритмическими и структурными выравниваниями от степени сходства между сравниваемыми последовательностями и выявлены причины расхождений между этими выравниваниями. В качестве количественной оценки качества алгоритмически полученных выравниваний использовались две взаимодополняющих меры: (1) *Точность выравнивания* (обозначение: *Ali_Acc)* равна отношению количества одинаковых сопоставлений (I) в обоих выравниваниях к общему количеству сопоставлений в эталонном выравнивании (G): Ali_Acc = I/G*100 %. (2) *Достоверность выравнивания* (обозначение: *Ali_Conf*), равна отношению количества одинаковых сопоставлений в обоих выравниваниях (I) к общему количеству сопоставлений в алгоритмически построенном выравнивании (A): Ali_Conf = I/A*100 %. Неформально говоря, точность *Ali_Acc* показывает, какую долю эталонного выравни-

вания удалось восстановить, а достоверность *Ali_Conf* – насколько можно доверять построенному выравниванию.

Как видно из рис. 3.1.2, алгоритм SW может строить выравнивания, хорошо совпадающие с эталонными, только при уровне сходства сравниваемых белков более 30-40%. Этот диапазон уровня гомологии (*ID* > 30%) примерно совпадает с известным порогом *%ID*, выше. которого можно достоверно восстановить структурное выравнивание, зная только последовательности [%56, %58-60]

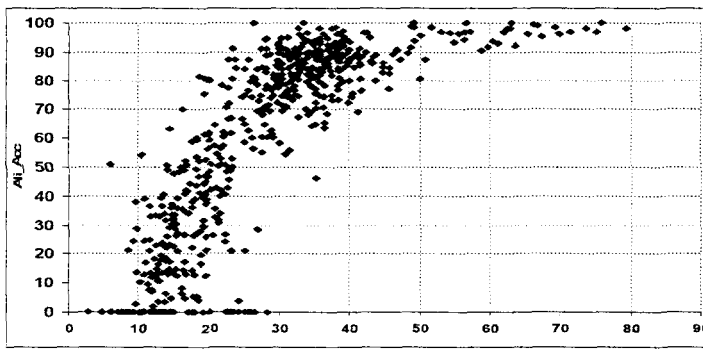

Рис. 3.1.2. Зависимость точности восстановления эталонных выравниваний методом SW от уровня сходства (%ID) сравниваемых белков. Каждая точка соответствует паре эталонных белков. X-координата точки равна *%ID* пары, а *Y*-равный координата – значению точности. Зависимость для достоверности практически такая же

При уровне гомологии меньше 10% метод SW не может восстановить правильное выравнивание даже частично. Для диапазона уровня гомологии от 10% до 30% выравнивания Смита-Уотермана показывают очень широкий разброс точности и достоверности. Для разных пар последовательностей с одинаковым уровнем сходства построенные SW-выравнивания могут иметь очень различные значения точности и достоверности. Это означает, что в этом диапазоне *%ID* качество алгоритмических выравниваний определяется не только уровнем сходства сравниваемых белков, но и «внутренними свойствами» их эталонных выравниваний. Эти «внутренние» свойства удобно формулировать в терминах т.н. «островов».

Определение 3.1.1. Островом в выравнивании $A = \langle u, v, Q \rangle$ называется непродолжаемая последовательность сопоставлений, не разделенных удалениями фрагментов. Весом острова называется сумма весов входящих в остров сопоставлений. Выравнивание можно представить как цепочку островов, разделенных делециями.

Эталонные выравнивания и SW-выравнивания имеют существенно различную структуру островов с точки зрения веса островов (см. рис. 3.1.5). Неожиданно много эталонных островов имеют очень низкий или даже отрицательный вес, в то время как алгоритмические выравнивания совсем не содержат островов малого веса. Стоит отметить, что суммарная длина таких «слабых» островов в эталонных выравниваниях достаточно велика (см. рис. 3.1.5 б)

Эталонные острова веса меньше 5 составляют 32% от всех островов и покрывают 20% всей длины эталонных выравниваний. Только 5% островов такого малого веса были восстановлены алгоритмом. Для выравниваний из серой зоны ($10 < \%ID \leq 30$) картина еще более критическая – восстановлено всего 2.5% островов веса меньше 5. Эти «слабые» острова обычно не имеют шансов быть восстановленными любым алгоритмом, использующим данную матрицу замен.

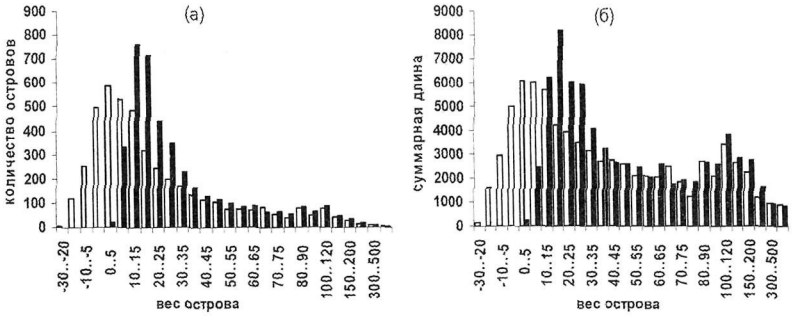

Рисунок 3.1.5. (а) Гистограмма распределения количества островов в эталонных выравниваниях (белый) и выравниваниях SW (черный) по весу острова. (б) Суммарная длина островов, имеющих вес в пределах диапазонов. Эталонные острова – белый, SW – черный

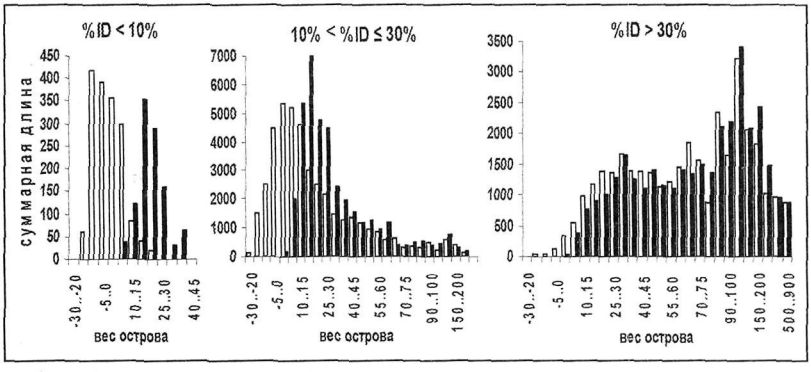

Рисунок 3.1.6. Более детальное представление данных рис. 3.1.5а. Гистограммы суммарных длин островов с весом в пределах указанных по оси X диапазонов отдельно для каждой их 3-х областей *%ID*.Данные по эталонным островам показаны белым, SW – черным

С увеличением степени сходства сравниваемых белков (см. рис. 3.1.6) различие в весе эталонных и построенных островов уменьшается. Однако даже для высокого уровня сходства белков ($ID > 30\%$) встречаются эталонные острова отрицательного веса.

Таким образом, даже наиболее точный из используемых в настоящее время алгоритмов, SW-алгоритм, не способен надежно воссоздать выравнивание

пространственных структур, если идентичность последовательностей ниже 30%. В разделе 3.2 представлен подход, который позволяет существенно повысить как точность, так и достоверность алгоритмических выравниваний первичных структур белков. Идея подхода состоит в явном учете сведений о вторичной структуре сравниваемых белков, при этом примерно с равным успехом может использоваться как экспериментально полученная вторичная структура, так и теоретически предсказанная. Таким образом, метод применим и для выравнивания белков с неизвестной пространственной структурой. Наша работа не была первой работой в этом направлении, однако мы впервые исследовали влияние учета сведений о вторичной структуре белков на точность и достоверность получаемых выравниваний и получили лучшее по сравнению с предшественниками качество выравниваний.

Представленный в разделе 3.2 алгоритм STRUSWER является модификацией алгоритма Смита-Уотермана. Отличие состоит в том, что при сопоставлении i-го аминокислотного остатка одной последовательности и j-го остатка другой к весу сопоставления добавляется бонус. Этот равен произведению коэффициента SBON, определяющего вклад вторичной структуры в вес сопоставления, на величину сходства элементов вторичной структуры.

Как указывалось выше, предложенный метод может использоваться как с экспериментально полученными, так и с теоретически предсказанными вторичными структурами. Для предсказания вторичной структуры использовалась программа PSIPRED [Bryson K, McGuffin LJ, Marsden RL, Ward JJ, Sodhi JS. & Jones DT. Protein structure prediction servers at University College London. Nucl. Acids Res. 2005. Vol. 33 (Web Server issue): W36-38] в двух режимах: совместного предсказания структуры для группы гомологичных белков ("full version") и предсказание структуры только по аминокислотной последовательности ("single version"). При этом точность предсказания для использованного набора белков составила соответственно 82% и 65%, что согласуется с результатами, приведенными на сервере EVA (http://cubic.bioc.columbia.edu/eva/).

Для каждой из этих версий использовалось два способа представления предсказанной вторичной структуры. В первом случае («тип_структуры»), каждому остатку аминокислотной последовательности приписывается определенный символ вторичной структуры (H – спираль; E – бета-структура; L – петля). Во втором («вероятность_структуры»), каждому остатку приписываются вероятности принадлежности остатка к каждому из трех типов вторичной структуры, которые также рассчитываются программой PSIPRED.

При тестировании качество выравниваний, полученных методом STRUSWER с различными способами разметки вторичной структуры (см. выше) сравнивалось с качеством выравниваний, полученных методом SW, а также методом WFMFL, представленный в работе [Wallqvist A, Fukunishi Y, Murphy L.R., Fadel A, Levy R.M. Iterative sequence/secondary structure search for protein homologs: comparison with amino acid sequence alignments and application to fold recognition in genome databases. Bioinformatics. 2000. V. 16. P. 988–1002]

В качестве эталонных выравниваний, как и в разделе 3.1, использовались выравнивания из базы BaliBase (см. выше); корпус эталонных парных выравниваний был разделен на обучающий и тестовый наборы. Источником данных об

экспериментально определенных вторичных структурах белков служила база данных DSSP [Kabsch W, Sander C. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers. 1983. V. 22, P. 2577–2637].

Таблица 3.2.2.

Точность (Acc) и достоверность (Conf) различных методов выравнивания при тестировании на тестовой выборке

Метод	SBON	GOP	GEP	Acc	Conf	ID < 30%	
						Acc	Conf
SW	не исп.	7	1	0.525	0.585	0.353	0.429
а) предсказание вторичной структуры по последовательности							
STRUSWER_SIN_S	2	10	1	0.578	**0.622**	0.428	**0.482**
STRUSWER_SIN_%	7	8	2	**0.602**	0.618	**0.461**	0.477
WFMFL_SIN	не исп.	13	1	0.399	0.488	0.263	0.346
б) предсказание вторичной структуры с привлечением данных о гомологичных белках							
STRUSWER_PSI_S	8	9	1	0.659	0.683	0.546	0.573
STRUSWER_PSI_%	17	6	2	**0.683**	**0.695**	**0.579**	**0.589**
WFMFL_PSI	не исп.	16	1	0.631	0.672	0.503	0.560
в) экспериментально известная структура							
STRUSWER_EXP	8	10	1	**0.677**	**0.700**	**0.577**	0.601
WFMFL_EXP	не исп.	15	1	0.638	0.698	0.527	**0.602**

Параметры (SBON, GOP, GEP) были подобраны на обучающей выборке для получения максимальной точности (Acc) каждым из методов. Представлены данные как для всей тестовой выборки (288 пар белков) так и для «серой зоны» (белки с гомологичностью ниже 30%, 182 пары). Для методов STRUSWER и WFMFL указаны способы разметки вторичной структуры: Exp – экспериментально полученная структура; PSI – предсказание структуры по гомологии ("full version" PSIPRED); SIN. Суффиксы _S и _% указывают способ представления предсказаний: «*тип_структуры*» (_S) или «вероятность_структуры» (_%). Метод WFMFL не приспособлен для использования предсказаний, представленных вероятностями структур

Результаты тестирования представлены в таблице 3.2.2. Параметры каждого из методов, были предварительно подобраны на обучающей выборке выравниваний. Отметим, что параметр SBON имеет смысл только для метода STRUSWER, а параметры аффинных штрафов за удаление (GOP, gap opening penalty, и GEP, gap elongation penalty) используются всеми методами. Значения параметров в таблице 3.2.2 были получены путем максимизации средней точности выравниваний по обучающей выборке. Результаты тестирования с параметрами, полученными при оптимизации достоверности, носят сходный характер.

Наилучшие результаты, как по точности, так и по достоверности показали методы, основанные на экспериментально известной вторичной структуре и методы, использующие предсказание вторичной структуры с привлечением данных о таковой в гомологичных белках. Однако эти результаты представляют, скорее, методический интерес. Как правило, экспериментально определенная вторичная структура предполагает наличие известной пространственной структуры, и тогда имеет смысл воспользоваться одной из программ выравнивания белков по их пространственной структуре. Вместе с тем, тесты на экспериментально известной вторичной структуре показывают примерный предел, которого

можно достичь в данном методе, используя вторичную структуру совместно с последовательностью. С другой стороны, привлечение гомологов и их вторичных структур для выравнивания пары белков, пусть и в форме предсказаний вторичной структуры, противоречит смыслу парного выравнивания. Результаты выравнивания двух белков с использованием таких «предсказаний по гомологии» следует сравнивать с результатами множественного выравниваниями использованной группы белков.

Таким образом, в качестве основного метода предсказания вторичной структуры следует рассматривать метод, делающий предсказание вторичной структуры по отдельной аминокислотной последовательности, в данной работе – программой PSIPREDsingle. Обладая меньшей, чем полный вариант PSIPREDfull, точностью предсказаний, он, тем не менее, имеет ряд преимуществ. Первое преимущество состоит в том, что программа PSIPREDsingle не основана на поиске гомологов, и поэтому STRUSWER в соответствующих режимах (STRUSWER_SIN_S и STRUSWER_SIN_%) можно использовать в случаях, когда произвести поиск гомологов по тем или иным причинам невозможно. Второе преимущество вытекает из первого, и заключается в существенно меньших затратах времени, необходимого для предсказания вторичной структуры. Подобный факт может оказаться решающим в больших вычислительных проектах. Выравнивания, сделанные алгоритмом STRUSWER_SIN, с использованием вторичной структуры, предсказанной по аминокислотной последовательности, превосходят по качеству аналогичные выравнивания, полученные алгоритмом SW и алгоритмом WFMFL_SIN, как по точности, так и по достоверности. Все указанные соотношения остаются верными, если мы ограничимся только слабогомологичными парами белков. При этом относительный выигрыш от использования вторичной структуры существенно возрастает.

В разделе 3.3. тема построения оптимального выравнивания биологических последовательностей, обогащенных сведениями о структуре соответствующих макромолекул, продолжена применительно к последовательностям РНК. Важное отличие РНК от белков состоит в том, вторичная структура РНК (рассматриваются только структуры не содержащие псевдоузлов) описывается не словом, а упорядоченным корневым деревом.

Определение 3.3.3. *РНК-дерево* – это такое корневое упорядоченное дерево, что

(1) листья помечены буквами РНК-алфавита {A, C, G, U};

(2) каждая внутренняя вершина имеет по меньшей мере двух сыновей; причем самый левый и самый правый из этих сыновей – листья, они называются *главными* сыновьями.

РНК-лесом $F = <T_1, ..., T_n>$ называется упорядоченное множество РНК-деревьев.

Говоря неформально, каждая внутренняя вершина соответствует «спариванию» (т.е. образованию водородной связи, pairing) своих главных сыновей. Порядок на ребрах, выходящих из одной вершины, индуцирует порядок на множестве листьев, тем самым, РНК-лес однозначно определяет последовательность РНК.

Представленный в разделе 3.3 алгоритм строит оптимальное выравнивание РНК-лесов при аффинных штрафах за удаление фрагментов РНК. Уточним постановку задачи.

Пусть $<S_1, P_1>$, $<S_2, P_2>$ - РНК-структуры без псевдоузлов, здесь S_i - последовательность, P_i – множество спариваний ($i=1, 2$); G – выравнивание S_1 и S_2. Будем говорить что в выравнивании G спаривание $(x_1, y_1) \in P_1$ сопоставлено спариванию $(x_2, y_2) \in P_2$, если x_1 сопоставлено x_2 и y_1 сопоставлено y_2.

Весовая система для выравнивания структур РНК – это пятерка $<M, g, d, b, c>$, где M – весовая матрица замен; g и d – это коэффициенты аффинной весовой функции удалений фрагментов; b – бонус за сопоставление спариваний, c – штраф за каждое спаривание, не участвующее в сопоставлениях.

Пусть задана весовая система $<M, g, d, b, c>$; $<S_1, P_1>$ и $<S_2, P_2>$ структуры и A – их выравнивание с набором сопоставленных позиций (склеек) $\{<p_k, q_k> \mid k = 1,..., N\}$. Пусть m – общее число удаленных фрагментов в A; l – суммарная длина этих фрагментов, k – количество спариваний в P_1 and P_2, которые не сопоставлены в A другим спариваниям: $t = (|P_1| + |P_2| - k)/2$ - количество сопоставлений спариваний. Тогда структурным весом выравнивания A называется величина

$$Score(A) = \Sigma_{k=1..N} M(S_1[p_k], S_2[q_k]) - g \cdot m - d \cdot l + b \cdot t - c \cdot k. \qquad (3.3.1)$$

Это определение соответствует определению веса глобального выравнивания последовательностей с аффинными весовыми функциями удаления фрагментов, к которому добавлены бонусы и штрафы за правильное (неправильное) сопоставление спариваний.

Представленный алгоритм сводит задачу построения оптимального выравнивания последовательностей РНК с известными вторичными структурами при заданной весовой системе $<M, g, d, b, c>$ к обобщению задачи выравнивания лесов на случай РНК-лесов. Специфика РНК-лесов заключается (1) в различной интерпретации листьев и внутренних вершин (нуклеотиды и спаривания), в частности, в понятии главных сыновей и (2) в использовании аффинной функции штрафов за удаление символов.

Определение 3.3.4. *Выравниванием* РНК-лесов F_1 и F_2 называется такое множество $A \subseteq Vertex(F_1) \times Vertex(F_2)$, что для каждой пары (v_1,w_1), $(v_2,w_2) \in A$ выполнено:

1) $v_1 = v_2$ тогда и только тогда, когда $w_1 = w_2$ (иными словами, A – взаимно однозначное соответствие между подмножествами $Vertex(F_1)$ и $Vertex(F_2)$).

2) вершина v_1 предшествует вершине v_2 в смысле левого обхода леса F_1 тогда и только тогда, когда вершина w_1 предшествует вершине w_2 в смысле левого обхода леса F_2.

3) если $(v, w) \in A$, то либо обе вершины v и w – листья, либо обе они – внутренние вершины;

4) Пусть $(v, w) \in Vertex(F_1) \times Vertex(F_2)$; v_1 и v_2 – главные сыновья вершины v; w_1 и w_2 – главные сыновья вершины w. В этих условиях $(v, w) \in A$ тогда и только тогда, когда (v_1,w_1), $(v_2,w_2) \in A$.

Первые два условия традиционны для выравнивания упорядоченных лесов общего вида. Условия (3) и (4) отражают специфику РНК-лесов. Определение веса выравнивания РНК-лесов очевидным образом следует из (3.3.1).

Алгоритм построения оптимального выравнивания РНК-лесов - это алгоритм динамического программирования, использующий парадигму «левый-правый», предложенную в работе Клейна [Klein P.N.. Computing the edit-distance between unrooted ordered trees // Proceedings of the 6th European Symposium on Algorithms(ESA). 1998. P. 91-102]. В указанной работе вес удаления фрагмента пропорционален его длине, что соответствует посимвольным удалениям. Для того, чтобы использовать аффинные штрафы за удаления фрагментов, для каждого промежуточного РНК-леса вычисляются значения дополнительных (по сравнению с алгоритмом Клейна) характеристик.

Определение 3.3.13. Пусть L, $R \subseteq \{1, 2\}$ и A - выравнивание лесов F_1 и F_2. Пусть $k_L(A)$ (соответственно, $k_R(A)$) - количество таких индексов $i \in L$ (соответственно, $i \in R$) , что лес F_i пуст или при выравнивания A в лесе F_i не удален самый левой (соответственно, правый) лист. Делеционным весом $WD(A, L, R)$ выравнивания A при ограничениях L, R называется величина

$$WD(A, L, R) = W(A) - (k_L + k_R) \cdot g,$$

где $W(A)$ – вес A относительно выбранной весовой системы $<M, g, d, b, c>$; g – штраф за удаление фрагмента, аналогичный *gap opening penalty* алгоритма Смита-Уотермана.

Определение 3.3.14. Пусть $<F_1, F_2>$ - пара лесов, L, $R \subseteq \{1, 2\}$. Через $Best(F_1, F_2, L, R)$ обозначается наибольшее возможное значение делеционного веса $WD(A, L, R)$ для выравниваний A лесов F_1 и F_2.

Очевидно,

$WD(A, \varnothing, \varnothing) = W(A)$

и $Best(F_1, F_2, \varnothing, \varnothing)$ – это вес оптимального выравнивания лесов F_1 и F_2.

Сложность работы алгоритма естественно выражается через количество промежуточных пар лесов K, обработанных в ходе работы алгоритма. Потребная память есть $O(K)$, а время - $O(K \log(K))$. Аналогично алгоритму Клейна, можно показать, что $K = O(m^2 n \lg(n))$, где $m \leq n$ – длины сравниваемых последовательностей. Таким образом, получаем для памяти оценку $O(m^2 n \log(n))$, а для времени работы - оценку $O(m^2 n \log^2(n))$.

В разделе 3.4 представлен алгоритм вычисления энергии внутренних петель в структурах РНК в рамках модели Цукера-Тернера-Мэтьюза (Nearest Neighbour Model, NNM) – общепринятой в настоящее время модели строения РНК. Такой алгоритм является необходимой частью общего алгоритма поиска оптимальной (т.е. имеющей минимальную свободную энергию) структуры для данной последовательности РНК в рамках NNM. Как было показано в предыдущих разделах главы 3, использование сведений о вторичной структуре, в том числе – теоретических предсказаний структуры, важно для сравнения биологических последовательностей.

В рамках модели NNM энергия структуры РНК представляется как сумма энергий т.н. петель (loop). Каждую петлю можно представить себе, как простой цикл в представлении вторичной структуры РНК в виде планарного графа. В этом графе вершинами являются нуклеотиды, а ребра (двух видов) изображают

соответственно валентные и водородные связи. Петли делятся на типы в зависимости от количества входящих в них водородных связей и длин спаренных участков (см. определения в п. 3.4.2.2 диссертации).

Фиксируем последовательность РНК длины L и множество U, состоящее из M спариваний позиций этой последовательности; спаривания из U будут называться *допустимыми*. Множество $ROW_B = \{(x, B) \in U\}$ называется B-й строкой.

Определение 3.4.2. Пусть $P \subseteq U$ – структура, $(A, B), (p, q) \in P$. Спаривания $(A, B), (p, q)$ образуют *внутреннюю петлю* (internal loop), если $A < p < q < B$ и ни один из нуклеотидов x, где $A < x < p$ или $q < x < B$, не участвует в спариваниях структуры P. Спаривание (A, B) называется замыкающим спариванием петли; спаривание (p, q) - внутренним спариванием петли; фрагменты $[A+1, p-1]$ и $[q+1, B-1]$ – крыльями петли.

Определение 3.4.4. Будем говорить, что спаривание (A, B) – *замыкающее спаривание* (closing pairing) в структуре P, если P не содержит пар (x, y), где $x < A$ или $y > B$.

Алгоритм, который предсказывает вторичную структуру РНК в рамках модели NNM, просматривает все допустимые спаривания строка за строкой в порядке возрастания номеров строк $B \in [1, L]$. При этом, для каждого спаривания $(A, B) \in ROW_B$ вычисляется оптимальная структура каждого типа $t \in \{0, 1, 2, 3\}$, тип структуры определяется типом петли, к которой принадлежит замыкающее спаривание структуры. Далее просмотром этих «частных» оптимальных структур вычисляется структура, имеющая наименьшую энергию $\Delta G(A, B)$ среди всех структур, не имеющих спаренных нуклеотидов вне сегмента $[A, B]$.

Энергия структуры, в которой замыкающая склейка (A, B) принадлежит внутренней петле, образованной склейками (A, B) и (x, y), задается формулой

$$\begin{aligned}&\Delta G_{Internal\_Loop}(A, B; x, y) = \\ &= f_{Len}(d_A+d_B) + f_{Diff}(d_A-d_B) + \Delta G(x, y) = \\ &= f_{Len}((B-A) - (y-x+2)) + f_{Diff}((B+A) - (y+x)) + \Delta G(x, y).\end{aligned} \qquad (3.4.1)$$

Здесь $f_{Len}(t)$ – фиксированная выпуклая функция, она хорошо аппроксимируется логарифмической функцией. Функция $f_{Diff}(t)$ определяется соотношениями:

$$\begin{aligned}&f_{Diff}(t) = base\_level, && \text{если } |t| \geq width, \\ &f_{Diff}(t) = (base\_level / width)*|t|, && \text{если } |t| < width.\end{aligned} \qquad (3.4.2)$$

Таким образом, энергия оптимальной структуры среди структур с замыкающей склейкой (A, B) при условии, что эта склейка принадлежит внутренней петле, задается формулой

$$\begin{aligned}&\Delta G_{Internal\_Loop}(A, B) = \\ &= min\{\Delta G_{Internal\_Loop}(A, B; x, y) \mid (x, y) \in U \ \& \ A<x<y<B\} = \\ &= min\{ f_{Len}((B-A) - (y-x+2)) + f_{Diff}((B+A) - (y+x)) + \Delta G(x, y) \mid \\ &\qquad\qquad \mid (x, y) \in U \ \& \ A<x<y<B\}.\end{aligned} \qquad (3.4.3)$$

Говоря неформально, $\Delta G_{Internal\_Loop}$ описывает увеличение (т.е. ухудшение – структура с большей энергией менее стабильна) энергии структуры РНК в связи с образованием внутренней петли. Это увеличение зависит от длин d_A и d_B крыльев петли и представляется в виде суммы двух слагаемых. Одно из слагае-

мых ($f_{Len}(t)$) зависит от суммарной длины крыльев, другое ($f_{Diff}(t)$) – от асимметрии петли, т.е. величины $| d_A - d_B |$. Если асимметрия мала, то $f_{Diff}(t)$ растет линейно; для значений асимметрии, превышающих порог *width*, значение $f_{Diff}(t)$ постоянно. Такой своеобразный вид функции $\Delta G_{Internal\_Loop}(A, B; x, y)$ и представляет основную трудность при анализе внутренних петель.

В первых алгоритмах построения оптимальной вторичной структуры минимум в (3.4.3) находился прямым перебором, что приводило к общему времени анализа внутренних петель $O(L^4)$; позднее был предложен алгоритм [Lyngsø *et al.* Fast evaluation of internal loops in RNA secondary structure prediction //Bioinformatics. 1999. V. 15. P. 440-445] со сложностью $O(L^3)$. Представленный в разделе 3.4 алгоритм AFOLD использует технику динамического программирования для разреженных матриц (sparse dynamic programming, см. [Eppstein *et al.* Sparse Dynamic Programming II: Convex and Concave Cost Functions J. ACM. 1992. V. 39, P. 546–567]) и имеет временную сложность $O(M^*\log^2(L))$, где L - длина последовательности, M - количество теоретически допустимых спариваний. Т.к. $M \le L^2$, эта оценка существенно улучшает оценку $O(L^3)$ лучшего из ранее предложенных алгоритмов. Отметим, что использованная алгоритмическая техника близка к технике выравнивания последовательностей и может быть использована при построении выравниваний геномов.

В заключение разбора раздела 3.4 сделаем два замечания. Во-первых, в отличие от предлагавшихся ранее подходов, в оценку времени нашего алгоритма явно входит число M допустимых склеек. Благодаря этому алгоритм хорошо сочетается с различными (как экспериментальными, так и теоретическими) способами предварительного отбора (фильтрации) допустимых склеек. Во-вторых, для некоторых классов относительно небольших молекул РНК известно, что их структуры не содержат ветвящихся петель. Для таких молекул наш подход позволяет построить оптимальную структуру за время $O(M^*\log^2(L))$.

Рассмотренные в главах 2 и 3 алгоритмы оптимального выравнивания имеют квадратичную временную сложность – их время работы (в худшем случае) пропорционально произведению длин последовательностей. Такое время слишком велико для многих биоинформатических приложений, в частности, - для сравнения синтенных фрагментов геномов (длина ~ 10^7 нуклеотидов. Глава 4 посвящена выравниванию последовательностей, не связанному с оптимизацией весовой функции, и возникающим в этой связи алгоритмическим задачам. В разделе 4.1 представлен иерархический алгоритм геномного выравнивания OWEN, основанный на построении систем коллинеарных локальных сходств. Разделы 4.2 и 4.3 посвящены поиску локальных сходств с помощью т.н. затравок – наиболее распространенному и быстродействующему из известных методов построения локальных сходств. В разделе 4.2 дано определение затравки в наиболее общем виде и рассмотрен простейший класс «неклассических» затравок – разреженные затравки, предложенные в [Ma, B., Tromp, J, and Li, M. PatternHunter: Faster and More Sensitive Homology Search // Bioinformatics. 2002. V. 18. P. 440-445, 2002]. Для этого класса указан вид оптимальной оптимальной затравки среди затравок с одной несущественной позицией. Раздел 4.3 посвящен введенному нами (совместно с Г.Кучеровым и Л. Ноэ) классу затравок – классификационным затравкам, и некоторым алгоритмическим проблемам, связанным

с вычислением чувствительности затравок этого класса. Задача вычисления чувствительности произвольных затравок рассмотрена в разделе 5.2 главы 5.

Задача выравнивания геномов, точнее – выравнивания синтенных (коллинеарных) участков генома, на рубеже веков стала одной из основных задач алгоритмической биоинформатики. Эта задача существенно отличается от рассмотренных в предыдущих главах задач выравнивания белков и относительно коротких фрагментов ДНК. В последнем случае существует сходство между последовательностями в целом, это сходство может быть представлено глобальным выравниванием, которое, в свою очередь, может быть (с погрешностями) восстановлено путем построения оптимального выравнивания относительно подходящей весовой функции. Существенным обстоятельством является и то, что для белков возможно построение структурных выравниваний, которые могут служить эталоном при проверке адекватности выбранной весовой функции (см. подробнее п.3.1).

Сходство синтенных участков геномов «средне близких» организмов (человек – мышь, *C.Elegance – C.Briggsae* и т.п.) существенно неоднородно и распадается на локальные сходства. Как правило, эти сходства являются статистически достоверными, т.е. имеют достаточно низкое значение *P*-значения (*P-value*) для заданных длин последовательностей и распределения вероятностей. Большинство статистически достоверных сходств коллинеарны, т.е. проекции на оба сравниваемых генома расположены в одинаковом порядке. Это объясняется тем, что основные эволюционные события (замена нуклеотида, удаление/вставка фрагмента) не нарушают коллинеарности. Возможные нарушения коллинеарности (*конфликты между сходствами, см. рис.* 4.1.1), связаны с более редкими эволюционными событиями.

Есть три основных источника нарушения коллинеарности локальных сходств или *микроколлинеарности* (термин «микроколлинеарность» используется, чтобы отличить коллинеарность локальных сходств, как кодирующих, так и некодирующих, от *макроколлинеарности* - коллинеарности генов в синтенных участков геномов). Этими источниками являются (1) локальная конвергентная эволюция, (2) консервативный повтор, присущий обоим геномам, но по-разному расположенный в них; (3) локальные перестановки в геноме.

В случае средне-близких геномов конвергентная эволюция, как правило, не приводит к достаточно высокому уровню сходства, и, следовательно, при конфликте такого сходства со сходством ортологичных сегментов именно последние будут иметь более высокий уровень значимости. Вставки повторов могут (хотя и редко) привести к появлению неортологичных сходств высокого уровня значимости, поэтому повторы желательно предварительно замаскировать. Таким образом, отдавая предпочтение более значимому из конфликтующих сходств, мы, как правило, выбираем сходство между ортологичными фрагментами геномов (см. рис.4.1.2).

В случае локальной перестройки оба сходства имеют равное право на включение в основную цепочку локальных сходств. Поэтому и в этом случае выбор на основе величины значимости оправдан.

Таким образом, геномное выравнивание можно строить, исходя из следующих принципов.

1. Все конфликты между статистически значимым локальным сходством и любым количеством индивидуально менее значимых сходств разрешаются в пользу первого. Таким образом, сходство, которое не имеет конфликтов с более значимыми сходствами, всегда включается в цепь локальных сходств, образующих глобальное выравнивание.

2. Описанный выше принцип применяется как при сравнении последовательностей в целом, так и при сравнении их ортологичных фрагментов («фрактальность»). Благодаря этому, после того, как выделена цепь локальных сходств, значимых на уровне последовательностей в целом, алгоритм может пополнить эту цепь менее сильными сходствами, расположенными в коллинеарных участках между уже выделенными сходствами. Эти «дополнительные» сходства не являлись значимыми при сравнении исходных последовательностей длиной, скажем, 10^7, но становятся значимыми при сравнении коллинеарных фрагментов длиной, скажем 10^3. Иными словами, более «сильные» локальные сходства поддерживают коллинеарные с ними более слабые сходства.

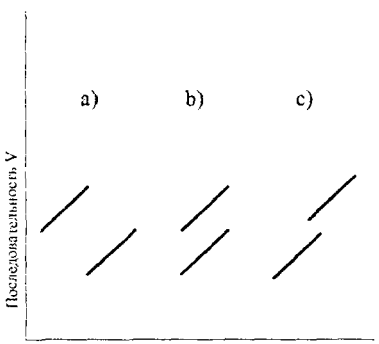

Рис. 4.1.1. Матрица сходств (dot-matrix), на которой показаны различные вида конфликтов между локальными сходствами. a) Неустранимый конфликт: сходные фрагменты на разных последовательностях расположены в разном порядке. b) Неустранимый конфликт: проекции сходств на последовательность *U* существенно перекрываются. c) Есть незначительное перекрытие проекций сходств на последовательность *U* В этом случае конфликт может быть разрешен путем обрезания концов сходств

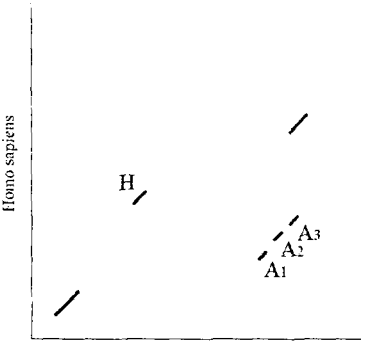

Рис. 4.1.2. Сравнение синтенных участков последовательностей мыши (AF139987) и человека (AF045555). Наиболее сильное сходство *H* – это ортологичное сходство, представляющее экзон 10 локусе Rfc2 (нуклеотиды 104514-104583 по последовательности мыши). Каждое из коллинеарных между собой сходств *A1, A2* и *A3* не является ортологичным. Однако их общий вес выше, чем вес сходства *H*

Цепь локальных сходств, построенную исходя из сформулированных выше принципов, будем называть *остовной* (backbone) цепью. Мы полагаем (см. аргументацию выше), что она близка к «эволюционно правильной» цепи, содержащей все ортологичные локальные сходства. В отличие от выравнивания белков (см. разделы 3.1, 3.2), при выравнивании геномов мы вынуждены полагаться

только на индивидуальную статистическую значимость локальных сходств. Описанный подход реализован в алгоритме *OWEN*, который описан в разделе 4.1.

Формальное определение остовной цепи основано на понятии *p*-достоверности ($p \in [0,1]$ – *порог значимости*).

Определение 4.1.1. *P-значением* сходства *H* между последовательностями *U* и *V* называется вероятность того, что независимые случайные последовательности длин $|U|$, $|V|$ имеют локальное сходства веса не менее *Score(H)*.

Определение 4.1.3. Пусть *F* – цепь локальных сходств в последовательностях *U*, *V* и *H* – элемент *F*. Сходство *H p-достоверно* в *F*, если его P-значение относительно фрагментов *U*, *V*, ограниченных ближайшими к *H* элементами цепи *F*, имеющими больший, чем *H*, вес, не превышает порога *p* (см. рис. 4.1.3).

Цепь локальных сходств *F* в последовательностях *U*, *V* называется *p-достоверной*, если каждое, входящее в *F* сходство *p*-достоверно в *F*.

Остовная цепь для *U*, *V* с уровнем значимости *p* – это максимальная *p*-достоверная цепь для *U*, *V* (уточнение понятия максимальности см. определение 4.1.6 в диссертации).

Построение *p*-остовной цепи для последовательностей *U*, *V* выполняется

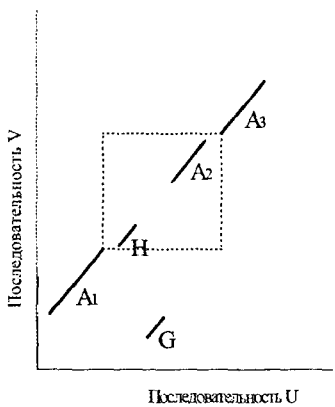

Рис. 4.1.3. *Иллюстрация понятия p-достоверности сходства относительно цепи. Пусть цепь* $F = \{A_1, H, A_2, A_3\}$ *и* $Score(H) < Score(A_2) < Score(A_3) < Score(A_1)$. *Тогда (1)* $F_H = \{A_1, A_2, A_3\}$, $F_{A_2} = \{A_1, A_3\}$, $F_{A_3} = \{A_1\}$, *и* F_{A_1} *пусто; (2)* A_2 *p-достоверно в* F, *если в выделенном прямоугольнике* $U[End(A_1,U)+1, Beg(A_3,U)-1] \times V[End(A_1,V)+1, Beg(A_3,V)-1]$, *ограниченном сходствами* A_1 *and* A_3; *(3)* A_1 *p-достоверно в F, если оно p-достоверно относительно последовательностей U and V в целом.*

иерархически. Сначала строятся все сходства, *p*-достоверные на уровне последовательностей *U*, *V* в целом и для этого множества сходств строится максимальная цепь Затем эта процедура рекурсивно применяется к промежуткам между соседними сходствами уже построенной подцепи искомой остовной цепи. Такой иерархический подход позволяет избежать затратного по времени анализа относительно слабых локальных сходств по последовательности в целом, что и определяет эффективность алгоритма.

Описанный алгоритм дважды использует парадигму жадных алгоритмов. Во-первых, описанная иерархическая (рекурсивная) процедура является жадной – «сильные» элементы остовной цепи выбираются без учета «слабых». Во-

вторых, построение остовной цепи для текущего множества сходств выполняется жадным алгоритмом. Это становится возможным ввиду приведенных выше принципов локального разрешения конфликтов между сходствами.

Время работы описанного алгоритма оценивается величиной $O(Z) + O(NlogN) + Time_{SetLoc}$, где Z – количество сходств в остовной цепи, N – общее количество локальных сходств, рассмотренных в ходе работы алгоритма, $Time_{SetLoc}$ – общее время построения локальных сходств. При обычно используемых малых значения параметра p ($p < 0.01$) значение $N = k\,T$, где k мало.

Построение локальных сходств при описанном подходе является функциональным параметром алгоритма геномного выравнивания. Таким образом, значение $Time_{SetLoc}$ определяется значением порога p и эффективностью алгоритма построения локальных сходств (см. ниже).

Выбор алгоритма построения локальных сходств и его параметров также определяет, считается ли некоторое сходство единым или распадается на цепочку более мелких (и, следовательно, менее значимых) сходств. Таким образом, мы не избавляемся полностью (что невозможно) от проблемы выбора параметров. Достоинство подхода в том, что анализ последовательностей «в малом» (внутри локальных сходств) отделен от анализа «в большом» (построение цепи локальных сходств). Параметры выравнивания, в частности, штрафы за удаления, используются только при поиске «в малом»; при этом есть возможность опираться на экспериментально подтвержденные выравнивания. При поиске «в большом» единственным параметром остается выбор порога для уровня значимости (сравни п. 2.4). Зависимость от этого параметра достаточно грубая, кроме того, в программной системе OWEN, реализующей описанный алгоритм, есть возможность «ручного» управления отбором локальных сходств.

Практически все используемые в настоящее время алгоритмы поиска локальных сходств основаны на фильтрации пространства поиска с помощью предварительного выделения т.н. затравочных сходств. Приведенные ниже определения обобщают определения из работ [Ma, B., Tromp, J, and Li, M. PatternHunter: Faster and More Sensitive Homology Search. // Bioinformatics. 2002. V. 18. P. 440-445], [Brown D. Optimizing multiple seeds for protein homology search. //IEEE Transactions on Computational Biology and Bioinformatics. 2005. Vol. 2 , P. 29 – 38.] и др.

Определение 4.2.1. *Затравкой* [seed] будем называть произвольное множество локальных выравниваний (=сходств), эти выравнивания называются *затравочными* выравниваниями. Затравка Z *допускает* выравнивание $G = \langle v, w, E\rangle$, если существует затравочное выравнивание $z_0 \in Z$ такое, что z_0 – подвыравнивание выравнивания G.

Простейший пример затравки – это множество точных совпадений заданной длины m. В этом случае затравка состоит из выравниваний вида $\langle u, u, S_0\rangle$, где u – слово длины m, S_0 – тривиальное выравнивание слова с самим собой.

Пусть π – затравка. Назовем слова w', w'' π-эквивалентными, если

$$\forall (v, w') \in \pi \Leftrightarrow (v, w'') \in \pi).$$

Очевидно, множество $C(\pi, v)$ всех слов w, образующих с данным словом v затравочное сходство в смысле затравки π , есть объединение классов π-

эквивалентности. На этом наблюдении основан стандартный способ поиска затравочных сходств для данных последовательностей v, w.. Он состоит из двух этапов.

1. Индексирование. Для каждого класса π-эквивалентными Z строится список всех вхождений элементов этого класса в последовательность v (т.н. индексный список).

2. Поиск. Просматриваем последовательность w слева направо. Чтобы найти все фрагменты последовательности v, образующие затравочное сходство с очередным фрагментом $w[x, x+d]$, достаточно просмотреть индексные списки для всех классов π -эквивалентности Z таких, что $Z \subseteq C(\pi, w[x, x+d)$.

Как правило, индексирование может быть выполнено за время, пропорциональное общей длине индексных списков, а поиск затравочных сходств – за время, пропорциональное их общему количеству. Таким образом, на построение каждого затравочного сходства тратится конечное время.

Предположим, что все (или почти все, см. ниже) *целевые* (т.е. представляющие интерес для исследователя) локальные сходства содержат затравочное подсходство, иными словами, допускаются затравкой. В этом случае возможна следующая двухэтапная схема поиска целевых локальных сходств.

Этап 1. Поиск затравок (см. выше).

Этап 2. Поиск *в окрестности затравок* локальных сходств, представляющих интерес с точки зрения поставленной биологической задачи.

Благодаря первому этапу, на котором эффективно осуществляется фильтрация пространства поиска, алгоритмы, построенные по такой схеме, работают достаточно быстро. Их время работы пропорционально общему количеству выделенных затравочных сходств. Платой за это является возможная потеря целевых локальных сходств, которые не содержат ни одного затравочного сходства.

Говоря неформально, затравка характеризуется двумя величинами – *чувствительностью*, т.е. долей значимых локальных сходств, в которых содержатся затравочные сходства, и *избирательностью*, т.е. вероятностью затравочного сходства в рамках выбранной вероятностной модели независимых случайных последовательностей.

Вычисление избирательности затравки определяется вероятностью того, что фрагменты независимых случайных последовательностей, которые начинаются в заданных позициях, образуют затравочное сходство. Например, для рассмотренной выше затравки «точное совпадение длины n» и бернуллиевских случайных последовательностей избирательность равна p^n. В то же время вычисление чувствительности затравки представляет определенную сложность. Вычисление чувствительности затравки означает вычисление вероятности (в рамках заданной *вероятностной модели)* того, что случайное выравнивание из заданного *множества целевых выравниваний* содержит подвыравнивание, принадлежащее затравке. Алгоритмы вычисления чувствительности затравок рассмотрены в разделе 5.2 главы 5.

Упомянутые выше точные совпадения были первым использовавшимся классом затравок, однако в последние годы был предложен ряд новых классов. Два таких класса рассмотрены в главе 4.

Первый из этих классов (см. раздел 4.2) – это разреженные затравки (spaced seeds) – первый класс затравок, отличный от «классических» идущих подряд n совпадений; этот класс был предложен в начале 2000-х годов (см. [Burkhardt S. and Karkkainen J. Better Filtering with Gapped q-Grams // Fundamenta Informaticae. 2003. Vol. 56, N. 1-2. P. 51-70], [Ma, B., Tromp, J, and Li, M. PatternHunter: Faster and More Sensitive Homology Search // Bioinformatics. 2002. V. 18. P. 440-445]. Неформально говоря, разреженная затравка, как и классическая затравка, требует наличия нескольких совпадений, но расположенных не подряд, а по описанной в затравке схеме. Более формально разреженные затравки описываются следующими определениями.

Определение 4.2.3. *Разреженным термом* E называется список положительных целых чисел $\{p_1, ...,p_d\}$ таких, что $p_1 < ... < p_d$ и $p_1=1$. *Размером* терма E называется величина $span(E) = p_d$; *весом* терма E называется число $weight(E) = d$. Отметим, что вес разреженной затравки напрямую связан с ее избирательностью.

Терм $E = \{p_1, ...,p_d\}$ часто для наглядности представляют словом t_E длины p_d в двухбуквенном алфавите $\{\#, -\}$ таким, что $t_E[i] = '\#' \Leftrightarrow i \in \{p_1, ...,p_d\}$. Символ '-' называют *джокером*.

Определение 4.2.5. Пусть G = безделеционное выравнивание слов v и w; $E = \{p_1, ...,p_d\}$ – разреженный терм. Выравнивание G *соответствует* терму E, если

1) $|v|=|w| = p_d$;

2) $\forall j \in \{1, ...d\}(v[k+ p_j+1] = w[k+ p_j+1])$.

Разреженная затравка, задаваемая термом E - это множество всех выравниваний, соответствующих терму E.

Т.к. разреженные затравки трактуют выравнивания только с точки зрения совпадений-несовпадений, то при работе с разреженными затравками выравнивания обычно перекодируются в двухбуквенный алфавит с буквами 1 (совпадение) и 0 (несовпадение).

В цитированной работе Ма и др. было показано, что при данной избирательности чувствительность разреженных затравок существенно выше, чем у классических «сплошных». При этом искомые разреженные затравки данной чувствительности (или, что то же самое, – данного веса) строились перебором. Мы впервые провели теоретическое исследование вида оптимальной затравки для простейшей ситуации – затравок с одним джокером при условии фильтрации без потерь,,т.е. с чувствительностью 1.

Определение 4.2.9. Выравнивание $g \in \{1, 0\}^m$ называется *(m, k)*-выравниванием, если в нем есть ровно k нулей и $m-k$ единиц. Затравка E *решает (m, k)-проблему*, если она допускает все *(m, k)*-выравнивания.

Определение 4.2.11. Пусть Q - затравка и k – допустимое количество несовпадений. *k-критическая длина* для Q – это минимальная длина m, при которой Q решает *(m, k)*-проблему. Затравка Q называется *k-оптимальной* в некотором классе затравок, если она имеет минимальную *k-критическую длину* в этом классе.

Утверждение 4.2.2. Рассмотрим класс затравок веса не менее d с одним джокером, k – натуральное число. Тогда *k-оптимальной* затравкой будет затравка $\#^{d-r}\text{-}\#^{r}$, где

$r = [d/2]$, если $k=1$,

$r = [d/3]$, если $k> 1$,

где $[x]$ – ближайшее целое к числу x и $[n+1/2] = n$.

Этот результат был обобщен на случай затравок с несколькими джокерами в [Farach-Colton, M., Landau, G.M., Sahinalp, S.C., Tsur, D: Optimal spaced seeds for faster approximate string matching // J. Comput. Syst. Sci. 2007. V. 73. P. 1035-1044].

Пример 4.2.1. Затравка $\#^4\text{-}\#^2$ – оптимальная в смысле приведенного выше определения среди затравок веса 6 с одним джокером. Это, в частности, означает, что она решает (m, 2) – проблему для всех $m \geq 16$ и это – наилучшая оценка в рассматриваемом классе. Аналогично, эта затравка решает (m, 3) – проблему для всех $m \geq 20$ и это – наилучшая оценка в рассматриваемом классе и т.д.

Естественным обобщением разреженных затравок являются классификационные затравки, представленные в разделе 4.3.

Пусть π – затравка. Слова равной длины w' и w'' называются π-эквивалентными, если для любого слова v той же длины безделиционные выравнивания (v, w') и (v, w'') либо одновременно являются затравочными сходствами для π (см. определение 4.2.1), либо одновременно не являются. Через $C(\pi, v)$ обозначим множество всех слов w, образующих с данным словом v затравочное сходство в смысле затравки π. Очевидно, множество $C(\pi, v)$ есть объединение классов π-эквивалентности.

Для произвольной разреженной затравки E каждый класс $C(E, v)$ состоит ровно из одного класса E-эквивалентности (говорят, что разреженные затравки *допускают однозначное индексирование*), что упрощает поиск соответствующих затравочных сходств. Возможность однозначного индексирования для разреженных затравок связана с тем, что эти затравки различают только два вида отношений между символами – совпадение и несовпадение. В [Brejova B., Brown, D., Vinar T. Vector seeds: an extension to spaced seeds allows substantial improvements in sensitivity and specificity // Proceedings of the 3rd International Workshop in Algorithms in Bioinformatics / Eds. Benson, G., Page, R. Lecture Notes in Computer Science. Vol. 812. Springer. 2003] была предложена более гибкая и общая модель затравок – векторные затравки, в которых критерий соответствия между затравкой и выравниванием базируется на интегральной характеристике - сумме вкладов разных позиций. Платой за эту общность является то, что для векторной затравки R класс $C(R, v)$ может состоять из десятков классов R–эквивалентности, что усложняет как поиск, так и индексирование.

Классификационные затравки занимают промежуточное положение между разреженными и векторными затравками. Они, с одной стороны, могут работать с алфавитом выравниваний произвольного размера (т.е. не обязательно различая только два вида сопоставлений «совпал» - «не совпал»), а, с другой стороны, сохраняют возможность однозначного индексирования. Приведем точные определения.

Фиксируем алфавит последовательностей Π. Пусть $M = \{(a, a) | a \in \Pi\}$ – множество всех пар-совпадений.

Определение. 4.2.12. *Классификационный* алфавит B – это алфавит, каждая буква $b \in B$ которого соответствует непустому подмножеству $\Pi_b \subseteq \Pi \times \Pi$, причем

1) $\forall b \in B\ (M \subseteq \Pi_b)$ (любая классификационная буква допускает совпадение);
2) В B есть буква #, для которой $\Pi_\# = M$ (# требует только совпадений, она ниже называется *единичной* буквой).

Очевидно, алфавит разреженных затравок {#, -} – классификационный, причем джокеру соответствует множество $\Pi \times \Pi$.

Определение. 4.3.1. *Классификационная* затравка – это слово в некотором классификационном алфавите B. Затравка $\pi = b_1 \ldots b_m \in B^m$ допускает фрагмент выравнивания $a_1 \ldots a_m \in (\Pi \times \Pi)^m$, если $\forall i \in \{1, \ldots, m\}\ a_i \in b_i$. Размером *span*($\pi$) затравки π называется ее длина m, #-весом – количество *sharp*(π) букв '#' среди $b_1, \ldots, b_m$

Определение. 4.3.2. Назовем буквы $x, y \in \Pi \times \Pi$ эквивалентными относительно алфавита B (B-эквивалентными), если $\forall b \in B\ (x \in \Pi_b \Leftrightarrow y \in \Pi_b)$.

Из определений 4.3.1, 4.3.2 следует, что при использовании классификационных затравок в алфавите B, исходный алфавит выравниваний $\Pi \times \Pi$ можно факторизовать по отношению B-эквивалентности. Полученный алфавит будем называть *алфавитом выравниваний, порожденным* классификационным алфавитом B. При работе с данным классификационным алфавитом B все выравнивания будут представляться, как слова в соответствующем алфавите выравниваний A.

Пример 4.3.1. Рассмотрим выравнивания нуклеотидных последовательностей, т.е. выравнивания над алфавитом последовательностей {*A, C, G, T*}. *Транзицией* называется одна из четырех пар {*(A, T), (T, A), (C, G), (G, C)*}. *Трансверсией* называется любое несовпадение, отличное от транзиции. Известно, что среди замен в нуклеотидных выравниваниях частота транзиций существенно выше, чем частота трансверсий. В качестве алфавита затравок возьмем алфавит B = { #, @, -}; элементы B представляют соответственно только совпадение (#), совпадение или транзицию (@), любое сопоставление (-). Очевидно, порожденный алфавитом B алфавит выравниваний A включает три множества: (1) совпадения (символ: *1*); (2) транзиции (символ: *h*); (3) трансверсии, т.е. несовпадения, отличные от транзиций (символ:*0*).

В отличие от веса разреженных затравок, #-вес классификационных затравок не определяет избирательность затравки. В отличие от разреженных затравок, определение веса квалификационных должно явно учитывать выбранную вероятностную модель.

Пусть B – алфавит классификационных затравок. Будем считать, что каждой букве $c \in B$ сопоставлена вероятность $\beta(c)$. Это можно сделать, например, так. Рассмотрим алфавит последовательностей Π, связанный с алфавитом B. Пусть на Π задано распределение вероятностей β: Π -> [0,1], где

$$\Sigma_{x \in \Pi} \beta(x) = 1.$$

Это распределение индуцирует распределение вероятностей на множестве пар $\Pi \times \Pi$ по правилу $\beta(x, y) = \beta(x)\,\beta(y)$. Так как буквы классификационного алфавита соответствуют подмножествам в $\Pi \times \Pi$, то, тем самым, зная распределение β, каждому классификационному символу c можно приписать вероятность $\beta(c)$. Отметим, что, вообще говоря,

$$\Sigma_{c \in B} \beta(c) \neq 1.$$

Определение 4.3.3. Пусть B – алфавит классификационных затравок и каждой букве $c \in B$ сопоставлена вероятность $\beta(c) \in [0, 1]$. Пусть # - единичный символ алфавита B, т.е. символ, соответствующий множеству совпадений $M = \{(x, x) | x \in \Pi\}$. *Весом* произвольного *символа* $c \in B$ называется величина

$$weight(c) = -\log(\beta(c))/\log(\beta(\#)).$$

Весом $weight(\pi)$ классификационной *затравки* π называется сумма весов входящих в нее букв.

Пример 4.3.2 (продолжение примера 4.3.1). Пусть на алфавите $\{A, C, G, T\}$ задано равномерное распределение вероятностей. Тогда $weight(\#) = 1$; $weight(@) = ½$; $weight(-) = 0$. Вес $weight(\pi)$ затравки $\pi = \#@-\#$ равен 2.5.

В п.4.3.4 диссертации показано (см. таблицы 4.3.3, 4.3.4), что при фиксированном весе, а следовательно, при фиксированной избирательности, классификационные затравки из рассмотренных выше примеров 4.3.1, 4.3.2 имеют более высокую чувствительность, чем разреженные затравки. В обеих таблицах множество целевых выравниваний – это множество всех безделеционных выравниваний длины 64. В таблице 4.3.1 распределение вероятностей на этом множестве – это бернуллиевское распределение, в таблице 4.3.4 – марковское распределение порядка 3. Параметры этих распределений получены из анализа парных геномных выравниваний для 40 бактериальных геномов. Ниже приведена таблица 4.3.3, данные таблицы 4.3.4 аналогичны.

Таблица 4.3.3.

Оптимальные затравки в классах разреженных затравок и классификационных затравок

Вес	Разреженные затравки	Чувствительность	Классификационные затравки, два @	Чувствительность
9	##-##-#-#-###	0.4183	###-#-#@#-@##	0.4443
10	##-##-##-#-###	0.2876	###-@#-@#-#-###	0.3077
11	###-#-#-###-###	0.1906	##@#-##-#-#-@###	0.2056
12	###-#-##-#-##-###	0.1375	##@#-#-##-#@-####	0.1481

Оптимальные затравки в классах разреженных затравок и классификационных затравок в алфавите {#, @, -} (см. пример 4.2.2) с не более, чем двумя символами @ для различных весов (от 9 до 12). Целевые выравнивания имеют длину 64. Распределение вероятностей – бернуллиевское, все символы равновероятны.

В п. 4.3.3 представлена конструкция т.н. классификационного автомата данной классификационной затравки π – автомата, который распознает множество всех безделеционных выравниваний, допустимых затравкой π. Как видно из результатов главы 5, количество состояний классификационного автомата определяет время работы алгоритма, вычисляющего чувствительность затравки π.

Утверждение 4.3.3. Пусть $\pi = b_1...b_m \in B^m$ - классификационная затравка длины m, w – количество символов # в π и $S_\pi = < A, Q, q_0, Q_F, \psi >$ - классификационный автомат затравки π. Тогда количество состояний автомата S_π не превосходит $(w+1)\bullet 2^{m-w}$.

Эта оценка не улучшаема, что показывает следующее

Утверждение 4.3.4. Пусть $A = \{\#, -\}$ и $\pi = \#-^r\#$ (между символами # расположено r джокеров). Тогда классификационный автомат S_π неприводим, т.е.

(i) любое состояние S_π достижимо из начального состояния q_0;

(ii) никакие два состояния не эквивалентны.

В силу определения 4.2.1, классификационный автомат затравки π эквивалентен автомату Ахо-Корасик для множества слов M_π, соответствующих затравке π. Для двухбуквенного алфавита выравниваний, соответствующего разреженным затравкам, оценка утверждения 4.3.3 совпадает с оценкой количества состояний автомата Ахо-Корасик из работы [Aho, A.V.; Corasick, M.J. Efficient String Matching. // Communications of the ACM. 1975. V. 18. P. 333–340.], которая дается формулой $(w+1)\bullet a^{m-w}$, где a – размер алфавита выравниваний. Таким образом, утверждение 4.3.3 дает верхнюю оценку для числа состояний приведенного автомата, соответствующего автомату Ахо-Корасик для множества слов M_π.

Таблица 4.3.1 представляет данные о среднем количестве состояний классификационного автомата соответственно для разреженных и классификационных затравок из примера 4.3.1. Для каждого веса w было подсчитано среднее количество состояний у автомата Ахо-Корасика (столбцы *«Ахо-Корасик»)* и классификационного автомата, описанного в разделе 4.2.4 (столбцы *«Классиф. автомат»).* В каждом случае показывалось как среднее количество состояний (столбцы «*Разм.»),* так и отношение этого среднего к среднему количеству состояний для соответствующего минимального автомата (столбцы «*Отн»).* Среднее количество состояний дли минимального автомата (это число одинаково для обеих конструкций, т.к. соответствующие автоматы эквивалентны) приведено в столбце *«Разм. мин. авт.».* Отметим, что классификационный автомат компактнее автомата Ахо-Корасик не только для 3-буквенного алфавита B (см. выше), но и для двухбуквенного алфавита R. При данном весе для регулярных затравок среднее подсчитано по всем затравкам веса w и длины не более $w+8$; для затравок примера 4.3.1 - по всем затравкам веса w, длины не более $w+5$ и содержащим не более двух символов @.

В главах 2-4 представлены алгоритмы построения парного выравнивания биологических последовательностей, ориентированные на широкий спектр постановок задач. Большинство из этих алгоритмов основано на методе динамического программирования. Как известно (см. [Finkelstein, A.V. and Roytberg, M.A. Computation of biopolymers: a general approach to different problems. // BioSystems. 1993. Vol. 30, P.1–19.]), динамическое программирование, наряду с построением оптимальных объектов, позволяет решать, в определенном смысле, двойственную задачу. Это задача вычисления т.н. обобщенных статистических сумм, которые характеризуют множество допустимых объектов в целом. Более формально, речь идет о следующей задаче.

Таблица 4.3.1.

Классификационные автоматы и автоматы Ахо-Корасик.

Вес	Разреженные затравки {#, -}					Классификационные затравки {#, @, -}				
	Ахо-Корасик		Классиф. автомат		Размер мин. авт.	Ахо-Корасик		Классиф. автомат		Размер мин. авт
	Разм.	Отн.	Разм.	Отн.		Разм.	Отн.	Разм.	Отн.	
9	345.9	3.1	146.3	1.3	113.2	1900.7	16.0	167.6	1.4	119.0
10	380.9	3.2	155.1	1.3	120.6	2104.0	16.5	177.9	1.4	127.5
11	415.4	3.3	163.8	1.3	127.6	2306.3	17.0	188.1	1.4	136.0
12	449.5	3.3	172.4	1.3	134.9	2507.9	17.4	198.1	1.4	144.0
13	483.3	3.4	180.9	1.3	141.8	2709.0	17.8	208.1	1.4	152.3

Дан конечный ациклический ориентированный граф G, для простоты будем считать, что у графа G один источник A и один сток Z. Полным путем в графе G называется путь из источника в сток. Ребра графа помечены элементами множества R, на котором определены две ассоциативные операции – «сложение» (обозначается $\oplus$) и «умножение» (обозначается $\otimes$), обладающие следующими свойствами:

1) нейтральные элементы 0 и 1:

$$\forall x \in M\ (0 \oplus x) = x; \qquad \forall x \in M\ (1 \otimes x) = x;$$

2) коммутативность сложения:

$$\forall x, y \in M\ (y \oplus x = x \oplus y);$$

3) дистрибутивность:

$$\forall x, y, z \in M\ (\ (x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)\ )\ ;$$

Весом пути называется «произведение» (относительно операции $\otimes$) весов входящих в него ребер, взятых в порядке следования от источника к стоку.

Обобщенной статистической суммой (ОСС) графа G называется «сумма» (относительно операции $\oplus$) весов всех его полных путей.

Если операции $\oplus$ и $\otimes$ - это обычные операции сложения и умножения, а веса ребер – это вероятности переходов, то ОСС графа G это суммарная вероятность полных путей из источника G этого графа.

Значение обобщенной статистической суммы может быть вычислено за время $O(Edge(G))$, где $Edge(G)$ – количество ребер в графе G.

Глава 5, заключительная глава диссертации, посвящена применению обобщенных статистических сумм в задачах биоинформатики. Наиболее распространенной интегральной характеристикой множества объектов является его вероятность. В разделе 5.1.1 приведены достаточные условия, при которых вычисление вероятности множества символьных последовательностей может быть сведено к

вычислению обобщенной статистической суммы. Этот подход применен к вычислению чувствительностей затравок (см. главу 4 и раздел 5.2) и вероятностей обнаружения сигналов в последовательностях (раздел 5.3).

Фиксируем алфавит A и длину последовательностей m. Пусть задано множество последовательностей $S \subseteq A^m$ и распределение вероятностей ρ на множестве A^m. В разделе 5.1 вычисление вероятности $P_\rho(S)$ сведено к вычислению обобщенной статистической суммы для некоторого графа. Этот граф строится на основе конечного детерминированного автомата (КДА), распознающего язык S и конечно-автоматного генератора вероятностей, задающего распределение ρ.

Множество $S \subseteq A^m$, вероятность которого требуется найти – конечное, и, следовательно, допускается конечным детерминированным автоматом K с не более, чем $m|S|$ состояниями. Специальные случаи построения такого автомата K, распознающего множество S рассматриваются ниже. «Конечно-автоматность» распределения ρ является достаточным условием применимости предлагаемого подхода, она эквивалентна существованию НММ, описывающей распределение ρ.

Говоря неформально, конечно-автоматный генератор вероятностей – это недетерминированный автомат без заключительных состояний, в котором для каждого перехода (q, a, q'), где q – состояние автомата, a – символ используемого алфавита, q' – новое состояние, задана соответствующая вероятность. Соответствие между конечно-автоматными генераторами вероятностей и НММ аналогично соответствию между автоматами Мили и Мура.

Определение 5.1.1. Пусть A – алфавит. Конечно-автоматный генератор вероятностей над алфавитом A – это четверка $G = \langle Q, q^0, A, \rho \rangle$, где Q – множество состояний; q^0- начальное состояние; A - алфавит, $\rho: Q \times A \times Q \to [0, 1]$ – функция генерации вероятностей такая, что

$$\forall q \in Q \; \Sigma_{q' \in Q, a \in A} \rho(q, a, q') = 1.$$

Переходом в G называется тройка $e = \langle q,a,q' \rangle$ такая, что $\rho(q, a, q') > 0$. Буква a называется меткой перехода и обозначается $label(e)$. Состояния q и q' называются соответственно начальным и конечным состоянием перехода и обозначаются соответственно $start(e)$, $end(e)$. *Меткой пути* $P = (e_1, \ldots, e_n)$ в G называется слово $label(e_1)\ldots label(e_n)$.

Генератор G называется детерминированным, если для любой пары $q \in Q$, $a \in A$ есть не более одного перехода вида $\langle q,a,q' \rangle$.

Путь P называется *инициальным*, если его начальное состояние – это состояние q^0 генератора G. *Вероятностью* $\rho(P)$ пути $P = (e_1, \ldots, e_n)$ называется произведение $\rho(P) = \Pi_{i=1,n} \rho(e_i)$. Вероятностью слова w относительно генератора G называется величина $P_G(w)$, равная сумме вероятностей всех инициальных путей с меткой w. Вероятностью конечного набора слов (языка) $L \subseteq A^*$ называется сумма $P_G(L)$ вероятностей всех слов $w \in L$. Очевидно, для любого n выполнено $P_G(A^n) = 1$.

Традиционно используемые способы задания распределений вероятностей могут быть переформулированы в терминах конечно-автоматных генераторов вероятностей. Бернуллиевское распределение описывается генератором с одним состоянием. В случае Марковских распределений порядка k вероятность очеред-

ного символа зависит от k предшествующих символов. Такое распределение может быть описано детерминированным генератором с не более, чем $|A|^k$ состояниями; эти состояния соответствуют словам длины менее k в алфавите A

Пусть даны КДА $K = <Q_K, q^0_K, Q^F_K, A, \varphi_K>$, - распознающий язык S и конечно-автоматный генератор $G= <Q_G, q^0_G, A, \rho_G >$, задающий распределение вероятностей ρ на A^m. Вычисление вероятности $P_G(L)$ сводится к вычислению ОСС для графа т.н. вероятностного распознающего автомата (ВР-автомата), который строится как *декартово произведение K и G*. Говоря неформально, ВР-автомат – *это недетерминированный конечно-автоматный генератор вероятностей, в котором выделено подмножество допускающих состояний.*

Определение 5.1.6. Пусть $K = <Q_K, q^0_K, Q^F_K, A, \varphi >$ - детерминированный конечный автомат без выхода; $G= <Q_G, q^0_G, A, \rho >$ - конечно-автоматный генератор вероятностей над алфавитом A. *ВР-автомат* $W = K \times G$ - это пятерка $W= =<Q_W, q^0_W, Q^F_W, A, , \rho_W>$, где $Q_W = Q_K \times Q_G$; $q^0_W = q^0_K \times q^0_G$;

$Q^F_W = \{<k, g> \in = Q_K \times Q_G \mid k \in Q^F_K\}$;

$\rho_W(<k, g>, a, <k', g'>) = \rho(g, a, g')$, если $\varphi(k, . a) = k'$;

$\rho_W(<k, g>, a, <k', g'>) = 0$ – в противном случае.

Все описанные ниже алгоритмы и оценки их сложности основываются на следующих утверждениях, которые следуют из определения 5.1.6.

1. Значение вероятности $P_G(L)$ равно значению обобщенной статистической суммы для графа ВР-автомата W.

2. Автомат W имеет $|Q_G| \cdot |Q_K|$ состояний и для каждой пары, состоящей из состояния q автомата W и символа $a \in A$ есть не более $|Q_G|$ исходящих из q ребер.

Для некоторых важных частных случаев оценки п.2 могут быть существенно улучшены, что приводит к различным оценкам сложности алгоритмов при различных допущениях. Некоторые из этих оценок приведены ниже.

Утверждение 5.1.4. Рассмотрим алфавит A и конечное множество $L \subseteq A^m$. Пусть $K = <Q_K, q^0_K, Q^F_K, A, \varphi_K>$ - КДА, распознающий множество L и $G= <Q_G, q^0_G, A, \rho_G >$ - конечно-автоматный генератор, задающий распределение вероятностей ρ на A^m. Тогда вероятность $P_G(L)$ множества L относительно распределения ρ_G может быть найдена за время $O(|Q_G|^2 \cdot |Q_K| \cdot |A|)$ с использованием памяти $O(|Q_G| \cdot |Q_K|)$. Если генератор G детерминированный, то для времени работы алгоритма верна оценка $O(|Q_G| \cdot |Q_K| \cdot |A|)$.

Следствие 1. Пусть распределение вероятностей на множестве A^m – бернуллиевское. Тогда для времени и памяти вычисления вероятности $P_G(L)$ верны оценки $Time_{Bern} \le O(|Q_K| \cdot |A|)$ и $Space_{Bern} \le O(|Q_K|)$

Следствие 2. Пусть распределение вероятностей на множестве A^m – это Марковское распределение порядка r. Тогда для времени и памяти вычисления вероятности $P_G(L)$ верны оценки $Time_{Markov} \le O(|Q_K| \cdot |A^{r+1}|)$ и $Space_{Markov} \le O(|Q_K| \cdot |A^r|)$

Утверждение 5.1.6. Рассмотрим алфавит A и конечное множество $L \subseteq A^m$. Пусть $B = <Q_B, q^0_B, Q^F_B, A, \varphi_B>$ - автомат, распознающий множество L' такое, что $L' \cap A^m = L$ и $G= <Q_G, q^0_G, A, \rho_G >$ - конечно-автоматный генератор, задающий распределение вероятностей ρ на A^m. Тогда вероятность $P_\rho(L)$ множества L

относительно распределения ρ может быть найдена за время $O((|Q_B|\cdot|Q_G|^2\cdot|A|\cdot m)$ с использованием памяти $O(|Q_B|\cdot|Q_G|)$. В случае детерминированного генератора G время работы равно $O((|Q_B|\cdot|Q_G|\cdot|A|\cdot m)$.

Следствие 1. Пусть распределение вероятностей на множестве A^m – Бернуллиевское. Тогда для времени и памяти вычисления вероятности $P_G(L)$ верны оценки $Time_{Bern} \le O(|Q_B|\cdot|A|\cdot m)$ и $Space_{Bern} \le O(|Q_B|)$

Следствие 2. Пусть распределение вероятностей на множестве A^m – это Марковское распределение порядка r. Тогда для времени и памяти вычисления вероятности $P_G(L)$ верны оценки $Time_{Markov} \le O(|Q_B|\cdot|A^{r+1}|\cdot m)$ и $Space_{Markov} \le O(|Q_B|\cdot|A^r|)$

Утверждение 5.1.8. Рассмотрим алфавит A и конечное множество $L \subseteq A^m$. Пусть M - конечное множество слов, множество L' – это множество

$$L' = \{w \in A^* | \ w \text{ содержит подслово из } M\}$$

и при этом выполнено $L = A^m \cap L'$.

Пусть далее, $B = <Q_B, q^0_B, Q^F_B, A, \varphi_B>$ - автомат Ахо-Корасик для множества M, распознающий множество L' и ρ – распределение вероятностей на A^m, задаваемое марковской моделью порядка r. Тогда вероятность $P_\rho(L)$ множества L относительно распределения ρ может быть найдена за время $O((|Q_B|+|A^r|)\cdot|A|\cdot m)$ с использованием памяти $O((|Q_B|+|A^r|))$.

В разделе 5.2 результаты предыдущего раздела применены для вычисления чувствительности данной затравки, т.е. вероятности (в рамках заданной вероятностной модели) того, что случайное выравнивание из заданного множества целевых выравниваний допускается затравкой. Все выравнивания в этом разделе считаются безделеционными и представляются словами в некотором алфавите A. В качестве множества целевых выравниваний рассматривается множество A^m всех выравниваний длины m. На A^m задано распределение вероятности ρ, отражающее вероятности замен в интересующем исследователя классе выравниваний. Как и в разделе 5.1, будем считать, что распределение вероятности ρ задано конечно-автоматным генератором вероятностей $G= <Q_G, q^0_G, A, \rho >$. Фиксируем затравку π.

Определение 5.2.1 Через $S(\pi) \subseteq A^*$ обозначается множество всех выравниваний, допускаемых затравкой π. *Автоматом затравки* π называется детерминированный конечный автомат без выхода, распознающий множество $S(\pi)$. Через $S(\pi, m) \subseteq A^m$ обозначается множество всех выравниваний, допускаемых затравкой π

Будем считать, что вместе с затравкой π задан и ее автомат $B(\pi) = <Q_B, q^0_B, Q^F_B, A, \varphi_B>$. Такой автомат для всех известных классов затравок может быть построен на основе автомата Ахо-Корасик, специальная конструкция автомата для классификационных затравок представлена в п.4.3.4. Из результатов раздела 5.1 вытекают следующие утверждения.

Утверждение 5.2.1.Пусть A – алфавит выравниваний, $G= <Q_G, q^0_G, A, \rho_G >$ - конечно-автоматный генератор, задающий распределение вероятностей ρ на A^m; π – затравка и $B(\pi) = <Q_B, q^0_B, Q^F_B, A, \varphi_B>$ - автомат затравки π. Тогда чувствительность $Sens(\pi, m)$ затравки π относительно множества выравниваний A^m может быть найдена методом динамического программирования за время

$O(|Q_G|^2 \cdot |Q_B| \cdot m \cdot |A|)$ с использованием памяти $O(|Q_G| \cdot |Q_B|)$. Если автомат $B(\pi)$ – детерминированный, то время работы равно $O(|Q_G| \cdot |Q_B| \cdot m \cdot |A|)$.

Следствие 1. Пусть распределение вероятностей на множестве A^m – бернуллиевское. Тогда для времени и памяти вычисления чувствительности $Sens(\pi, m)$ верны оценки $Time_{Bern} \leq O(|Q_B| \cdot m \cdot |A|)$ и $Space_{Bern} \leq O(|Q_B| \cdot m)$

Следствие 2. Пусть распределение вероятностей на множестве A^m – это Марковское распределение вероятностей ρ порядка r. Тогда для времени и памяти вычисления вероятности $P_G(L)$ верны оценки $Time_{Markov} \leq O(|Q_B| \cdot m \cdot |A^{r+1}|)$ и $Spac_{Markov} \leq O(|Q_B| \cdot m \cdot |A^r|)$

Утверждение 5.2.3.Пусть π – затравка в алфавите последовательностей Π, A – алфавит выравниваний, согласованный с затравкой π и автомат $B(\pi) = <Q_B, q_B^0, Q^F{}_B, A, \varphi_B>$ является автоматом Ахо-Корасик. Пусть далее на множестве A^m задано Марковское распределение вероятностей ρ порядка r. Тогда чувствительность $Sens(\pi, m, \rho)$ затравки π относительно множества выравниваний A^m и распределения ρ может быть найдена методом динамического программирования за время $O((|Q_B|+|A^r|) \cdot |A| \cdot m)$ с использованием памяти $O((|Q_B|+|A^r|))$.

В разделе 5.3 приведен еще один пример приложения техники, описанной в разделе 5.1, - вычисление значимости найденного кластера регуляторных сайтов. Факторы регуляции транскрипции (ФРТ) связываются с фрагментами ДНК, содержащими специфичную для данного фактора последовательность нуклеотидов. Как правило, все последовательности, с которыми может связываться данный ФРТ имеют одну и туже длину. Множество таких последовательностей называется мотивом связывания ФРТ. В геноме область связывания ФРТ расположена перед регулируемым геном и называется цис-регуляторным модулем (ЦРМ). В последние годы было показано, что регуляция транскрипции с помощью ФРТ носит кумулятивный характер: в регуляции может участвовать несколько ФРТ, причем для каждого из них может быть несколько потенциальных мест связывания (вхождений соответствующих мотивов). Существующие программы распознавания ЦРМ основаны на поиске фрагментов генома, в котором перепредствлены соответствующие мотивы. Для практического использования этих программ необходимы средства оценки значимости полученных результатов. Формально задача может быть поставлена следующим образом.

Фиксируем алфавит ДНК $Д=\{a, c, g, t\}$. Как и в разделе 5.2, будем считать, что на A^m (здесь m – длина предполагаемого ЦРМ) есть распределение вероятностей ρ, которое задается конечно-автоматным генератором $G= <Q_G, q^0{}_G, A, \rho_G>$. В частности, это распределение может быть Бернуллиевским или Марковским некоторого порядка $r > 0$.

Определение 5.3.1. *Мотив* длины t – это множество слов $M \subseteq Д^t$.

Определение 5.3.2. Пусть M - мотив длины t, $D \in Д^m$ – последовательность ДНК длины m; $m \geq t$. *Вхождение* мотива M в последовательность D – это фрагмент $D[x, x+t-1]$ такой, что $D[x, x+t-1] \in M$.

Пусть $D[x_1, x_1+t_1-1]$ – вхождение мотива M_1 и $D[x_2, x_2+t_2-1]$ – вхождение мотива M_2. Эти вхождения *пересекаются*, если пересекаются отрезки $[x_1, x_1+t_1-1]$ и $D[x_2, x_2+t_2-1]$.

Определение 5.3.2. Пусть дан набор $\boldsymbol{M} = < M_1, ..., M_s >$, состоящий из s мотивов M_i длины t_i $(i=1, ..., s)$ и вектор $\boldsymbol{k} = < k_1, ..., k_s >$, состоящий из s нату-

ральных чисел k_i $(i=1, ..., s)$. Пусть $D \in Д^m$ – последовательность ДНК длины m. Набор мотивов $\boldsymbol{M}$ имеет $\boldsymbol{k}$ *–вхождение* в последовательность D, если для каждого $i \in \{1, ..., s\}$ в D есть не менее k_i (возможно, пересекающихся) вхождений мотива M_i.

Введем следующие обозначения:

$F(\boldsymbol{M}, \boldsymbol{k})$ - множество всех последовательностей из $Д^*$, для которых существует $\boldsymbol{k}$ –вхождение набора $\boldsymbol{M}$;

$F(\boldsymbol{M}, \boldsymbol{k}, m)$ - множество всех последовательностей из $Д^m$, для которых существует $\boldsymbol{k}$ –вхождение набора $\boldsymbol{M}$;

В качестве меры достоверности потенциального ЦРМ длины m, содержащего k_i вхождений мотива M_i $(i=1, ..., s)$ при заданном распределении вероятностей ρ на $Д^m$ используется величина $P_\rho(F(\boldsymbol{M}, \boldsymbol{k}, m))$, где $\boldsymbol{M} = < M_1, ..., M_s >$ и $\boldsymbol{k} = < k_1, ..., k_s >$. С помощью техники, описанной в п. 5.1, вычисление этой вероятности можно свести к вычислению обобщенной статистической суммы и, следовательно, найти искомую вероятность алгоритмом, основанным на методе динамического программирования.

Приведенные ниже утверждения дают оценки сложности этого алгоритма для различных вариантов задания распределения вероятностей ρ. Во всех этих оценках в качестве параметра фигурирует количество $|Q_C|$ состояний автомата Ахо-Корасика C для множества $M_0 = M_1 \cup ... \cup M_s$. В цитированной выше работе Ахо и Корасик показано, что

$$|Q_C| \le L(\boldsymbol{M}), \qquad (5.3.1)$$

где $L(\boldsymbol{M})$ – суммарная длина всех слов из множества M_0. В приведенных ниже утверждениях фигурирует величина $|Q_C|$, а не более естественная величина поскольку, как показано в разделе 4.3, оценка (5.3.1) недостаточно точна.

Утверждение 5.3.3. Пусть $\boldsymbol{M} = < M_1, ..., M_s >$ - набор мотивов в алфавите A; $\boldsymbol{k} == < k_1, ..., k_s >$ - вектор с натуральными компонентами, $C = <Q_C, q^0_C, Q^F_C, A, \varphi_C>$ - автомат Ахо-Корасик для множества $M_0 = M_1 \cup ... \cup M_s$ и $G = <Q_G, q^0_G, A, \rho_G >$ - конечно-автоматный генератор, задающий распределение вероятностей ρ на A^m.

Тогда вероятность $P_\rho(F(\boldsymbol{M}, \boldsymbol{k}, m))$ множества $F(\boldsymbol{M}, \boldsymbol{k}, m)$ относительно распределения ρ может быть найдена методом динамического программирования за время $O(|Q_C| \cdot k_1 \cdot ... \cdot k_s \cdot m \cdot |Q_G|^2 \cdot |A|)$ с использованием памяти $O(|Q_C| \cdot k_1 \cdot ... \cdot k_s \cdot |Q_G|)$. Если генератор G – детерминированный, то оценка времени может быть понижена до $O(|Q_C| \cdot k_1 \cdot ... \cdot k_s \cdot m \cdot |Q_G| \cdot |A|)$

Следствие. Пусть распределение вероятностей на множестве A^m – Бернуллиевское. Тогда для времени и памяти вычисления вероятности $P_\rho(F(\boldsymbol{M}, \boldsymbol{k}, m))$ верны оценки $Time_{Bern} \le O(|Q_C| \cdot k_1 \cdot ... \cdot k_s \cdot m \cdot |A|)$ и $Space_{Bern} \le O(|Q_C| \cdot k_1 \cdot ... \cdot k_s)$.

Утверждение 5.3.5. Пусть $\boldsymbol{M} = < M_1, ..., M_s >$ - набор мотивов в алфавите A; $\boldsymbol{k} == < k_1, ..., k_s >$ - вектор с натуральными компонентами, $C = <Q_C, q^0_C, Q^F_C, A, \varphi_C>$ - автомат Ахо-Корасик для множества $M_0 = M_1 \cup ... \cup M_s$. и ρ – распределение вероятностей на A^m, задаваемое марковской моделью порядка r.

Тогда вероятность $P_\rho(F(\boldsymbol{M}, \boldsymbol{k}, m))$ множества $F(\boldsymbol{M}, \boldsymbol{k}, m)$ относительно распределения ρ может быть найдена методом динамического программирования за время $O((|Q_C| \cdot k_1 \cdot ... \cdot k_s + |A|^r) \cdot m \cdot |A|)$ с использованием памяти $O(|Q_C| \cdot k_1 \cdot ... \cdot k_s + |A|^r)$

Основные результаты

1. На основе конечно-автоматного представления семейств последовательностей и конечно-автоматного описания вероятностных моделей предложен единый подход к вычислению вероятностей сигналов в случайных последовательностях и их выравниваниях. Разработаны методы построения затравок для поиска локальных сходств в нуклеотидных и аминокислотных последовательностях.

2. Предложен алгоритм оптимального парного выравнивания символьных последовательностей при штрафах за делеции, задаваемых кусочно-линейными функциями. Время работы этого алгоритма пропорционально произведению длин последовательностей. Предложен алгоритм построения множества Парето-оптимальных выравниваний символьных последовательностей. Это множество может быть построено за время, не превосходящее $O(m^2 nlog(n))$.

3. Точность алгоритмического выравнивания аминокислотных последовательностей белков относительно выравнивания этих последовательностей, основанных на пространственной структуре белков, лимитируется наличием в структурных выравниваниях участков, имеющих отрицательный вес. Это ограничение может быть преодолено за счет учета сведений о вторичной структуре сравниваемых белков. Соответствующий алгоритм имеет время работы, пропорциональное произведению длин последовательностей.

4. Предложен алгоритм предсказания внутренних петель при построении оптимальной вторичной структуры РНК с временной сложностью $O(P^*\log^2 n)$, где P – количество потенциальных спариваний нуклеотидов, n – длина последовательности РНК. Предложен алгоритм построения оптимального выравнивания последовательностей РНК с заданной вторичной структурой относительно аффинных штрафов за делеции с временной сложностью $O(m^2 nlog^2(n))$, где m, n – длины сравниваемых последовательностей.

5. Выравнивание геномов может быть проведено, не используя оптимизацию выравнивания в целом относительно какой-либо весовой функции и учитывая только оценки значимости отдельных локальных сходств. Предложен эффективный иерархический алгоритм построения геномного выравнивания, основанный на указанном подходе.

Основные результаты диссертации опубликованы в работах:

1. Roytberg M.A. A Search for Common Patterns in many Sequences // Comput. Appl. Biosci. 1992. Vol. 8, N 1. P. 57 – 64.
2. Roytberg M.A. Fast algorithm for optimal aligning of symbol sequences // In: Mathematical methods of the analysis of biopolymer sequences. AMS, Providence. 1992. P. 103-117.
3. Beridze T., Tsirekidze N., Roytberg M.A. On the tertiary structure of satellite DNA // Biochimie. 1992. Vol.74, N.1. P. 187-194.
4. Roytberg M.A. Similarity Search in Biological Sequences. // Modelling and Computer Methods in Molecular Biology and Genetics / Eds. V.A.Ratner and N.A. Kolchanov. NY: Nova Science Publishers, Inc. 1992. P. 81-86.

5. Gelfand M.S., Roytberg M.A. Prediction of the exon-intron structure by a dynamic programming approach // Biotechnology Software. 1992. P. 13-18.
6. Finkelstein, A.V. and Roytberg, M.A. Computation of biopolymers: a general approach to different problems // BioSystems. 1993. Vol.30. P.1–19.
7. Gelfand M.S., Podolsky L.I., Astakhova T.V. and Roytberg M.A. Prediction of the exon-intron structure and multicriterial optimization. //Bioinformatics and Genome Research / Eds. H.A.Lim, C.R.Cantor. Singapore: World Scientific Publ. Co. 1995. P. 173-183.
8. Gelfand M.S., Roytberg M.A. A dynamic programming algorithm for prediction of the exon-intron structure // BioSystems. 1993. Vol.30, P.173-182.
9. Nazipova N.N., Shabalina S.A., Ogurtsov A.Yu., Kondrashov A.S., Roytberg M.A., Buryakov G.V., Vernoslov S.E. SAMSON: a software package for the biopolymer primary structure analyses // Comput. Appl. Biosci. 1995. Vol. 11. P. 423-426.
10. Gelfand M.S., Podolsky L.I., Astakhova T.V., Roytberg M.A. Recognition of genes in human DNA sequences // Journal of Computational Biology. 1996. Vol. 3, N. 2. P. 223-234.
11. Ройтберг М.А., Астахова Т.В., Гельфанд М.С. Алгоритм высокоспецифичного распознавания белок-кодирующих областей в последовательностях высших эукариот // Молекулярная биология. 1997. Т. 31, № 1. С. 25-31.
12. M.A.Roytberg, T.V.Astahova, M.S.Gelfand. Combinatorial approaches to gene recognition // Computers and Chemistry. 1998). Vol. 1, P. 229-236.
13. Sze S.H, Roytberg M.A., Gelfand M.S., Mironov A.A., Astakhova T.V., Pevzner P.A. Algorithms and software for support of gene identification experiments // Bioinformatics. 1998. Vol.1, N. 14. P. 14-19.
14. Mironov A.A., Roytberg M.A. Pevzner P.A., Gelfand M.S. Performance guarantee gene predictions via spliced alignment //Genomics. 1998. Vol. 51, P. 332-339.
15. Ройтберг М.А., Симеоненков М.Н., Таболина О.Ю. Парето-оптимальные выравнивания символьных последовательностей //Биофизика. 1998. Т. 44, №4. С. 581-594.
16. Mironov AA, Koonin EV, Roytberg MA, Gelfand MS. Computer analysis of transcription regulatory patterns in completely sequenced bacterial genomes //Nucleic Acids Res. 1999. Vol.15, N.27. P. 2981-2989.
17. Ramensky V.E., Makeev V.Ju., Roytberg M.A., Tumanyan V.G. DNA segmentation through the Bayesian approach //J Comput Biol. 2000. Vol.7, N.1-2. P. 215-231.
18. Ramensky V.E., Makeev V.Y., Roytberg M.A., Tumanyan V.G. Segmentation of long genomic sequences into domains with homogeneous composition with BASIO software // Bioinformatics. 2001. Vol.17, N.11. P. 1065-1066.
19. Kister A.E, Roytberg M.A, Chothia C., Gelfand I.M. The sequence determinants of cadherin molecules // Protein Science. 2001. Vol. 10. P. 1801-1810.
20. Roytberg, M.A., Ogurtsov A.Y., Shabalina S.A., Kondrashov A.S. A hierarchical approach to aligning collinear regions of genomes // Bioinformatics. 2002. Vol. 18. P.1673–1680.

21. Ogurtsov A.Y., Roytberg M.A., Shabalina S.A. and Kondrashov A.S. OWEN: aligning long collinear regions of genomes // Bioinformatics. 2002. Vol. 18. P. 1703-1704.
22. Sunyaev S.R., Bogopolsky G.A., Oleynikova N.V., Vlasov P.K., Finkelstein A.V., Roytberg M.A. 2004. From analysis of protein structural alignments toward a novel approach to align protein sequences // Proteins. 2004. Vol. 54, P.569–582.
23. М.А.Ройтберг. Сравнительный анализ первичных структур нуклеиновых кислот и белков // Молекулярная биология. 2004. Т. 38, № 1. С. 92-103.
24. Kucherov, G., Noé, L. Roytberg, M. Multi-seed lossless filtration. // IEEE/ACM Transactions on Computational Biology and Bioinformatics. 2005. Vol. 2, N. 1, P. 51-61.
25. Kucherov G., No´e L., and Roytberg M. A unifying framework for seed sensitivity and its application to subset seeds //Journal of Bioinformatics and Computational Biology. 2006. Vol. 4, N. 2. P. 553–570
26. Astakhova T.V., Petrova S.V. , Tsitovich I.I., Roytberg M.A. Recognition of coding regions in genome alignment // Bioinformatics of Genome Regulation and Structure II. / Eds. N.Kolchanov and R. Hofestaedt. NY: Springer Science+Business Media. 2006. P. 3-10.
27. Ogurtsov, A.Yu, Shabalina, S.A., Kondrashov, A.S., Roytberg M.A. Analysis of internal loops within the RNA secondary structure in almost quadratic time // Bioinformatics. 2006, Vol. 22, No. 11, P. 1317-1324
28. Литвинов И.И., Лобанов М.Ю., Миронов А.А., Финкельштейн А.В., Ройтберг М.А. Информация о вторичной структуре белка улучшает качество выравнивания // Молекулярная биология. 2006. Т. 40, № 3. С. 533-540.
29. Backofen R, Chen S, Hermelin D, Landau GM, Roytberg MA, Weimann O, Zhang K. Locality and gaps in RNA comparison // J Comput Biol. 2007. Vol. 14. N 8. P.1074-1087.
30. Asthana S., Roytberg M., Stamatoyannopoulos J., Sunyaev S. Analysis of sequence conservation at nucleotide resolution // PLoS Comput Biol. 2007. V. 3, N 12. P. 254.
31. Boeva V, Clement J, Regnier M, Roytberg MA, Makeev VJ. Exact p-value calculation for heterotypic clusters of regulatory motifs and its application in computational annotation of cis-regulatory modules // Algorithms Mol Biol. 2007. Vol. 2. P.13-27
32. Polyanovsky V.O., Roytberg M.A., Tumanyan V.G. Reconstruction of genuine pair-wise sequence alignment // J. Comput. Biol. 2008. V. 15. N 4. P. 379-391.
33. Поляновский, В.О., Ройтберг, М.А., Туманян, В.Г. Новый подход к оценке достоверности выявления вставок-делеций в парном выравнивании // Биофизика. 2008. Т. 53, №4. С.533-537.
34. Корзинов О.М., Астахова Т.В., Власов П.К., Ройтберг М.А Статистический анализ участков ДНК в окрестности сайтов сплайсинга // Молекулярная биология. 2008. Т. 42. № 1: С. 150-162
35. Roytberg, M., Gambin, A., Noé, L., Lasota, S., Furletova, E., Szczurek, E., Kucherov, G. On subset seeds for protein alignment // IEEE/ACM Transactions on Computational Biology and Bioinformatics. 2009. V. 6. N 3. P. 483-494.

36. Regnier, M., Kirakosyan, Z., Furletova E. and Roytberg, M. A Word Counting Graph // London Algorithmics 2008: Theory and Practice. / Eds. Chan, J., Daykin, J.W. and Rahman M.S. London: College Publications. 2009. P. 10-43.
37. Ройтберг М.А. Вычисление вероятностей семейств биологических последовательностей // Биофизика. 2009. Т.54. № 5. С. 718-724

Подписано в печать 7 сентября 2009 г.
Формат 60x90/16
Объём 2,7 п.л.
Тираж 100 экз.
Заказ № 211009251

Оттиражировано на ризографе в ООО «УниверПринт»
ИНН/КПП 7728572912\772801001
Адрес: 119333, г. Москва, Университетский проспект, д. 6, кор. 3.
Тел. 740-76-47, 989-15-83.
http://www.univerprint.ru