*Информационные процессы, Том 3, № 1, 2003, стр. 47–60.*




# Поиск консервативных вторичных структур РНК[1]

**В.А. Любецкий, К.Ю. Горбунов**

*Институт проблем передачи информации, Российская академия наук*
*101447, Москва, ГСП-4, Б.Каретный пер., 19, Россия*
*e-mail: lyubetsk@iitp.ru, gorbunov@iitp.ru*

Поступила в редколлегию 24.12.2002

**Аннотация—**Описана формальная постановка задачи поиска консервативных вторичных структур РНК и алгоритм ее решения. Приведены результаты тестирования алгоритма на наборах фрагментов последовательностей РНК.

## 1. ВВЕДЕНИЕ

Роль вторичной структуры РНК хорошо известна. Она важна и для структурных РНК, таких как рибосомные или малые ядерные РНК, и для РНК ряда вирусных геномов, и для рибозимов, и для регуляции экспрессии генов. Большинство известных систем регуляции в бактериях на уровне матричных РНК работают по принципу аттенуации, когда реализуется одна из двух альтернативных структур. “Запрещающая” на уровне транскрипции содержит терминатор транскрипции, а на уровне трансляции блокирует сайт связывания рибосом. Альтернативная (“разрешающая”) структура соответственно не позволяет образоваться терминатору транскрипции или не позволяет блокировать сайт связывания рибосом. При этом часто “разрешающая” структура стабилизируется различными молекулами: белком в случае регуляции экспрессии рибосомных белков, низкомолекулярными лигандами (например, FMN) в случае регуляции синтеза рибофлавина и тиамина, другими РНК в случае Т-бокса.

Вторичная структура состоит из *спиралей*, а спираль состоит из двух упорядоченных наборов (каждый из одинакового числа) *отрезков* нуклеотидов — левого и правого наборов. В каждом из этих наборов между соседними отрезками имеются какие-то *промежутки* — либо не имеющие парного промежутка в другом наборе (тогда этот промежуток называется *выпячиванием*), либо имеющие таковой (тогда эта пара промежутков называется *внутренней петлей*). Промежуток между самими наборами называется *внешней петлей*. Каждый $i$-ый отрезок от *начала левого набора* должен быть комплементарным $i$-ому отрезку от *конца правого* набора. В таком порядке эти отрезки всегда нумеруются.

Концы левого набора будем *обозначать* A и B, а концы правого набора — C и D (нумерация нуклеотидов везде от начала исходной последовательности к ее концу). Исходную последовательность часто называют еще *фрагментом*, имея в виду, что она является частью какой-то еще более длинной нуклеотидной последовательности. Отрезки от A до B и от C до D называют *плечами* спирали (соответственно левым и правым плечами). Таким образом, плечо — это набор отрезков вместе со всеми выпячиваниями и внутренними петлями (точнее, “половинками” петель) в нем.

Пару таких $i$-ых отрезков назовем *простой спиралью*, т.е. простая спираль — это спираль, у которой плечи содержат по одному отрезку. Всякая спираль однозначно образуется последовательным присоединением простых спиралей, считая их от петли (т.е. от B к началу и от C к концу фрагмента). А именно, присоединением $k$ простых спиралей в случае исходной спирали с $k$ отрезками в одном и другом плече; при таком присоединении каждая очередная простая спираль имеет все более длинную внешнюю петлю, в которой последовательно расположены все предыдущие спирали.

[1] Эта работа была поддержана грантом Минпромнауки.

Вообще отношение “спираль E целиком содержит в своей петле спираль F” *обозначается* E>F и играет здесь существенную роль. Другое важное отношение состоит в том, что спираль E расположена целиком левее спирали F, в этом случае будем *писать* E/F. Понятно, что взаиморасположение четырех спиралей клеверного листа целиком описывается этими двумя отношениями. *Перекрытием (*= 2-*перекрытием)* назовем такое взаимное расположение двух спиралей с плечами L1,R1 и L2,R2, что плечо L1 целиком лежит левее L2, а плечо R1 целиком лежит левее R2 (при этом плечи R1 и L2 могут пересекаться или, как иногда говорят, “зацепляться” на определенное число нуклеотидов, которое назовем *коэффициентом зацепления* перекрытия).

3-*Перекрытием* назовем три спирали, последовательно образующие два перекрытия: первая со второй и вторая с третьей. Такова, например, широко распространенная трехшпилечная структура. Здесь существенно еще большее число характеристик типа коэффициента зацепления (т.е. фактически характеристик взаиморасположения) первой спирали со второй и второй с третьей. Условимся нумеровать сами спирали в 3- и т.п. перекрытиях от конца к началу фрагмента. Это связано с тем, что обычно последняя спираль является терминатором, перед ней находится антитерминатор, а еще левее расположен спесифайер или паузная шпилька. Таким образом, в нашей нумерации 1-ая спираль — это терминатор, 2-я — антитерминатор, 3-я — спесифайер или аналогичная ему спираль.

*Псевдоузлом* называется такое взаимное расположение двух спиралей с плечами L1,R1 и L2,R2, когда плечи расположены в порядке L1, L2, R1, R2, т.е. во внешней петле первой спирали содержится ровно одно (левое) плечо второй спирали.

## 2. ПОСТАНОВКИ ЗАДАЧИ

Пусть дан набор из n фрагментов РНК. Часто это — набор регуляторных областей ортологичных генов. Далее переменные $i$ и $j$ пробегают номера этих фрагментов от 1 до $n$. Ищется набор подобных (= гомологичных) вторичных структур, содержащихся в этих фрагментах; может быть (редко), по несколько структур берется из одного фрагмента и (часто) не берется ни одной структуры из некоторых (так называемых, “*мусорных*”) фрагментов.

Говоря образно, интересующую нас ситуацию можно описать так. Когда-то в прагеноме была регуляторная область к определенному гену или оперону, которая в ходе эволюции разошлась по геномам разных организмов. В ходе этого процесса сам этот праген, его прарегуляторная область и, главное, интересующая нас правторичная структура (а также ее спирали) изменялись. Однако, остались, во-первых, некоторые консервативные нуклеотиды в определенных позициях относительно этой вторичной структуры (типа: “на таком-то интервальном расстоянии перед левым плечом”, или “во внешней петле 2-ой спирали” и т.п. — такой конкретики довольно много и ее нужно учитывать). Такие наборы консервативных нуклеотидов называются *боксами*. Во-вторых, осталось нечто общее (тут менее ясно, что же именно) в строении самой вторичной структуры и ее частей вследствие развития структуры от общего предка.

Один такой набор гомологичных вторичных структур назовем *сигналом*. Представляет интерес поиск и нескольких “лучших” сигналов. Часто интересно и достаточно было бы найти только ранжирование спиралей в каждой не мусорной последовательности, при котором спираль с меньшим номером (из $i$-ой последовательности) имела бы большие основания претендовать на вхождение в искомую вторичную структуру этой $i$-ой последовательности. В такой постановке вопрос о нахождении самой вторичной структуры в $i$-ой последовательности не ставится (он может возникать только как средство узнать, правильно ли были ранжированы спирали в наборе последовательностей).

С другой стороны, вопрос о том, как данный набор спиралей (может быть, вместе с их ранжированием по значимости) в данной последовательности складывается в ней в “наилучшую” структуру (иногда, структуры) сам представляет интерес и обсуждается ниже. Конечно, здесь принципиальное значение имеет указание функционала, который фактически и определяет, что такое “наилучшая” структура.

Если ограничиться структурами без псевдоузлов и их спиралями, то указанную выше постановку задачи в терминах поиска набора попарно изморфных подграфов в исходных графах вторичных структур можно переформулировать следующим более простым образом.

Дан набор множеств $F_i$, а именно, из $i$-й последовательности в $F_i$ включаются какие-то из ее спиралей (все или только часть), а $i$ пробегает номера всех исходных последовательностей. Будем считать, что все спирали в каждом $F_i$ перенумерованы переменной $k$. Пусть также дана *функция похожести (сходства)* любых двух спиралей $k_1$ и $k_2$ из этих (различных) множеств.

**Первая задача.** В большинстве этих множеств найти по подмножеству, для каждого из которых его элементы располагаются в виде (плоского) дерева с соблюдением следующих двух условий. Каждый предок A содержит потомка B в своей спирали, т.е. A$>$B. И вершина B, которая расположена в дереве "лексикографически левее, но не является предком" вершины A, находится в отношении "спираль B целиком левее спирали A", т.е A$/$B. При этом требуется, чтобы все так полученные деревья были попарно изоморфны друг другу (вариант: все они были изоморфны наперед заданному дереву), а соответствующие при изоморфизме их вершины (т.е. соответствующие спирали) были максимально похожи друг на друга. Эффективный алгоритм для решения этой задачи возможен, он будет представлен в другой публикации. Здесь мы рассмотрим следующую еще более упрощенную постановку, для которой ниже приведем алгоритм ее решения.

**Вторая задача.** По каждому множеству $F_i$ образуем множество $F_i'$, состоящее из плеч всех спиралей, входящих в $F_i$. Точнее, $F_i'$ состоит из всех пар вида $\langle k, d\rangle$, где переменная $k$ нумерует все спирали из $F_i$, а булевская переменная $d$ один раз равна 0, а один раз равна 1 для каждого $k$. Пара $\langle k, 0\rangle$ обозначает просто левое плечо спирали $k$, а пара $\langle k, 1\rangle$ — правое плечо той же спирали $k$. Линейно упорядочим множество $F_i'$ в соответствии с тем порядком, в котором плечи расположены п последовательности РНК. Требуется в каждом $F_i'$ найти некоторое подмножество, чтобы все подмножества имели одинаковую мощность и их элементы, стоящие на одинаковых (в смысле индуцированного линейного порядка) местах были бы максимально похожи друг на друга (в том числе, имели бы одинаковое значение переменной $d$).

## 3. БАЗИСНОЕ ОПИСАНИЕ АЛГОРИТМА

Представим себе, что каждое из множеств $F_i'$ как-то линейно упорядоченно, т.е. превращено также в последовательность, которую мы будем называть *$i$-й последовательностью плеч*, соответствующей исходной $i$-й последовательности нуклеотидов.

Таким образом, индекс $i$ нумерует все последовательности плеч (как и исходные нуклеотидные последовательности, здесь $i$ пробегает от 1 до $n$). Если теперь последовательности плеч выравнить в духе алгоритма Смита-Уотермена (или иным алгоритмом) попарно или множественно, все сразу, то можно предполагать, что плечи, которые попарно выравниваются друг на друга, будут соответствовать в каждой нуклеотидной последовательности искомой вторичной структуре в ней.

Например, пусть речь идет о попарном выравнивании 1-й последовательности с каждой $j$-ой (где $j$ пробегает от 2 до $n$). Припишем $l$-му плечу из 1-ой последовательности сумму весов, которые оно получает при каждом из этих выравниваний; эту сумму назовем весом $l$-го плеча. Пусть максимум весов плеч из 1-ой последовательности достигается на каком-то плече $l_0$, которое мы обозначим $l_0(1)$.

Заменяя число 1 на произвольный номер последовательности $i$, аналогично получим плечо $l_0(i)$. Естественно предполагать, что функция $l_0(i)$ дает хороших кандидатов во вторичные структуры каждой исходной $i$-й последовательности. Более того, последовательности, для которых вес плеча $l_0(i)$ ниже некоторого порога, объявим мусорными и удалим из исходного набора нуклеотидных последовательностей. Аналогично находится следующее после $l_0(i)$ выделенное плечо, его обозначим $l_1(i)$; и т.д. в каждой из остающихся (не мусорных) нуклеотидных последовательностей образуем некоторое фиксированное число "наилучших" плеч $l_2(i), \ldots, l_c(i)$. Таким образом, в каждой исходной последовательности будут отобраны и ранжированы некоторые плечи.

Описанный подход предполагает определенные оговорки. При выравнивании последовательностей плеч фиксируется *функция сходства* двух спиралей, играющая такую же роль как призы и штрафы при выравнивании последовательностей из букв. С ее помощью разрешается выравнивание только левого плеча с левым и правого с правым. Если при выравнивании двух последовательностей плеч оба плеча одной $k$-й спирали выравнились, соответственно, с двумя плечами какой-то $r$-й спирали (это, конечно, может быть и не так), то спирали $k$-ю и $r$-ю назовем *выравненными*. К весу $l_p(i)$ плеча добавляется приз, если происходит выравнивание не только этого плеча, но и его спирали. Аналогично, если $k$-я спираль выравнилась с $r$-й, а $r$-я выравнилась с $s$-й (из какой-то третьей нуклеотидной последовательности), то в случае, если при этом и $k$-я выравнилась с $s$-й, то всем участвующим в этом плечам добавляется еще приз ("*правило треугольника*").

Наконец, существенной добавкой к изложенной идее является следующее. Кроме множеств $F'_i$ плеч рассмотрим и множества $G_i$ консервативных боксов из той же $i$-й последовательности. Пусть теперь образована последовательность плеч и боксов из $i$-й последовательности, которую будем обозначать также $F'_i$. В этой последовательности отражено взаиморасположение уже и плеч, и боксов. Будем называть ее *последовательностью плеч-боксов*. Действуем с этой последовательностью также как выше; это приводит к одновременному выравниванию как плеч спиралей, так и консервативных боксов.

Изложенная выше идея алгоритма основана на том, что у консервативных структур гомологичные спирали часто имеют и гомологичные плечи, поэтому выравнивание "последовательностей" спиралей можно заменить на выравнивание последовательностей плеч (или, лучше, последовательностей плеч-боксов). Подробное описание этого алгоритма выделено в приложение 1.

Таким образом, задача 2 проще задачи 1. Такое упрощение имеет свои недостатки: в терминах задачи 2 труднее учитывать важное отношение "быть плечами одной спирали".

Один из оптимальных компромиссов между большей биологической содержательностью постановки и возможностью эффективного алгоритма содержится в задаче 1, сформулированной выше.

## 4. ТЕСТИРОВАНИЕ АЛГОРИТМА

Приведем результат тестирования алгоритма на 18 фрагментах тРНК кишечной палочки длины каждая около 100 нуклеотидов. Ниже в ответах после номера организма и названия антикодона перечисляются найденные нашим алгоритмом спирали этой вторичной структуры в каждом из данных фрагментов.

При этом используются следующие сокращения. Буква Ч обозначает акцепторную спираль (черенок), Д — D-спираль, А — антикодонную, П — псевдоуридиновую спирали. Отсутствие буквы в каком-то ответе означает, что соответствующая спираль не была найдена алгоритмом.

Числа в квадратных скобках после какой-то из этих букв означают погрешности на двух концах внешней петли, которые допустил алгоритм по сравнению с биологическим ответом. Отсутствие таких чисел означает, что соответствующая максимально продолженная спираль найдена алгоритмом точно.

Число в круглых скобках указывает, какое число ложных (отсутствующих в биологическом ответе) спиралей выдал алгоритм.

Например, для случая DH1660 GTG сообщается, что алгоритм нашел все четыре спирали, но спираль П нашлась по сравнению с природной в части концов внешней петли с ошибкой в 1 нуклеотид слева и с ошибкой в 2 нуклеотида справа; при этом алгоритм выдал одну ложную спираль.

DA1660 TGC: Ч,Д,А,П (1); DA1661 GGC: Ч,Д,А,П (1); DC1660 GCA: Ч,А,П (0);
DD1660 GTC: Ч,А,П (1); DE1660 TTC: Ч,П (2); DF1660 GAA: Ч,Д,А,П (0);
DG1660 TCC: Ч,А,П (1); DG1661 GCC: Ч,Д,А,П (1); DG1662 CCC: Ч,Д,А,П (0);
DH1660 GTG: Ч,Д,А,П[1,2] (1); DI1660 GAT: Ч,Д,А,П (0); DI1661 CAT: Ч,Д,А,П (1);

DK1660 TTT: Ч,Д,А,П (0); DL1660 CAG: А,П (2); DL1661 TAG: Ч,П (2);
DL1662 CAA: Ч,А,П (2); DL1663 GAG: Ч,А,П (1); DL1664 TAA: Ч,А,П (0).

Как видно из результатов счета, псевдоуридиновую спираль алгоритм нашел в 100% исходных последовательностей, акцепторную спираль — в 94%, антикодонную спираль — в 83% и D-спираль — в 50%. D-спираль находится с трудом, поскольку она короткая. Алгоритм не использовал никакой специфики тРНК-го случая.

Кроме случая **клеверного листа** алгоритм тестировался для других случаев: RFN-структур [1], Т-боксов, S-боксов. **RFN-структура** регулирует экспрессию генов биосинтеза и транспорта рибофлавина. Она состоит из черенка и четырех спиралей (другие спирали обычно не столь консервативны, и нами не искались).

Ниже приведен результат счета на выборке из 20 RFN-фрагментов длины примерно 150, причем левое плечо спирали бралось с окрестностью по 10 нуклеотидов с каждой стороны.

Ниже перед каждым результатом указано сокращенное название генома и гена, из $5'$ области которого вырезан соответствующий фрагмент. Цифра 1 обозначает черенок, а цифры 2,3,4,5 — спирали (занумерованные по часовой стрелке). Смысл остальных сокращений тот же, что и выше.

BS ribD: 2[1,4],3[2,2],5[2,2](3); BQ ribD: 2[9,6],3,5(5); BE ribD: 1,3[0,12],4[0,3],5[2,2](4);
HD ribD: 2,5[2,2](4); CA ribD: 1,2,3[3,0],5[2,2](3); DF ribD: 1,2,3[2,2],4[0,6],5[2,2](4);
SA ribD: 1,2[5,0],3[2,2],5[2,2](3); LLX ribD: 1,2,3[2,2],4,5(4);
PN ribD: 1,2,3,4[0,6],5[2,2](4);TM ribD: 1,2[1,4],3,5[2,2](4); BS ypaA: 3,5[2,2](6);
BQ ypaA: 1,2[1,4],4,5[2,2](4); BE ypaA: 3[1,16],4[12,2],5[4,5](4);
SAypaA: 1,2[1,4],3[2,2],5[2,2](3); DF ypaA: 1,2[6,0],3[2,2],4,5[2,2](4);
EF ypaA: 1,3,5[2,2](4); LLX ypaA: 2,3[1,11],5(1); ST ypaA: 1,2,3,5(3);
SA ypaA: 2[5,0],3[2,2],4[2,5],5[2,2](3); AMI ypaA: 2,3[1,5](2).

Видно, что черенок найден в 60% последовательностей, спираль 2 — в 80%, спираль 3 — в 90%, спираль 4 — в 40% и спираль 5 — в 95%. Четвертая спираль выделяется существенно хуже остальных. Это объясняется низкой консервативностью соответствующего участка РНК.

Если отказаться от поиска черенка и задать достаточно малый максимальный размер петли, то можно рассчитывать на улучшение качества работы алгоритма за счет значительного уменьшения числа рассматриваемых спиралей. Также, значительно уменьшается время работы алгоритма, что позволяет обрабатывать выборки большего размера. Ниже приведены результаты запуска программы на выборке из 39 RFN-фрагментов, расширяющей предыдущую выборку (максимальный размер петли равен 40).

BS ribD: 2,3,5[2,2](2); BQ ribD: 2,3,5[2,2](2); BE ribD: 3[0,12],5[8,1](5);
HD ribD: 2,3,5[2,2](2); CA ribD: 2,3,4,5[2,2](2); DF ribD: 2,3,4,5[2,2](3);
SA ribD: 2,3[2,2],5[2,2](2); LLX ribD: 2,3[5,8],4,5[2,2](2); PN ribD: 2,3,4,5[2,2](4);
TM ribD: 2[6,0],3,5[6,2](5); BS ypaA: 2,3,4,5[2,2](3); BQ ypaA: 2[9,6],3,4,5[2,2](4);
BE ypaA: 2,3,5[8,1](3); CA ypaA: 2,3,5[2,2](4); DF ypaA: 2,3,4,5[2,2](3);
EF ypaA: 2,3,4[1,5],5[2,2](3); LLX ypaA: 3,5[2,2](2); ST ypaA: 2,3,5[2,2](2);
SA ypaA 2[1,2],3[0,7],5[2,2](2); AMI ypaA: 2,3[1,5](2); DR ribD: 2,3[1,2](2);
TQ ribD: 2,3,4,5[11,0](3); AO* ribD: 2,3,4,5[2,2](3); DU* ribD: 2,3,5[2,2](2);
CAU ribD: 2[4,0],3[2,6],4,5[6,2](3); FN ribH2: 2,3, 4,5[2,2](4);
FN ribE: 2,3,4[1,5],5[2,2](1); TFU ribE: 3,5[3,7](2); SX ribE: 2,3[5,9]5[1,7](2);
BME ribH2: 2,3,4,5[5,6](3); REU ribB: 2,4[6,6](4); EC ribB:2,3,4,5[2,2](4);
KP ribB: 2,3,4,5[6,1](4); VK ribB: 2,4,5[2,2](4); VC ribB: 2,3[0,13],4,5[6,2](1);
BPA ribB: 2[2,4],4(2); AC ribB: 2,3,4,5[2,2](3); MLO ribH2: 2,4,5[10,10](5);
PA ribE: 2,3[1,8],4,5[5,9](3).

Как видно, результат улучшился: спираль 2 найдена в 92% последовательностей, спираль 3 — в 90%, спираль 4 — в 59%, спираль 5 — в 90%.

Другой способ уменьшить число спиралей, участвующих в выравниваниях — увеличить минимальную длину спирали.

Теперь рассмотрим **структуру Т-бокса**, которая регулирует гены, связанные с транспортом, биосинтезом и усвоением аминокислот в Грам-положительных бактериях. В этой структуре алгоритм искал три шпильки. Первая спираль (спесифаер) обычно состояла из 4-5 отрезков, в среднем три из которых имели длину строго больше трех. Вторая спираль (антитерминатор) состояла из двух отрезков, хотя бы один из которых имел длину строго больше трех. Третья спираль (терминатор, перекрывающийся со второй спиралью) состояла из одного или двух отрезков, по крайней мере один из которых был достаточно длинным. (Эта регуляторная система образует две альтернативных конформации: одна из них соответствует низкой концентрации аминокислоты и содержит спесифаер и антитерминатор; другая соответствует высокой концентрации аминокислоты и содержит спесифаер и терминатор.)

В таких условиях естественно задать минимальную длину спиралей, равной 4, так как расположение недостающих спиралей длины 3 можно легко восстановить по остальным спиралям.

Кроме того, задавались четыре консервативных бокса, о которых известно, что они с небольшими изменениями всегда встречаются в структуре данного типа. Это AGTA{1}, AGAGA{1}, (A|G)(A|C|G|T)TG{0}, GGGTGG{1}; здесь в фигурных скобках указано число возможных замен букв в слове, а в круглых через вертикальную черту перечисляются допустимые на данной позиции буквы, если их больше одной.

Левое плечо бралось с окрестностью радиуса 15. Ниже приводится результат работы алгоритма на выборке из 16 фрагментов длины около 300. Обозначение i.j означает $j$-ую спираль $i$-ой шпильки (когда терминатор состоит лишь из одной спирали, мы пишем “3” без точки), остальные обозначения аналогичны предыдущим.

BS serS: 1.1, 1.3, 1.4, 2.1, 2.2, 3.1 (6); BS serS: 1.4, 2.1[0,9], 2.2[0,5], 3 (9);
HD serS: 2.1, 3 (11); BQ serS: 1.1, 2.1, 2.2, 3 (9);
HD serA: 1.1, 1.4, 1.5, 2.1[1,10] (4); BE tyrS: 1.2, 1.4, 2.1, 2.2, 3 (4);
BS tyrZ: 1.2[10,9], 1.3, 3.1 (9); HD tyrZ: 1.5, 2.1, 2.2, 3.1 (4);
BS trpS: 1.1[3,1], 1.2, 1.3, 2.1, 3.2 (5); BQ trpS1: 2.1, 2.2, 3 (6);
BQtrpS2: 1.1, 1.3, 1.4, 2.1, 3 (2); BQ aroF: 1.2, 1.4, 2.1, 2.2, 3 (4);
BS pheS: 1.4, 2.1[3,11],3 (5); BE pheS: 1.2, 1.4, 2.1, 2.2, 3.1 (4);
HD pheS: 2.1, 2.2, 3.1 (5); BQ pheS: 1.4, 2.1[5,8], 3.1, 3.2 (5).

Как видно, среди спиралей длины 4 или больше алгоритм нашел 52% спиралей специфаера, 75% спиралей антитерминатора и 75% спиралей терминатора.

Заметим еще, что отличить истинные спирали от ложных часто можно по их качеству; это видно на примере, который приводится ниже. Повторим еще раз, что выше применялся один и тот же алгоритм, не использующий специфики каждого из рассмотренных случаев.

Теперь приведем результат тестирования нашего алгоритма на 20 искусственных последовательностях типа RFN длины 190 (черенок с петлей длины 160 и в ней четыре спирали с петлями длины 15). Все спирали имели длину 6. Выборка была сформирована так: сначала задавались последовательности, у которых все однотипные спирали совпадали, а затем в каждой спирали в двух случайных местах проводились случайные замены, сохраняющие комплиментарность (ковариантные замены).

Ниже содержимое каждой пары круглых скобок соответствует одной последовательности: слева от вертикальной черты перечислены качества полученных алгоритмом истинных спиралей, справа — качества полученных им ложных спиралей.

(56,43,33,12|20,11); (55,35,20,9|17,12); (40,32|30,22,15,12,12); (37,26,26,25|12);
(38,18,15|20,17,8,8); (41,35,16|18,8,8,8); (52,18|19,19,11); (33,21,21,13|22,11);
(61,52,33|12,11,8,8,8); (40,17,8|13,12,12); (30,25|14,12,12); (65,26,24|12,7);
(107,37,16|28,12,9,8); (76,51,43,12|8); (29,26|29,16,13,12); (72,21,19|8);
(29,27,12|17,12,11); (47,45,18,8|13,8,8); (34,11|16,12,8); (100,38,25,24,17|17,9,8).

Как видно, в подавляющем большинстве случаев по крайней мере две из трех спиралей с наибольшим качеством являются истинными.

На этой же выборке тестировался специальный вариант нашего алгоритма, в котором можно указывать топологию искомой структуры, задаваемую последовательностью левых и правых плеч спиралей этой структуры в том порядке, в каком они встречаются в последовательности РНК (например, структуре типа RFN соответствует последовательность плеч “0010101011”, где число 0 обозначает левое плечо, а 1 — правое). Полученный результат несколько лучше предыдущего. Приводим его в том же виде:

(25,18,9|29,16,5); (45,33,20,12,8|4,4); (21|20,12,13,11); (68,41,14,14,5|ложных нет);
(70,34,27,17,9|8); (42,40,12,4|5,4); (32,24,17,4|13,9,4); (17,16,14,6|17,14);
(38,23,16,13,6|14,8,8); (47,28,4|24,16); (16,9|26,22,13,5); (80,24,23,9|ложных нет);
(33,28,16|9,9,9,8); (49,35,18|14,9,5); (22,18,12|9,8,5); (58,37,32,32,4|4);
(20|25,13,12,4); (73,30,18,13|16,4); (59,36,20|13); (62,53,28,27,18|5,5,5).

Алгоритм сохраняет работоспособность, когда часть исходных последовательностей “мусорные”, т.е. не содержат искомых структур. Так, программа тестировалась на фрагментах, первые десять из которых — те же, что и выше, а последние десять — случайные по равномерной мере.

Приводим результат такого счета (для первых десяти фрагментов).

(26,11|13,12,11,9,8); (27,23,13,8|13,8); (26,11|15,12,8); (28,23,17|16,8);
(16[2,1],11|24,10,8); (20[1,3],19,17,16,8|16); (22,21,20,12|11); (23,21,16|16,14);
(34,8|19,8); (16|20,13,12,12,12,7).

Выделить “мусорную” последовательность также можно по ее качеству, т.е. по сумме качеств выделенных в ней алгоритмом спиралей.

Например, при счете специального варианта алгоритма на искусственной выборке типа RFN, включающей 15 последовательностей, имеющих искомую структуру (спирали длины 7 с двумя ковариантными заменами) и 5 случайных последовательностей, суммы качеств распределились следующим образом:
неслучайные: 128, 133, 107, 106, 102, 101, 98, 89, 69, 67, 66, 59, 59, 58, 40;
случайные: 43, 43, 33, 28, 22.

Как уже отмечалось, часто достаточно выделить спирали, которые с большой вероятностью принадлежат искомой структуре.

Так, алгоритм тестировался на наборе из 20 последовательностях **типа S-box** средней длины около 350. Эта структура содержит 7 простых спиралей, первые три из которых обладают большой консервативностью. Каждое плечо спирали бралось с окрестностью радиуса 10, минимальная длина спирали равнялась 4, максимальное выпячивание равнялось 7, так что расположенные недалеко друг от друга вторая и третья спирали иногда сливались в одну (как в 24-й спирали).

Приведем первые 30 спиралей из общего списка спиралей, набравших наибольшее качество. Ниже после качества q спирали приводится ее номер в структуре (если она истинная) и (в скобках) сокращенное название генома. Истинные спирали помечены знаком $+$, немного сдвинутые — знаком $\sim$ (с указанием размеров сдвига), а ложные — знаком $-$.

+1:q=1774,N3(RCA03456);+2:q=1743,N3(RSA01535);
+3:q=1708,N1(RZCO1284_YAEC);+4:q=1692,N3(RBS03888);
∼ 5[5,1]:q=1621;N2(RZCO2886_YJCI);
+6:q=1539,N1(RZC04628_YUSC);+7:q=1539,N3(RCA02165);
+8:q=1413,N1(RZCO5182);+9:q=1395;N1(RBS01188);
+10:q=1369,N3(RDF01337);+11:q=1350,N1(RCA02165);
+12:q=1325,N1(RSA01535);+13:q=1295,N1(RZCO5182);
+14:q=1295,N1(RBE04103);+15:q=1187,N1(RZCO5367);
∼ 16[5,1]:q=1086,N2(RBS01855);+17:q=1053,N3(RCA01265);
+18:q=1009,N3(RBS03889);+19:q=961,N1(RSA02024);
+20:q=936,N1(RBS01558);+21:q=853,N1(RZCO2886_YJCI);
+22:q=815,N3(RDF00647);+23:q=782,N1(RBS01855);
+24:q=776,NN2,3(RSA02024);−25:q=770(RBS01855);
+26:q=758,N1(RDF00647);+27:q=753,N1(RCA03456);
−28:q=744(RCA01265);∼ [5,1]29:q=672,N2(RBS01188);
∼ [6,0]30:q=649,N2 (RBE04103).

Как видно, алгоритм из 30 спиралей выдал 26 истинных, 4 немного сдвинутых и 2 ложных.

Авторы благодарны А.А. Миронову за многочисленные пояснения биологического содержания задачи и обсуждения результатов.

## 5. ПРИЛОЖЕНИЕ 1: ПОДРОБНОЕ ОПИСАНИЕ АЛГОРИТМА

Итак, пусть дано $n$ нуклеотидных последовательностей, длины которых могут быть различными.

**Этап 1. Составление списков спиралей.** Для каждой последовательности составляется список спиралей, удовлетворяющих следующим требованиям.

1) Длина каждого отрезка не меньше minpodryad (обычно, minpodryad равен 3 или 4).

2) Длина внешней петли лежит в диапазоне от minpetlya до maxpetlya (обычно, $minpetlya = 3$, часто maxpetlya между 20 и 200).

3) Количество G-T пар в отрезке не превышает maxGT (обычно, $maxGT = 1$ или 2).

4) Длина каждого промежутка между отрезками не превышает параметра maxvypyach (мы часто брали maxvypyach=0, ограничиваясь, таким образом, простыми спиралями).

Если заданы последовательности консервативных нуклеотидов (их описания называются *боксами*), то дополнительное требование может иметь такой вид:

5) Если некоторая характеристика спирали (например, длина внешней петли) удовлетворяет таким-то ограничениям, то в такой-то окрестности такого-то плеча спирали лежит такой-то бокс. Отметим, что бокс — это описание-слово, в котором могут быть не только буквы A,G,T,C, но и любая их дизъюнкция. Кроме того, для каждого бокса может быть задано максимальное число возможных нарушений в нем, например бокс A(G/C)GA с одним нарушением может иметь вид ATGA. Боксы известны частью заранее из биологических соображений, а частью могли бы искаться до работы описываемого здесь алгоритма, например, каким-либо алгоритмом множественного выравнивания исходного набора нуклеотидных последовательностей.

Сначала строятся простые спирали. Они берутся максимально продолженными в обе стороны за исключением случая, когда продолжение на очередную G-T пару нарушает ограничение на количество G-T пар, а также, возможно, случая, когда G-T пара находится с краю. (Конечно, в случае необходимости можно обрезать спираль, используя энергетические соображения.) Затем, если maxvypyach>0, то в спирали объединяются простые спирали, расположенные относительно друг друга так, что их левые и, соответственно, правые плечи отстоят друг от друга на расстояние не более maxvypyach.

В процессе составления списка спиралей подсчитывается энергия каждой спирали. В простейшем случае она равна сумме энергий отрезков минус штрафы за выпячивания и внутренние петли. Энергия же каждого отрезка равна сумме чисел, по одному для каждой пары соседних пар склеенных нуклеотидов в отрезке. (Вообще говоря, эти числа зависят от температуры, но энергия стандартно берется при температуре 37 градусов.) Для алгоритма важны лишь позиции четверки A, B, C, D концов плеч, а не внутреннее строение спирали. Поэтому для каждой возможной такой четверки в списке оставляется лишь одна спираль с этими краями, имеющая максимальную энергию. После составления всех списков спиралей, спирали в каждом списке упорядочиваются по возрастанию энергии и список обрезается с конца до заданного размера.

**Этап 2. Составление базы данных сходств спиралей и боксов.** База сходств спиралей хранит для каждой пары спиралей $(H_1, H_2)$ число $S(H_1, H_2)$, отражающее степень их сходства, которое подсчитывается следующим образом. Пусть $A_i$, $D_i$ — внешние концы соответственно левого и правого плеч спирали $H_i$, а $B_i$, $C_i$ — внутренние. Тогда для каждой из спиралей $H_i$ берутся два слова: $W_{il} = [A_i - O_1, B_i + O_2]$ и $W_{ir} = [C_i - O_3, D_i + O_4]$ — плечи спиралей вместе с некоторыми их окрестностями. Если maxvypyach>0, то в алгоритме предусмотрена возможность вырезания из этих слов всех выпячиваний или, наоборот, вырезания всех склеенных участков. Размеры окрестностей, т.е. числа $O_1$, $O_2$, $O_3$, $O_4$, задаются как параметры (обычно, они не более 15). Полагаем $S(H_1, H_2) = S'(W_{1l}, W_{2l}) + S'(W_{1r}, W_{2r})$, где $S'$ — функция сходства двух слов. Она подсчитывается выравниванием этих слов, производимым известным алгоритмом Смита-Уотермена; точнее, тем вариантом этого алгоритма, в котором ищутся два наиболее близких в смысле выравнивания подслова данных слов. Если подсчитанное таким образом сходство оказывается меньше некоторого заданного порога (параметр limit), то оно заменяется на сильно отрицательное число (у нас равное -1000). Иначе, это сходство корректируется наложением штрафа за различие длин leng1 и leng2 внешних петель спиралей $H_1$ и $H_2$ (а также, возможно, штрафов просто за длинные внешние спирали, за различия длин плеч, призов и штрафов за наличие или отсутствие консервативных нуклеотидов в определенных местах около этих плеч и т.п.).

В случае, когда известны боксы, кроме базы сходств спиралей составляется база боксов, хранящая координаты положения боксов в последовательностях (координатой бокса считается полусумма номеров позиций его начала и конца), а также указание, на каком расстоянии влево или вправо от такой-то спирали находится ближайший такой-то бокс. Эту базу можно составлять вместе с составлением списков спиралей. В качестве еще одного параметра нужно задавать функцию на боксах, значение которой равно призу за выравнивание этих двух боксов друг к другу (см. этап 4б).

**Этап 3. Предварительное прореживание списков спиралей.** Для каждой спирали $H$ подсчитывается число $k$ списков (кроме списка, содержащего $H$), в которых имеется хоть одна спираль, имеющая с $H$ положительное сходство по базе (т.е. сходство не менее limit по алгоритму Смита-Уотермена; это число можно вычислять одновременно с составлением базы). Если $k$ мало, то $H$ удаляется из списка.

**Этап 4. Выравнивание последовательностей из плеч (спиралей) и боксов (нуклеотидных последовательностей).** Каждый список спиралей преобразуется в список из их плеч, причем все плечи упорядочиваются по их координате. Координатой плеча считается полусумма номеров позиций его концов или (если искомая структура содержит перекрытия) номер позиции конца, примыкающего к внешней петле. Если боксы учитываются, то и они добавляются в этот упорядоченный список (напомним, что координатой бокса считается полусумма позиций его концов). Элементы с совпадающими координатами упорядочиваются произвольным образом. Получившиеся слова из плеч и боксов выравниваются друг относительно друга. Возможны два следующих варианта выравнивания.

а) Множественное выравнивание. Как известно, практические алгоритмы множественного выравнивания являются эвристическими в том смысле, что они не гарантируют оптимизации какого-либо функционала. Однако на хорошо обусловленных данных эти алгоритмы позволяют получать каче-

ственные результаты. В данном случае их задача состоит в таком выравнивании, чтобы любой столбец состоял (за исключением пробелов) только из левых плеч или только из правых, или только из соответствующих боксов. Если столбец состоит из плеч, то любые два из них должны иметь положительное сходство. Желательно, чтобы среди столбцов из плеч выделялись такие, которые имеют небольшое число пробелов, и чтобы эти хорошие столбцы распадались на пары левых и соответствующих им правых плеч.

б) Попарное выравнивание. В алгоритме предусмотрено некоторое фиксированное число итераций, внутри каждой из которых происходит выравнивание каждой пары последовательностей плечбоксов относительно друг друга. Естественно, левым плечам разрешается выравниваться только с левыми (к тому же их сходство должно быть положительным), правым — с правыми, а боксам — с боксами (напомним, что имеется функция сходства на боксах, которая соответствует функции сходства на спиралях). В списках спиралей сначала присутствует много "лишних" спиралей, поэтому сначала штраф за делецию плеча делается нулевым или малым (например, 0.1 от энергии соответствующей спирали), однако с увеличением номера итерации штраф увеличивается. Штраф за делецию бокса зависит от оценки вероятности случайной встречи такого бокса, обычно он невелик. Желательное условие: если, например, левое плечо L1 выровнялось с некоторым левым плечом L2, то соответствующее ему правое плечо P1 выровнялось бы именно с соответствующим плечом P2 — не налагается. Это связано с трудностью оптимизации качества выравнивания при таком условии. Возможны два варианта попарного выравнивания.

б1) Общий вариант. Здесь нет ограничений на число пар выровненных плеч и боксов, на комбинацию выравненных плеч, как "левых" или "правых". Общий вариант применяется, когда не известна топология искомой структуры или нет уверенности, что все ее спирали имеют в соответствующей окрестности достаточно консервативные участки. Выравнивание производится тем же вариантом алгоритма Смита-Уотермена, который упоминался в этапе 2. В программе предусмотрена возможность следующего отклонения от общего варианта. Пусть ищется структура, в которой одна спираль имеет длинную внешнюю петлю, внутри которой расположены несколько спиралей с относительно короткими внешними петлями (такую структуру назовем простейшей; например, таковы структуры клеверного листа или RFN). Тогда можно указать предполагаемую границу (параметр razdel) между длиной петель первого и второго типа. Теперь, если выровнялись два левых плеча, то выравнивание соответствующих правых плеч производится лишь в том случае, если либо
а) длины обеих петель меньше, чем razdel, и внутри петель выровнялись не более заданного числа (например, нуля) спиралей, расположенных последовательно друг относительно друга либо
б) длины обеих петель не меньше, чем razdel, и внутри петель выровнялись не менее заданного числа спиралей, расположенных последовательно друг относительно друга (программа допускает различные уточнения понятий "в петлях", "последовательно" и "выровнялись").
Отметим еще, что в процессе выравнивания подсчитывается и выводится в файл результатов число пар спиралей, выровнявшихся как левыми, так и правыми плечами. Это число обычно отражает качество выравнивания и помогает в подборе параметров алгоритма.

б2) Специальный вариант. В этом варианте заранее фиксируется слово $T$ в алфавите $\{l, r\}$, указывающее на расположение в искомой структуре левых и правых плеч (как они должны идти в структуре слева направо). Так, клеверному листу соответствует слово llrlrlrr, структуре типа RFN — слово llrlrlrlrlrr и т.д. Производится такое выравнивание, у которого расположение левых и правых плеч таково, как в заданном слове $T$. Если боксы учитываются, то для каждого из них указывается, между какими плечами он расположен, и задается максимально возможное расстояние от него до соседних плеч или боксов — в этом случае слово $T$ состоит не только из букв $l$, $r$, но и из боксов.

Кратко опишем алгоритм попарного выравнивания в случае специального варианта. Внутри циклов по началам выравниваемых слов (по их буквам $k$ и $l$) присутствует еще цикл по началам слова $T$ (по буквам $t$) и индуктивно подсчитывается функция $f(k, l, t)$ от трех аргументов (а не от двух,

как обычно). Если $k$ и $l$ — плечи типа, соответствующего $t$-ой букве слова $T$, то проверяются все требования относительно присутствия на заданных расстояниях слева и справа от них определенных боксов. Если эти требования удовлетворяются (а также в случае, когда $k$ и $l$ — боксы нужного типа) полагаем $f(k,l,t)$ равным наибольшему из четырех чисел: 1) $f(k-1,l-1,t-1)$ плюс сходство $k$ и $l$, 2) $f(k-1,l,t)$ минус штраф за делецию $k$, 3) $f(k,l-1,t)$ минус штраф за делецию $l$, 4) ноль. Если же что-либо не соответствует $t$-ой букве слова $T$ или не удовлетворяются требования на присутствие боксов, то полагаем $f(k,l,t)$ равным наибольшему из трех чисел: 1) $f(k-1,l,t)$ минус штраф за делецию $k$, 2) $f(k,l-1,t)$ минус штраф за делецию $l$, 3) ноль. Максимальное по всем $k$ и $l$ значение $f(k,l,length(T))$ будет равно качеству искомого выравнивания, а само это выравнивание находится обратным ходом (в процессе вычисления $f$ поставим пометки о том, какой именно случай дал максимальное значение $f$ для каждой тройки).

**Этап 5. Подсчет качеств спиралей.** После проведения всех итераций выравниваний подсчитывается качество каждой спирали. Это — число, отражающее то, как часто плечи какой-то спирали выравнивались с другими плечами другой спирали (чтобы удобнее его подсчитывать, еще в процессе выравниваний заполняется база, хранящая соответствующую информацию). Параметр liber указывает, в скольких первых итерациях качество подсчитать либерально, когда приз дается за частичное выравнивание спиралей (выравнивание только по одному плечу). Эта надбавка равна взятому из базы сходству двух спиралей; в этом случае за полное выравнивание дается в два раза большая надбавка. Часто мы задавали liber, равным нулю. Еще один приз дается за каждый “треугольник”, когда две спирали, полностью выровненные с рассматриваемой спиралью $H$, полностью выровнялись и между собой. Она равна среднему сходству спирали $H$ с этими двумя спиралями. После подсчета качеств всех спиралей в каждой последовательности $s$ считается сумма качеств всех рассмотренных спиралей из $s$. Часто ее можно рассматривать как качество самой последовательности, и, таким образом, выделять “мусорные” последовательности.

Спирали с качеством меньше заданного удаляются из списков (возможно и удаление самих последовательностей с низким качеством). Оставшиеся спирали принимают участие в следующей итерации выравниваний, после которой качества спиралей подсчитываются заново. Программа позволяет выполнять любое заданное количество итераций, но тестирование показало, что обычно достаточно двух итераций.

После завершения всех итераций составляется общий список оставшихся спиралей (длину его можно ограничивать), упорядоченных по убыванию качества. Отметим, что по количеству удаленных и оставшихся спиралей удобно подбирать значения параметров limit и kolraz. Как уже отмечалось, часто достаточно найти ранжирование спиралей в оставшихся последовательностях по вероятности их вхождения в искомую вторичную структуру. В этом случае можно ограничиться описанными пятью этапами. Заметим, что псевдоузлы в искомой структуре не влияют на описанные выше этапы работы алгоритма.

**Этап 6. Построение структур.** На каждой последовательности структура строится независимо от других последовательностей нашей модификацией алгоритма Нуссинов-Джекобсона. Как известно, этот алгоритм строит в данной нуклеотидной последовательности наиболее мощную структуру (в смысле количества пар склеенных нуклеотидов или в смысле энергии). Наша модификация состоит в следующем. Во-первых, первичными элементами структуры являются не нуклеотиды, а плечи спиралей, тогда сама структура — это некоторое подмножество множества спиралей. Во-вторых, вместо наиболее мощной структуры в простейшем случае мы строим структуру с максимальной суммой качеств входящих в нее спиралей. В более сложном случае возможен также учет их энергии, взаимного расположения и расстояний между ними и т.д.

Таким образом, алгоритм работает на последовательности (упорядоченных по их координате) плеч спиралей, которые не были удалены из соответствующего списка на предыдущих этапах. Координатой плеча считается его внешний конец (либо середина), если искомая структура не содержит

перекрытий, и его внутренний конец, если она содержит перекрытия. Как известно, в алгоритме Нуссинов-Джекобсона структура строится на каждом участке исходной последовательности индукцией по длине этого участка. В нашей модификации индуктивный шаг становится проще: если концы $i$, $j$ отрезка $[i, j]$ — это плечи какой-то одной спирали $H$, то искомая структура на данном отрезке состоит из спирали $H$ и спиралей той структуры, которая была построена на отрезке $[i+1, j-1]$. Перебор случаев здесь отсутствует, так как в последовательности из плеч каждое плечо может спариться лишь с одним плечом (в отличие от случая нуклеотидной последовательности).

В нашем алгоритме предусмотрена возможность отступления от указанного варианта, аналогичная той, которая упоминалась в этапе попарного выравнивания. А именно, для простейших структур можно указать такое число $c > 1$, что концы $i$ и $j$ отрезка будут спариваться только, если структура, построенная на отрезке $[i + 1, j - 1]$ пустая или состоит из не менее, чем $c$ спиралей. Если желательно учитывать взаимное расположение спиралей и расстояния между ними, то мы использовали здесь алгоритм, подобный описанному, например в [2, pp. 160-165], алгоритму построения структуры с минимальной энергией; который применяли, естественно, к последовательностям плеч. Для этого в последовательности плеч указываются интервальные расстояния между соседними плечами. В нашем случае этот алгоритм также существенно упрощается. Например, в его классическом варианте подсчет энергии $VBI(i, j)$ оптимальной структуры на отрезке $[i, j]$, где пара $(i, j)$ предполагается склейкой, крайней к некоторому выпячиванию (или внутренней петле) с дальней от внешней петли стороны, дает оценку четвертой степени (на время работы всего алгоритма): $VBI(i, j) = min_{i<i'<j'<j}[eL(i, j, i', j') + V(i', j')]$(если не ограничиваться формулами специального вида для подсчета энергии внутренних петель, см. [2, pp. 166-178]). Здесь $eL(i, j, i', j')$ — энергия внутренней петли (или выпячивания), ограниченного (склеенными) парами $(i, j)$ и $(i', j')$, $V(i, j)$ — энергия оптимальной структуры на отрезке $[i, j]$, при условии, что $i$ и $j$ спарены. Отметим, что здесь мы не накладываем никаких ограничений на размеры выпячиваний и внутренних петель; таким образом, в категорию выпячиваний и внутренних петель попадает любая пара промежутков между соседними простыми спиралями структуры, где одна спираль расположена во внешней петле другой. В нашем же случае не нужно перебирать пары $(i', j')$, достаточно перебрать левые плечи $i'$, где $i < i' < j$, пользуясь тем, что для каждого плеча $i'$ возможно лишь одно парное плечо $j'$. Таким образом, время нашего алгоритма ограничено третьей степенью от длины последовательности плеч (при условии, что энергия мультипетли (т.е. отрезка $[i, j]$ со спаренными концами, на котором последовательно расположены более одной спирали, не находящейся во внешней петле другой спирали этого отрезка) подсчитывается способом, аналогичным, описанному в [2, p. 165]).

В конце этого этапа мы выводим в файл результатов построенные структуры. При этом для каждой спирали указываются: номера позиций концов всех ее отрезков, длины плеч без учета G-T пар, ее энергия и подсчитанное на предыдущем этапе ее качество. В случае, если применялся специальный вариант алгоритма, то для каждого плеча спирали указывается также средний номер того места, которое это плечо занимало при выравнивании среди пар выровненных плеч (это проясняет расположение спирали в структуре).

**Этап 7. Построение консенсусной структуры.** Под *консенсусной структурой* обычно понимается некоторая абстрактная вторичная структура вместе с частичными отображениями множества ее спиралей в множество спиралей каждой построенной алгоритмом вторичной структуры. Однако, в соответствии с идеологией алгоритма, нам удобнее находить консенсусную структуру как состоящую не из спиралей, а из плеч. Теоретически это может, конечно, привести к возникновению “уродливой” структуры, в которую входит лишь одно плечо некоторой спирали или в которой плечи одной спирали отображаются в плечи разных спиралей. Но, как нам представляется, такие явления будут свидетельствовать скорее о плохой обусловленности исходных данных, чем о недостатках алгоритма. Если применялось множественное выравнивание, то столбцы с достаточно большим числом плеч и образуют консенсусную структуру. Если же применялось попарное выравнивание, то консенсус-

ную структуру приходится строить (хотя в специальном варианте алгоритма прояснить ее могут уже средние номера мест плеч, упоминавшиеся ранее). Для построения консенсусной структуры спирали всех построенных структур (точнее, их плечи) поступают на дополнительную итерацию выравниваний. По ее результатам строится $n$-дольный граф $G$, в котором вершины — это плечи и два плеча соединены ребром, если они выравнивались друг с другом при соответствующем выравнивании. Естественно предполагать, что плечам консенсусной структуры соответствуют достаточно плотные подграфы в $G$. Для поиска плотных подграфов мы применяли алгоритм, разработанный А.В. Селиверстовым. Основная идея этого алгоритма в следующем. Из данного графа последовательно удаляются вершины, соединенные ребрами с небольшим числом долей, и удаляются ребра, принадлежащие небольшому числу треугольников и тетраэдров. Эти вершины и ребра заведомо не могут лежать в плотном подграфе. Если таких вершин или ребер нет, то удаляется вершина, принадлежащая наименьшему числу неупорядоченных пар различных треугольников с общим ребром. При каждом из таких удалений проверяются на плотность все подграфы, состоящие из вершины степени $q$ (где $q$ — фиксированный параметр) и всех смежных с ней вершин.

**Модификация этапа 6 для псевдоузлов.** При наличии псевдоузлов этап 6 нуждается, конечно, в изменении. Рассматрим здесь лишь случай максимизации суммы качеств спиралей (без учета их взаимного расположения и расстояний между ними). Тогда мы применяли следующую модификацию.

Определим понятие $p$-допустимой структуры, где $p$ — параметр, понимаемый, как максимальная степень "псевдозаузленности" структуры. Это определение построено так, чтобы любое множество спиралей при некотором значении параметра $p$ являлось $p$-допустимой структурой.

**Определение.** *Допустимой структурой на отрезке* $[i, j]$ (в последовательности плеч) назовем структуру, которая может быть построена последовательным применением любого конечного числа раз и в любом порядке операций, описанных ниже в пунктах 1, 2, 3, 4 (отметим, что в нашем случае любая структура допустима). *Степень псевдозаузленности*, обозначаемая $PZ$, приписывается такой структуре как наименьшая $PZ(C)$, которая ей приписывается для каждого такого построения $C$. При этом $PZ(C)$ определяется ниже индуктивно вместе с описанием самих операций. *$p$-допустимой структурой* назовем такую допустимую структуру, у которой степень псевдозаузленности не более, чем $p$. Знак объединения обозначает операцию объединения над множествами спиралей, из которых состоят структуры.

0) Степень псевдозаузленности пустой структуры по определению равна нулю.

1) *Операция замыкания*: если $i$ и $j$ — плечи одной спирали $H$ и уже построена структура $S$ на отрезке $[i+1, j-1]$, то $S \cup H$ — структура на $[i, j]$ и $PZ(S \cup H) = PZ(S)$.

2) *Операция последовательного соединения*: если уже построены структуры $S_1$ и $S_2$ на отрезках соответственно $[i, k]$ и $[k+1, j]$, то $S = S_1 \cup S_2$ — структура на $[i, j]$ и $PZ(S) = max(PZ(S_1), PZ(S_2))$.

3а) *Операция одинарного псевдозаузливания слева*: если уже построена структура $S$ на отрезке $[i+1, j]$ и парное к плечу $i$ плечо $i'$ ему принадлежит, то $S' = S \cup H$ — структура на $[i, j]$, где $H$ — спираль $(i, i')$, и $PZ(S') = PZ(S) + 1$.

3б) *Операция одинарного псевдозаузливания справа*: если уже построена структура $S$ на отрезке $[i, j-1]$ и парное к плечу $j$ плечо $j'$ ему принадлежит, то $S' = S \cup H$ — структура на $[i, j]$, где $H$ — спираль $(j', j)$, и $PZ(S') = PZ(S) + 1$.

4) *Операция двойного псевдозаузливания*: если построена структура $S$ на отрезке $[i+1, j-1]$ и парные к плечам $i$ и $j$ плечи $i'$ и $j'$ ему принадлежат, то $S' = S \cup H_1 \cup H_2$ — структура на $[i, j]$, где $H_1$ — спираль $(i, i')$, $H_2$ — спираль $(j', j)$, и $PZ(S') = PZ(S) + 2$. (Конец определения.)

Итак, пусть задана максимальная степень псевдозаузленности $p$ некоторой структуры. В исходном алгоритме строится наилучшая структура на каждом отрезке $[i, j]$ по возрастанию длины этого отрезка. Теперь кроме параметров $i$ и $j$ введем новый параметр: на отрезке $[i, j]$ ищется наилучшая структура $S$ со степенью псевдозаузленности не более, чем $p'$ (где $p' \leq p$ и при фиксированных $i$ и $j$ параметр $p'$ перебирается по возрастанию). Перебор того, какая операция была самой последней при

построении искомой структуры, доказывает, что следующее алгоритмическое решение является правильным. Если $i$ и $j$ — плечи одной спирали $H$, то последней могла быть лишь операция замыкания. В этом случае по существу никаких изменений в исходном алгоритме не должно быть — спираль $H$ добавляется к (уже построенной) структуре с параметрами $(i+1, j-1, p')$. Иначе, последней могла быть как операция последовательного соединения, так и одинарного псевдозаузливаия слева (если $i'$ лежит в $[i, j]$) или справа (если $j'$ лежит в $[i, j]$), или двойного псевдозаузливания (если $i'$ и $j'$ лежат в $[i, j]$). В любом случае структуры на меньших отрезках, из которых получается $S$, являются лучшими среди тех, у которых степень псевдозаузленности не превышает соответственно $p'$ (для последовательного соединения), $p'-1$ (для одинарного псевдозаузливания) и $p'-2$ (для двойного псевдозаузливания). По предположению индукции эти структуры уже построены, и нужно из всех возможных вариантов лишь выбрать наилучший.

## СПИСОК ЛИТЕРАТУРЫ

1. Vitreschak A.G., Rodionov D.A., Mironov A.A., Gelfand M.S. Regulation of bacterial riboflavin genes by a conserved RNA structural element. *Proceedings of the Third International Conference on bioinformatics of genom regulation and structure*, vol. 2, 2002, pp. 44-46, IC&G, Novosibirsk.
2. Christian N.S. Pedersen. Algorithms in Computational Biology. Pn.D. Dissertation, 2000. (http://www.brics.dk)