МОЛЕКУЛЯРНАЯ БИОЛОГИЯ, 2007, том 41, № 4, с. 711–718

МАТЕМАТИЧЕСКАЯ И СИСТЕМНАЯ БИОЛОГИЯ

УДК 577.1

МЕТОД ПОИСКА КОНСЕРВАТИВНЫХ СТРУКТУР РНК

© 2007 г. А. А. Миронов[1, 2]*

[1]Факультет биоинформатики и биоинженерии Московского государственного университета им. М.В. Ломоносова, Москва, 119992

[2]Институт проблем передачи информации им. А.А. Харкевича Российской академии наук, Москва, 127994

Поступила в редакцию 06.07.2006 г.
Принята к печати 31.10.2006 г.

Предсказание вторичной структуры РНК – одна из классических проблем биоинформатики. Наиболее эффективные подходы к этой проблеме основаны на идее сравнительного анализа. Обычно используется множественное выравнивание исследуемых последовательностей и в нем ищется общая вторичная структура. В статье описан новый алгоритм решения этой задачи. Алгоритм не требует заранее определенного множественного выравнивания. Основная идея алгоритма основана на MEME-подобной итеративной процедуре, применяемой к обобщенному профилю на разных уровнях. На первом уровне ищутся общие блоки в последовательностях РНК. На следующем этапе алгоритм уточняет цепочки из общих блоков. На последнем этапе алгоритм ищет множества общих спиралей, согласованных с блоками выравнивания. Алгоритм тестировался на множестве тРНК, содержащем дополнительные мусорные последовательности, и на последовательностях рибопереключателей RFN. Алгоритм реализован в виде веб-сервера (http://bioinf.fbb.msu.ru/RNAAlign/).

Ключевые слова: вторичная структура РНК, множественное выравнивание, предсказание.

A METHOD FOR PREDICTION OF CONSERVED RNA SECONDARY STRUCTURES, by *A. A. Mironov[1, 2]** ([1]Department of Bioengineering and Bioinformatics, Moscow State University, Moscow, 119992 Russia, *e-mail: mironov@bioinf.fbb.msu.ru; [2]Kharkevich Institute for Information Transmission Problems, Russian Academy of Sciences, Moscow, 127994 Russia). The RNA secondary structure prediction is a classical problem in bioinformatics. The most efficient approach to this problem is based on the idea of a comparative analysis. In this approach the algorithms utilize multiple alignment of the RNA sequences and find common RNA structure. This paper describes a new algorithm for this task. This algorithm does not require predefined multiple alignment. The main idea of the algorithm is based on MEME-like iterative searching of abstract profile on different levels. On the first level the algorithm searches the common blocks in the RNA sequences and creates chain of this blocks. On the next step the algorithm refines the chain of common blocks. On the last stage the algorithm searches sets of common helices that have consistent locations relative to common blocks. The algorithm was tested on sets of tRNA with a subset of junk sequences and on RFN riboswitches. The algorithm is implemented as a web server (http://bioinf.fbb.msu.ru/RNAAlign/).

Key words: RNA secondary structure, multiple alignment, prediction.

Роль вторичной структуры РНК в молекулярной биологии трудно переоценить. Вторичная структура РНК определяется ее последовательностью и, в свою очередь, определяет ее пространственную структуру. Она играет существенную, а во многих случаях определяющую роль для ее функционирования. Классическим примером роли вторичной структуры РНК являются тРНК [1, 2] и рРНК [3]. Она важна для функционирования рибозимов [4], упаковки вирусов, сборки различных рибонуклеиновых комплексов [5], формирования различных сигналов, например, сигналов внутренней инициации трансляции [6]. В последнее время все яснее становится роль структуры РНК для разного рода регуляторных систем. Здесь следует упомянуть регуляцию на уровне микроРНК [7], рибопереключатели [8, 9], которые в последнее время привлекают все большее внимание исследователей.

Задача предсказания структур РНК по последовательности – одна из классических задач биоинформатики. При разработке различных методов предсказания в биоинформатике необходимо иметь "золотой стандарт" – множество объектов, для которых из эксперимента известно правильное решение. В случае предсказания вторичной структуры РНК в качестве "золотого стандарта" часто принято использовать хорошо известную вторичную структуру тРНК.

* Эл. почта: mironov@bioinf.fbb.msu.ru

Для решения задачи предсказания вторичных структур РНК применяли различные подходы. Наибольшую популярность приобрели методы, основанные на минимизации энергии [10, 11]. В основе этих подходов лежит идея о том, что правильная вторичная структура РНК должна быть термодинамически наиболее стабильной и, следовательно, обладать наименьшей свободной энергией. При решении задачи минимизации энергии необходимы правила подсчета энергии для любой структуры и эффективный алгоритм минимизации энергии. Сделано много попыток построения правил подсчета свободной энергии на основе множества экспериментальных данных [12], созданы соответствующие эффективные алгоритмы, основанные на динамическом программировании. Хотя эти методы часто используются, однако их применение к "золотому стандарту" показывает не очень хорошие результаты [13]. У этого может быть несколько причин. Во-первых, оптимальная структура может быть отделена от доступного пространства высоким энергетическим барьером, на преодоление которого надо потратить время, превышающее время жизни молекулы, и поэтому биологически активная структура не обязательно оптимальна. Во-вторых, при расчете энергий структур учитывается только часть взаимодействий, а значительная часть энергетических вкладов игнорируется – например, третичные взаимодействия, взаимодействия с другими молекулами и т.п. Есть ряд других алгоритмов поиска оптимальной структуры, использующих методы стохастической оптимизации (например, генетические алгоритмы [14, 15]), позволяющие, в частности, учитывать псевдоузлы.

Другой подход основан на анализе кинетики сворачивания РНК в процессе ее синтеза [16–18]. При этом, в отличие от методов минимизации, ищутся не наиболее стабильные структуры, а структуры, кинетически доступные для сворачивания, поскольку можно предположить, что оптимальная структура может быть отделена от кинетически доступной очень высоким энергетическим барьером. Для этих подходов пока не проводилось массового анализа, однако, несмотря на физическую ясность подхода, этот метод содержит в себе довольно много неучтенных факторов. Поэтому можно предположить, что он даст несколько лучшие результаты, но до полного решения проблемы еще далеко.

Наконец, третий подход можно назвать "биологическим". Он основан на идее, что биологически важные вторичные структуры должны сохраняться в процессе эволюции. При таком подходе анализируется не одна последовательность, а множество последовательностей, выполняющих одну биологическую роль. Этот подход позволил предсказать структуры множества РНК, включая тРНК, рРНК, рибопереключатели, и множество других. В основе этого подхода лежит анализ множественного выравнивания последовательностей и обнаружение в этом выравнивании компенсаторных замен, сохраняющих уотсон-криковское спаривание. Наиболее популярным алгоритмом, реализующим этот метод, служит анализ ковариаций в разных позициях [19], одна из реализаций такого метода представлена на сервере http://www.genebee.msu.su/services/rna2_reduced.html. Применение этого метода вызывает ряд проблем. Это, прежде всего, построение правильного выравнивания последовательностей, что не всегда удается. Например, построение множественного выравнивания тРНК на некоторых выборках с помощью программы CLUSTAL приводит зачастую к неправильному выравниванию, что, в свою очередь, не позволяет восстановить консервативную структуру. Здесь существенную роль играют правильные эволюционные расстояния между последовательностями. Если последовательности слишком близки, то выравнивание строится надежно, но разнообразие последовательностей слишком мало для расчета ковариаций. С другой стороны, слишком далекие последовательности не позволяют построить правильное выравнивание. В литературе рассматривали и другие подходы к проблеме поиска консервативных вторичных структур. Так, в работах [20, 21] для описания и поиска консервативных вторичных структур применяются контекстно-свободные грамматики. Однако для поиска параметров стохастических контекстно-свободных грамматик требуется достаточно большое количество последовательностей. В работах [22, 23] используется оптимальная вторичная структура для каждой последовательности, которые затем сравниваются, и находятся общие подструктуры. Здесь следует отметить, что поиск оптимальной (в смысле свободной энергии) структуры, во-первых, представляет собой достаточно трудную задачу для вычислений, а во-вторых, часто приводит к неправильным структурам.

В настоящей работе предложен метод поиска консервативных вторичных структур, основанный на анализе множества последовательностей. Первым этапом алгоритма является построение множественного выравнивания, однако, в отличие от традиционных подходов, множественное выравнивание допускает невыравненные участки в последовательностях. Это позволяет рассматривать более далекие последовательности. Наличие невыравненных участков не позволяет напрямую использовать ковариации. Вместо этого метод ищет списки потенциальных спиралей, согласованных с выравниванием. Алгоритм реализован в виде web-сервера (http://bioinf.fbb.msu.ru/RNAAlign/).

АЛГОРИТМ

В основе алгоритма лежит многократное (на разных уровнях) применение метода поиска сигналов MEME. Чтобы пояснить основную идею алгоритма MEME, введем понятие обобщенного профиля. Под обобщенным профилем будем понимать правило, согласно которому некоторой структуре на последовательности ставится в соответствие число, которое обычно называют весом структуры. Другим важным свойством обобщенного профиля является то, что его можно построить по набору структур на последовательностях. В качестве примеров профилей можно привести стандартную позиционную весовую матрицу (в этом случае в качестве структуры используется позиция слова заданной длины на последовательности), или HMM профиль (в качестве структуры рассматривается фрагмент последовательности с указанием вставок и делеций). В этой работе мы будем использовать и другие определения профилей.

Говоря более формально: *структурой на последовательности* называется некоторый набор позиций. *Профилем* называется отображение $P : S \longrightarrow R$, причем для любого непустого множества структур на последовательностях $\{S\}$ можно построить профиль: $\{S\} \longrightarrow P$. *Наилучшим наложением профиля P* на последовательность или *наилучшим вхождением профиля* в последовательность будем называть такую структуру S^* на последовательности, которая обеспечивает наибольший вес: $S^* = \arg\max_S P(S)$.

Основная идея алгоритма состоит в последовательном (итеративном) уточнении профиля:

1. Пусть на первом этапе мы имеем профиль.

2. Находим на каждой последовательности структуру, обеспечивающую максимальное значение *P*.

3. По набору наилучших вхождений вычисляем новый профиль.

4. Если новый профиль не отличается от предыдущего, или, если исчерпано заранее определенное число итераций, то заканчиваем процесс, иначе переходим к п. 2.

В результате алгоритма получаем некий локально оптимальный профиль. Разумеется, не для любого определения профиля этот алгоритм сходится, но в большинстве разумных случаев можно показать его сходимость.

В целом алгоритм можно представить в виде нескольких последовательных этапов.

1. Поиск консервативных блоков.

2. Построение цепочек консервативных блоков. Эти цепочки являются множественным выравниванием последовательностей. Участки последовательностей, не покрытые блоками, могут рассматриваться как невыравненные фрагменты.

3. Поиск потенциальных спиралей.

4. Поиск консервативных (согласованных с блоками) спиралей – "колонок спиралей".

На каждом этапе используется свое определение профиля – на первом этапе это будет стандартная весовая матрица, на втором этапе профиль будет описывать цепочку из консервативных блоков, на четвертом этапе профиль будет описывать консервативные спиральные участки.

Поиск консервативных блоков

В качестве профиля P_w используется стандартная весовая матрица.

Определение 1. *αl-Блоком* будем называть совокупность фрагментов последовательностей одинаковой длины *l*. При этом: 1) в каждой из последовательностей может быть не более одного фрагмента, принадлежащего блоку; 2) доля последовательностей, содержащих фрагменты, принадлежащие одному блоку, составляет не менее α ($\alpha \le 1$). Блок определяется длиной фрагмента *l* и набором позиций $\{p_i\}$. Для блока можно определить число, характеризующее его качество, например, полное информационное содержание.

Поиск консервативных блоков осуществляется с помощью стандартного алгоритма MEME [24]. Задаемся длиной блока. Далее для каждого слова заданной длины из каждой последовательности выполняем следующую процедуру. 1. По одному слову строим частотный профиль с учетом псевдосчетчиков [25]. 2. Затем в каждой последовательности ищем наилучшее вхождение, соответствующее профилю. Если наилучшее вхождение имеет вес меньше заданного значения, то оно в формировании профиля не участвует. 3. По этим вхождениям строим новый профиль. 4. Переходим к п. 2. Процесс заканчивается при сходимости или при исчерпании заданного числа итераций (обычно 3–5). В результате этой процедуры мы получаем большое количество блоков, многие из которых бессмысленны. Задавшись порогом на информационное содержание блока, можно отобрать наиболее значимые блоки.

Построение цепочек

Определение 2. Блок B_1 называется *β-предшествующим* блоку B_2, если доля последовательностей, в которых представлены фрагменты обоих блоков, составляет не менее β и в которых позиции фрагментов первого блока B_1 меньше позиций фрагментов второго блока B_2 составляют не менее β ($\beta \le 1$). В этом случае будем говорить, что $B_1 <_\beta B_2$. Два блока называются *β-сравнимыми*, если один из них *β-предшествует* другому.

Отметим, что определение 2 не задает частичного порядка на множестве блоков, поскольку из

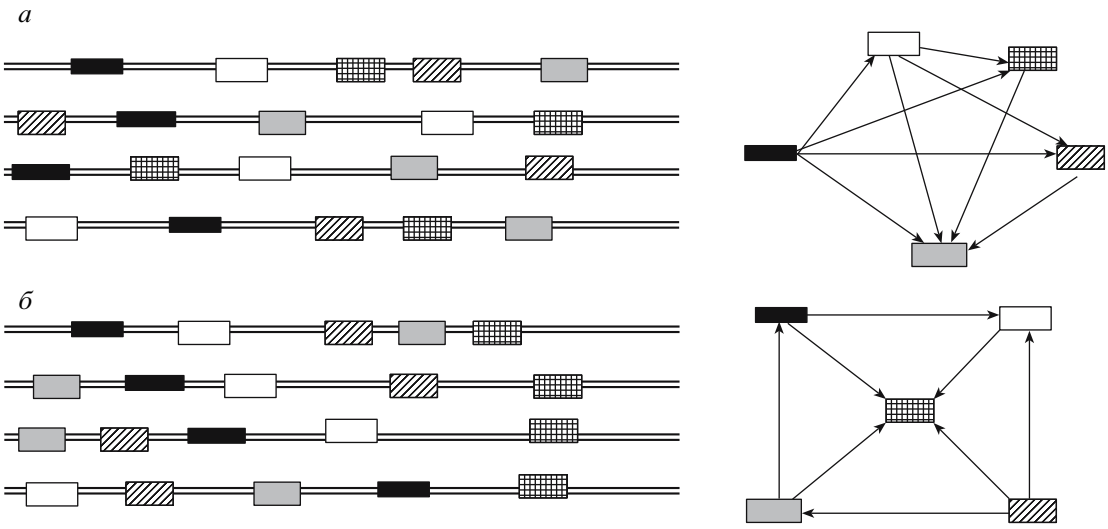

Рис. 1. Консервативные блоки и соответствующий граф для поиска наилучшей β-цепочки (β = 0.75). *а* – Нормальный случай без циклов; *б* – граф с циклом.

условий $B_1 <_\beta B_2$ и $B_2 <_\beta B_3$ не следует, что $B_1 <_\beta B_3$, т.е. не выполняется условие транзитивности.

Определение 3. *β-цепочкой* блоков называется упорядоченный массив блоков $\{B_i\}$ такой, что для любого $i < j$ $B_i <_\beta B_j$. Вес *β-цепочки* определяется как сумма весов блоков, ее составляющих.

Определение 4. β-цепочка $C = \{C_i\}$ называется *β-предшествующей* блоку B, если для любого элемента цепочки C_i выполняется $C_i <_\beta B$.

На основе этих определений можно построить алгоритм поиска *β-цепочки* наибольшего веса. Определим ориентированный граф (рис. 1): каждому блоку поставим в соответствие вершину. Из вершины A в вершину B проводится ребро, если $A <_\beta B$. Задача заключается в поиске самого мощного пути в этом графе, состоящем из β-взаимносовместимых вершин (блоков). Можно применить стандартный алгоритм поиска в глубину, причем на каждом шаге поиска в глубину просматриваются только те вершины, которые совместимы с предыдущими вершинами. Наилучший путь можно найти, проверив каждую вершину графа в качестве стартовой. Разумеется, этот алгоритм можно оптимизировать, но обычно количество вершин в описанном графе невелико, и поиск оптимального пути можно провести достаточно быстро.

Разрешение противоречий. Поскольку определение β-цепочки допускает небольшое противоречие в порядке следования фрагментов последовательностей: если два блока $A <_\beta B$, то в нескольких последовательностях допустимо, что фрагменты, принадлежащие A, лежат правее фрагментов, принадлежащих B. Такое может возникнуть, если алгоритм поиска блоков допустил ошибку. Чтобы восстановить согласованность, используется следующий алгоритм. Каждому фрагменту последовательности из колонки (слову) можно сопоставить число N_{err} – количество колонок, с которыми нарушена согласованность. Выбираем слово с наибольшим значением N_{err} и это слово удаляем из блока. Процедура повторяется до тех пор, пока не исчезнут все противоречия.

В результате предыдущей процедуры получена цепочка непротиворечивых блоков. Некоторые блоки могут пересекаться. Если расстояния между словами пересекающихся блоков постоянны, то эти блоки являются частями блока большего размера и их можно объединить. Если же расстояния между словами непостоянны, то они частично противоречивы на нуклеотидном уровне. Все блоки, которые имеют противоречия, можно представить как совокупность колонок (колонкой называются блоки с длиной слова 1). Теперь можно построить непротиворечивую цепочку колонок максимального информационного содержания, используя уже описанный алгоритм. Затем соседние колонки можно опять собрать в блоки.

Уточнение выравнивания

Комбинированный профиль. Итак, мы получили цепочку непротиворечивых блоков. По ним можно построить комбинированный профиль. Профиль каждого блока определяется как обычная позиционная весовая матрица. Наложение комбинированного профиля на последовательность определяется набором позиций $\{pos_i\}$ каждого блока на последовательности. Вес W такого наложения можно определить как сумму весов профилей блоков минус некоторый штраф за делеции $D(\{pos_i\})$, зависящий от расстояний между

наложенными блоками (вес за делеции будет определен ниже):

$$W(\{pos_i\}) = \sum_{i=1}^{n_{\text{bloks}}} W_i(pos_i) - \sum_{i=2}^{n_{\text{bloks}}} D_i(pos_i - pos_{i-1}).$$

Задача заключается в том, чтобы найти наилучшее наложение комбинированного профиля на последовательность:

$$\{pos_i\} = \arg\max W(\{pos_i\}).$$

С этой целью можно применить алгоритм динамического программирования.

Вес делеций. После построения и дальнейшей прочистки блоков мы для каждого блока получили набор позиций. Ясно, что если разнообразие расстояний между сопряженными блоками невелико, то при наложении комбинированного профиля на последовательность расстояние между наложенными блоками не должно сильно отличаться от наблюдаемых расстояний. С другой стороны, если разнообразие расстояний велико, то и штраф за отклонение от этих расстояний не должен быть велик. Предлагается следующая величина штрафа:

$$D_i(\delta) = \alpha \ln\left(1 + \frac{(e_i - \delta)^2}{\sigma_i^2}\right), \qquad (1)$$

где $e_i = \sum (p_i - p_{i-1})/(n_{\text{blocks}} - 1)$ – среднее расстояние между соседними блоками, $\sigma_i^2 = \sum((p_i - p_{i-1} - e_i)^2 + 1)/(n_{\text{blocks}} - 1)$ – дисперсия расстояния между соседними блоками, p_i – позиции слов блока в последовательностях, определенные при построении блока, α – параметр штрафа.

Комбинированный профиль (в дальнейшем будем называть его просто профилем) позволяет провести итеративное уточнение выравнивания. Полученный на предыдущем этапе профиль можно оптимальным образом наложить на каждую из последовательностей. Далее, по полученным наложениям можно заново построить профиль. Повторив процедуру несколько раз, получим уточненное выравнивание.

После применения процедуры уточнения выравнивания мы получаем набор блоков. Межблочные фрагменты последовательностей можно также попытаться выравнять по той же схеме, стартуя с более коротких слов.

Поиск потенциальных спиралей

При поиске потенциальных спиралей можно использовать различные алгоритмы, например, алгоритм поиска локальных выравниваний Смита–Ватермана. При этом надо задать ограничение на максимальный размер делеции (что в нашем случае соответствует выпячиванию – *bulge*), а также на минимальную мощность спирали. В результате поиска потенциальных спиралей каждая последовательность получила список спиралей $\{h_s\}_i$. Каждая спираль характеризуется четырьмя числами: $h = (f_l, t_l, f_r, t_r)$, определяющими начало и конец (f, t) левого и правого сегментов (l, r).

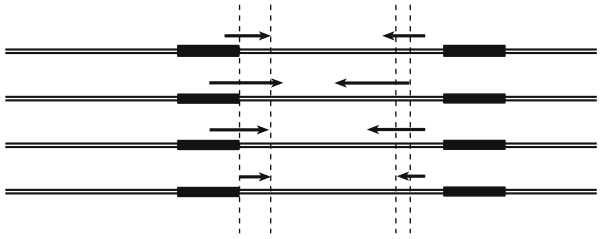

Рис. 2. Относительное положение спиралей и блоков выравнивания.

Определение 5. *α-колонкой спиралей* будем называть такое множество потенциальных спиралей *CH*, что: 1) в каждой последовательности представлено не более одной спирали из *CH* и 2) доля последовательностей, содержащих спирали, не менее чем заданное число α ($\alpha \le 1$).

Задача состоит в том, чтобы найти α-колонки спиралей, которые наиболее согласованы с блоками выравнивания последовательностей (рис. 2). Сначала для этого надо определить меру согласованности колонки спиралей и выравнивания. Каждая спираль (и каждая колонка спиралей) состоит из двух частей – левого и правого сегмента спирали, а те, в свою очередь, определяются двумя наборами чисел. Мера согласованности определяется как сумма мер для каждого типа чисел (левой границы левого плеча спирали, правой границы левого плеча спирали и т.д.).

В качестве меры рассогласования позиции t колонки спиралей H и колонки выравнивания A выбираем

$$D(H^t, A) = \eta \sum_{i=\text{helices}} \log\left(1 + \frac{(\delta_i^t - e^t)^2}{(\sigma^t)^2}\right),$$

где t – тип позиции спирали (начало левого плеча, конец левого плеча …), δ_i^t – расстояние от соответствующего конца спирали в последовательности i до блока в колонке выравнивания, e^t – среднее расстояние позиции типа t спирали до колонки выравнивания, σ^t – среднеквадратичное отклонение расстояний соответствующего типа позиции спирали до колонки выравнивания; η – параметр. Качество позиции t колонки спиралей определяем как минимум от меры рассогласованности, а ка-

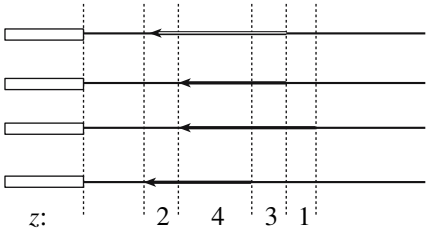

Рис. 3. Подсчет информационного содержания колонки спиралей. При подсчете информационного содержания колонки спиралей в каждой позиции множественного выравнивания подсчитывается количество z спиралей из колонки, покрывающих эту позицию.

чество колонки спиралей как сумму мер рассогласованностей позиций:

$$D(H^t) = \min_A D(H^t, A); \quad D(H) = \sum_t D(H^t).$$

Колонка спиралей, привязанная к выравниванию, может рассматриваться в качестве профиля. Этот профиль определяется следующим набором параметров $\{A^t, e^t, \sigma^t\}$. Этот набор данных обладает двумя важными свойствами: 1) его можно построить по набору спиралей; 2) для каждой спирали можно определить ее качество относительно профиля. Поэтому можно применить итеративный метод построения профиля спиралей:

1. Выбираем очередную спираль и строим по ней профиль. При этом в качестве A^t используем ближайшие колонки выравнивания, e^t – расстояние от соответствующей позиции спирали до блока в колонке выравнивания, σ^t – некоторое произвольное число (например, 1).

2. По заданному профилю находим оптимальные спирали в каждой последовательности. Если рассогласование превышает заданный порог, то спираль не принимается.

3. По набору спиралей строим новый профиль.

4. Если число итераций не превысило заданное количество, то переходим к п. 2.

5. Если остались необработанные спирали, то переходим к п. 1.

В результате применения этого алгоритма получаем набор колонок спиралей, среди которых есть эквивалентные или почти эквивалентные. Если две колонки спиралей содержат хотя бы одну общую спираль, то ту колонку, которая обладает худшим качеством, удаляем.

Информационное содержание для колонки спиралей можно определить следующим образом. Выравняем плечи спиралей относительно соответствующей колонки выравнивания. Спроецируем их на одну линию и подсчитаем в каждой позиции количество спиралей, которые покрывают соответствующую позицию (см. рис. 3). В результате получим массив $\{z\}$. С помощью этого массива подсчитаем информационное содержание для плеча спирали.

$$I = \sum_i f_h(i)\log_2\frac{f_h(i)}{p_0} + f_n(i)\log_2\frac{f_n(i)}{q_0},$$

$$f_h(i) = \frac{z_i + p_0}{N_{\text{seq}} + 1}, \quad f_n = 1 - f_h.$$

Эта величина отражает степень выравненности плеч спиралей. При полностью выравненных спиралях эта величина примерно равна $2L_{\text{helix}}$, где L_{helix} – длина спирали, однако, если спирали плохо выравниваются, то она примерно равна $2L_{\text{core}}$, где L_{core} – длина общей части спиралей.

ПАРАМЕТРЫ ПРОГРАММЫ

Набор параметров, которыми управляется программа, можно разбить по этапам выполнения алгоритма.

Поиск консервативных блоков

• Исходная длина консервативного блока (Start word length). При работе алгоритма значение этого параметра будет уменьшаться. Влияние этого параметра на качество работы неоднозначно – иногда его уменьшение приводит к ухудшению качества работы, поскольку появляется большое количество паразитных блоков, а иногда приводит к улучшению качества.

• Допустимое количество замен (Max N missmatch). Этот параметр используется для сокращения времени построения профиля блока. Уменьшение параметра может привести к потере значимых блоков, но при этом приводит к ускорению работы программы.

• Минимальный вес профиля блока при наложении на последовательность (Min score). В случае, если вес последовательности относительно профиля блока меньше этой величины, то считается, что данный блок не представлен в последовательности. Увеличение параметра приводит к ухудшению качества.

Построение оптимальных цепочек блоков

• Параметр α в формуле для оценки веса делеции в формуле (1) (Gap parameter). Увеличение параметра приводит к более жестким штрафам за делеции.

• Минимальное информационное содержание колонки (Min information). Колонки с информационным содержанием меньше, чем заданное значение, считаются невыравненными.

Таблица 1. Параметры программы

Имя параметра	Описание параметра	Рекомендуемое значение	Значение при тестировании
Start word length	Исходная длина консервативного блока	10–20	16
Max N missmatch	Допустимое количество замен при поиске похожих участков в других последовательностях	6–8	8
Min score	Минимальное значение веса при наложении профиля на последовательность	1.5–2.5	1.5
Gap parameter	Параметр α в формуле для оценки веса делеции	0.5–2.0	1
Min information	Минимальное информационное содержание колонки	0.5–1.2	0.6
Iterations	Максимальное число итераций при уточнении профиля	3–5	3
Max Loop	Максимальный размер петли спирали	50–150	80
Min Energy	Минимальная по абсолютной величине энергия спирали	5.0–6.5	6.0

• Максимальное число итераций при уточнении профиля (Iterations).

Поиск спиралей

• Максимальный размер петли спирали (Max Loop). Увеличение этого параметра приводит к значительному замедлению работы программы. С другой стороны, при уменьшении этого параметра можно потерять некоторые консервативные спирали.

• Минимальная по абсолютной величине энергия спирали (Min Energy). Уменьшение этого параметра может привести к существенному замедлению работы программы.

В табл. 1 приведен список параметров, которые используются в программе.

РЕАЛИЗАЦИЯ АЛГОРИТМА И ТЕСТИРОВАНИЕ

Алгоритм реализован на языке Java в виде веб-ресурса http://www.bioinf.fbb.msu.ru/RNA/MultAl. Сервер принимает набор последовательностей в FASTA формате и в результате выдает множественное выравнивание и набор потенциальных колонок спиралей. Пользователь может отметить те колонки спиралей, которые его интересуют, и увидеть разметку выравнивания.

Тестирование алгоритма проводили на выборке транспортных РНК из *E. coli* (86 последовательностей) при параметрах, указанных в табл. 2. Выборку строили как набор фрагментов из генома *E. coli*, отмеченных ключом tRNA с фланками по 10 нуклеотидов с 3' и с 5' концов. Расчет консервативных вторичных структур занял около 2 мин на стандартном компьютере. Предсказание структур в целом соответствует стандартному представлению о структуре тРНК. Найдено 20 колонок спиралей, 4 из которых отвечают стандартным спиралям структуры тРНК, причем они являются лидерами по информационному содержанию. Наблюдаются весьма консервативные взаимно комплементарные участки между *D* петлей и *Tψ* петлей. Все найденные спирали полностью соответствуют представлению о структуре тРНК. Добавление случайных фланков размером 25 нуклеотидов не меняло предсказания. При геномных исследованиях регуляции зачастую приходится иметь дело с данными, содержащими достаточно много шума. Чтобы проверить устойчивость алгоритма по отношению к добавлению лишних последовательностей, применена следующая процедура. К выборке из 14 негомологичных тРНК из *E. coli* добавляли случайные некодирующие последовательности из того же генома. В табл. 2 показано влияние добавленных последо-

Таблица 2. Зависимость качества предсказания консервативной структуры от количества ложных последовательностей

Количество лишних последовательностей	Ранг элемента				Количество найденных колонок спиралей
	"стебель"	антикодоновая спираль	спираль при *D*-петле	спираль при T_ψ-петле	
0/14	1	2	5	3	20
1/14	1	3	4	2	12
2/14	1	2	9	3	9
3/14	1	2	8	3	9
4/14	1	3	5	2	12
5/14	–	1	2	3	6
6/14	1	3	–	2	5
7/14	1	–	–	2	4
8/14	1	–	–	2	2

Примечание. Указан ранг каждого канонического элемента структуры в списке найденных колонок спиралей.

вательностей на качество предсказания структуры. Видно, что добавление до 20% лишних последовательностей не влияет существенно на качество предсказания, а некоторые спирали надежно находятся даже при добавлении до 50% лишних последовательностей. Немонотонность зависимости числа найденных колонок спиралей объясняется тем, что некоторые "мусорные" последовательности могут содержать слова, похожие на слова в правильных последовательностях и по случайным причинам в них могут возникать спирали, которые поддерживают возникновение колонки заданной мощности.

Я благодарен М.С. Гельфанду и М.А. Ройтбергу за полезные обсуждения, а также А. Витрещаку и Д. Родионову за использование программы и ее тестирование.

Работа частично поддержана Программами Российской академии наук ("Молекулярная и клеточная биология" и "Происхождение и эволюция биосферы"), Медицинским институтом Ховарда Хьюза (55000309) и Российским фондом фундаментальных исследований (04-04-49438).

СПИСОК ЛИТЕРАТУРЫ

1. Spirin A.S. 2002. Ribosome as a molecular machine. *FEBS Lett*. **514**, 2–10.
2. Steitz T.A., Moore P.B. 2003. RNA, the first macromolecular catalyst: the ribosome is a ribozyme. *Trends Biochem Sci*. **28**, 411–418.
3. Gutell R.R., Larsen N., Woese C.R. 1994. Lessons from an evolving rRNA: 16S and 23S rRNA structures from a comparative perspective. *Microbiol Rev*. **58**, 10–26.
4. Doherty E.A., Doudna J.A. 2000. Ribozyme structures and mechanisms. *Annu Rev Biochem*. **69**, 597–615.
5. Buratti E., Baralle F.E. 2004. Influence of RNA secondary structure on the pre-mRNA splicing process. *Mol. Cell. Biol*. **24**, 10505-10514.
6. Lopez-Lastra M., Rivas A., Barria M.I. 2005. Protein synthesis in eukaryotes: the growing biological relevance of cap-independent translation initiation. *Biol. Res*. **38**, 121–146.
7. Kim V.N., Nam J.W. 2006. Genomics of microRNA. *Trends Genet*. **22**, 165–173.
8. Vitreschak A.G., Rodionov D.A., Mironov A.A., Gelfand M.S. 2004. Riboswitches: the oldest mechanism for the regulation of gene expression? *Trends Genet*. **20**, 44–50.
9. Nudler E., Mironov A.S. 2004. The riboswitch control of bacterial metabolism. *Trends Biochem. Sci*. **29**, 11–17.
10. Zuker D.P., Zuker M. 1989. Computer prediction of RNA structure. *Methods Enzymol*. **180**, 262–288.
11. Zuker M. 2003. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res*. **31**, 3406–3415.
12. Freier S.M., Kierzek R., Jaeger J.A., Sugimoto N., Caruthers M.H., Neilson T., Turner D.H. 1986. Improved free-energy parameters for predictions of RNA duplex stability. *Proc. Natl. Acad. Sci. USA*. **83**, 9373–9377.
13. Layton D.M., Bundschuh R. 2005. A statistical analysis of RNA folding algorithms through thermodynamic parameter perturbation. *Nucleic Acids Res*. **33**, 519–524.
14. Gultyaev A.P., van Batenburg F.H., Pleij C.W. 1995. The computer simulation of RNA folding pathways using a genetic algorithm. *J. Mol. Biol*. **250**, 37–51.
15. Titov I.I., Vorobiev D.G., Kolchanov N.A. 2002. Mass analysis of RNA secondary structures using a genetic algorithm. Proceeding of the second international conference of bioinformatics of genome regulation and structure (BGRS-2000). **2**, 138–142.
16. Danilova L.V., Pervouchine D.D., Favorov A.V., Mironov A.A. 2006. RNAKinetics: a web server that models secondary structure kinetics of an elongating RNA. *J. Bioinf. Comp. Biol*. **4**, 589–596.
17. Mironov A., Kister A. 1986. RNA secondary structure formation during transcription. *J. Biomol. Struct. Dyn*. **4**, 1–9.
18. Xayaphoummine A., Bucher T., Thalmann F., Isambert H. 2003. Prediction and statistics of pseudoknots in RNA structures using exactly clustered stochastic simulations. *Proc. Natl. Acad. Sci. USA*. **100**, 15310–15315.
19. Chen Y., Carlini D.B., Baines J.F., Parsch J., Braverman J.M., Tanda S., Stephan W. 1999. RNA secondary structure and compensatory evolution. *Genes Genet. Syst*. **74**, 271–286.
20. Knudsen B., Hein J. 2003. Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Res*. **31**, 3423–3428.
21. Klein R.J., Eddy S.R. 2003. RSEARCH: finding homologs of single structured RNA sequences. *MC Bioinformatics*. **4**, 44.
22. Bouthinon D., Soldano H. 1999. A new method to predict the consensus secondary structure of a set of unaligned RNA sequences. *Bioinformatics*. **5**, 785–798.
23. Pavesi G., Mauri G., Stefani M., Pesole G. 2004. NAProfile: an algorithm for finding conserved secondary structure motifs in unaligned RNA sequences. *Nucleic Acids Res*. **32**, 3258–3269.
24. Bailey T.L., Noble W.S. 2003. Searching for statistically significant regulatory modules. *Bioinformatics*. **Suppl 2**, II16–II25.
25. Дурбин Р., Эдди Ш., Крог А., Митчисон Г. 2006. Анализ биологических последовательностей. Москва–Ижевск: НИЦ "Регулярная и хаотическая динамика".