Российская Академия наук

Институт проблем передачи информации РАН

ЛЮБЕЦКАЯ ЕЛЕНА ВАСИЛЬЕВНА

МАССОВЫЙ ПОИСК АТТЕНЮАТОРНОЙ РЕГУЛЯЦИИ В ГЕНОМАХ ПРОТЕОБАКТЕРИЙ

05.13.17 – Теоретические основы информатики,

03.00.28 – Биоинформатика

ДИССЕРТАЦИЯ

на соискание ученой степени

кандидата физико-математических наук

Научный руководитель –

член-корреспондент РАН,

доктор биологических наук,

профессор Л.М. Чайлахян

Москва – 2004

СОДЕРЖАНИЕ

ВВЕДЕНИЕ. ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. Текущие концентрации молекул в клетке в значительной мере зависят от протекающих биохимических реакций в ней (в работе рассматривается случай прокариот). Как правило, реакция протекает по мере поступления в цитоплазму клетки соответствующего набора ферментов, что зависит от экспрессии соответствующих групп генов (регулонов и их полицистронных случаев – оперонов). Таким образом, текущая жизнь клетки в значительной степени состоит в регуляции групп генов в зависимости от ее внутренних и внешних условий жизнедеятельности. Известны разные типы регуляции: основанные на белок-ДНКовом взаимодействии (позитивная или негативная, когда активатор или репрессор связывается с соответствующим сайтом в лидерной области оперона, обычно расположенным внутри промотера или вблизи него); или основанные на образовании специфических вторичных структур мРНК (например, альтернативных и, в частности, аттенюаторных – в последнем случае механизм регуляции зависит от взаимного расположения РНК-полимеразы и рибосомы в параллельно идущих процессах транскрипции и трансляции); регуляция аллостерическая (когда конечный продукт катализируемой ферментами реакции ингибирует работу одного из ферментов) и другие. В процессах мРНКовой регуляции большую роль играют: лидерный пептид с регуляторными кодонами, стабилизирующие белки, тРНК, молекулы-эффекторы и т.п. Часто в управлении одного оперона участвуют несколько разных типов регуляции.

Регуляции с помощью белков-репрессоров или активаторов, а также аллостерическая регуляции изучаются сравнительно давно. Фундаментальная важность альтернативной регуляции обнаружена недавно, когда были найдены новые ее примеры.

В настоящее время расшифровано и доступно более 100 полных геномов, несколько сотен полных геномов секвенируются и будут доступны в ближайшее время, не говоря уже о секвенировании частей геномов. Такой огромный объем информации

делает невозможным лабораторный биохимический анализ подавляющего большинства геномов, поэтому необходимы **алгоритмы** компьютерного анализа геномов и, в частности, *поиска потенциальных аттенюаторных структур* в достаточно полно секвенированных геномах, которые были бы применимы для **массового анализа** сразу всех организмов из данной таксономической группы.

В основном для поиска *регуляторных сигналов* до сих пор применялись **два подхода**: составлялось распознающее правило (по выборке лидерных областей, содержащих достаточно сходные регуляторные сайты); или такой сигнал искался непосредственно в каждой из последовательностей, входящих в выборку, на основе существенной консервативности сигнала. Оба эти подхода плохо применимы в случае массового поиска аттенюаторных структур. Ситуация особенно усложняется, когда речь идет о поиске регуляторного сигнала для генов с неизвестной функцией или для геномов, у которых еще не выяснена структура интересующего нас оперона.

Цель работы. Создание алгоритмов и компьютерной программы для массового поиска аттенюаторной регуляции экспрессии генов. Тестирование эффективности алгоритмов и программы на различных искусственных и биологических данных. Применение этих алгоритмов для решения биологической задачи поиска аттенюаторных сигналов регуляции у бактерий.

Методика исследования. Построение алгоритмов и компьютерных программ для поиска аттенюаторной регуляции в ***одном*** исходном нуклеотидном фрагменте, и затем применение их для массового поиска аттенюаторной регуляции у бактерий. Сначала методы сравнительной геномики применяются для построения предполагаемой оперонной структуры генов (в нашем случае биосинтеза некоторых аминокислот), отсюда в основном вручную выделяются потенциальные регуляторные области. К ним по отдельности применяются разработанные нами программы. Для подтверждения полученных потенциальных аттенюаторных структур для данного оперона у ряда более, а иногда и мало родственных бактерий проводилось выравнивание участ-

ков регуляторных областей, содержащих найденные структуры (с некоторыми полями влево и вправо) так, чтобы оказались выровненными терминаторы, антитерминаторы и паузные шпильки, лидерные пептиды и другие аттенюаторные элементы. После такого подтверждения найденных сигналов (или после указания алгоритма об их отсутствии) уточнялись оперонные структуры в геномах и проводился окончательный анализ соответствующего метаболического пути биосинтеза аминокислот на его полноту и непротиворечивость.

Таким образом, был проведен массовый поиск аттенюаторной регуляции в протеобактериях и в некоторых группах грамположительных бактерий.

Научная новизна. Предложенные алгоритмы – одни из первых для поиска регуляторных аттенюаторных сигналов по одной исходной нуклеотидной последовательности. Алгоритмы были реализованы в виде программного приложения, разнообразно тестированы и применены в задаче поиска сигналов аттенюаторной регуляции в геномах протеобактерий и грамположительных бактерий. Для последних эта задача до сих пор не рассматривалась.

Основные результаты. В диссертации получены следующие основные результаты:

- Предложены и реализованы в виде компьютерной программы, *называемой далее* **LLLM**, алгоритмы построения потенциальных структур аттенюаторной регуляции в геномах бактерий.
- Показана практическая эффективность и надежность созданной программы LLLM на основе ее детального тестирования на искусственных и биологических нуклеотидных последовательностях.
- Проведен массовый поиск и во многих случаях найдены потенциальные сигналы аттенюаторной регуляции (а в иных случаях алгоритм указал на предположительную причину их отсутствия) у гамма-, альфа- и бета-протеобактерий (биосинтез разветвленных и ароматических аминокислот, гистидина и треонина, фенилаланил-тРНК-синтетазы), а также и у грамположительных бактерий: из групп

тРНК-синтетазы), а также и у грамположительных бактерий: из групп Bacillales, Lactobacillales, Clostridiales, Bacteroidetes/Chlorobi и Thermotogales (биосинтез гистидина); описана эволюционная динамика аттенюаторной регуляции транскрипции.

- Установлены потенциальные оперонные структуры для генов биосинтеза некоторых аминокислот (триптофана, фенилаланина, треонина, гистидина и разветвленные аминокислот) в различных геномах.

- Предсказано новое семейство гистидиновых транспортеров – ортологов *yuiF* у *B. subtilis* (например, HI0325 у *H. Influenzae*) и два гистидиновых транспортера BC0629 у *B. cereus* (ортолог *yvsH* у *B. subtilis*) из белкового семейства APA и новый ген у *L. lactis* (ортолог *lysQ* у *E. coli*) из семейства APC.

- Получена предположительная функциональная аннотация ряда генов, кодирующих ферменты и находящихся под аттенюаторной регуляцией, а именно:

  - гену *ygeA* у *Pasteurella multocida* приписана функция рацемазы разветвленных аминокислот;

  - не ортологичные гены *vatB*, *actX2* и *actX3*, соответственно, у *Pasteurella multocida*, *Mannheimia haemolytica*, *Polaribacter filamentus* кодируют ацетилтрансферазы, участвующие в метаболизме гистидина.

- Показано, что биосинтез изолейцина у Xanthomonadales использует треонин дегидратазу TdcB, в отличие от IlvA у *E. coli*.

- Показано, что у Pasteurellales бифункциональный ген *thrA* аспартат киназы/гомосерин дегидрогеназы регулируется не только треонинином и изолейцином (как это имеет место у *E. coli*), но и метионином.

- Предсказано, что у альфа-протеобактерий ацетолактат синтаза IlvIH регулируется аттенюацией с регуляторными кодонами лейцина, изолейцина и валина.

- Предсказано, что оперон *his* биосинтеза гистидина регулируется гистидин-зависимыми аттенюаторами у *Bacillus cereus* и *Clostridium difficile*, но и в тоже время регулируется гистидиновыми T-боксами у *Lactococcus lactis* и *Streptococcus mutans*.

Хорошее выравнивание этих биологических предсказаний рассматривается нами как еще одно подтверждение правильности работы программы LLLM.

Теоретическая и практическая ценность. До недавнего времени алгоритмы для поиска потенциальных альтернативных структур *в одной регуляторной области* не предлагались (наши алгоритмы излагаются в Главе 1). Публикации автора по таким алгоритмам были одними из первых работ, сравнение наших алгоритмов с немногочисленными другими приводится в следующем пункте. Исследование аттенюаторной регуляции в классе протеобактерий начато сравнительно недавно, см., например, работы[1,3]; а проведенный нами массовый поиск в классе грамположительных бактерий является первым. Нами предсказаны (Глава 3) новые регуляторные сигналы этого типа (включая лидерные пептиды и регуляторные кодоны, терминаторы и часто антитерминаторы и паузные шпильки) у гамма-, альфа- и бета-протеобактерий, у фирмикутов из групп Bacillales, Lactobacillales и Clostridiales, у бактерий из групп Bacteroidetes/Chlorobi и Thermotogales. Соответствующие выравнивания нуклеотидных участков исходных последовательностей показали хорошую согласованность между собой предсказанных нами регуляторных сигналов. В Главе 2 приводятся результаты систематического тестирования наших алгоритмов для аттенюаторных и Т-бокс структур. В частности, нами были независимо найдены все ранее известные случаи аттенюаторной регуляции в классе гамма-протеобактерий[1,2].

Работы, содержащие алгоритмы или методы поиска аттенюаторных сигналов. В работе [1] рассмотрен метод поиска консервативной вторичной структуры РНК с помощью двух программ FOLDALIGN и COVE, описание которых дано в предыдущих статьях этих авторов. Для выравнивания нуклеотидных последовательностей без учё-

[1] Panina, E.M., Vitreschak, A.G., Mironov, A.A. and Gelfand, M.S. (2001) Regulation of aromatic amino acid biosynthesis in gamma-proteobacteria. J. Mol. Microbiol. Biotechnol. 3, 529-543.

[2] Landick, R., Turnbough, C.L. and Yanovsky, C. (1994) Transcriptional attenuation. In: Escherichia coli and Salmonella. Cellular and molecular biology (Neidhardt, F.C., Ed.), p. 1263-1286. American Society for Microbiology, Washington, DC.

та вторичной структуры РНК ими использовалась программа CLUSTAL_W. Однако, такой метод часто не позволяет выделить одно выравнивание среди многих вариантов, близких по качеству. С другой стороны, существуют программы для поиска консервативных вторичных структур РНК в наборе последовательностей, заранее выровненных по нуклеотидному составу. Одна из таких программ – COVE. Она основана на методе представления вторичных структур с помощью стохастических контекстно-свободных грамматик SCFG. Особенностью этого метода является то, что алгоритм ищет общее (глобальное) выравнивание на всём протяжении исходных последовательностей. В действительности, консервативная структура, имеющая биологическое значение может располагаться лишь на небольших, но неизвестных участках исходных РНК. Поэтому применение алгоритмов, основанных на стохастических контекстно-свободных грамматиках эффективно лишь после предварительной обработки данных. Хотя теоретически такое выравнивание можно искать в любом данном наборе РНК, ответ обычно не удовлетворителен с биологической точки зрения.

Программа FOLDALIGN является расширением программы типа CLUSTAL_W. Она позволяет выравнивать последовательности с учётом как нуклеотидного состава, так и вторичной структуры РНК. Программа FOLDALIGN сначала проводит попарное выравнивание последовательностей, используя метод динамического программирования. Затем проводится поиск множественного выравнивания, при котором множество выровненных последовательностей постепенно увеличивается, а новые последовательности сравниваются с консенсусом уже построенного выравнивания. Но программа FOLDALIGN учитывает лишь локальную структуру последовательностей. Поэтому найденная FOLDALIGN консервативная вторичная структура РНК может оказаться неудачной в целом. Результаты «локального» выравнивания программой FOLDALIGN и «глобального» программой COVE могут значительно отличаться. Локальный характер работы программы FOLDALIGN связан с тем, что она работает очень медленно и не пригодна для больших объёмов данных.

Новый метод основан на комбинации двух программ FOLDALIGN и COVE. Сначала происходит поиск хорошего выравнивания на некоторой части входных последовательностей с помощью локального метода FOLDALIGN. Потом это выравнивание уточняется программой COVE. При этом выборка хорошо выровненных последовательностей может расшириться. К ней можно вновь применить FOLDALIGN и т.д. В результате выравниваются все исходные последовательности. И это выравнивание учитывает как нуклеотидный состав, так и вторичную структуру РНК в целом. Проверка такого метода была проведена на большой выборке рибосомальных РНК, а также на на ферритин IRE-элементах, имеющих нетривиальную консервативную вторичную структуру.

В работе [2] описан алгоритм для поиска вторичной структуры одной РНК, если она аналогична уже известной вторичной структуре некоторой РНК без псевдоузлов. Алгоритм основан на методе представления вторичных структур с помощью стохастических контекстно-свободных грамматик SCFG. Емкостная сложность этого алгоритма равна $O(N^2*\log N)$. Его проверка была проведена на большой выборке рибосомальных РНК.

В работе [3] описаны свойства терминаторов (GC-богатых шпилек с полем остатков урацила) и их роль в регуляции транскрипции. Различаются два типа регуляций, вовлекающая терминаторы: аттенюация, в которой терминатор расположен перед геном, и антитерминация, в которой терминатор расположен после гена. Статья носит обзорный характер и не содержит каких-либо алгоритмов.

Автору был доступен неопубликованный алгоритм множественного выравнивания Миронова, описание которого выполненное автором приводится ниже для полноты картины. Пусть дан набор нуклеотидных последовательностей. Фиксируем некоторое число *l*.

Колонкой ширины l в этом наборе назовем выборку, содержащую не более одного подслова длины *l* в каждой из этих последовательностей. *Представителем ко-*

лонки C в какой-то одной из этих последовательностей назовем подслово из этой последовательности, входящее в *C* (если его нет, то говорим, что *представитель пустой*). *Качеством колонки* назовем какую-то фиксированную меру того, насколько её слова попарно похожи друг на друга, а также, возможно, и того, сколь велико число слов в ней (например, это – взятая со знаком минус сумма энтропий вероятностных распределений на буквах, задаваемых столбцами этой колонки). *Сходством двух колонок* назовём какую-то фиксированную меру того, насколько слова одной колонки похожи на слова другой (например, это – взятая со знаком минус сумма информационных дивергенций распределений на соответствующих столбцах; фактически нам понадобится лишь случай, когда одна из этих двух колонок состоит из одного и того же слова). Скажем, что в последовательности «*одно подслово лежит левее другого подслова*» той же длины, если начало первого подслова строго левее начала второго подслова. Скажем, что «*колонка C_1 расположена целиком левее колонки C_2*», если в любой последовательности, где обе эти колонки имеют непустых представителей, представитель для C_1 лежит левее представителя для C_2 (таким образом, два представителя могут перекрываться, но не могут совпадать). *Расстоянием между подсловами* одной последовательности назовём расстояние между их началами. *Расстоянием между двумя колонками*, одна из которых лежит целиком левее другой, назовём среднее расстояние между их представителями по тем последовательностям, в которых оба представителя непустые. *Согласованностью* упорядоченной пары $<C_1,C_2>$ колонок назовём какую-то фиксированную меру, отражающую, насколько мал разброс расстояний между их представителями в тех последовательностях, где представитель C_1 лежит левее представителя C_2 (например, взятую со знаком минус дисперсию этих расстояний).

Основная часть описываемого алгоритма решает следующую задачу.

Дан набор $S_1, \ldots, S_n$ нуклеотидных последовательностей. Найти упорядоченный слева направо такой набор $C_1, \ldots, C_m$ колонок ширины l (напомним, что l задаётся в качестве параметра, а m не фиксировано и не задано), что:

Для любых $i<j$ колонка C_i лежит целиком левее колонки C_j, и качества колонок и согласованности пар соседних колонок как можно больше (здесь возможны разные формальные уточнения).

Теперь опишем основные этапы этого алгоритма.

Этап 1. Построение начального множества колонок. По каждого подслову w (длины l) из какой-то последовательности S_i строим колонку, в которую сначала включаем слово w из S_i, а из каждой последовательности S_j $(j \neq i)$ включаем наиболее похожее на w слово (если сходство больше некоторого порога). Полученное начальное множество колонок обозначим M_0.

Этап 2. Итерационное уточнение колонок. Каждую колонку C из M_0 уточняем следующим образом. В каждой последовательности S_i находим слово $w(i)$, у которого максимальное сходство с колонкой C (если это сходство меньше некоторого порога, то полагаем $w(i)$ пустым). Затем для каждого i заменяем представителя колонки C в S_i на слово $w(i)$. Получаем новую колонку, для которой применяем ту же процедуру уточнения. После заданного числа итераций отберём из всех рассматривавшихся колонок наилучшую, или несколько наилучших (скорее всего процесс будет сходиться к наилучшей колонке). Среди всех отобранных (при разных C) колонок, оставим лишь те, качество которых не меньше порога. Их множество обозначим M_1.

Этап 3. Построение ориентированного графа на колонках. Строим ориентированный граф G, вершины которого – колонки из M_1. Из колонки C_1 в колонку C_2 проводим ребро, если количество последовательностей, где представитель C_2 лежит левее представителя C_1, не превосходит $\varepsilon * n$, где ε – заданное достаточно малое число. Дополнительно возможна разметка вершин числами, отражающими качества соответствую-

щих колонок, и разметка рёбер числами, отражающими согласованность пар соответствующих колонок.

Этап 4. Поиск максимального пути в орграфе. Ищем в *G максимальный путь* (в случае неразмеченного графа максимизируем число рёбер, составляющих путь, в случае размеченного – сумму числовых пометок на вершинах и рёбрах пути).

Как известно, если в ориентированном графе нет циклов, то максимальный путь ищется обычной процедурой динамического программирования. Если циклы имеются (легко видеть, что в G могут быть лишь циклы длины не меньшей, чем $1/\varepsilon$), то возможны разные эвристические подходы. Например, уменьшать ε или удалять вершины с наименьшим качеством до тех пор, пока не исчезнут циклы. Возможно, также использование какого-либо эвристического алгоритма, допускающего существование циклов. Линейно упорядоченное множество колонок на таким образом найденном пути обозначим M_2.

Этап 5. Удаление противоречивых представителей. Для каждого представителя w каждой колонки из M_2 подсчитаем уровень его противоречивости $U(w)$. Это – число других представителей в той же последовательности, расположение которых относительно w не соответствует отношению линейного порядка на колонках. Затем удаляем представителя с максимальным уровнем противоречивости и пересчитываем функцию U. Продолжаем так до тех пор, пока все представители не будут иметь нулевой уровень противоречивости. Теперь линейный порядок на полученном множестве M_3 соответствует отношению «одна колонка целиком лежит левее другой». Мощность множества M_3 обозначим через m.

Этап 6. Итерационное уточнение набора колонок. Смысл одной итерации в том, чтобы для каждой последовательности S_i решить, нет ли в ней лучшего набора представителей, чем текущий. Для этого строим ориентированный граф G, вершины которого – все подслова из S_i длины l. Проводим ребро из вершины w_1 в вершину w_2, если слово w_1 лежит левее слова w_2 (заметим, что в G нет циклов). Каждую вершину w по-

мечаем набором $<c_1, \ldots, c_m>$, где c_j – число, отражающее сходство слова w с j-ой слева колонкой из M_3. Каждое ребро помечаем набором $<d_1, \ldots, d_{m-1}>$, где d_j – число, отражающее отклонение расстояния между w_1 и w_2 от расстояния между j-ой и (j+1)-ой колонками (взятое со знаком минус).

Ищем в G *максимальный путь*, содержащий ровно m вершин. Точнее, максимизируем сумму пометок $c_1(1)+d_1(1)+c_2(2)+d_2(2)+\ldots+c_{m-1}(m-1)+d_{m-1}(m-1)+c_m(m)$, где в скобках указаны номера вершин и рёбер на пути (от его начала), в которых берутся соответствующие компоненты разметки. Легко видеть, что поиск такого пути может быть проведён простой модификацией стандартной процедуры поиска максимального пути в ориентированном графе. Найденному пути соответствует новый набор представителей колонок в S_i.

После того, как для каждой S_i найден новый набор представителей, заменяем все текущие наборы на новые, получая, таким образом, новый набор колонок. После заданного числа итераций отберём из всех рассматривавшихся наборов колонок наилучший. При постановке задачи мы отмечали, что возможны разные формальные уточнения понятия «наилучший набор»; следует ожидать, что при естественном уточнении процесс должен сходиться к наилучшему набору.

Из окончательного набора колонок выбрасываем представителей, у которых сходство со «своими» колонками ниже порога.

Этап 7 (дополнительный). Слияние близких колонок. Просматривая пары соседних колонок, производим слияние двух соседних колонок в одну, если расстояние между этими колонками мало, а их согласованность велика. Получившиеся в результате слияния колонки будут, конечно, нестандартными, так как их представители могут иметь разную длину (больше l).

Итак, описан алгоритм поиска колонок. Укажем его возможные применения.

Применение 1. Алгоритм может служить основой для поиска консервативных вторичных структур. А именно, для каждой спирали из каждой последовательности S за-

писываем информацию о том, как расположены плечи этой спирали относительно представителей колонок в *S* (достаточно, видимо, указывать номера колонок, ближайших к плечу слева и справа представителей и расстояния до них). После этого каким-либо алгоритмом выделяем в (объединённом) множестве всех спиралей кластеры – подмножества спиралей с похожей информацией (например, можно использовать процедуру, аналогичную этапам 1 и 2 описанного алгоритма). Естественно ожидать, что спирали из одного кластера (возможно, после отбрасывания некоторого числа лишних спиралей) составят набор консервативных однотипных спиралей искомой структуры.

Применение 2. Алгоритм даёт процедуру множественного выравнивания последовательностей, состоящих из объектов произвольной природы. Действительно, замена подслов длины *l* на другие объекты не препятствует работе алгоритма. Его можно применять, например, при выравнивании последовательностей из плеч и боксов в алгоритме поиска консервативных вторичных структур из [4].

Применение 3. Найденные алгоритмом колонки из боксов могут поступать на вход описанного в [4] алгоритма.

Апробация. Результаты диссертации докладывались на:

- 3-ей международной конференции «Проблемы управления и моделирования в сложных системах», Самара, РАН, 4-9 сентября 2001.
- 3rd International Conference on Bioinformatics of Genome Regulation and Structure, BGRS’2002, 14-20 July 2002, Novosibirsk, Russia.
- Moscow Conference on Computational Molecular Biology (MCCMB'03), 22-25 July 2003, Moscow, Russia.
- Научном семинаре по биоинформатике Института проблем передачи информации РАН под руководством профессора, члена-корреспондента РАН Л.М. Чайлахяна.

- Научном семинаре по алгоритмам в геномике Московского государственного университета им. Ломоносова (механико-математический факультет) под руководством профессора В.А. Любецкого.
- Московском семинаре по компьютерной генетике Института молекулярной биологии им. В.А. Энгельгардта РАН.

Публикации. По теме диссертации опубликовано 6 печатных работ.

Структура и объем работы. Диссертация состоит из 3 глав и 3 приложений; последние содержат основные результаты работы предложенных нами алгоритмов. Объем работы 110 страниц машинописного текста, в том числе, 13 таблиц и 26 рисунков.

Глава 1 содержит биологическую и математическую постановки задачи, описание полученных автором алгоритмов поиска аттенюаторной регуляции, краткое описание реализующей их компьютерной программы, сравнение двух полученных алгоритмов между собой.

Глава 2 содержит результаты компьютерного тестирования наших двух алгоритмов на биологических последовательностях, для которых аттенюаторы были известны, и на искусственных последовательностях. Тестирование этих алгоритмов продолжается в Главе 3 на задаче массового поиска аттенюаторной регуляции в обширных классах бактерий.

В главе 3 последовательно рассматриваются результаты массового поиска потенциальных сигналов аттенюаторной регуляции у протеобактерий и фирмикутов, у бактерий из групп Bacteroidetes/Chlorobi и Thermotogales для случаев биосинтеза разветвленных и ароматических аминокислот, гистидина и треонина. Для каждого из этих случаев приводится и кратко комментируется соответствующая потенциальная оперонная структура. Затем приводятся выравнивания найденных РНК-структур.

В конце работы приведены выводы, список использованной литературы, включая работы автора по теме диссертации, и 3 приложения.

ГЛАВА 1. Два алгоритма и компьютерная программа поиска потенциальных аттенюаторных регуляторных структур мРНК

Рассматривается задача поиска аттенюаторной регуляции в лидерной области гена. Математическая постановка этой задачи такова. Дана последовательность фиксированной длины в алфавите {A, C, T, G}; нужно найти определенную уточняемую ниже комбинацию подслов в ней, соответствующую биологическому понятию аттенюаторной структуры. А именно, аттенюаторная структура искалась как состоящая из лидерного пептида с регуляторными кодонами, участка остатков урацила и трех шпилек: терминатора, антитерминатора и паузной, определенным образом расположенных относительно друг друга (см. рис. 1 в Главе 3). К этому традиционному описанию нами было добавлено представление о трех словах – «основах» этих трех шпилек. Наш первый алгоритм реализует идею поиска именно таких трех слов, «порождающих» три соответствующие шпильки; описание этого приведено ниже.

Условимся *обозначать* плечи паузной шпильки 1 и 2, плечи терминатора 3 и 4, а плечи антитерминатора 2' и 3'.

§1.1. Первый алгоритм

Первый алгоритм применяется для исследования вопроса о наличии аттенюаторной регуляции в **одной** данной лидерной области: в ней ищется сама аттенюаторная структура или обоснование ее отсутствия (т.е. ищется алгоритмическое указание на отсутствующий элемент такой структуры). Нами обычно рассматривались случаи, когда лидерная область имеет длину в 450-800 нуклеотидов. Алгоритм ищет *структуру*, состоящую из следующих элементов в исходной нуклеотидной последовательности: рамки считывания, кодирующей лидерный пептид с набором регуляторных кодонов в нем; тройки слов s_1, s_2, s_3, которые служат «основой» для построения плеч: s_1 плеч 2 и 2', s_2 плеч 3 и 3', s_3 плеча 4; и трех шпилек – терминатора, антитерминатора и паузной (в соответствии с этими плечами); участок остатков урацила.

В алгоритме используются следующие основные параметры: минимальная длина лидерного пептида; длина участка, содержащего остатки урацила; минимальное количество самих остатков урацила в этом участке; длина каждого из слов s_1, s_2, s_3; максимальное расстояние (разница между началом первой буквы следующего слова и последней буквой предыдущего слова) между словами s_1 и s_2, s_2 и s_3; максимальное число различных (в паре s_1 и s_3) и некомплементарных (в парах s_1 и s_2, s_2 и s_3) нуклеотидов; максимальная расстояние между началом слова s_3 и концом слова s_2; максимальное число отбираемых алгоритмом троек слов s_1, s_2, s_3; максимальное расстояние между началом участка остатков урацила и концом слова s_3; минимальное отношение числа пар GC к числу пар AT в словах s_2 и s_3; интервальное значение длины фрагмента до начала петли антитерминатора – паузная шпилька ищется в этом фрагменте.

Значения этих параметров подбирались с помощью стандартных приемов статистики, один из них (поиск доверительного интервала с помощью распределения Стьюдента) в качестве примера указан ниже. Для результатов, которые описаны в Главе 3, характерны следующие значения этих параметров. Перечисляются в том же порядке, как выше: 30 нуклеотидов, 5, 8, 100, 2, 30, 15, 9, 1.3, от 100 до 50.

Первый алгоритм последовательно последовательно находит.

1. Кандидатов в лидерные пептиды. Для этого перебираются все открытые рамки считывания, в каждой из которых ищется скопление регуляторных кодонов соответствующей аминокислоты (в процессе счета список этих кодонов может расширяться – он также является параметром алгоритма).

Открытой рамкой считывания считается участок последовательности, удовлетворяющий условиям: 1. Его первый кодон является старт-кодоном (обычно это ATG, реже GTG или TTG); 2. Последний кодон является стоп-кодоном (одним из кодонов TAA, TAG, TGA). 3. Длина участка в нуклеотидах кратна 3 (т.е. старт-кодон, стоп-кодон и кодоны, лежащие между ними, находятся в рамке считывания). 4. Расстояние

между старт и стоп-кодонами не менее *мин_дл_пепт*. Поиск кандидатов выполняется тривиально:

A. В последовательности находим стоп-кодоны. Отступая влево от каждого стоп-кодона на расстоянии, большем *мин_дл_пепт* и кратном 3, ищем старт-кодоны. Исключаем рамки считывания содержащие другие стоп-кодоны.

B. Стоп-кодоны, для которых не найден старт-кодон, отсеиваются. Для каждого стоп-кодона находится множество всех возможных старт-кодонов, из которого выбирается один с наибольшей координатой начала.

C. Для каждой пары старт-кодон и стоп-кодон находящиеся между ними триплеты размечаются как кодоны соответствующих аминокислот.

D. Среди так полученных рамок считывания отбираем кандидатов в лидерные пептиды. Для этого в каждом рамке ищем непрерывное поле соответствующих регуляторных кодонов. Если специфической для данного оперона аминокислоте соответствует малое количество различных кодонов (например, 1-2), то поле может быть более коротким (2 и более кодона). Если аминокислоте соответствует более 2-х кодонов, то ищем поле из 4 и более кодонов. Известно, что некоторые опероны регулируются с помощью лидерных пептидов, содержащих в таком поле специфические кодоны «чужих» аминокислот (например, биосинтез треонина является изолецин-зависимым и т.д.) – в таком случае появление в поле кодонов «чужих» аминокислот не считается разрывом.

Если ни в одной рамке не нашлось поля специфических кодонов, алгоритм выдает сообщение об отсутствии аттенюаторной регуляции.

Для каждого найденного кандидата в лидерные пептиды ищутся все подходящие участки остатков урацила.

Для каждой пары таких объектов ищутся тройки слов s_1, s_2, s_3; тройки сортируются в порядке увеличения расстояния между s_2 и s_3, из них отбирается заданное число первых. Тройку слов $<s_1, s_2, s_3>$ ищем как комбинацию подслов последовательности оди-

наковой длины *длина_слова* (обычно *длина_слова* = 8), находящихся на расстоянии друг от друга не более константы *макс_длина_шпильки* (обычно *макс_длина_шпильки* = 100) нуклеотидов и удовлетворяющих условиям:

1. Слова s_1, s_2 комплементарны с точностью до замены любых *макс_разница* (*макс_разница* < *длина_слова*) нуклеотидов. 2. Слова s_2, s_3 комплементарны с точностью до замены любых *макс_разница* (*макс_разница* < *длина_слова*) нуклеотидов. 3. Слова s_1, s_3 совпадают с точностью до замены любых *макс_разница* (*макс_разница* < *длина_слова*) нуклеотидов.

Для каждого подслова s_1 длины *длина_слова* на отрезке [s_1+ *длина_слова*; s_1+ *длина_слова* + *макс_длина_шпильки*] ищутся комплементарные слова s_2. Если находятся, то для каждого слова s_2 на отрезке [s_2+ *длина_слова*; s_2+ *длина_слова* + *макс_длина_шпильки*] ищутся комплементарные слова s_3. Далее проверяется совпадение слов s_1 и s_3 ; выдаются тройки.

Из всех полученных троек слов выбираем одну, выполняя следующие действия.

Отсортируем эти тройки по возрастанию разницы между координатами первой буквы слова s_3 и последней буквы слова s_2. Ищем слова, удовлетворяющие следующим условиям:

1. Разница между координатами первой буквы слова s_3 и последней буквы слова s_2 меньше *макс_дельта* (обычно *макс_дельта* = 30; иначе не образуется шпилька-терминатор), а среди них отбираются первых *количество_ответов* ответов (обычно *количество_ответов* = 15)

2. Координата начала слова s_3 меньше координаты начала участка урацилов U и отстоит от нее не более, чем на *расстояние_до_поля_T* нуклеотидов (обычно *расстояние_до_поля_T* равнялось 9).

Алгоритм отслеживает и такое дополнительное условие:

3. Суммарное число комплементрных пар GC в паре слов s_2 и s_3, поделенное на число комплементарных пар AT не меньше x (обычно x = 1.3), т.е. эта пара слов должна быть GC-насыщенной.

В случае нахождения такой тройки алгоритм переходит к следующему шагу и сообщает «аттенюаторная регуляция обнаружена». Если такая тройка не находится, то алгоритм выдает сообщение об отсутствии аттенюации.

Затем последовательно по s_2 и s_3 строится терминатор из условия минимума энергии, и аналогично – антитерминатор и паузная шпилька (при заданных параметрах).

Среди так полученных структур отбираются по одному представителю из каждого класса «подобных» структур (определяемого нами отношением эквивалентности). Для наших случаев, как правило, фактически оказывалось по одному-двум, но не более трех таких представителей.

Итак, терминатор образуется спариванием слов s_2 и s_3 и продолжением спаривания в обе стороны. Альтернативность терминатора и антитерминатора поддерживается за счет слова s_2: антитерминатор – это шпилька, включающая спаривание s_1 и s_2 с изменением свободной энергии сворачивания, меньшим некоторого параметра (например, –10 ккал/моль, что обеспечивает возможность сворачивания антитерминатора).

Правильность работы первого алгоритма подтверждается сравнением результатов счета с известными биологическими данными, а также результатами совместных выравниваний найденных нами новых и уже известных сигналов аттенюаторной регуляции. Выравнивания говорят, в частности, о консервативности указанных троек слов. Действительно:

1. При нашем выравнивании для многих оперонов и независимо от алгоритма определяются слова s_1, s_2, s_3 и они оказываются консервативными. Так, для *trp*-оперона эти слова выровнялись даже у гамма-, альфа- и бета-протеобактерий. Для каждого из 8 изученных оперонов у организмов из одной группы эти слова оказыва-

лись одинаковыми с точностью до нескольких букв (см. рисунки 3-6 в Приложении 2).

2. Для многих оперонов (например, *thrABC* у гамма-протеобактерий) антитерминаторы, полученные алгоритмом для разных организмов, имели сходную структуру: близкие значения числа отрезков и их длин, близкие значения длин выпячиваний и их типов (односторонние или двусторонние), и т.д.

Алгоритм обеспечивает следующие условия:

1. Наличие лидерного пептида с полем регуляторных кодонов соответствующих аминокислот.

2. Пересечение терминатора и антитерминатора.

3. Отсутствие пересечения паузной шпильки и терминатора.

4. Расстояние от конца поля регуляторных кодонов до начала левого плеча антитерминатора более 5 нуклеотидов.

Кроме того, алгоритм проверяет полученные им ответы на выполнение условия, которое наблюдалось во многих известных случаях аттенюаторной регуляции: расстояние от конца лидерного пептида до начала левого плеча антитерминатора от –3 до +3 нуклеотидов.

Упомянутое хорошее выравнивание по разным причинам не является обязательным условием наличия аттенюаторной регуляции. В некоторых организмах (например, *Y. pestis*) фрагменты последовательности, участвующие в такой регуляции, не являются консервативными в других гамма-протеобактериях.

Для нахождения численных значений перечисленных выше параметров первого алгоритма (и, в меньшей степени, для нахождения параметров второго алгоритма, описанного ниже) применяются стандартные методы статистики. По известным значениям X этих параметров, которые, в основном, брались из результатов работы [38], а также из наших уже подтвержденных выравниванием результатов, находились значения параметра и доверительный интервал. Распределение X было близко к нор-

мальному, что проверялось стандартным образом; или в некоторых случаях мы работали с другими (например, log-нормальным) распределениями. За само значение бралось среднее X^* (обычно, среднее арифметическое, иногда иное среднее) и находилось квадратическое отклонение *s*. Обычно выбиралась надежность $\gamma = 0{,}95$ и искался радиус доверительного интервала, равный $t_\gamma \cdot \frac{s}{\sqrt{n}}$, где *n* – объем наблюденных значений и t_γ находился стандартным образом по таблице, отражающей соотношения $P\left(\left|\frac{X^* - a}{s/\sqrt{n}}\right| < t_\gamma\right) = 2\int_0^{t_\gamma} S(t,n)dt = \gamma$. Здесь $S(t,n)$ - плотность распределения Стьюдента. Для наших параметров значения радиуса было небольшим.

В заключение приведем схему первого алгоритма.

§1.2. Второй алгоритм

Второй алгоритм основан на идее, предложенной в работе[3], и применяется, как и первый, для поиска аттенюаторного сигнала в **одной** нуклеотидной последовательности. Его применение целесообразно в случае слабо выраженного сигнала: он выдает больше вариантов ответа, но может находить альтернативный сигнал, мало похожий на «классическую» аттенюацию.

Этот алгоритм (примерно квадратичный от размера исходных данных) по любой нуклеотидной последовательности выдает список потенциальных аттенюаторных структур в ней. Каждая такая структура состоит теперь из тройки шпилек <терминатор, антитерминатор, пауза>. Ниже в качестве примера указываются численные значения, которые, конечно, являются параметрами этого алгоритма и могут варьироваться.

Алгоритм состоит из двух этапов, которые в автореферате описаны схематично, а в тексте диссертации подробно.

Этап 1: *порождение множества локально оптимальных шпилек*. Количество элементов в этом множестве приблизительно равно длине исходной последовательности. Этап 1 состоит из следующих двух шагов.

1. Последовательность делится на фрагменты фиксированной длины и внутри каждого из них индуктивно порождаются локально оптимальные шпильки (т.е. максимальные элементы в смысле некоторого фиксированного отношения частичного порядка).

2. На индуктивном шаге переопределяются лишь параметры шпилек. Только малое количество специально отобранных алгоритмом шпилек порождается полностью, т.е. как совокупность пар комплементарных нуклеотидов.

Структура **этапа 1** второго алгоритма.

---

[3] Верещагин Н.К., Любецкий В.А. Алгоритм определения вторичной структуры РНК. Труды научно-исследовательского семинара логического центра ИФ РАН, выпуск 14, Москва, Издательство РАН, 2000, с. 99-109.

1. Глобальная переменная X пробегает исходную последовательность, и для каждого значения X находится основные параметры тех локально-оптимальных шпилек, для которых X находится в правом основании. Параметры шпилек при фиксированном X находятся рекурсивно.

При решении переборной задачи с помощью рекурсии число шагов алгоритма уменьшается. При хорошем подборе параметров рекурсии возможна значительная оптимизация алгоритма, что далее и делается.

Нами замечено, что в результате такой оптимизации число массивов с локально-оптимальными шпильками равно длине последовательности.

При фиксированном X массивы с такими шпильками ищем в промежутке [*начало*, ..., *конец*], рис. 1.

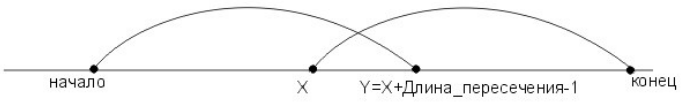


Рис. 1

2. По полученным параметрам строятся сами шпильки. Сейчас строятся все локально-оптимальные шпильки, но существует вариант более избирательного критерия отбора шпилек по их параметрам и построения только части шпилек.

Рассмотрим структуру этапа 1. Этап состоит из двух частей.

Часть 1. Получаем на вход исходную последовательность и по каждому значению X выдаем некоторые параметры локально-оптимальных шпилек;

Часть 2. Получаем на вход параметры ранее выбранных (всех локально-оптимальных шпилек) по всем значениям X и строим сами шпильки.

Описание **части 1** (для фиксированного значения X) этапа 1.

Этап 1 порождает параметры (называемые «качества») всех локально-оптимальных шпилек при всех значениях X. Для всех таких шпилек обеспечивается условие B не меньше X. Рассмотрим, как это происходит при фиксированном X.

Сначала определим отношение порядка, по которому будут отбираться локально-оптимальные шпильки. Ниже буквы m и M обозначают мощность шпильки (т.е. количество комплементарных пар в ней) и A, B, C, D – параметры шпильки, а именно, соответственно начала и концы левого и правого плеч.

Отношение порядка на парах. Определим отношение порядка для пар (m,C) как $(m_1,C_1) > (m,C)$, если $m_1 > m$ и $C_1 > C$, т.е. порядок покоординатный. Если $m_1 > m$, $C_1 < C$, то пары несравнимы.

Отношение порядка на тройках. Определим отношение порядка для пар ((m,C),D) как $((m_1,C_1),D_1) > ((m,C),D)$, если ($m_1 > m$ и $C_1 > C$) или ($m_1 = m$ и $C_1 = C$ и $D_1 < D$), т.е. порядок лексикографический. Если $m_1 > m$, $C_1 < C$, то пары не сравнимы. Оба отношения – частичные порядки.

Параметры, которые мы хотим найти для каждой локальной-оптимальной шпильки, образуют вектор (M,A,C,D). Результат работы части 1 алгоритма (т.е. множество таких параметров) будем хранить в массиве *максшпилька* (или *максшпилька2*).

Определение массива *максшпилька*. Массив *максшпилька* хранит качества (M,A,C,D) всех локально-оптимальных шпилек с началом в A от A = *Начало* до A = X, лежащих в отрезке [*начало, конец*], у которых B не менее X. Из способа организации этого массива видно, что в нем хранятся качества всех локально-оптимальных шпилек. Массив *максшпилька* индексируется по A и состоит из столбцов, соответствующих каждому значению A от A = *Начало* до A = X. Каждый столбец *максшпилька*[A], содержит все максимальные несравнимые качества троек с началом в A и имеет вид, показанный на рис. 2.

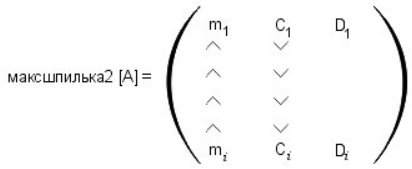


Рис. 2

Таким образом, задача части 1 алгоритма состоит в заполнении массива *максшпилька2*.

Для нахождения массива *максшпилька* необходимо определить промежуточные массивы. В определении этих массивов используется переменная К – некоторое фиксированная координата из промежутка [X, *конец*].

Определение массива *продолж*. Массив *продолж* индексируется переменной А; *продолж*[А] хранит максимальные несравнимые качества шпилек с началом в А и с концом, лежащем в отрезке [К-*максвыпячивание*, ..., К]. Качествами являются пары (m,C), так как параметр D – конец шпильки однозначно определяется из массива. Очевидно, что именно такие шпильки могут быть продолжены новыми отрезками при увеличении К (рис. 3).

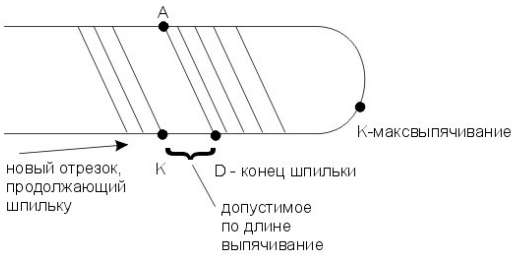


Рис. 3

Продолж[А] организован следующим образом: это очередь длины *максвыпячивание* , где i-тый элемент хранит максимальные качества (пары (m,C)) шпилек с началом в А и концом в К+*максвыпячивание*-i.

Определение массива *предшпилька*. Массив *предшпилька* индексируется по переменной A; *предшпилька*[A] хранит максимальные несравнимые качества предшпилек (пары (m,C)) с началом в A и концом в K. Предшпилькой кратности r называется шпилька, у которой первый отрезок (начинающийся в A) содержит r нуклеотидов. Определяются предшпильки кратности r = 1, .., *минподряд*-1. Предшпильки при шаге индукции достраиваются до шпилек. *Предшпилька*[A] организована следующим образом: это антиочередь длины *минподряд*-1, где i-ый элемент хранит максимальные качества предшпилек кратности i.

Массивы перезаполняются индукцией по K. На каждом шаге индукции хранятся максимальные несравнимые качества локально-оптимальных шпилек, лежащих в отрезке [*Начало*, K]. На последнем шаге индукции, при K = *конец*, получим искомый массив *максшпилька*; массивы *продолж* и *предшпилька* являются вспомогательными.

Начальное значение индукции (при K = X) находится очевидно, так как шпилек и предшпилек, у которых B > X, не существует. Опишем индуктивный переход от K к K+1. То есть необходимо перезаполнить массивы *продолж*, *максшпилька2* и *предшпилька* при каждом A. При этом возможны два случая.

Случай 1. Пара нуклеотидов (A ,K+1) не комплементарна. Тогда, в массиве *продолж*[A] удаляется первый элемент (т.к. шпильки с концом в K-*максвыпячивание* не могут быть продолжены) и последний элемент равен пустому множеству (так как шпилек с концом в K+1 не существует). Массив *максшпилька*[A] не изменяется, так как новых шпилек не образовалось. Массив *предшпилька*[A] заполняется пустыми множествами, так как предшпилек с началом в A и концом в K+1 не существует.

Случай 2. Пара нуклеотидов (A ,K+1) комплементарна. Тогда, из массива *продолж*[A] удаляется первый элемент, а последний заполняется следующим образом: сливаются последний элемент массива *продолж*[A+1] и последний элемент *предшпилька*[A+1] и в результатах слияния мощности увеличиваются на единицу (т.е.

учитываем пару (A,K+1)). Слиянием называется операция, составляющая из нескольких упорядоченных массивов пар (или троек) единый массив пар (или троек) с максимальными несравнимыми качествами в смысле вышеописанного отношения порядка. Получившееся множество назовем set. Если пары нуклеотидов (A, K+1) комплементарны, то у нас есть возможность получить новый отрезок. Каковы возможные способы образования отрезка? *Предшпилька*[A+1] (еще с концом в K) может быть достроена до отрезка; следовательно, берем предшпильки наибольшей кратности *минподряд*-1; они хранятся в последнем элементе массива *предшпилька*[A+1], рис. 4.

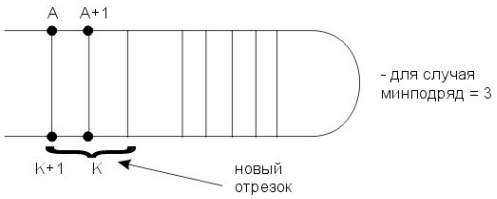


Рис. 4

Другой способ достройки – продолжение отрезка с началом в A и с концом в K+1. Качества шпилек с такими отрезками хранятся в последнем элементе массива *продолж*[A+1], рис. 5.

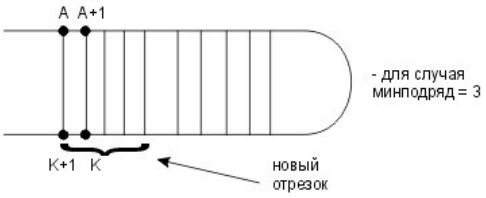


Рис. 5

Операция слияния позволяет выбрать лучшие качества из обоих вышеописанных случаев.

В массив *максшпилька*[A] вливается множество sets. На каждом шаге мы вливаем в *максшпильку*[A] только качества шпилек с концом в K. Но так как операция слияния позволяет оставлять «лучшие» качества, то очевидно, что качества лучших

шпилек останутся в окончательном массиве *максшпилька*[A]. Массив *предшпилька*[A] перезаполняется следующим образом: в предшпильки кратности r > 1 кладутся соответствующие (r–1-ые) столбцы массива *предшпилька*[A+1] (еще с концом в K). Заполним, например, второй столбец массива *предшпилька*[A], то есть максимальные качества предшпилек кратности 2. Очевидно, такие предшпильки получаются из предшпилек с началом в A+1 , концом в K и кратности 1 (рис. 6). Значит, берем первый столбец массива *предшпилька*[A+1], увеличиваем в нем мощности на 1 и кладем его во второй столбец массива *предшпилька*[A].

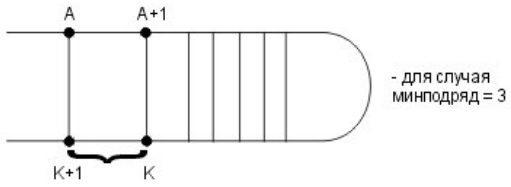


Рис. 6

Первый столбец массива *предшпилька*[A], то есть предшпильки кратности 1, заполняется следующим образом: необходимо к паре (A, K+1) подобрать наилучшую по мощности шпильку. Для этого сливаем элементы массивов *продолж*[A+1], .., *продолж*[A+ *максвыпячивание*+1] и увеличиваем мощности полученного множества на 1 с тем, чтобы учесть пару (A, K+1), рис. 7.

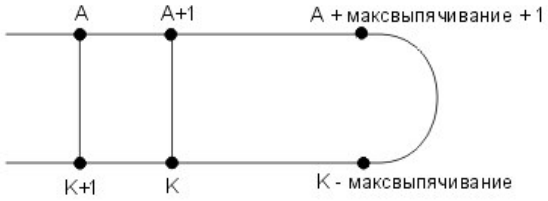


Рис. 7

Очевидно, что качества шпилек, которые может продолжить пара (A, K+1) имеют начало в отрезке [A+1, .., A+*максвыпячивание*+1] и конец в отрезке [K-

максвыпячивание, .., K]. Качества таких шпилек лежат в массивах *продолж*[A+1], .., *продолж*[A+*максвыпячивание*+1].

Описание **части 2** этапа 1 второго алгоритма. Часть 2 получает на вход вектор параметров некоторой локально-оптимальной шпильки (M,A,C,D) и находит минимальный параметр B, такой, что существует шпилька с параметрами (M,A,B,C,D), а затем строит саму шпильку. Очевидно, что во многих случаях существует более одной шпильки с параметрами (M,A,B,C,D); алгоритм строит одну произвольную шпильку из этого множества. Приведем схему работы части 2.

1. Для фиксированного (M,A,C,D) находится минимальное B такое, что (A,B,C,D) – параметры локально-оптимальной шпильки мощности M; находятся массивы *полн* и *неполн*.

2. Используя массивы *полн* и *неполн*, строится шпилька с указанными параметрами.

Определим массивы *полн* и *неполн*. Массив ***полн*** индексируется переменными B и c; *полн*[B,c] хранит максимальную мощность шпильки с концами A, B, c, D. В части 2 под предшпилькой понимается немного другая структура – шпилька, у которой может не быть выполнено условие (c-B <= *макспетля*).

Двумерный массив ***неполн*** также индексируется переменными B, c; *неполн*[B,c] является антиочередью длины *минподряд*-1, i-ый элемент которой хранит максимальную мощность внутренней предшпильки кратности i. Внутренней предшпилькой кратности i (i=1,..,*минподряд*-1) называется шпилька, у которой отрезок ***c*** концами B и ***c*** имеет длину i и *c*-B+1 может быть больше константы *макспетля***.**

Представим двумерный массив *полн* как таблицу. Пусть по столбцам расположены значения *c*, где *c* принадлежит промежутку [C,D]; по строкам расположены значения B от B=A. Для фиксированного B будем заполнять строку для всех *c*. Предположим, что, зная строки для всех B, где B принадлежит промежутку [A,B], умеем заполнить строку для B+1. Тогда очевидно, что нужно заполнять новые строки, пока

в столбце *c* = C не встретится B, удовлетворяющее всем условиям шпильки мощности M (т.е. *минпетля не больше* C–B+1 не больше *макспетля* и т.д.). Такое B будет минимальным.

Рассмотрим индукцию по B для случая *минподряд* = 3.

Начальный шаг, когда B=A. Тогда *полн*[B,c] = –∞ для всех *c* (*c* принадлежит [C,D]), так как шпилек с параметрами A=A, B=A, *c*=*c*, D=D не существует. Аналогично *неполн*[B,D] = (1, –∞), для остальных *c* значение *неполн* равно множеству (–∞, –∞).

Опишем индуктивный переход от B к B+1.

Случай 1. Пара нуклеотидов (B+1,***c***) не комлементарна. Тогда *полн*[***c***,B+1] равен –∞ (так как шпилек с такими концами не существует), и аналогично *неполн* равен антиочереди, состоящей из элементов , равных –∞.

Случай 2. Пара нуклеотидов (B+1,*c*) комлементарна. Тогда массив *полн*[*c*,B+1] заполняется максимумом из *полн*[*c*+1,B] и последнего элемента *неполн*[*c*+1,B]. Массив *неполн* перезаполняется следующим образом: в i-ый столбец где i=2, .., *минподряд*-1, нужно положить элементы i-1 столбца массива *неполн*[*c*+1,B], увеличенные на единицу. Осталось перезаполнить первый элемент массива *неполн*[*c*,B+1]. В него кладем максимум элементов *полн*[u,v], где u = *c*, .., *c*+*максвыпячивание* и v = B, .., B–*максвыпячивание*, увеличенный на единицу.

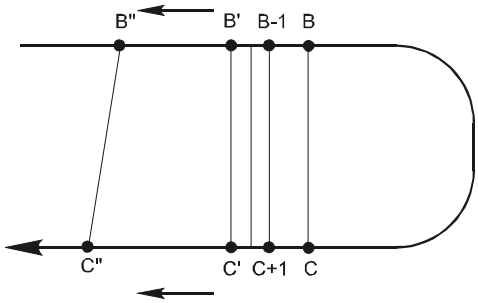


Рис. 8

Теперь по A, B, C, D и массивам *полн* и *неполн* восстановим шпильку (рис. 9). Встанем в точки B, C – начальное положение и проверим, что *полн*[B,C] = M. Тогда запомним пару (B,C) – она входит в рассматриваемую шпильку. Увеличим C на 1, и B уменьшим на 1, т.е. встанем в позицию (B-1, C+1) и будем искать *полн*[B–1,C+1] = M–1. Пара (B–1,C+1) также входит в рассматриваемую шпильку. Таким образом, восстановим первый отрезок (сначала по массиву *полн*, а последние нуклеотиды отрезка по

массиву *неполн*). Затем находим начало нового отрезка. Пусть восстановили m нуклеотидов шпильки и остановились в позиции (B',C'). Тогда определим цикл, который от B', уменьшаясь, и от C', увеличиваясь, ищет такие B'', C'', что *полн*[B'', C''] = M–m'. Эти B'', C'' будут началом нового отрезка.

Число шагов второго алгоритма равно O(*длина_ последовательности*длина_ шпильки *длина_шпильки*максвыпячивание*). Число шагов в части 2 второго алгоритма равно O(*длина_последовательности*длина_шпильки*максвыпячивание** (*длина_шпильки+максвыпячивание*)).

Этап 2 второго алгоритма: *построение самой аттенюаторной структуры*, который состоит из следующих двух шагов.

1. В множестве всех локально оптимальных шпилек отбираются те внешние петли (этих шпилек), начала и концы (B,C) которых повторяются более чем $p = 9$ раз. Такие пары (B,C) будем называть *частыми*.

2. В так полученном списке внешних петель отбираются петли, у которых частые пары (B,C) расположены рядом с участком остатков урацила U. Они считаются петлями будущих терминаторов. Затем для каждой такой петли строится сам терминатор, а для него последовательно порождаются еще две оставшиеся шпильки – сначала антитерминатор и затем паузная шпилька.

Построение терминатора для данных (B,C) и участка остатков урацила U происходит «вытягиванием» отрезков от (B,C), т.е. последовательным спариванием как можно большего числа нуклеотидов в 1-2 отрезка.

При построении антитерминатора и паузы используется понятие ядра. *Ядром* для данного множества шпилек с одной и той же внешней петлей (B,C) с координатами начала B и конца C петли называется консенсусная шпилька, т.е. шпилька наилучшим образом согласованная со всеми шпильками из этого множества. Например, в качестве ядра нами бралась шпилька, состоящая из пар нуклеотидов, которые являются спаренными в более чем половине от всех шпилек из этого множества. Для на-

ших случаев, как правило, получалось ядро, состоящее из двух отрезков (иногда оно содержало от 1 до 5 отрезков). Заметим, что в большинстве случаев так вычисляемые ядра несколько короче биологических шпилек, участвующих в аттенюации.

Итак, для поиска антитерминатора при уже полученном терминаторе берется частая пара (B_1,C_1) слева и ближайшая к петле (B,C) терминатора. Рассмотрим все локально оптимальные шпильки с данной парой (B_1,C_1) , включая и их подшпильки; и среди них в качестве антитерминатора отберем все те, у которых плечо C_1D_1 имеет не менее 5 общих нуклеотидов с плечом AB, и также $A > B_1$, $C > D_1$. Если таким образом нашлось пустое множество, то в качестве антитерминатора возьмем все ядра, построенные для множества всех локально оптимальных шпилек с данным (B_1,C_1).

Паузную шпильку построим по каждому уже полученному антитерминатору аналогично тому, как антитерминатор строился по полученному терминатору. Для каждого терминатора выбираем одну соответствующую ему пару антитерминатор-пауза как пару с наибольшей суммарной мощностью. Так полученные структуры ранжируются по мощности терминатора, а при одинаковой его мощности по возрастанию координаты B его петли.

Алгоритмы реализованы на языке Object Pascal в среде[4] Delphi 5; вспомогательные алгоритмы (а также вариант первого алгоритма) – на языке Perl.

Подходя к концу описания наших алгоритмов заметим, что рассматриваемая задача представляется NP-полной. При построении наших алгоритмов мы ориентировались на результаты из математической логики и теории алгоритмов. Например, такие как реализация булевой функции от *n* булевых переменных нормальным алгоримом (по Маркову) в *m*-буквенном алфавите (со сложностью порядка $2^n / \log_2 m$) и машиной Тьюринга с –буквенным внешним алфавитом (со сложностью порядка $2^n / n \cdot (m-1)$) и другими результатами о разрешимых проблемах.

[4] Алгоритмы также реализованы на языке ANSI C для параллельной вычислительной архитектуры с протоколом MPI – этот результат не включается в диссертационную работу. Часть вычислений велась на суперкомпьютере МВС-1000М (в МСЦ Миннауки, РАН, МГУ и РФФИ).

§1.3. Сравнение результатов работы первого и второго алгоритмов.

Отметим общие результаты сравнения их работы.

1. Терминаторы с выраженными признаками (т.е. наличие участка остатков урацила U, отсутствие выпячиваний, GC-насыщенность, большое число комплементарных пар нуклеотидов) одинаково хорошо находятся с помощью обоих алгоритмов.

2. Первый алгоритм, в отличие от второго, с хорошей точностью находит антитерминаторы и паузные шпильки.

3. Если в исходной последовательности отсутствует лидерный пептид или антитерминатор неканонический, то первый алгоритм не находит структуру, но она может успешно быть найденной вторым алгоритмом. Например, в случае регуляции гена *pheA* в организме *Y. pestis* перед аттенюаторной структурой встраивается мобильный элемент, и лидерный пептид хотя, конечно, существует, но находится на большом расстоянии от вторичной структуры и фактически не попадает в исходную последовательность разумной длины. В этом примере, антитерминатор также не содержит в последнем от его петли отрезке комплементарных слов s_1 и s_2, что рассматривается нами как случай неканонического антитерминатора, и первый алгоритм его не находит.

Продолжая сравнение алгоритмов, можно добавить следующее: первый алгоритм оперирует с б′ольшим числом параметров и объектов, поэтому «классическая» аттенюаторная регуляция находится им более точно. Второй алгоритм из биологических особенностей использует только энергетические соображения, наличие сгущения нуклеотидов U и т.п. Поэтому, если искомая структура содержит не все стандартные элементы классической аттенюаторной регуляции, то второй алгоритм, в отличие от первого, может ее найти. Для успешной работы второго алгоритма желательно присутствие мощного терминатора в исходной последовательности, а для успешной работы первого алгоритма важно наличие там лидерного пептида.

Аттенюаторные структуры, найденные вторым алгоритмом и не найденные первым среди 18 тестовых примеров, перечисленных в таблице 2.5 Приложения 1, приведены в таблице 1. В качестве этих тестовых брались лидерные области, в которых аттенюация известна[5]; в случаях *E. coli_ilvBN* (и *E. coli_G*) она получена экспериментально[6].

Таблица 1.

Название организма и гена	Наличие лидерного пептида	Комплементарная тройка слов – существенный параметр первого алгоритма
Y. pestis_trpE	да	нет
Y. pestis_pheA1	нет	да
Y. pestis_pheA2	нет	нет
E. coli_ilvBN	да	нет

В заключение приведем схемы первого и второго этапов алгоритма. В схемах используются следующие обозначения: «A = K+1» означает комплементарность нуклеотидов с координатами A и (K+1) в исходной последовательности. Символ «©» означает слияние конечного числа множеств, т.е. формирование множества с максимальными несравнимыми качествами (в смысле отношения порядка во втором алгоритме) из конечного числа множеств, также содержащих максимальные несравнимые качества.

[5] Panina, E.M., Vitreschak, A.G., Mironov, A.A. and Gelfand, M.S. (2001) Regulation of aromatic amino acid biosynthesis in gamma-proteobacteria. J. Mol. Microbiol. Biotechnol. 3, 529-543.

[6] Landick, R., Turnbough, C.L. and Yanovsky, C. (1994) Transcriptional attenuation. In: Escherichia coli and Salmonella. Cellular and molecular biology (Neidhardt, F.C., Ed.), pp. 1263-1286. American Society for Microbiology, Washington, DC.

ГЛАВА 2. Тестирование алгоритмов

Тестирование двух приведенных выше алгоритмов проводилось для следующих семи типов условий. Для каждого типа тестировался один из алгоритмов.

§2.1. Второй алгоритм: тестирование на случайных последовательностях. На вход алгоритма подавалась случайная бернуллиевская последовательность длины 450 (порождаемая с помощью генератора случайных чисел стандартной библиотеки языка Perl). На выходе появлялось сообщение вида <Терминатор, Антитерминатор или -, Пауза или -, Участок U или ->, в котором знак «-» говорит об отсутствии соответствующего элемента (или сообщение об ее отсутствии). Участком U считались фрагменты последовательности длины 7, в которых нуклеотид U встречался не менее 5 раз и допускались разрывы: два из одной буквы или один из двух букв. Найденными считались структуры, содержащие терминатор из не менее, чем трех пар комплементарных нуклеотидов, в то время как антитерминатор и/или паузная шпилька, участок U могли отсутствовать.

Было проведено 67 запусков алгоритма с указанными выше параметрами. Результаты приведены в таблице 2.1.1. Из них видно, что в случайной последовательности, по крайней мере, терминатор находится почти в 40% случаев.

Таблица 2.1.1. Второй алгоритм: поиск аттенюаторных структур в бернуллиевских случайных последовательностях.

Аттенюаторная структура в случайной последовательности:	Число соответствующих случаев:
Найдена (с участком U)	29
Не найдена (но участок U имеется)	21
Не найден даже участок U	17

Рассмотрим число пар (B,C), повторяющихся более чем p раз в множестве локально оптимальных шпилек, которое образовано вторым алгоритмом по исходной нуклеотидной последовательности. Такую пару (B,C) мы назвали *частой*. Число частых пар при значении порога $p = 9$ для тех же случайных и 20 тестовых биологических последовательностей приводятся в таблице 2.1.2, третий столбец. Из нее и других наших вычислений видно, что число различных пар (B,C) в множествах локально оптимальных шпилек примерно на порядок меньше длины исходной последовательности, и число частых пар примерно в 4 раза меньше числа всех пар (B,C).

Таблица 2.1.2. Число всех пар (B,C) и число частых пар (B,C) в множествах локально оптимальных шпилек, построенных вторым алгоритмом по бернулиевским случайным и биологическим последовательностям. В качестве случайных взяты 20 последовательностей из 67 упомянутых в таблице 2.1.1.

Тип исходной последовательности		Число различных пар (B,C) в множестве локально оптимальных шпилек, найденных по исходной последовательности	Число частых пар (B,C) в том же множестве, т.е. повторившихся в нем более чем p раз ($p = 9$)
случайные	*Случайная*	54	5
	Случайная	62	12
	Случайная	54	10
	Случайная	55	11
	Случайная	44	10
	Случайная	56	12
	Случайная	50	10
	Случайная	51	13
	Случайная	49	11
	Случайная	47	12
	Случайная	53	13
	Случайная	47	12
	Случайная	54	13
	Случайная	58	11

	Случайная	53	10
	Случайная	57	8
	Случайная	57	7
	Случайная	59	10
	Случайная	61	13
	Случайная	49	11
биологические	*AB_pheA*	57	13
	EC_pheA	43	12
	EC_pheS	47	8
	EC_trpE	44	12
	HI_pheA	48	14
	HI_pheST	39	10
	HI_trpBA	57	12
	HI_trpE	38	15
	SH_pheS	46	12
	SH_pheA	43	9
	SH_trpE	49	10
	ST_pheA	41	11
	ST_pheS	40	11
	ST_trpE	50	11
	VC_pheA	37	8
	VC_trpE	48	13
	YP_pheA1	41	8
	YP_pheA2	53	8
	YP_pheS	47	14
	YP_trpE	32	7

§2.2. Второй алгоритм: тестирование на случайных последовательностях с участком остатков урацила U. На вход алгоритма подавались 20 бернуллиевских случайных последовательностей, использованных в §2.2., в которые дополнительно в случайных местах вставлялись 1-2 фрагмента из 5 нуклеотидов U, которые находились от начала последовательности на расстоянии не менее 100 нуклеотидов (с тем,

чтобы алгоритм имел возможность построить структуру перед участком U). На выходе появлялось такое же как выше сообщение <Терминатор мощности не менее 3, Антитерминатор или -, Пауза или -, Участок U или -> или сообщение об отсутствии такого терминатора.

В этом случае структуры находились чаще, чем в пункте 1, что, конечно, нежелательно. Поэтому было предложено указанное ниже дополнительное ко второму алгоритму условие, которое позволяет отличить последовательность, случайную даже с вставленным в нее (как выше) участком U, от биологической последовательности, содержащей аттенюацию. Таким образом, второй алгоритм, дополненный этим условием, может применяться и для решения задачи различения случайной последовательности от биологической, содержащей аттенюацию. Численные значения указанных ниже параметров соответствуют рассмотренному нами случаю последовательностей длины 450. Это *условие* состоит в следующем. Биологическая последовательность, содержащая аттенюаторную регуляцию, отличается от случайной последовательности с вставленными в нее 1-2 участками U, если для структур, найденных в ней вторым алгоритмом, выполняется одно из трех следующих свойств 1-3.

1. Ответ содержит ровно одну структуру и она «хорошая» в смысле:

а) участок U около терминатора найденной структуры содержит не менее 6 нуклеотидов подряд;

б) найденная структура имеет пересекающиеся (не менее чем на 5 нуклеотидов) терминатор и антитерминатор, а также паузную шпильку;

в) терминатор имеет не менее 5 пар комплементарных нуклеотидов.

2. Среди двух или трех найденных структур хотя бы одна удовлетворяет условиям 1б и 1в.

3. Найдено не менее 4 структур с различными парами (B,C) координат петли терминатора.

Из таблицы 2.2 видно, что случайная последовательность с участком U устойчиво отличается от биологической последовательности, содержащей аттенюацию. Причем, как правило, эта аттенюация выдается нашим алгоритмом в качестве *первого* ответа, когда все находимые ответы *ранжируются* по убыванию мощности терминатора, а при одинаковой мощности по возрастанию координаты B из пары (B,C) терминатора. Напомним, что *мощность* шпильки это – число комплементарных пар в ней.

Таблица 2.2. Первый столбец содержит название организма и оперона, к регуляторной области которого применялся второй алгоритм вместе с нашим условием; или указывает на случайность исходной последовательности. В каждой ячейке второго столбца перечисляются все найденные алгоритмом структуры (для исходной последовательности) и для каждой из них указывается выполнение отдельных пунктов нашего условия. А именно, для каждой структуры после ее номера в указанном ранжировании перечисляются: знак «+», если терминатор и антитерминатор пересекаются не менее чем на 5 нуклеотидов и алгоритм нашел паузную шпильку (иначе «–»); затем знак «+», если участок U около терминатора имеет длину не менее 6 нуклеотидов подряд (иначе «–»); и мощность терминатора. Подчеркиванием выделяется ответ, совпавший с известным биологическим. В третьем столбце указывается выполнение в целом нашего условия на исходную последовательность (Real, если выполнено, иначе False).

Имя организма и гена	Параметры условия для каждой из найденных структур	Выполнение условия
1. *AB_pheA*	<u>1 + + 14</u> 2 + + 4 3 + – 3 4 + – 3	Real
2. *EC_pheA*	1 + + 8 2 – – 4	Real

	3 + – 3 4 + – 5 5 + + 5	
3. *EC_pheS*	1 + + 9 2 – + 3	Real
4. *EC_trpE*	1 + + 7 2 – + 3	Real
5. *HI_pheA*	1 + + 15 2 + – 3 3 – – 4	Real
6. *TY_pheA*	1 + + 13 2 + + 8	Real
7. *TY_trpE*	1 + + 6 2 – – 3	Real
8. *TY_pheS*	1 + + 12 2 + – 9 3 + + 4	Real
9. *VC_pheA*	1 + + 12 2 – + 10 3 – + 4	Real
10. *VC_trpE*	1 + + 7 2 – – 3	Real
11. *YP_pheA1*	1 + – 10 2 + + 13 3 – – 4	Real
12. *YP_pheA2*	1 – – 8 2 + – 3 3 – – 3	False
13. *YP_pheS*	1 – – 9 2 + – 6 3 + – 4 4 + + 3 5 + + 3	Real
14. *YP_trpE*	1 + + 8	Real
Случайная	–	False

Случайная	1 + – 9	False
Случайная	–	False
Случайная	1 + – 4	False
Случайная	–	False
Случайная	1 + – 6	False
Случайная	1 + – 4	False
Случайная	1 + – 4 2 + – 3	False
Случайная	1 + – 5	False
Случайная	–	False
Случайная	–	False
Случайная	1 + + 5	Real
Случайная	1 + – 5 2 + – 3	Real
Случайная	–	False
Случайная	1 + – 3	False
Случайная	1 + – 4	False
Случайная	1 + – 5	False
Случайная	–	False
Случайная	1 + – 5	False
Случайная	–	False

§2.3. Второй алгоритм: тестирование на случайных последовательностях, содержащих биологически значимые терминаторы. На вход алгоритму подавались случайные бернуллиевские последовательности, использованные в пункте 2 тестирования, в которые перед участком U на расстоянии в 3 нуклеотида помещался фрагмент, содержащий биологически значимый терминатор из регуляторной области гена *trpE* организма *E. coli*, а именно, фрагмент `CAGCCCGCCTAATGAGCGGGC`. На выходе алгоритм выдавал такое же как выше сообщение.

На таких последовательностях второй алгоритм не дает удовлетворительных результатов: он не может систематически отличать такие последовательности от биологически значимых. Это видно из таблицы 2.3.

Таблица 2.3. Результат работы второго алгоритма и параметры нашего условия для случайных последовательностей с участком U и терминатором из регуляторной области оперона *trpE* организма *E. coli*. Обозначения такие же, как в таблице 2.2.

Название гена и организма	Параметры условия для каждой из найденных структур	Выполнение условия
Случайная	–	False
Случайная	1 + + 7	Real
Случайная	–	False
Случайная	1 + + 7	Real
Случайная	1 – + 7	False
Случайная	1 + + 7 2 + – 6	Real
Случайная	1 + – 4	False
Случайная	1 + + 7 2 + – 4	Real
Случайная	1 + + 7 2 + – 5	Real
Случайная	1 – + 7	False
Случайная	1 – + 7	False
Случайная	1 + + 7	Real
Случайная	1 + + 7 2 + – 5 3 + – 3	Real
Случайная	1 + + 7	Real
Случайная	1 + + 7	Real
Случайная	1 + – 4	False
Случайная	1 + + 7	Real
Случайная	1 + + 7	Real
Случайная	–	False
Случайная	1 + + 7	Real

§2.4. Первый и второй алгоритм: тестирование на биологических последовательностях, содержащих аттенюаторную структуру. На вход второму алгоритму подавались регуляторные области длины 450, содержащие уже известную аттенюаторную регуляцию. На выходе алгоритма выдавалось такое же сообщение. Результаты тестирования приводятся в таблице 2.4, из которой видно, что алгоритм находит аттенюацию с высокой точностью. Естественно, что он находит терминатор с более высокой точностью, чем антитерминатор, и последний с более высокой точностью, чем паузу.

Таблица 2.4. Результат работы второго алгоритма для биологических последовательностей, содержащих известную аттенюаторную регуляцию. Первый столбец содержит название организма и гена, перед которым располагалается регуляторная область. Во втором столбце ставится знак «+», если алгоритм правильно нашел биологический терминатор; иначе ставится знак «–». Третий столбец содержит тройку чисел 0, 1, 2, которые оценивают точность совпадений биологических терминатора, антитерминатора и паузной шпильки соответственно с одноименными шпильками, найденными алгоритмом. Оценка «2» ставится алгоритмически найденной шпильке, если максимум модулей разностей по координатам B и C петель (B,C) ее и соответствующей биологической шпильки составляет не более чем 5 нуклеотидов, и аналогичный максимум по началам и концам A и D ее и биологической шпильки не более чем 7 нуклеотидов. Оценка «1», если максимальная разность координат петель B и C найденной и биологической шпилек составляет не более чем 5 нуклеотидов, и максимальная разность координат начала и конца рассматриваемых шпилек (A,D) более чем 7 нуклеотидов. Оценка «0», если максимальная разность координат B и C, A и D петли найденной и биологической шпилек составляет более чем соответственно 5 и 7 нуклеотидов. Пометка «н/и» ставится в тех позициях ячейки, когда не были известны биологические антитер-

минатор и пауза (но был известен биологический терминатор). Знак «–» указывает на то, что алгоритм не нашел ответа; иначе знак «+».

Название гена и организма	Нахождение ответа	Точность ответа
1. EC_trpE	+	(2,1,0)
2. TY_trpE	+	(2,1,0)
3. YP_trpE	+	(2,1,0)
4. VC_trpE	+	(2,1,0)
5. HI_trpE	–	–
6. HI_trpE	–	–
7. EC_pheS	+	(2,2,1)
8. TY_pheS	+	(2,1,2)
9. YP_pheS	+	(2,1,1)
10. HI_pheS	–	–
11. EC_pheA	+	(1,2,1)
12. TY_pheA	+	(2,1,0)
13. YP_pheA1	+	(2,2,1)
14. YP_pheA2	+	(2,1,0)
15. EC_hisG	+	(2,н/и, н/и)
16. EC_thrA	+	(2,н/и, н/и)
17. EC_leuA	+	(2,н/и, н/и)
18. EC_ilvG	+	(2,н/и, н/и)
19. EC_ilvB	+	(2,н/и, н/и)

Первый алгоритм также тестировался на биологических последовательностях, содержащих аттенюаторную структуру. На вход алгоритму подавались регуляторные области длины 450 из предыдущего пункта, содержащие аттенуаторную регуляцию. На выходе алгоритм выдавал четверку <Лидерный пептид, Терминатор, Антитерминатор, Пауза> или сообщение об ее отсутствии. В подавляющем большинстве случаев алгоритм нашел ответ с высокой точностью, что видно из таблицы 2.5.

Таблица 2.5. Результаты тестирования первого алгоритма на регуляторных областях, содержащих аттенюаторную регуляцию. В первом столбце указаны назва-

ние организма и ген, регуляторная область которого подавалась на вход алгоритма. Второй столбец содержит знак «+», если для алгоритмически найденной и биологической структур выполнилось: лидерные пептиды совпадают точно, а три шпильки совпадают с точностью до 5 нуклеотидов по координатам A, B, C, D; иначе ставится «–».

Название организма и гена	Результат
1. EC_trpE	+
2. TY_trpE	+
3. YP_trpE	–
4. VC_trpE	+
5. HI_trpE	–
6. EC_pheS	+
7. TY_pheS	+
8. YP_pheS	+
9. HI_pheS	–
10. EC_pheA	+
11. TY_pheA	+
12. YP_pheA1	–
13. YP_pheA2	–
14. EC_hisG	+
15. EC_thrA	+
16. EC_leuA	+
17. EC_ilvG	+
18. EC_ilvB	–

§2.5. Второй алгоритм: тестирование на биологических последовательностях, содержащих альтернативные структуры типа Т-бокс. На вход алгоритма подавались регуляторные области длины 350-550, содержащие альтернативную регуляцию типа Т-бокс. На выходе алгоритм выдавал пары шпилек вида <Антитерминатор, Терминатор> или сообщение об их отсутствии. В этом случае терминаторы нашлись с точностью до 1-2 нуклеотидов; точность нахождения антитерминатора указана в таблице 2.6 и обычно она была весьма высокой.

Таблица 2.6. Результат тестирования второго алгоритма на регуляторных областях, содержащих структуры типа Т-бокс. Первый столбец содержит имя организма и гена, для которых искалась эта альтернативная регуляция. Во втором столбце указан знак «+», когда терминатор найден точно и максимальное покоординатное расстояние между координаторами A, B, C, D биологического и алгоритмически найденного антитерминаторов составляло не более чем 3 нуклеотида; указан знак «±», когда терминатор нашелся точно и антитерминатор имеет аналогичное расстояние не более чем на 5 нуклеотидов по координатам B и C и не более чем на 7 нуклеотидов по координатам A и D; знак «–», когда терминатор нашелся точно и антитерминатор найден с меньшей точностью, чем указано выше.

Имена организма и гена	Качество соответствующего антитерминатора
BS_serS	+
BE_serS	+
HD_serS	–
BQ_serS	–
SA_serS	–
EF_serS	+
LO_serS	+
LLX_serS	+
BE_alas	+
BE_cysE	–
BE_glyQ	+
BE_hisS	±
BE_ileS	+
BE_ilvB	±
BE_leuS	+
BE_pheS	±
BE_proB	±
BE_thrS	+
BE_trpS	–

BE_tyrS	+
BE_valS	±
BH_0807	–

§2.6. Второй алгоритм: тестирование на биологических последовательностях, не содержащих аттенюаторов по результатам работы первого алгоритма. На вход второму алгоритму подавались лидерные области генов, в которых первый алгоритм не нашел аттенюаторной регуляции (в частности, не нашел лидерного пептида). Из таблицы 2.7 видно, что примерно в 80% случаев второй алгоритм также не нашел в них аттенюаторной регуляции.

Таблица 2.7. Результаты работы второго алгоритма на лидерных областях, в которых первый алгоритм указал на отсутствие аттенюаторной регуляции. Обозначения такие же, как в таблице 2.2.

Название гена	Качество ответов	Оценка алгоритма
1. AB_pheA	1 + + 14 2 + + 4 3 + – 3 4 + – 3	Real
2. AB_hisG	1 + – 7 2 – – 4 3 + – 3 4 + – 3	Real
3. AB_ilvG	1 + – 5 2 – – 4	Real
4. AB_pheS	1 + – 6 2 + + 6	Real
5. AB_thrC	–	False
6. AB_trpB	1 + – 3	False
7. AB_trpC	–	False
8. AB_trpD	1 – – 3	False
9. AB_trpE	1 – – 8	False

10. EO_ilvB	1 – – 5 2 – – 4 3 + – 4 4 – – 3	Real
11. HI_trpBA	1 + – 4 2 + – 4 3 – – 3 4 + – 4	Real
12. HI_trpE	1 + – 6 2 – – 3 3 + – 3	Real
13. HI_ilvB	1 + + 11 2 – – 6 3 – – 6 4 + + 5 5 + + 4 6 + + 3	Real
14. HI_ilvA	1 + + 15 2 + – 3 3 – – 4	Real
15. HI_pheS	1 + – 15 2 + – 4 3 + – 4 4 – – 3	Real
16. KP_hisD	–	False
17. KP_leuA	–	False
18. KP_leuC	1 + – 6	False
19. KP_pheT	–	False
20. KP_thrA	–	False
21. KP_trpB	1 + – 4	False
22. KP_trpE	1 + – 3	False
23. PA_hisG	–	False
24. PA_leuA	–	False
25. PA_leuB	–	False
26. PA_leuC	–	False

27. PA_leuD	–	False
28. PA_pheA	–	False
29. PA_pheS	1 + – 13	False
30. PA_thrA	–	False
31. PA_thrC	–	False
32. PA_trpE	–	False
33. VK_pheA	1 + + 15	Real
34. VK_pheS	1 + – 6 2 + – 5 3 – – 3 4 – + 4	Real
35. VK_trpB	1 + – 5 2 + – 3	Real
36. VK_trpD	1 + – 6 2 + – 6 3 + – 4 4 + – 3	Real
37. VK_trpE	1 – – 3 2 + – 5 3 – + 3	Real
38. SH_pheS	1 + – 4 2 + – 3 3 + – 3 4 + – 3	Real
39. TY_trpC	1 + – 5 2 + – 6 3 + – 3	Real
40. VC_pheS	–	False
41. XCA_hisG	–	False
42. XCA_pheA	1 + + 3	False
43. XCA_pheS	1 + – 12	False
44. XCA_trpA	1 + – 4	False
45. XCA_trpB	–	False
46. XCA_trpE	–	False
47. XFA_hisG	1 + – 4	False

48. XFA_pheA	1 – – 6 2 + – 4 3 + – 4 4 + + 3 5 + + 3	Real
49. XFA_pheS	1 + – 4 2 + – 5	Real
50. XFA_thrA	1 + – 6	False
51. XFA_trpB	–	False
52. XFA_trpE	1 + – 3 2 + + 4	False

ГЛАВА 3. Массовый поиск аттенюаторной регуляции

В этой главе разработанная нами компьютерная программа LLLM, реализующая два выше описанных и тестированных алгоритма, применялась для массового поиска аттенюаторной регуляции у протеобактерий, у фирмикутов и у бактерий из групп Bacteroidetes/Chlorobi и Thermotogales.

Процедура поиска заключалась в следующем: полные и частично секвенированные последовательности бактериальных геномов выгружались из базы данных Genbank (института NCBI). Список рассмотренных нами геномов приводятся в таблице 3.1. Похожесть белков определялась с помощью алгоритма Смита-Уотермана, реализованного программой GenomeExplorer[7]. Ортологичные белки находились как наилучшие двунаправленные хиты[8] сразу для нескольких геномов. В некоторых случаях приходилось дополнительно строить филогенетические деревья белков. Например, у *P. multocida* был найден ген, высоко гомологичный двум генам *ilvG* и *ilvB* из *E. coli*, и только филогенетическое дерево построенное для данного семейства белков позволило аннотировать его как *ilvG*. Филогенетические деревья строились методом максимального правдоподобия, реализованного в пакете[9] Phylip, множественные выравнивания выполнялись программой[10] CLUSTAL_W. Для аннотации генов также применялась программа поиска трансмембранных сегментов (TMpred, http://www.ch.embnet.org/software/TMPRED_form.html) и использовались базы, содержащие функ-

[7] Mironov, A.A., Vinokurova, N.P. and Gelfand, M.S. (2000) GenomeExplorer: software for analysis of complete bacterial genomes. Mol. Biol. 34, 222-231.

[8] Tatusov, R.L., Natale, D.A., Garkavtsev, I.V., Tatusova, T.A., Shankavaram, U.T., Rao, B.S., Kiryutin, B., Galperin, M.Y., Fedorova, N.D. and Koonin, E.V. (2001) The COG database: new developments in phylogenetic classification of proteins from complete genomes. Nucleic Acids Res. 29, 22-28.

[9] Felsenstein, J. (1981) Evolutionary trees from DNA sequences: a maximum likelihood approach. J. Mol. Evol. 17, 368-376.

[10] Thompson, J.D., Gibson, T.J., Plewniak, F., Jeanmougin, F. and Higgins, D.G. (1997) The CLUSTAL_X windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools. Nucleic Acids Res. 25, 4876-4882.

циональную и структурную аннотацию белков, в частности[9,11] COG и InterPro. Затем, с учетом найденных сигналов регуляции, были получены оперонные структуры для рассмотренных нами организмов и генов биосинтеза аминокислот (приведены в таблицах 3.2-3.5).

Выявление потенциальных оперонных структур позволило найти предполагаемые лидерные области оперонов для поиска в них различных типов регуляции. Поиск аттенюаторных регуляторных структур выполнялся с помощью нашей программы LLLM, а также (в случае белок-ДНКовой регуляции) с помощью программы[12] IRSA или иными средствами поиска других упомянутых в таблицах 2-5 типов регуляции. На защиту выносятся только случаи аттенюаторной регуляции, поэтому другие регуляции подробно не обсуждаются.

Средством подтверждения обоснованности наших результатов об аттенюаторной регуляции служило выравнивание лидерных участков, содержащих соответствующие структуры (с небольшими полями) от начала лидерного пептида до окончания терминатора и участка остатков урацила (при его наличии). Эти выравнивания приведены на рисунках 3.3- 3.6. Соответствующие метаболические карты приведены на рисунке 3.2.

§3.1. Особенности аттенюаторной регуляции биосинтеза гистидина, треонина, разветвленных и ароматическмх аминокислот

Биосинтез изолейцина, лейцина и валина (сокращенно ILV). Новые возможные транскрипционные аттенюаторы найдены у гамма-, альфа-, бета-протеобактерий. Возможные аттенюаторы оперонов *ilvGMEDA* найдены у энтеробактерий, у Pasteurellales (только в *P. multocida*), у Vibrionales, Alteromonadales (в *S.*

[11] Mulder, N.J., Apweiler, R., Attwood, T.K., Bairoch, A., Bateman, A., Binns, D., Biswas, M., Bradley, P., Bork, P., Bucher, P., Copley, R., Courcelle, E., Durbin, R., Falquet, L., Fleischmann, W., Gouzy, J., Griffith-Jones, S., Haft, D., Hermjakob, H., Hulo, N., Kahn, D., Kanapin, A., Krestyaninova, M., Lopez, R., Letunic, I., Orchard, S., Pagni, M., Peyruc, D., Ponting, C.P., Servant, F. and Sigrist, C.J. (2002) InterPro: an integrated documentation resource for protein families, domains and functional sites. Brief Bioinform. 3, 225-235.

[12] Данилова Л.В., Горбунов К.Ю., Гельфанд М.С., Любецкий В.А. (2001) Алгоритм выделения регуляторных сигналов в последовательностях ДНК. Мол. биол., том 35, № 6, с. 987-995.

oneidensis), у Xanthomonadales. У Pseudomonadales найдена возможная аттенюация отдельно лежащего гена *ilvA*. У энтеробактерий найдена аттенюаторная регуляция оперона *ilvBN*, содержащего гены, кодирующие один из изозимов ацетолактат-синтазы. Гены лидерных пептидов оперона *ilvG* содержат регуляторные кодоны трех аминокислот – изолейцин, лейцин и валин (как и в экспериментально изученном случае *E. coli*), а оперона *ilvBN* регуляторные кодоны только двух аминокислот – лейцина и валина.

Структура потенциальных оперонов биосинтеза ILV весьма разнообразна. Например, у энтеробактерий и Vibrionales она имеет классический вид *ilvGMEDA*, а у Xanthomonadales – вид *ilvCGM-tdcB-leuA*. Здесь ген *tdcB* предположительно регулируется вместе с этим опероном и кодирует треонин дегидратазу, которая участвует в биосинтезе изолейцина.

У *P. multocida* отмечен ген с неизвестной функцией (гомологичный гену *ygeA* у *E. coli*), который локализуется в *ilv*-опероне *ilvGM-ygeA-ilvDA*. Ген *ygeA* слабо похож на аспартат рацемазу *racX* из *B. subtilus*; весьма вероятно, что он кодирует новый тип рацемазы разветвленных аминокислот.

Среди гамма-протеобактерий оперон *leu* регулируется аттенюацией у энтеробактерий, Pasteurellales, Vibrionales, Alteromonadales, но не у Pseudomonadales и других видов. Регуляторными во всех этих случаях являются лейциновые кодоны.

В группе альфа-протеобактерий впервые обнаружена потенциальная аттенюаторная регуляция единственной ацетолактат синтетазы *ilvIH* у Rhizobiales (*Sinorhizobium meliloti, Agrobacterium tumefaciens, Mesorhizobium loti, Bradyrhizobium japonicum, Rhodopseudomonas palustris, Brucella melitensis*), у *Rhodobacter* spp., *Magnetospirillum magnetotacticum* и у *Caulobacter crescentus*. Регуляторными кодонами этой аттенюации являются кодоны изолейцина, лейцина и валина. Заметим, что у гамма-протеобактерий опероны, участвующие в биосинтезе двух ацетолактат синтетаз IlvGM и IlvBN (у энтеробактерий), но не IlvIH, регулируются аттенюацией.

Группа ортологов гена *leuA* изопропилмалат синтазы из *E. coli* обнаружена у гамма-протеобактерий (исключая Pseudomonadales) и у некоторых альфа-протеобактерий. Группа генов, названная нами *leuA2*, которые похожи на гены, кодирующие изопропилмалат синтазу у *Corynebacterium glutamicum*, была найдена у альфа-протеобактерий, у некоторых бета-протеобактерий и у Pseudomonadales. В альфа-протеобактериях оба типа 2-изопропилмалат синтазы LeuA и LeuA2 имеют потенциальные аттенюаторы. Эти предполагаемые аттенюаторы имеют лидерные пептиды с лейциновыми регуляторными кодонами, но их терминаторы слабые и не имеют участков остатков урацила. Эта ситуация кажется подобной регуляции оперонов *trpE* и *trpGDC* у Pseudomonadales, где имеется аттенюаторная регуляция в отсутствие ро-независимой терминаторной структуры.

Биосинтез гистидина. Потенциальные аттенюаторы были найдены у многих гамма-протеобактерий, у фирмикутов и у бактерий из групп Bacteroidetes/Chlorobi, и Thermotogales. В большей части гамма-протеобактерий (энтеробактерии, Pasteurellales, Vibrionales, Alteromonadales) имеется единый *his*-оперон, предположительно регулируемый аттенюаторами с гистидиновыми регуляторными кодонами, и имеется выраженная терминатор/антитерминаторная структура. Аттенюация не была найдена для *his*-генов у Pseudomonadales, Xanthomonadales и у некоторых других гамма-протео-бактерий.

У Bacillus/Clostridium, Bacteroidetes/Chlorobi, Thermotogales гистидиновые регулоны включают большую часть обычных *his*-генов и также ген *hisZ* или *hisS* гистидил-тРНК-синтетазы; они регулируются аналогично.

Отметим разнообразие механизмов регуляции *his*-генов. Например, у *Lactococcus lactis* и *Streptococcus mutans his*-оперон регулируется с помощью антитерминаторного механизма с образованием структуры Т-бокса[13], а у *Bacillus cereus* и

[13] Delorme, C., Ehrlich, S.D. and Renault, P. (1999) Regulation of expression of the Lactococcus lactis histidine operon. J. Bacteriol. 181, 2026-2037. Витрещак А.Г. Частное сообщение.

Clostridium difficile он предположительно регулируется классической аминокислотной аттенюацией; в тоже время, другие *Streptococcus* spp., а также *Entrerococcus* spp. лишены *his*-генов.

Что касается гистидил-тРНК-синтетазы, то у *Bacillus cereus* имеются три кодирующие ее гена *hisZ*, *hisZ2* и *hisS*. Оба гена *hisZ* и *hisZ2* регулируются аттенюацией: один в составе *his*-оперона, другой как отдельный ген. Третий ген *hisS*, как и его ортологи у *Bacillus* spp., *Listeria* spp., *Enterococcus* spp. и *Lactococcus lactis*, (предположительно) регулируется с помощью антитерминаторного механизма с образованием структуры Т-бокса[14].

Показана регуляция гистидином следующих генов. У *H. influenzae* ген *HI0325* кодирует возможный транспортер (с 10 трансмембранными сегментами) и имеет потенциальный аттенюатор. В ряде геномов (в частности, у *Fusobacterium nucleatum* и *Bacillus halodurans*) этот ген кластеризуется с генами утилизации гистидина (*hut* локус). Возможно, этот ген и его ортологи (например, *yuiF* у *B. subtilis*) образуют новое семейство гистидиновых транспортеров.

У *B. cereus* ген *BC0629*, возможно, регулируется аттенюатором с гистидиновыми регуляторными кодонами. Этот ген ортологичен *yvsH* у *B. subtilis*, гомологичен аргинин:орнитин антипортеру *arcD* у *Pseudomonas aeruginosa* и лизиновому транспортеру *lysI* у *Corinobacterium glutamicum.* Эти белки относятся к семейству APA антипортеров аминокислот и полиаминов.

У *B. cereus* имеются два паралога *yvsH* (BC0629 и BC0865), из которых первый имеет аттенюаторную регуляцию с гистидиновыми регуляторными кодонами, а второй (предполагаемый лизиновый транспортер) регулируется лизином с помощью лизин-специфичного регуляторного элемента[15]. Подобная ситуация наблюдается у *L.*

---

[14] Chopin, A., Biaudet, V. and Ehrlich, S.D. (1998) Analysis of the Bacillus subtilis genome sequence reveals nine new T-box leaders. Mol. Microbiol. 29, 662-664. Витрещак А.Г. Частное сообщение.

[15] Rodionov, D.A., Vitreschak, A.G., Mironov, A.A. and Gelfand, M.S. (2003) Regulation of lysine biosynthesis and transport genes in bacteria: yet another RNA riboswitch? Nucleic Acids Res. 31, 6748-6757.

lactis, где видны два паралогических транспортера LysP и LysQ. Оба белка подобны (более, чем на 50%) экспериментально подтвержденной лизиновой пермеазе LysP из *E. coli*. У *L. lactis* ген *lysP* регулируется *LYS*-элементом (и, по видимому, участвует в транспорте лизина), а ген *lysQ* предположительно регулируется аттенюатором с гистидиновыми регуляторными кодонами. Таким образом, эти два транспортера могут иметь различное сродство к лизину и гистидину, и потому по разному регулироваться.

Все гены гистидинового регулона были найдены во всех анализируемых бактериях, за исключением гистидинол фосфатазы HisB у *Pseudomonas* spp.

Найдены три негомологичных гена с неизвестной функцией (*actX2*, *vatB*, *actX3* соответственно у *Mannheimia haemolytica, Pasteurella multocida, Polaribacter filamentus*), возможно кодирующих ацетилтрансферазы и регулируемые совместно с *his*-генами. Эти белки могли бы катализировать превращение гистамина в 4-бета-ацетиламиноэтил-имидазол (EC 2.3.1.-).

Биосинтез треонина. В опероне биосинтеза треонина у энтеробактерий, Pasteurellales, Vibrionales, Alteromonadales и Xanthomonadales наблюдается обычный порядок генов *thrABC*. У Pseudomonadales и еще у некоторых бактерий гены треонинового биосинтеза разбросаны по геному. Более того, у энтеробактерий, Pasteurellales, Vibrionales, Alteromonadales и Xanthomonadaels ген *thrA* кодирует бифункциональный белок – аспартат киназу/гомосерин дегидрогеназу, а у Pseudomonadales и у некоторых других гамма-протеобактерий *thrA2* (аспартат киназа) и *hom* (гомосерин дегидрогеназа) расположены в разных локусах. У Pseudomonadales наблюдались два гена гомосерин киназы *thrB2* и *thrH*, которые не гомологичны *thrB* из *E. coli*.

Треониновые опероны регулируются аттенюацией у энтеробактерий, Pasteurellales, Vibrionales, Alteromonadales и *Xanthomonas campestris*. Все предполагаемые ат-

тенюаторы имеют треониновые и изолейциновые регуляторные кодоны, а также выраженные терминаторы и антитерминаторы.

Нами установлено, что у Pasteurellales (*Haemophylus influenzae, Pasteurella multocida, Actinobacillus actinomycetemcomitans* и *Mannheimia haemolytica*) регуляторными кодонами служат не только треонин и изолейцин, но и метионин. В самом деле, предполагаемая регуляция *thr*-оперона у Pasteurellales концентрациями треонина, изолейцина и метионина может быть обоснована наличием у Pasteurellales только одной аспартат киназы/гомозерин дегидрогеназы, вместо двух изозимов ThrA и MetL у других гамма-протеобактерий, которая входит в метаболические пути синтеза этих трех аминокислот.

Третья монофункциональная аспартат киназа LysC имеется у трех из пяти Pasteurellales, а именно, у *P. multocida, Haemophylus ducrei* и *M. Haemolytica*; и экспрессия гена *lysC* предположительно регулируется лизином посредством *LYS*-элементов (как у *E. coli*).

Биосинтез ароматических аминокислот (триптофана, фенилаланина) и фенилаланил-тРНК-синтетазы. Были найдены предполагаемые *trp*-, *pheA*- и *pheST*-опероны у альфа-, бета- и у многих новых гамма-протеобактерий.

Аттенюаторная регуляция *trp* была показана для энтеробактерий, Vibrionales, Alteromonadales с регуляторными кодонами триптофана и при наличии сильных терминаторов и антитерминаторов. Ген *trp(E/G)*, возникший в результате слияния и кодирующий две компоненты антранилат синтазы (первый шаг триптофанового биосинтеза), возможно, регулируется аттенюацией во всех проанализированных нами альфа-протеобактериях из группы Rhizobiales кроме *Brucella melitensis*.

Оперон *pheA*, возможно, регулируется аттенюаторами (с регуляторными кодонами фенилаланина) у энтеробактерий, Vibrionales, Alteromonadales, а оперон *pheST* регулируется таким же образом только у энтеробактерий.

Опероны *trpE* и *trpGDC* у Pseudomonadales имеют некоторые особенности: несмотря на экспериментальные данные об аттенюаторной регуляции этих оперонов[16], нами найдены только потенциальные лидерные пептиды с парой близких триптофановых кодонов, которые хорошо выравниваются для пяти Pseudomonadales, но наша программа не нашла для них ро-независимых терминаторов. Возможно, это объясняется тем, что в этих случаях имеются менее выраженные и стабильные терминаторы и антитерминаторы.

§3.2. Метаболические пути, оперонные структуры и выравнивания аттенюаторов изученных оперонов.

Таблица 1. Список изученных геномов с таксономией и обозначениями. Неполные геномы помечены знаком #.

Phylum/Class	Order	Bacteria	Abbr.
α-proteobacteria	**Rhizobiales**	*Sinorhizobium meliloti*	SM
		Agrobacterium tumefaciens	ATU
		Rhizobium leguminosarum	LE
		Mesorhizobium loti	MLO
		Bradyrhizobium japonicum	BJA
		Rhodopseudomonas palustris	RPA
		Brucella melitensis	BME
	Sphingomonadales	*Sphingomonas aromaticivorans #*	SAR
	Rhodobacteralles	*Rhodobacter sphaeroides #*	RS
	Rhodospirillales	*Magnetospirillum magnetotacticum #*	MMA
		Rhodospirillum rubrum #	RR
	Rickettsiales	*Rickettsia prowazekii*	RP
	Caulobacterales	*Caulobacter crescentus*	CO
β-proteobacteria		*Burkholderia pseudomallei #*	BPS
		Ralstonia solanacearum	RSO
		Nitrosomonas europaea	NE
		Bordetella pertussis	BP
		Neisseria meningitidis	NM
γ-proteobacteria	**Enterobacteriales**	*Escherichia coli*	EC
		Salmonella typhi	TY
		Klebsiella pneumoniae #	KP
		Erwinia carotovora	EO
		Yersinia pestis	YP
	Pasteurellales	*Haemophylus influenzae*	HI
		Pasterella multocida	VK
		Actinobacillus actinomycetemcomitans #	AB
		Mannheimia haemolytica #	PQ
	Vibrionales	*Vibrio cholerae*	VC
		Vibrio vulnificus	VV
		Vibrio parahaemolyticus	VP
	Alteromonadales	*Shewanella oneidensis*	SON

[16] Olekhnovich, I. and Gussin, G.N. (2001) Effects of mutations in the Pseudomonas putida miaA gene: regulation of the trpE and trpGDC operons in P. putida by attenuation. J. Bacteriol. 183, 3256-3260.

		Microbulbifer degradans #	MDE
	Pseudononadales	*Pseudomonas aeruginosa*	PA
		Pseudomonas putida	PP
		Pseudomonas fluorescens #	PU
		Pseudomonas syringae	PY
		Azotobacter vinelandii #	AV
		Acinetobacter spp. #	AC
	Xanthomonadales	*Xanthomonas campestris*	XCA
		Xylella fastidiosa	XFA
Firmicutes	**Bacillales**	*Bacillus subtilis*	BS
		Bacillus cereus	ZC
		Bacillus halodurans	HD
		Bacillus stearothermophilus #	BE
		Oceanobacillus iheyensis	OI
	Lactobacillales	*Enterococcus faecalis*	EF
		Enterococcus faecium	EFA
		Streptococcus mutans	SM
		Streptococcus pyogenes	ST
		Streptococcus pneumoniae	SPY
		Streptococcus equi #	SEQ
		Streptococcus agalactiae	SAQ
	Clostridiales	*Clostridium acetobutylicum*	CA
		Clostridium perfringes	CP
		Clostridium botulinum	CB
		Clostridium difficile #	DF
		Clostridium tetani	CT
		Clostridium thermocellum	CTE
Bacteroidetes /Chlorobi		*Bacteroides fragilis* *Porphyromonas gingivalis*	BX PFI
Thermotogae		*Thermotoga maritima* *Petrotoga miotherma*	TM PMI

Таблица 2. Предсказанные типы регуляции и оперонные структуры ILV генов. Предсказанные аттенюаторы помечены знаками & или % (если шпильки структуры менее стабильные). Концы контигов помечены квадратными скобками. Известные регуляторы из LysR семейства и возможные (знак REG) помечены жирным шрифтом.

α	*Sinorhizobium meliloti*	SM	*leuB; leuC; leuD; %leuA2; ilvE; ilvD; ilvA; **REG2**-ilvC; &ilvIH;*
	Agrobacterium tumefaciens	ATU	*leuDB; leuC; %leuA2; ilvE; ilvD; ilvA; **REG2-**ilvC &ilvIH;*
	Mesorhizobium loti	MLO	*leuD-//-leuB; leuC; %leuA2; ilvE; ilvD; ilvA; **REG2**-ilvC &ilvIH;*
	Bradyrhizobium japonicum	BJA	*leuB; leuC; leuD; %leuA; ilvE; ilvD; ilvA; ilvC; &ilvI-x-H;*
	Rhodopseudomonas palustris	RPA	*leuB; leuC-//-leuD; %leuA; leuA2; ilvE; ilvD; ilvA; &ilvIH-//-ilvC;*
	Brucella melitensis	BME	*leuB; leuC; leuD; %leuA2; ilvE; ilvD; ilvA; &ilvIH-//-ilvC;*
	Sphingomonas aromaticivorans #	SAR	*leuC-//-D; -leuB-; %leuA2; ilvE; ilvD; ilvA; ilvIHC;*
	Rhodobacter sphaeroides #	RS	*leuCD; leuB; %leuA; ilvE; ilvD; ilvA; ilvC; &ilvIH;*
	Magnetospirillum magnetotacticum #	MMA	*leuCDB; %leuA; leuA2; ilvE; ilvD; ilvA; &ilvIHC-//-leuA;*
	Rhodospirillum rubrum #	RR	*leuCDB; leuA2; leuA2; ilvE; ilvD; ilvA; ilvIHC;*
	Rickettsia prowazekii	RP	-
	Caulobacter crescentus	CO	*leuCD-x-B; %leuA; ilvE; ilvD; ilvA; ilvC; &ilvIH;*
β	*Burkholderia pseudomallei #*	BPS	*leuCDB leuA2; ilvE; ilvD; ilvA; ilvIHC-//-%leuA;*
	Ralstonia solanacearum	RSO	*leuCDB; leuA2; ilvE; ilvD; ilvA; ilvIHC-//-%leuA;*
	Nitrosomonas europaea	NE	*leuCDB; ilvE; ilvD; ilvA; ilvIHC-//-leuA;*
	Bordetella pertussis	BP	*leuCDB; leuA2; %leuA2; ilvE; ilvD; ilvA; ilvIHC;*
	Neisseria meningitidis	NM	*leuCD; leuB; leuA; ilvE; ilvD; ilvA; ilvIHC;*
γ	*Escherichia coli*	EC	***leoO**<-->&leuABCD; &ilvGMEDA--**ilvY**<->ilvC; &ilvBN; ilvIH;*
	Salmonella typhi	TY	***leoO**<-->&leuABCD; &ilvGMEDA--**ilvY**<->ilvC; &ilvBN; ilvIH;*
	Klebsiella pneumoniae #	KP	***leoO**<-->&leuABC[&ilvGMEDA--**ilvY**<->ilvC; &ilvBN;*
	Yersinia pestis	YP	***leoO**<-->&IS-leuABCD; &ilvGMEDA--**ilvY**<->ilvC; &ilvBN; ilvIH;*
	Erwinia carotovora	EO	*&leuABCD; &ilvGMEDA--**ilvY**<->ilvC; &ilvBN; ilvIH;*
	Haemophylus influenzae	HI	*&leuABCD; ilvGDA; ilvE; **ilvY**<->ilvC; ilvIH;*
	Pasterella multocida	VK	*&leuABCD; &ilvGM-ygeA-DA; ilvE; **ilvY**<->ilvC; ilvIH;*
	Mannheimia haemolytica #	PQ	*&leuA; leuB;]leuCD; ilvGM; -ilvE-; [ilvD[; ilvA-; ilvC; ilvIH;*
	Vibrio cholerae	VC	***leoO**<-->&leuABCD; &ilvGMEDA; **ilvY**<->ilvC; ilvIH;*
	Vibrio vulnificus	VV	***leoO**<-->&leuABCD; &ilvGMEDA; **ilvY**<->ilvC; ilvIH;*
	Vibrio parahaemolyticus	VP	***leoO**<-->&leuABCD; &ilvGMEDA; **ilvY**<->ilvC; ilvIH;*
	Shewanella oneidensis	SON	*&leuABCD; &ilvGMDA; -ilvE-; **ilvY**<->ilvC; ilvIH;*
	Pseudomonas aeruginosa	PA	***REG**<-->leuCD-yafE-leuB; leuA2; -ilvE-; ilvD; ilvD; X-ilvA; ilvA; ilvIHC;*
	Pseudomonas putida	PP	***REG**<-->leuCD-yafE-leuB; leuA2; ilvE; ilvD; &ilvA; ilvA; ilvIHC;*

	Pseudomonas fluorescens	PU	***REG***<-->*leuCD-x-B; leuA2; ilvE; ilvD; &ilvA; ilvA; ilvIHC;*
	Pseudomonas syringae	PY	***REG***<-->*leuCDB; leuA2; ilvE; ilvD; &ilvA; ilvIHC;*
	Acinetobacter spp.	AC	***REG***<-->*leuCxD--B; leuA2; -ilvE-; ilvD; ilvD; ilvA; ilvIHC;*
	Azotobacter vinelandii #	AV	***REG***<-->*leuCDB; -leuA- -ilvE-; ilvD; ilvD; ilvA; ilvIHC;*
	Microbulbifer degradans #	MDE	***REG***<-->*leuCDB; -leuA- -ilvE-;]ilvD ilvA;* ***ilvY***<->*ilvC; ilvIH;*
	Xanthomonas campestris	XCA	***REG***<-->*leuCD-yafE-leuB &ilvCGM-tdcB-leuA; ilvE; ilvD; ilvA;*
	Xylella fastidiosa	XFA	*leuCD--B; &ilvCGM-tdcB-leuA; ilvE; ilvD;*

Таблица 3. Возможные оперонные структуры и предсказанная регуляция HIS генов. Обозначения как в таблице 2. Слияние генов *hisI* и *hisE* помечено (I/E). Гистидин-зависимые структуры с образованием Т-бокса помечены знаком Т.

γ	*Escherichia coli*	*&hisGDCBHAF(I/E);*
	Salmonella typhi	*&hisGDCBHAF(I/E);*
	Klebsiella pneumoniae #	*&hisGDCBHAF(I/E);*
	Yersinia pestis	*&hisGDCBAHAF(I/E);*
	Erwinia carotovora	*&hisGDCBHAF(I/E);*
	Vibrio cholerae	*&hisGDCBHAF(I/E);*
	Vibrio vulnificus	*&hisGDCBHAF(I/E);*
	Vibrio parahaemolyticus	*&hisGDCBHAF(I/E);*
	Shewanella oneidensis	*&hisGDCBHAF(I/E);*
	Haemophylus influenzae	*&hisGDCBHAF(I/E); &HI0325;*
	Pasterella multocida	*&hisG-x-vatB-DCB--HAF--(I/E);*
	Actinobacillus actinomycetemcomitans #	*his(I/E);*
	Mannheimia haemolytica #	*]hisGD; &x-actX2-hisCB[ ]hisH-x-AF-x-(I/E);*
	Pseudomonas aeruginosa	*-hisGDC; his(B1)H-x1-AF; hisIE;*
	Pseudomonas putida	*-hisGDC; his(B1)H-x1-AF; hisIE;*
	Pseudomonas fluorescens #	*hisGDC; his(B1)H-x1-AF; hisIE;*
	Pseudomonas syringae	*hisGDC; his(B1)H-x1-AF; hisIE;*
	Azotobacter vinelandii #	*-hisGDC; his(B1)H-x1-AF; hisIE;*
	Acinetobacter spp. #	*-hisG[ ]hisDC; his(B1)H-x-A; hisF; his(I/E);*
	Microbulbifer degradans #	*-hisGD; hisC; his(B1)HAF; hisIE;*
	Acidithiobacillus ferrooxidans #	*-hisGDC(B1)HAF(I/E);*
	Xanthomonas campestris	*x2-hisGDCBHAF(I/E);*
	Xylella fastidiosa	*x2-hisGDCBHAF(I/E);*
α	*Caulobacter crescentus*	*&hisZ-x-hisG; x-his(B1)HAFE; -hisD-; hisC; hisI-;*

B/C	*Bacillus cereus*	*&hisZ-hisGD(B1)HAFIE-hisB2; x-hiB2; hisC; -hisC-; &hisZ2; ThisS-aspS; &yvsH1; LyvsH2;*
	other Bacillus sp. (BS, BE, HD)	*hisZ-hisGD(B1)HAF(I/E); -hisC-; ThisS-aspS;*
	Clostridium difficile #	*&hisZ-hisGC(B1)HAF(I/E); hisD; hisB2-x; hisB2; hisB2; -hisS-aspS;*
	Lactococcus lactis	*ThisC-hisZ-hisGD-x-(B1)-x-HAF(I/E); ThisS-aspS &lysQ; LlysP;*
	Streptococcus mutans	*ThisC-hisZ-hisGD-x-(B1)-x-H--AF(I/E); hisS—aspS*
	other Streptococcus sp. (SP, SPY, SEQ, SAG) Enterococcus sp. (EF, EFA)	*- hisS—aspS*
Therm	*Thermotoga maritima*	*&IS-hisS-hisGDC(B1)HAF(I/E); -hisB2;*
	Petrotoga miotherma #	*&hisS-hisGDC(B1)HAF(I/E); -hisB2;*
CFB	*Polaribacter filamentus #*	*&hisGDCBHAF-actX3-(I/E)*
	Bacteroides fragilis	*&hisGDCB; hisHAF(I/E);*

Таблица 4. Возможные оперонные структуры и предсказанная регуляция THR генов. Обозначения как в таблице 2.

γ	*Escherichia coli*	*&thrABC;*
	Salmonella typhi	*&thrABC;*
	Klebsiella pneumoniae #	*&thrABC;*
	Yersinia pestis	*&thrABC;*
	Erwinia carotovora	*&thrABC;*
	Vibrio cholerae	*&thrABC; thrA2;*
	Vibrio vulnificus	*&thrABC;*
	Vibrio parahaemolyticus	*&thrABC;*
	Shewanella oneidensis	*&thrABC; thrA2;*
	Haemophylus ducrei	*-*
	Haemophylus influenzae	*&thrABC;*
	Pasterella multocida	*&thrABC;*
	Actinobacillus actinomycetemcomitans #	*&thrAB-x-C;*
	Mannheimia haemolytica #	*&thrAB;]thrC;*
	Pseudomonas aeruginosa	*thrA2; hom-thrC; -thrB2; thrH;*
	Pseudomonas putida	*thrA2; hom-thrC; -thrB2;*
	Pseudomonas fluorescens #	*thrA2; thrA2; hom-thrC; -thrB2; thrH;*
	Pseudomonas syringae	*thrA2; hom-thrC; -thrB2; thrH;*
	Acinetobacter spp. #	*thrA2; hom] thrB2; thrH;*
	Azotobacter vinelandii #	*thrA2; -hom-; -thrC-; x-thrB2;*
	Microbulbifer degradans #	*thrA2; thrA2-; hom-thrC; thrH;*
	Xanthomonas campestris	*&thrAB—thrC;*
	Xylella fastidiosa	*thrABC;*

Таблица 5. Предсказанные операционные структуры и регуляции для генов TRP, PHE и *pheST* генов. Обозначения как в таблице 2. Слияние генов *trpE* и *trpG* помечено как *(E/G)*. Слияние генов *trpC* и *trpF* помечено как *(C/F)*.

α Rhizobiales	*Mesorhizobium loti*	*&trp(E/G);*	*-trpDC-*	*trpFBA;*	*-pheA;*	*pheS-x-T;*
	Brucella melitensis	*&trp(E/G);*	*-trpDC-*	*trpFBA;*	*-pheA;*	*pheST;*
	Sinorhizobium meliloti	*&trp(E/G);*	*-trpDC-*	*trpFBA;*	*-pheA;*	*pheST;*
	Agrobacterium tumefaciens	*&trp(E/G);*	*-trpDC-*	*trpFBA;*	*-pheA;*	*pheST;*
	Rhodopseudomonas palustris #	*&trp(E/G);*	*-trpDC-*	*trpFBA;*	*-pheA;*	*pheS-x-T;*
	Rhizobium leguminosarum #	*&trp(E/G);*	*-trpDC-*	*trpFBA;*	*]pheA;*	*pheS[;*
	Bradyrhizobium japonicum	*&trp(E/G);*	*-trpDC*	*trpFBA;*	*-pheA;*	*pheST;*
β	*Bordetella pertussis*	*&trpEGDC;*			*pheA;*	*pheST;*
γ	*Escherichia coli*	*&trpE(G/D)(C/F)BA;*			*&pheA;*	*&pheST;*
	Salmonella typhi	*&trpE(G/D)(C/F)BA;*			*&pheA;*	*&pheST;*
	Klebsiella pneumoniae #	*&trpEGD(C/F)BA;*			*&pheA;*	*&pheST;*
	Yersinia pestis	*&trpEGD(C/F)BA;*			*IS-pheA;*	*&pheST;*
	Erwinia carotovora	*&trpEGD(C/F)BA;*			*&pheA;*	*&pheST;*
	Haemophylus influenzae	*trpEG-x1-trpD(C/F)-*		*-trpBA;*	*pheA;*	*pheST;*
	Pasteurella multocida	*trpEG<-x-D(C/F)-x-trpBA;*		*trpBA;*	*pheA;*	*pheS-x-T;*
	Actinobacillus actinomycetemcomitans #	*trpEG-x1-trpD(C/F)-*		*-trpBA;*	*pheA;*	*pheS-x-T;*
	Mannheimia haemolytica #	*trpEG-x1-trpD)-*	*trp(C/F)[*	*-trpBA;*	*pheA;*	*pheST;*
	Vibrio cholerae	*&trpEGD(C/F)BA;*			*&pheA;*	*pheST;*
	Vibrio vulnificus	*&trpEGD(C/F)BA;*			*&pheA;*	*pheST;*
	Vibrio parahaemolyticus	*&trpEGD(C/F)BA;*			*&pheA;*	*pheST;*
	Shewanella oneidensis	*&trpEGD(C/F)BA;*			*&pheA;*	*pheST;*
	Pseudomonas aeruginosa	*%trpE;*	*%trpGDC;*	*-trpF;*	*trpI<->trpBA;*	*-pheA-*	*pheST;*
	Pseudomonas putida	*%trpE;*	*%trpGDC;*	*-trpF;*	*trpI<->trpBA;*	*-pheA-*	*pheST;*
	Pseudomonas fluorescens #	*%trpE;*	*%trpGDC;*	*-trpF;*	*trpI<->trpBA;*	*-pheA-*	*pheST;*
	Pseudomonas syringae	*%trpE;*	*%trpGDC;*	*-trpF;*	*trpI<->trpBA;*	*-pheA-*	*pheST;*
	Azotobacter vinelandii #	*%trpE;*	*%trpGDC;*	*-trpF;*	*trpI<->trpBA;*	*-pheA-*	*pheST;*
	Acinetobacter spp. #	*trpE;*	*trpGDC;*	*trpFB--A;*		*pheA-*	*pheST;*
	Microbulbifer degradans #	*trpE;*	*trpGDC;*	*trpFBA-*		*-pheA-*	*pheST;*
	Acidithiobacillus ferrooxidans #	*trpEGDC-;*		*-trpF;*	*trpBA;*	*]pheA-*	*]pheST[;*
	Xanthomonas campestris	*trpE;*	*trpGxDC-*	*-trpF;*	*trpI<->trpB-x-trpA;*	*-pheA-*	*pheST;*
	Xylella fastidiosa	*trpEGDC-x2;*		*-trpF--BA*		*-pheA-*	*pheST;*

Рис. 1. Схематическое изображение типичной «классической» аттенюаторной регуляции.

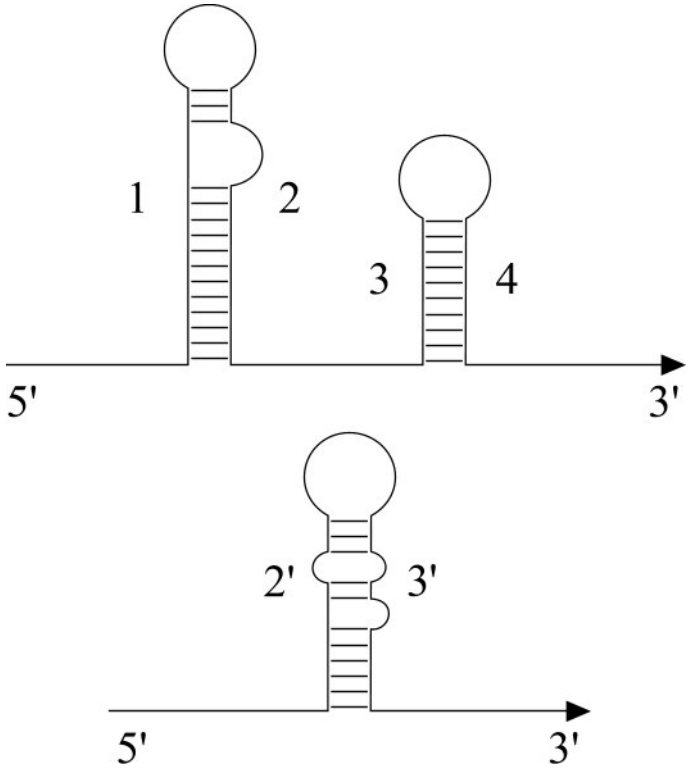


Рис. 2. Пути биосинтеза аминокислот у гамма- и альфа-протеобактерий для четырех случаев: (a) ILV (изолейцина, лейцина, валина), (b) HIS (гистидина), (c) THR (треонина), (d) ароматических аминокислот (триптофана, тирозина, фенилаланина).

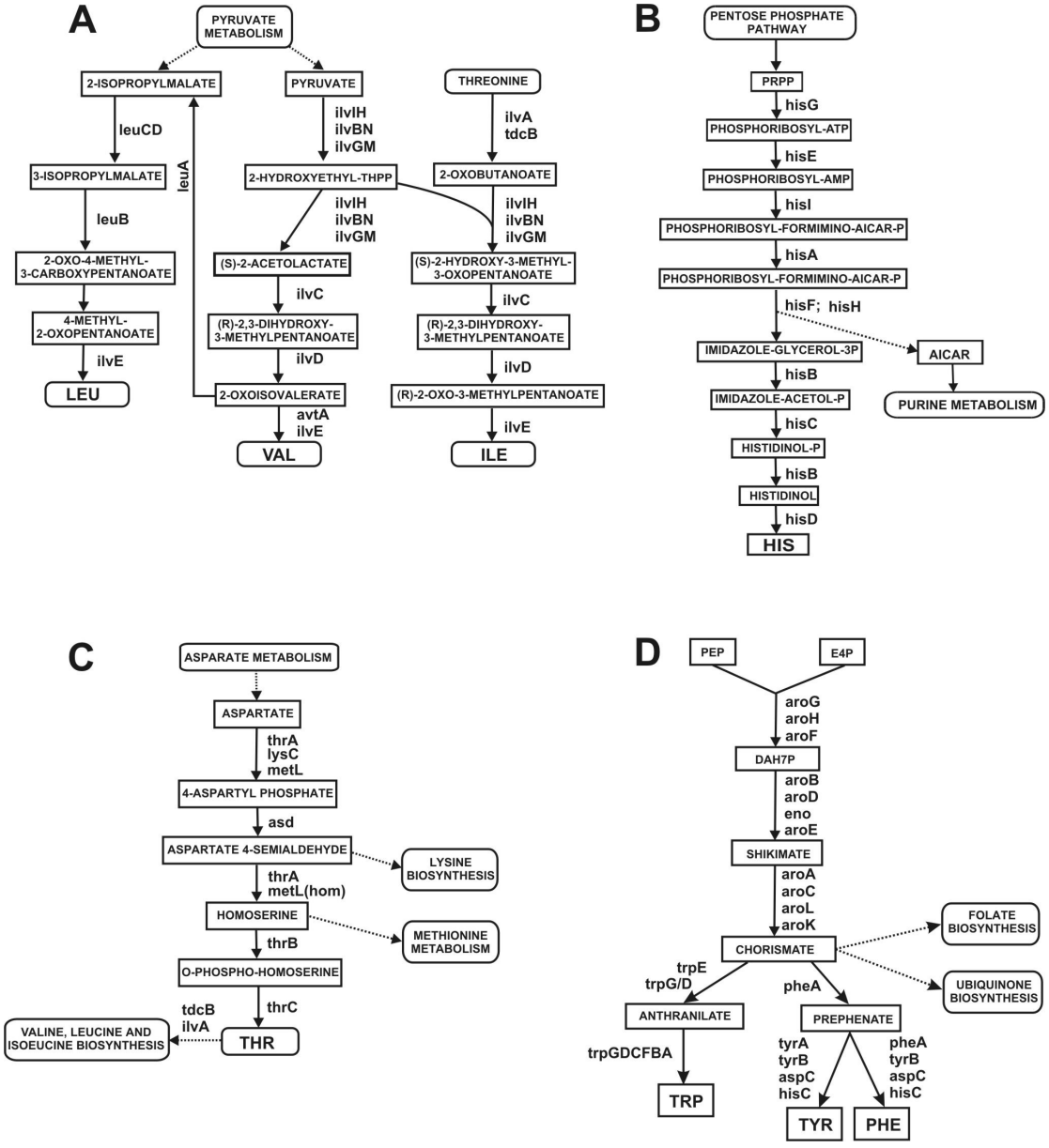
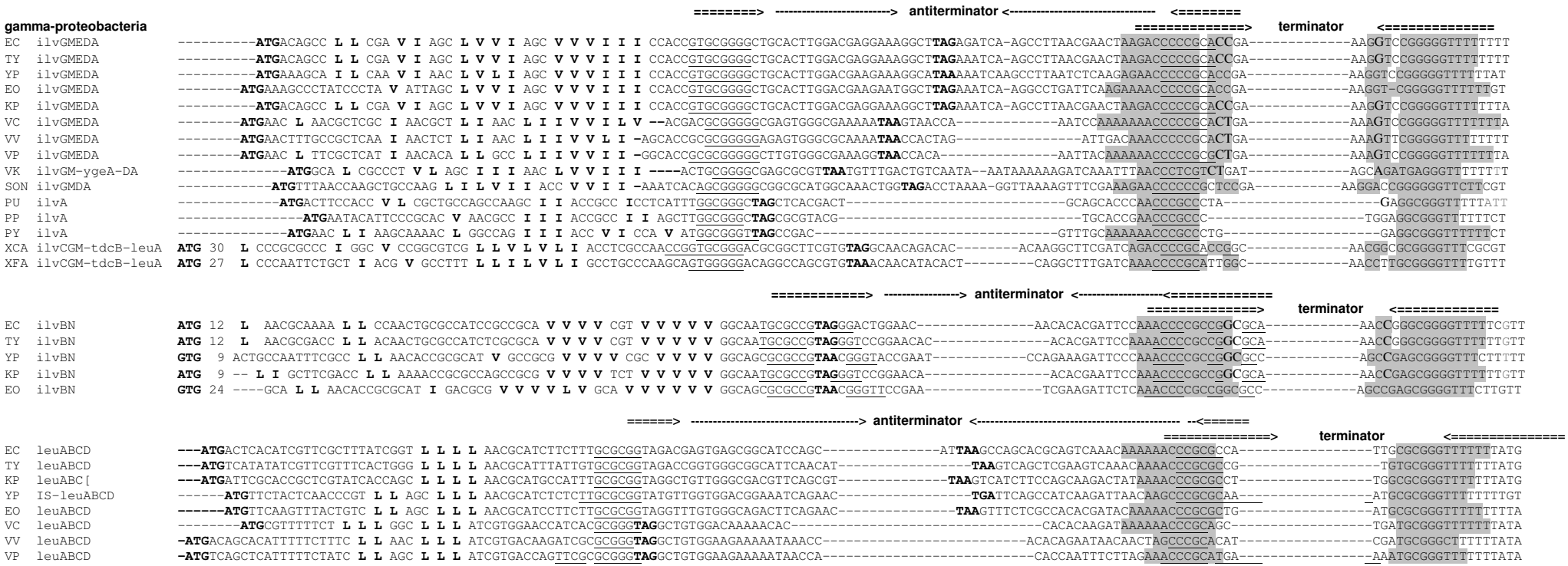


Рис. 3. Выравнивание предсказанных структур транскрипционной аттенюации биосинтеза разветвленных аминокислот у гамма- и альфа-протеобактерий. Выделены: шпильки аттенюаторных структур, лидерные пептиды и регуляторные кодоны в них, консервативные участки в регуляторных шпильках. Слева указаны сокращенные названия организмов, как в таблице 1, и регулируемые опероны. Справа указаны элементы самих аттенюаторов. Терминаторы выделены серым цветом, части антитерминаторов подчеркнуты. Кодоны аминокислот обозначены: L – лейцин, V – валин, I – изолейцин, H – гистидин, T – треонин, M – метионин, W – триптофан и F – фенилаланин. Цифры в лидерных пептидах означают количество пропущенных кодонов.

При выравнивании использовалась программа ClustalW со следующими параметрами: открытие делеции 10.0, удлинение делеции 0.05, вес транзиции 0.5.



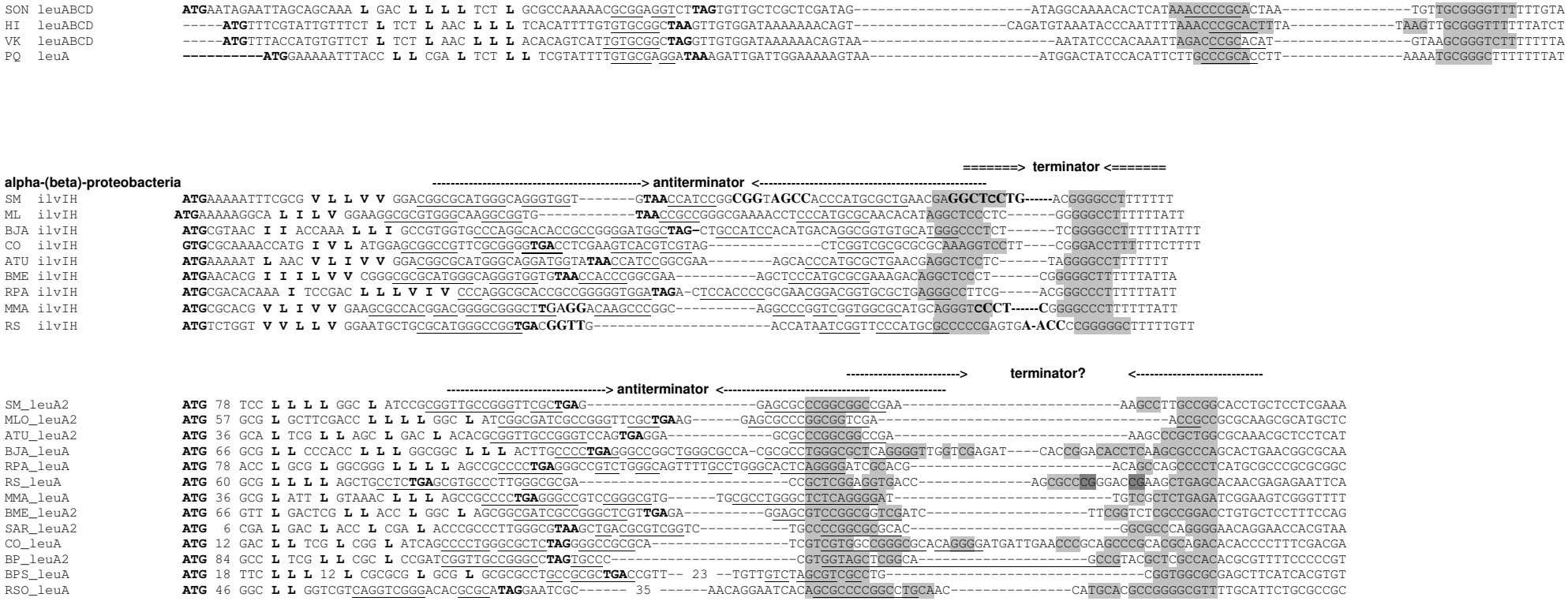
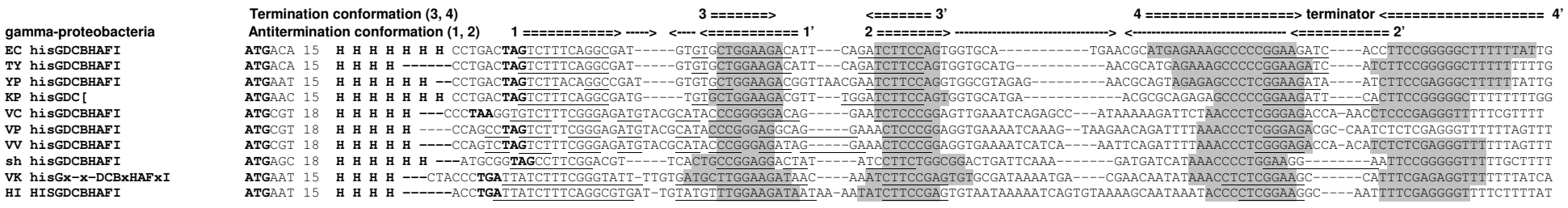


Рис. 4. Выравнивание предсказанных структур транскрипционной аттенюации биосинтеза гистидина у гамма- и альфа-протеобактерий, фирмикут и других бактерий. Обозначения как к рис. 3.



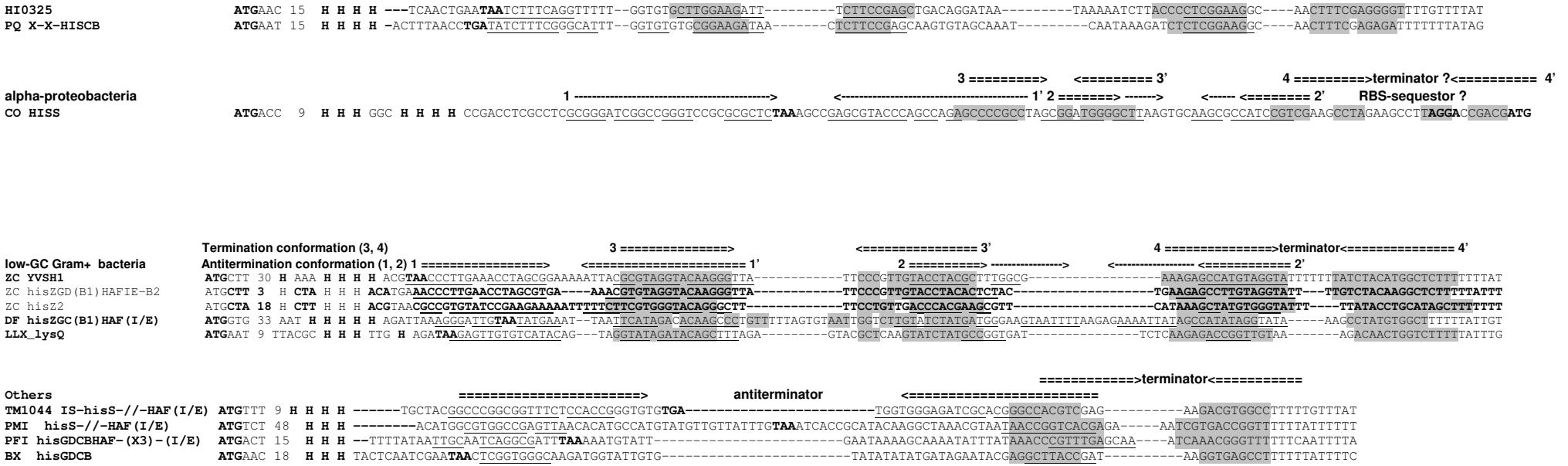

HI0325 ATGAAC 15 H H H H ---TCAACTGAATAATCTTTCAGGTTTTT--GGTGTGCTTGGAAGATT---------TCTTCCGAGCTGACAGGATAA---------TAAAAATCTTACCCCTCGGAAGGC----AACTTTCGAGGGGTTTGTTTTAT
PQ X-X-HISCB ATGAAT 15 H H H H -ACTTTAACCTGATATCTTTCGGGCATTT--GGTGTGTGCGGAAGATAA-------CTCTTCCGAGCAAGTGTAGCAAAT---------CAATAAAGATCTCTCGGAAGGC----AACTTTCGAGAGATTTTTTTATAG
3 ==========> <========== 3' 4 ==========>terminator ?<========== 4'
alpha-proteobacteria 1 ----------> <---------- 1' 2 ========> -------> <------ <========= 2' RBS-sequestor ?
CO HISS ATGACC 9 H H H GGC H H H H CCGACCTCGCCTCGCGGGATCGGCCGGGTCCGCGCGCTCTAAAGCCGAGCGTACCCAGCCAGAGCCCCGCCTAGCGGATGGGGCTTAAGTGCAAGCGCCATCCGTCGAAGCCTAGAAGCCTTAGGAACCGACGATG
Termination conformation (3, 4) 3 ==========> <========== 3' 4 ==========>terminator<========== 4'
low-GC Gram+ bacteria Antitermination conformation (1, 2) 1 ==========> <========== 1' 2 ==========> ----------> <---------- <========== 2'
ZC YVSH1 ATGCTT 30 H AAA H H H H ACGTAACCCTTGAAACCTAGCGGAAAAATTACGCGTAGGTACAAGGGTTA----------TTCCCGTTGTACCTACGCTTTGGCG----------AAAGAGCCATGTAGGTATTTTTTTATCTACATGGCTCTTTTTTTAT
ZC hisZGD(B1)HAFIE-B2 ATGCTT 3 H CTA H H H ACATGAAACCCTTGAACCTAGCGTGA----AAACGTGTAGGTACAAGGGTTA----------TTCCCGTTGTACCTACACTCTAC----------TGAAGAGCCTTGTAGGTATT---TTGTCTACAAGGCTCTTTTTATTT
ZC hisZ2 ATGCTA 18 H CTT H H H ACGTAACGCCGTGTATCCGAAGAAAAATTTTTCTTCGTGGGTACAGGGCTT----------TTCCTGTTGACCCACGAAGCGTT----------CATAAAGCTATGTGGGTATTT----TTATACCTGCATAGCTTTTTTT
DF hisZGC(B1)HAF(I/E) ATGGTG 33 AAT H H H H H AGATTAAAGGGATTGTAATATGAAAT--TAATTCATAGACACAAGCCCTGTTTTTAGTGTAATTGGTCTTGTATCTATGATGGGAAGTAATTTTAAGAGAAAATTATAGCCATATAGGTATA-----AAGCCTATGTGGCTTTTTTATTGT
LLX_lysQ ATGAAT 9 TTACGC H H H TTG H AGATAAGAGTTGTGTCATACAG---TAGGTATAGATACAGCTTTAGA---------GTACGCTCAAGTATCTATGCCGGTGAT----------TCTCAAGAGACCGGTTGTAA-------AGACAACTGGTCTTTTTATTTG
==========>terminator<==========
Others ==========> antiterminator <==========
TM1044 IS-hisS-//-HAF(I/E) ATGTTT 9 H H H H ------TGCTACGGCCCGGCGGTTTCTCCACCGGGTGTGTGA----------TGGTGGGAGATCGCACGGGCCACGTCGAG----------AAGACGTGGCCTTTTTGTTTAT
PMI hisS-//-HAF(I/E) ATGTCT 48 H H H --------ACATGGCGTGGCCGAGTTAACACATGCCATGTATGTTGTTATTTGTAAATCACCGCATACAAGGCTAAACGTAATAACCGGTCACGAGA-----AATCGTGACCGGTTTTTTATTTTTT
PFI hisGDCBHAF-(X3)-(I/E) ATGACT 15 H H H --TTTTATAATTGCAATCAGGCGATTTAAAAATGTATT----------GAATAAAAGCAAAATATTTATAAACCCGTTTGAGCAA----ATCAAACGGGTTTTTTCAATTTTA
BX hisGDCB ATGAAC 18 H H H TACTCAATCGAATAACTCGGTGGGCAAGATGGTATTGTG----------TATATATATGATAGAATACGAGGCTTACCGAT----------AAGGTGAGCCTTTTTTATTTTC

Рис. 5. Выравнивание предсказанных структур транскрипционной аттенюации биосинтеза треонина у гамма-протеобактерий. Обозначения как к рис. 3.

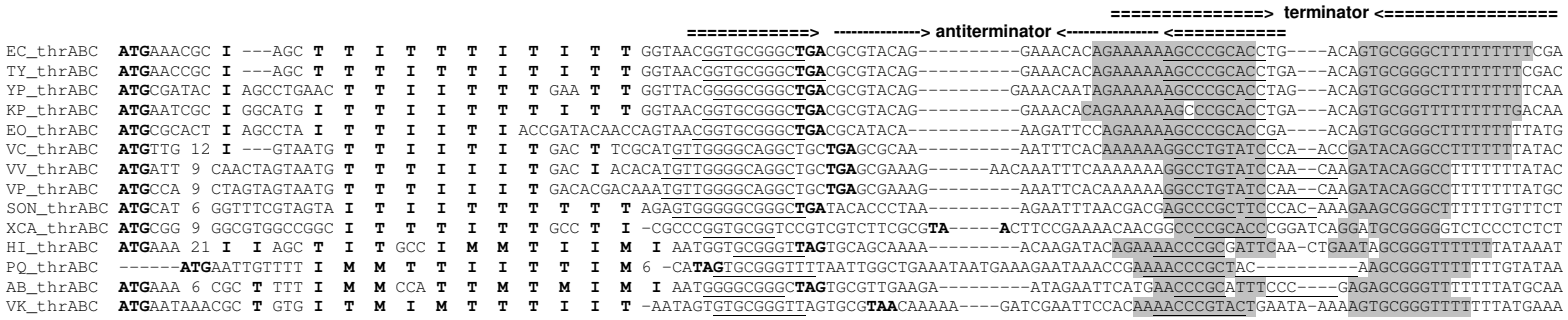
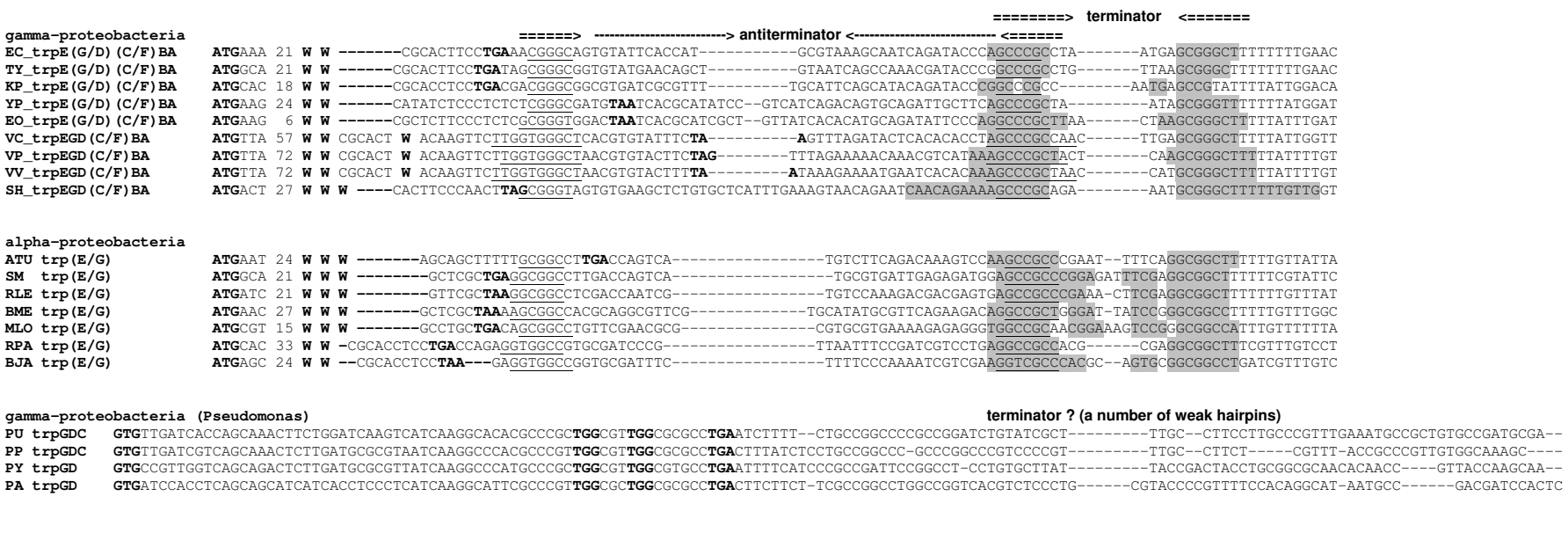


Рис. 6. Выравнивание предсказанных структур транскрипционной аттенюации *trp*, *pheA* and *pheST* оперонов у гамма- и альфа-протеобактерий. Обозначения как к рис. 3.




AV trpGDC
PA trpE
PP trpE
PU trpE
PY trpE
AV trpEG
terminator
antiterminator
gamma-proteobacteria
EC pheA
TY pheA
KP pheA
EO pheA
VV pheA
VP pheA
VV pheA
SON pheA
EC pheST
TY pheST
KP pheST
YP pheST
EO pheST

§3.3. Характерные аттенюаторные структуры, найденные алгоритмом.

Ниже приведены наиболее характерные аттенюаторные структуры, найденные алгоритмом.

Структуры представленные в виде рисунков, сделанных с помощью модуля визуализации. На рисунках слева направо изображаются шпильки структуры (терминатор, антитерминатор и паузная; в некоторых случаях паузная шпилька не найдена и соответственно не указана). В каждой шпильке горизонтальной чертой соединены комплементарные пары нуклеотидов; петля шпильки располагается сверху; выпячивания показаны как несоединенные нуклеотиды. Снизу от каждой шпильки указывается значение, характеризующее ее свободную энергию (без знака – и деления на 10, после чего получаются единицы Ккал/моль). В компьютерном выводе координата любого нуклеотида (при наведении курсора на него) появляется на экране.

Сверху справа схематически изображено отрезками взаимное расположение шпилек в лидерной области (в компьютерном выводе отображаются координаты концов отрезков при наведении на них курсора). Справа показаны лидерный пептид (серым цветом) и участок лидерной области, содержащий аттенюаторную структуру. В компьютерной визуализации кодоны закрашены различными цветами, при этом выдается названия соответствующих аминокислот.

Если алгоритм нашел несколько вариантов лидерных пептидов, то все они указываются отдельно. Паузная шпилька (если она найдена) выделяется курсивом, антитерминатор – жирным, а терминатор – подчеркнутым шрифтом. Например, пересечение терминатора и антитерминатора показано жирным и подчеркнутым шрифтом. Здесь шпильки размечаются без учета отрезков и выпячиваний.

Итак, для многих рассмотренных в работе геномов в лидерной области находится не более одной (существенно отличной) аттенюаторной структуры регуляции биосинтеза рассмотренных аминокислот. Если находится несколько вариантов несколько лидерных пептидов или троек слов, то соответствующие структуры отличаются несущественно.

В нахождении аттенюации важную роль играл подбор параметров наших алгоритмов. Одни параметры были известны из экспериментальных данных, например, интервальные ограничения на длины лидерного пептида и шпилек. Другие параметры вычислялись, например, длина и плотность поля регуляторных кодонов, длина и число пропусков в тройках слов. Они подбирались из статистического анализа, в частности, обучающей выборки из 17 последовательностей, взятой из статьи [38].

Число найденных структур в филогенетически далеких организмах и их выравнивание говорит об адекватности нашей модели и корректном подборе параметров алгоритмов: с одной стороны, по-видимому, не находится шума, т.е. биологически не значимых структур. А с другой стороны, алгоритмы обнаруживают многие случаи потенциальной биологической аттенюации.

В некоторых случаях алгоритмы не нашли паузных шпилек, которые, однако, являются наименее значимым элементом аттенюаторной структуры, что выражается малым весом «качества» паузной шпильки в целевом функционале F.

ЗАКЛЮЧЕНИЕ

1. Предложены новые алгоритмы и соответствующая компьютерная программа для поиска потенциальных структур аттенюаторной регуляции в геномах бактерий. Показана эффективность и надежность этой программы на основе ее детального тестирования на искусственных и биологических последовательностях.

2. Предсказано 108 предполагаемых аттенюаторных структур для оперонов биосинтеза разветвленных и ароматических аминокислот, гистидина и треонина у протеобактерий, у фирмикут и у бактерий из групп Bacteroidetes/Chlorobi и Thermotogales. В некоторых из этих групп аттенюация обнаружена впервые.

3. Получена предположительная функциональная аннотация ряда генов, кодирующих ферменты и находящихся под аттенюаторной регуляцией, а именно:
- гену *ygeA* у *Pasteurella multocida* приписана функция рацемазы разветвленных аминокислот;
- не ортологичные гены *vatB*, *actX2* и *actX3*, соответственно, у *Pasteurella multocida*, *Mannheimia haemolytica*, *Polaribacter filamentus* кодируют ацетилтрансферазы, участвующие в метаболизме гистидина.

4. Показано, что биосинтез изолейцина у Xanthomonadales использует треонин дегидратазу TdcB, в отличие от IlvA у *E. coli*.

5. Предсказано новое семейство гистидиновых транспортеров – ортологов *yuiF* у *B. subtilis* (например, HI0325 у *H. Influenzae*) и два гистидиновых транспортера BC0629

у *B. cereus* (ортолог *yvsH* у *B. subtilis*) из белкового семейства APA и ген у *L. lactis* (ортолог *lysQ* у *E. coli*) из семейства APC.

6. Предсказано, что оперон *his* биосинтеза гистидина регулируется гистидин-зависимыми аттенюаторами у *Bacillus cereus* и *Clostridium difficile*, и в тоже время регулируется гистидиновыми T-боксами у *Lactococcus lactis* и *Streptococcus mutans*.

7. Показаны следующие особенности аттенюаторной регуляции:

ген *thrA* аспартат киназы/гомосерин дегидрогеназы у Pasteurellales регулируется не только треонинином и изолейцином (как у *E. coli*), но и метионином;

ацетолактат синтаза IlvIH у альфа-протеобактерий имеет регуляторные кодоны лейцина, изолейцина и валина.

ЛИТЕРАТУРА

1. Горбунов К.Ю., Любецкая Е.В., Любецкий В.А. О двух алгоритмах поиска альтернативной вторичной структуры РНК // Информационные процессы, РАН, том 1, №2, 2001, стр. 178-187.

2. Леонтьев Л.А., Любецкая Е.В., Любецкий В.А. Модифицированный алгоритм поиска альтернативных вторичных структур РНК и результаты счета // Информационные процессы, РАН, 2002, том 2, №1, с. 100-105.

3. Lyubetsky E.V., Lyubetsky V.A. Algorithm for searching alternative secondary RNA structures. Proceedings of the third international conference of bioinformatics of genome regulation and structure, BGRS'2002, Novosibirsk, Russia, July 14-20, 2002, v. 3, p. 15-17.

4. Любецкая Е.В., Леонтьев Л.А., Гельфанд М.С., Любецкий В.А. Поиск альтернативных вторичных структур РНК, регулирующих экспрессию бактериальных генов // Молекулярная биология, том 37, № 5, 2003, с. 834-842.

5. Любецкая Е.В., Леонтьев Л.А., Любецкий В.А. Поиск альтернативных вторичных структур в классе гамма-протеобактерий // Информационные процессы, РАН, том 3, № 1, 2003, с. 23-38.

6. Lyubetskaya E.V., Leontiev L.A., Lyubetsky V.A. Algorithm for detecting alternative secondary RNA structures and mass analysis attenuator regulation in proteobacteria, MCCMB'03, 2003, p. 144-145.

7. Gorodkin, J., Stricklin, S.L., Stormo, G.D. (2001) Discovering common stem-loop motifes in unligneted RNA sequences // Nucleic Acids Reseach, Vol. 29, No. 10, p. 2135-2144

8. Eddy, S.R. (2002) A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure // BMC Bioinformatics, 3:18, p.1-16

9. Lathe, III, W.C., Suyama, M., Bork, P. (2002) Identification of attenuation and anti-termination regulation in prokaryotes. Genome Biology, 3: preprint 0003.1-0003.60.

10. Горбунов, К.Ю., Миронов, А.А., Любецкий, В.А. (2003) Поиск консервативных вторичных структур РНК // Молекулярная биология , том 37, № 5, с. 850-860.

11. Gollnick, P and Babitzke, P. (2002) Transcription attenuation. Biochim. Biophys. Acta. 1577, 240-250.

12. Vitreschak, A.G., Rodionov, D.A., Mironov, A.A. and Gelfand, M.S. (2004) Riboswitches: the oldest mechanism for the regulation of gene expression? Trends in Genetics. (in press, private letter).

13. Landick, R., Turnbough, C.L. and Yanovsky, C. (1994) Transcriptional attenuation. In: Escherichia coli and Salmonella. Cellular and molecular biology (Neidhardt, F.C., Ed.), pp. 1263-1286. American Society for Microbiology, Washington, DC.

14. Gelfand, M.S. (1999) Recognition of regulatory sites by genomic comparison // Res. Microbiol. 150, 755-771.

15. Osterman, A. and Overbeek, R. (2003) Missing genes in metabolic pathways: a comparative genomics approach // Curr. Opin. Chem. Biol. 7, 238-251.

16. Koonin, E.V. and Galperin, M.Y. (2003) Sequence – Evolution – Function: Computational approaches in comparative genomics, Kluwer Academic Publishers, Boston.

17. Umbarger, H.E. (1994) Biosynthesis of branched chain amino acids. In: Escherichia coli and Salmonella. Cellular and molecular biology (Neidhardt, F.C., Ed.), pp. 442-458. American Society for Microbiology, Washington, DC.

18. Lawther, R.P., Lopes, J.M., Ortuno, M.J. and White, M.C. (1990) Analysis of regulation of the ilvGMEDA operon by using leader-attenuator-galK gene fusions // J. Bacteriol. 172, 2320-2327.

19. Bartkus, J.M., Tyler, B. and Calvo, J.M. (1991) Transcription attenuation-mediated control of leu operon expression: influence of the number of Leu control codons // J. Bacteriol. 173, 1634-1641.

20. Wek, R.C. and Hatfield, G.W. (1988) Transcriptional activation at adjacent operators in the divergent-overlapping ilvY and ilvC promoters of Escherichia coli // J. Mol. Biol. 203, 643-663.

21. Rhee, K.Y., Opel, M., Ito, E., Hung, S., Arfin, S.M. and Hatfield, G.W. (1999) Transcriptional coupling between the divergent promoters of a prototypic LysR-type regulatory system, the ilvYC operon of Escherichia coli // Proc. Natl. Acad. Sci. U S A. 96, 14294-14299.

22. Jafri, S., Chen, S. and Calvo, J.M. (2002) ilvIH operon expression in Escherichia coli requires Lrp binding to two distinct regions of DNA // J. Bacteriol. 184, 5293-5300.

23. Rhee, K.Y., Parekh, B.S. and Hatfield G.W. (1996) Leucine-responsive regulatory protein DNA interactions in the leader region of the ilvGMEDA operon of Escherichia coli // J. Biol. Chem. 271, 26499-26507.

24. Winkler, M.E. (1996) Biosynthesis of histidine. In: Escherichia coli and Salmonella. Cellular and molecular biology (Neidhardt, F.C., Ed.), pp. 485-505. American Society for Microbiology, Washington, DC.

25. Blasi, F. and Bruni, C.B. (1981) Regulation of the histidine operon: translation-controlled transcription termination (a mechanism common to several biosynthetic operons) // Curr. Top. Cell. Regul. 19, 1-45.

26. Patte, J.C. (1994) Biosynthesis of threonine and lysine. In: Escherichia coli and Salmonella. Cellular and molecular biology (Neidhardt, F.C., Ed.), pp. 528-541. American Society for Microbiology, Washington, DC.

27. Green, R.C. (1994) Biosynthesis of methionine. In: Escherichia coli and Salmonella. Cellular and molecular biology (Neidhardt, F.C., Ed.), pp. 542-561. American Society for Microbiology, Washington, DC.

28. Lynn, S.P., Burton, W.S., Donohue, T.J., Gould, R.M., Gumport, R.I. and Gardner, J.F. (1987) Specificity of the attenuation response of the threonine operon of Escherichia coli is determined by the threonine and isoleucine codons in the leader transcript // J. Mol. Biol. 194, 59-69.

29. Rodionov, D.A., Vitreschak, A.G., Mironov, A.A. and Gelfand, M.S. (2003) Regulation of lysine biosynthesis and transport genes in bacteria: yet another RNA riboswitch? // Nucleic Acids Res. 31, 6748-6757.

30. Grundy, F.J., Lehman, S.C. and Henkin, T.M. (2003) The L box regulon: lysine sensing by leader RNAs of bacterial lysine biosynthesis genes // Proc. Natl. Acad. Sci. U S A. 100, 12057-12062.

31. Sudarsan, N., Wickiser, J.K., Nakamura, S., Ebert, M.S. and Breaker, R.R. (2003) An mRNA structure in bacteria that controls gene expression by binding lysine // Genes Dev. 17, 2688-2697.

32. Pittard, A.J. (1996). Biosynthesis of aromatic amino acids. In: Escherichia coli and Salmonella. Cellular and molecular biology (Neidhardt, F.C., Ed.), pp. 458-484. American Society for Microbiology, Washington, DC.

33. Somerville, R. (1992) The Trp repressor, a ligand-activated regulatory protein // Prog. Nucleic Acid Res. Mol. Biol. 42, 1-38.

34. Landick, R., Yanofsky, C., Choo, K. and Phung, L. (1990) Replacement of the Escherichia coli trp operon attenuation control codons alters operon expression // J. Mol. Biol. 216, 25-37.

35. Springer, M., Mayaux, J.F., Fayat, G., etc (1985) Attenuation control of the Escherichia coli phenylalanyl-tRNA synthetase operon // J. Mol. Biol. 181, 467-478.

36. Gavini, N. and Davidson, B.E. (1991) Regulation of pheA expression by the pheR product in Escherichia coli is mediated through attenuation of transcription // J. Biol. Chem. 266, 7750-7753.

37. Bae, Y.M. and Stauffer, G.V. Genetic analysis of the attenuator of the Rhizobium meliloti trpE(G) gene // J Bacteriol. 1991 Jun; 173(11): 3382-3388.

38. Panina, E.M., Vitreschak, A.G., Mironov, A.A. and Gelfand, M.S. (2001) Regulation of aromatic amino acid biosynthesis in gamma-proteobacteria. J // Mol. Microbiol. Biotechnol. 3, 529-543.

39. Benson, D.A., Karsch-Mizrachi, I., Lipman, D.J., Ostell, J. and Wheeler, D.L. (2003) GenBank // Nucleic Acids Res. 31, 23-27.

40. Overbeek, R., Larsen, N., Walunas, T., etc (2003) The ERGO genome analysis and discovery system // J. Bacteriol. 185, 5673-5684.

41. Mironov, A.A., Vinokurova, N.P. and Gelfand, M.S. (2000) GenomeExplorer: software for analysis of complete bacterial genomes // Mol. Biol. 34, 222-231.

42. Tatusov, R.L., Natale, D.A., Garkavtsev, I.V., etc (2001) The COG database: new developments in phylogenetic classification of proteins from complete genomes // Nucleic Acids Res. 29, 22-28.

43. Felsenstein, J. (1981) Evolutionary trees from DNA sequences: a maximum likelihood approach // J. Mol. Evol. 17, 368-376.

44. Thompson, J.D., Gibson, T.J., Plewniak, F., Jeanmougin, F. and Higgins, D.G. (1997) The CLUSTAL_X windows interface: flexible strategies for multiple sequence alignment aided by quality analysis tools // Nucleic Acids Res. 25, 4876-4882.

45. Mulder, N.J., Apweiler, R., Attwood, T.K., Bairoch, A., Bateman, A., etc (2002) InterPro: an integrated documentation resource for protein families, domains and functional sites // Brief Bioinform. 3, 225-235.

46. Popham, D.L. and Setlow, P. (1993) Cloning, nucleotide sequence, and regulation of the Bacillus subtilis pbpE operon, which codes for penicillin-binding protein 4 and an apparent amino acid racemase // J. Bacteriol. 175, 2917-2925.

47. Okada, H., Yohda, M., Giga-Ham,a Y., Ueno, Y., Ohdo, S. and Kumagai, H. (1991) Distribution and purification of aspartate racemase in lactic acid bacteria // Biochim. Biophys. Acta. 1078, 377-382.

48. Tarleton, J.C., Malakooti, J. and Ely, B. (1994) Regulation of Caulobacter crescentus ilvBN gene expression // J. Bacteriol. 176, 3765-3774.

49. Patek, M., Krumbach, K., Eggeling, L. and Sahm, H. (1994) Leucine synthesis in Corynebacterium glutamicum: enzyme activities, structure of leuA, and effect of leuA inactivation on lysine synthesis // Appl. Environ. Microbiol. 60, 133-140.

50. Casalone, E., Barberio, C., Cavalieri, D. and Polsinelli, M. (2000) Identification by functional analysis of the gene encoding alpha-isopropylmalate synthase II (LEU9) in Saccharomyces cerevisiae // Yeast. 16, 539-545.

51. Olekhnovich, I. and Gussin, G.N. (2001) Effects of mutations in the Pseudomonas putida miaA gene: regulation of the trpE and trpGDC operons in P. putida by attenuation // J. Bacteriol. 183, 3256-3260.

52. Delorme, C., Ehrlich, S.D. and Renault, P. (1999) Regulation of expression of the Lactococcus lactis histidine operon // J. Bacteriol. 181, 2026-2037.

53. Chopin, A., Biaudet, V. and Ehrlich, S.D. (1998) Analysis of the Bacillus subtilis genome sequence reveals nine new T-box leaders // Mol. Microbiol. 29, 662-664.

54. Kreimer, D.I., Malak, H., Lakowicz, J.R., Trakhanov, S., Villar, E., Shnyrov and V.L. (2000) Thermodynamics and dynamics of histidine-binding protein, the water-soluble receptor of histidine permease. Implications for the transport of high and low affinity ligands // Eur. J. Biochem. 67, 4242-4252.

55. Steffes, C., Ellis, J., Wu, J. and Rosen, B.P. (1992) The lysP gene encodes the lysine-specific permease // J. Bacteriol. 174, 3242-3249.

56. Patte, J.C., Clepet, C., Bally, M., Borne, F., Mejean, V. and Foglino, M. (1999) ThrH, a homoserine kinase isozyme with in vivo phosphoserine phosphatase activity in Pseudomonas aeruginosa // Microbiology. 4, 845-853.