Применение метода ветвей и границ в задаче восстановления матрицы расстояний между цепочками ДНК

Б. Ф. Мельников, М. А. Тренина

Аннотация—**В настоящей статье мы продолжаем рассмотрение одной из задач биокибернетики – задачи восстановления матрицы расстояний между последовательностями ДНК. В этой задаче на входе алгоритма известны не все элементы рассматриваемой матрицы (обычно 50% и менее элементов). Основой разработки алгоритма восстановления такой матрицы служит разработанный и исследованный нами ранее метод сравнительной оценки алгоритмов расчёта расстояний между последовательностями ДНК, основанный на специальном анализе матрицы расстояний. В этом анализе применялось определённое нами значение badness каждого из треугольников, соответствующего матрице.**

Продолжая улучшать алгоритмы решения этой задачи, мы рассматриваем применение в ней метода ветвей и границ. Для этого для некоторой известной последовательности незаполненных элементов мы применяем алгоритмы, рассмотренные нами в предыдущих публикациях, однако теперь сами последовательности мы выбираем с помощью разработанного нами варианта метода ветвей и границ. В нашей интерпретации этого метода в качестве множества допустимых решений берутся все возможные последовательности неизвестных элементов верхней треугольной части матрицы. В каждой текущей подзадаче в качестве разделяющего элемента берётся любой из незаполненных элементов матрицы, а в качестве границы берётся сумма значений badness для всех треугольников, которые уже сформировались к моменту рассмотрения этой подзадачи. Таким образом, определение элементов неполностью заполненной матрицы происходит в такой последовательности, при которой итоговый показатель badness для всех треугольников выбирается с помощью жадной эвристики, полностью вписывающейся в рамки классических вариантов описания метода ветвей и границ.

В результате применения такого алгоритма мы получаем результаты, которые с точки зрения значения badness являются минимально возможными (в случае завершённого варианта метода ветвей и границ), либо близкими к оптимальным (в случае его незавершённого варианта). При этом в проведённых нами вычислительных экспериментах время работы алгоритма практически совпадает со временем работы алгоритма, рассмотренного нами в предыдущей публикации (превышает его не более, чем на 10%), а значение badness обычно уменьшается на 20–40% от исходного значения. Таким образом, мы получаем возможность быстро и эффективно восстановить матрицу ДНК – часто даже в случае её заполнения менее, чем на 40%.

Ключевые слова—**метод ветвей и границ, последовательности ДНК, метрика, матрица расстояний, частично заполненная матрица, восстановление.**

Статья получена 1 июля 2018.

Борис Феликсович Мельников, Российский государственный социальный университет (email: bf-melnikov@yandex.ru).

Марина Анатольевна Тренина, Тольяттинский государственный университет (email: trenina.m.a@yandex.ru).

I. Введение

Настоящая статья является продолжением работы над одной из задач биокибернетики – задачи восстановления матрицы расстояний между последовательностями ДНК [1]. В этой задаче на входе алгоритма известны не все элементы рассматриваемой матрицы (обычно 50% и менее элементов). Основой разработки алгоритма восстановления такой матрицы служит разработанный и исследованный нами ранее метод сравнительной оценки алгоритмов расчёта расстояний между последовательностями ДНК, основанный на специальном анализе всех возможных треугольников, получающихся в матрице расстояний. В этом анализе применялось определённое нами значение badness каждого из треугольников матрицы [2], [3].

Рассмотренный нами метод сравнительного анализа для полученной в результате работы какого-либо алгоритма вычисления расстояний между геномами матрицы основан на рассмотрении всех возможных треугольников. При этом мы исходим из предположения, что в идеале они должны быть остроугольными равнобедренными (см. [4]). Для ответа на вопрос, насколько «правильной» является матрица, полученная в результате некоторого эвристического алгоритма, мы предлагаем использовать в качестве «характеристики отхода» полученных треугольников от «вытянутых равнобедренных» вышеупомянутое значение badness.

В разработанном нами ранее простом методе восстановления матрицы ДНК (см. [1]) неизвестные элементы матрицы рассматриваются и определяются «слева-направо и сверху-вниз» – и только в этой единственной последовательности, а само восстановление происходит в результате осуществления нескольких вычислительных проходов. На каждом из проходов для некоторых пока незаполненных (неизвестных) элементов матрицы получаются разные оценки; эти оценки специальным образом усредняются – и результат усреднения берётся в качестве значения неизвестного элемента.

Применение описанного метода для заполнения матрицы расстояний между последовательностями ДНК позволяет, прежде всего, значительно сократить время её заполнения, а отклонение полученных таким образом элементов матрицы от значений вычисленных, например, с помощью алгоритма Нидлмана – Вунша, в среднем составило менее, чем 2%, а в наихудшем случае – около 10%.

Вычисление неизвестных элементов *неполностью* за-

полненной матрицы на основе использования значения показателя badness осуществляется из расчёта того, что он в идеале должно быть равно нулю, а определение элементов «слева-направо и сверху-вниз», т.е. в строго определённой последовательности, может привести к тому, что для ранее определённых треугольников значение показателя badness значительно увеличится. Одной из форм сглаживания такой ситуации было использование функции риска, которая производит специальное усреднение [1], [5], [6].

Продолжая улучшать алгоритмы решения этой задачи, мы рассматриваем применение в ней метода ветвей и границ. Для этого для некоторой известной последовательности незаполненных элементов мы применяем алгоритмы, рассмотренные нами в предыдущих публикациях, однако

> *сами последовательности мы выбираем с помощью разработанного нами варианта метода ветвей и границ.*

В нашей интерпретации этого метода мы выполняем следующие действия.

- В качестве *множества допустимых решений* берутся все возможные последовательности неизвестных элементов верхней треугольной части матрицы.
- В каждой текущей подзадаче в качестве *возможных разделяющих элементов* берутся ещё не заполненные элементы матрицы.
- В качестве *границы* берётся сумма значений badness для всех треугольников, которые уже сформировались к моменту рассмотрения этой подзадачи.
- Вспомогательный *алгоритм выбора разделяющего элемента* заключается в том, что среди возможных разделяющих элементов мы выбираем такой, для которого максимально возможным образом получается граница между формируемыми правой и левой подзадачами.

Подробные описания см. в разделе III.

Таким образом, определение элементов неполностью заполненной матрицы происходит в такой последовательности, при которой итоговый показатель badness для всех треугольников выбирается с помощью жадной эвристики, полностью вписывающейся в рамки классических вариантов описания метода ветвей и границ.

В результате применения такого алгоритма мы получаем результаты, которые с точки зрения значения badness являются:

- минимально возможными – в случае завершённого варианта метода ветвей и границ;
- либо близкими к оптимальным – в случае его незавершённого варианта, т.н. truncated branches and bounds method.

При этом в проведённых нами вычислительных экспериментах время работы алгоритма практически совпадает со временем работы алгоритма, рассмотренного нами в предыдущей публикации (превышает его не более, чем на 10%), а значение badness обычно уменьшается на 20–40% от исходного значения. Таким образом, мы получаем возможность быстро и эффективно восстановить матрицу ДНК – часто даже в случае её заполнения менее, чем на 40%.

II. Предварительные све́дения

Как уже говорилось во введении, рассматриваемые матрицы, подлежащие восстановлению, мы будем называть неполностью заполненными матрицами расстояний. Мы в [1] ввели этот *термин* для матрицы, из которой «вычеркнуто» некоторое количество элементов.

Итак, в рассматриваемой задаче мы имеем неполностью заполненную матрицу расстояний между последовательностями ДНК, некоторые её элементы неизвестны. Задача состоит в том, чтобы определить эти значения на основе известных элементов матрицы.

Как уже отмечалось во введении, в основе алгоритма решения этой задачи лежит использование разработанного и исследованного нами ранее метода сравнительной оценки алгоритмов расчёта расстояний между последовательностями ДНК. Этот метод основан на анализе всех возможных треугольников, при этом мы исходим из предположения, что в идеале они должны быть остроугольными равнобедренными (см. [1], [4]). Для «характеристики отхода» полученных треугольников от «вытянутых равнобедренных» треугольников вводится показатель badness, для вычисления которого в описываемом нами случае используется формула

$$\sigma = \frac{a-b}{c} \tag{1}$$

где a, b и c – стороны треугольника, причём мы предполагаем, что $a \geq b \geq c$.

Далее выполняются следующие вычисления. Для определения неизвестного элемента мы рассматриваем все возможные треугольники, образованные из элементов этой матрицы, для которых одна из сторон неизвестна. Для каждого такого треугольника из того условия, что он является равнобедренным остроугольным (при этом значение badness равнялось бы 0), мы получаем *одно из возможных значений* этой неизвестной стороны, и среднее арифметическое этих значений мы считаем окончательным значением рассматриваемой стороны (неизвестного элемента).

Аналогично предыдущей работе, после выполнения алгоритма *для оценки его эффективности* мы используем такой показатель, как невязка, который характеризует отклонение полученной матрицы от исходной. Мы применяем следующую естественную формулу:

$$d = \frac{\sqrt{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(a_{ij}-\widetilde{a_{ij}})^2}}{n(n-1)/2}, \tag{2}$$

где:

- $\widetilde{a_{ij}}$ – элементы матрицы, полученной в результате применения некоторого алгоритма подсчёта расстояний между парой геномов (в нашем случае – алгоритма Нидлмана – Вунша);
- a_{ij} – элементы матрицы, восстановленной в результате работы вышеописанного алгоритма.

Стоит ещё раз отметить, что и в разработанном ранее алгоритме восстановления матриц расстояний между цепочками ДНК, в котором перебор незаполненных клеток производился только единственным образом «слева-направо и сверху-вниз» (и далее в таком же порядке, если сделанный проход не дал окончательных результатов), были получены достаточно приемлемые результаты; мы

делаем этот вывод в первую очередь потому, что при разных входных данных у нас получаются удачные значения невязки, см. [1], а также далее в настоящей статье. Однако, конечно, нет гарантии, что эти результаты нельзя улучшить – т.е. уменьшить вычисляемое значение невязки. Как обычно в подобных ситуациях (т.е. в схожих задачах дискретной оптимизации), очень большие улучшения даёт применение метода ветвей и границ.

III. Метод ветвей и границ в задаче восстановления матрицы расстояний

Метод ветвей и границ предназначен для разработки алгоритмов решения оптимизационных задач, и мы рассматриваем его интерпретацию близко к [7], [8]. Он относится к методу разработки алгоритмов, известному как *программирование с отходом назад* и исследует древовидную модель пространства допустимых решений. Цель применения метода ветвей и границ в нашем случае, т.е. для восстановления матрицы, – найти оптимальную *последовательность* вычисления неизвестных элементов так, чтобы в итоге невязка оказалось бы минимальной (или близкой к минимальной). Минимизация невязки напрямую связана с минимизацией значения badness всей матрицы, поэтому задачей применения метода ветвей и границ для восстановления матрицы является минимизация значения badness всей матрицы.

При использовании метода ветвей и границ должны быть описаны следующие вспомогательные алгоритмы:

- процедура ветвления области допустимых решений;
- процедура нахождения нижних и верхних границ целевой функции.

В решаемой нами задаче пространство допустимых решений – это все возможные последовательности определения неизвестных элементов верхней половины матрицы. Ветвление происходит по известным с самого начала работы алгоритма разделяющим элементам, которые в нашем случае совпадают с незаполненными элементами матрицы. Ветвление на множестве допустимых решений мы будем производить следующим образом: для выбранного элемента все последовательности разделяются на две последовательности:

- начинающиеся с этого элемента;
- и не начинающиеся с него.

Одна из вспомогательных эвристик – эвристика выбора разделяющего элемента a_{ij} $(i < j)$ – состоит в том, чтобы:

- найти все такие k, для которых a_{ki} и a_{kj} известны и $k \neq i, k \neq j$;
- вычислить разность

$$|a_{ki} - a_{kj}|. \tag{3}$$

Мы сначала отбираем элементы, для которых k принимает наибольшее значение, а потом среди них в качестве разделяющего элемента выбираем тот, для которого разность (3) наибольшая.

Правой подзадачей очередного шага метода ветвей и границ мы, согласно [7], [9], называем задачу, полученную при уменьшении размерности; в нашем случае это соответствует заполнению новой клетки матирицы. А другую альтернативу мы называем, соответственно, *левой подзадачей* этого шага. В нашем случае она соответствует решению о запрете заполнения этого элемента в оптимальном решении; конечно же, этот запрет «действует» до окончания рассмотрения всех оставшихся незаполненных элементов.

Не будет преувеличением сказать, что смысл *всех* модификаций метода ветвей и границ (в том числе незавершённого) состоит в том, что с помощью некоторых эвристик мы добиваемся того, чтобы вероятность наличия оптимального решения была больше для правой задачи, чем для левой, – так как размерность левой задачи обычно на 1 меньше, чем правой. (При этом мы не будем подробно писать здесь про вспомогательные эвристики для оценки этих двух вероятностей: это для каждой задачи дискретной оптимизации может быть темой отдельной публикации. Неявно эти эвристики описаны в алгоритме 3.)

Границы, как уже было отмечено выше, формируются как сумма значений badness для всех треугольников, которые уже сформированы в матрице. Отсюда следует, что заполнение матриц новыми элементами приводит к увеличению (точнее, неуменьшению) этого значения.

Применяемая нами естественная вспомогательная эвристика выбора разделяющего элемента такова: мы выбираем новый элемент таким образом, чтобы граница левой подзадачи увеличилась бы минимально.

С учётом большого числа образующихся треугольников и для наглядности вычислений мы от показателя badness для треугольника перейдём к аналогичному показателю, но для элементов матрицы. Для каждого элемента, лежащего выше главной диагонали, рассматриваем все треугольники, построенные на нём и вычисляем значение показатель σ каждого треугольника. А элементу a_{ij} $(i < j)$ ставим в соответствие значение

$$\sigma_{ij} = \max_{1 \leq k \leq n} \sigma_k, \tag{4}$$

где σ_k – показатель badness треугольника, построенного на элементах a_{ij}, a_{ki} и a_{kj}.

Для всей матрицы ДНК мы определяем сумму значений badness по всем элементам, т.е.

$$\sigma = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (\sigma_{ij}). \tag{5}$$

Цель восстановления матрицы – минимизация показателя (5).

Разработанный и приведенный в работе [1] алгоритм 1 в несколько изменённом виде будем использовать в качестве вспомогательного для алгоритма восстановления неполностью заполненной матрицы расстояний между цепочками ДНК методом ветвей и границ. (Отметим, что согласно [9], непосредственное применение алгоритма 1 может быть названо жадным алгоритмом построения *последовательности правых подзадач*.) В дальнейшем, для удобства, мы продолжим эту нумерацию алгоритмов, поэтому вспомогательный алгоритм назовём алгоритмом 2, а общий алгоритм восстановления матрицы расстояний методом ветвей и границ – алгоритмом 3.

Ниже приведём описание вспомогательного алгоритма 2 (применяемого для определения выбранного в качестве разделяющего неизвестного элемента), а также алгоритма 3. В приложении 1 приведены блок-схемы

обоих этих алгоритмов, а также исходного алгоритма 1, блок-схема которого не приводилась в [1].

Алгоритм 2 (Вспомогательный алгоритм присваивания значения выбранному элементу)

Вход: 1. Неполностью заполненная матрица $A = a_{ij}$ (все равные нулю элементы вне главной диагонали считаем неизвестными).
2. Выбранный неизвестный элемент a_{ij}.

Описание алгоритма.

for $k := 0$ to n do begin
  if $k \neq i$ and $k \neq j$ and $a_{ki} \neq 0$ and $a_{kj} \neq 0$ then
    if $a_{ki} > a_{kj}$ then $a_{ij} := a_{ij} + a_{ki}$
    else $a_{ij} := a_{ij} + a_{kj}$;
end;
$a_{ij} := a_{ij}/kol_{ij}$.

Выход: Значение элемента a_{ij}. □

Алгоритм 3 (Восстановление матрицы методом ветвей и границ)

Вход: Неполностью определенная матрица $A = a_{ij}$ (все равные нулю элементы вне главной диагонали считаем неизвестными).

Описание алгоритма.

Шаг 1: Инициализация: «обнуление» списка подзадач и текущего псевдооптимального решения.

Шаг 2: Добавление в список подзадач задачи соответствующей исходной матрице.

Шаг 3: Если список подзадач пуст, то выход 1.

Шаг 4: Если время работы алгоритма закончилось, то выход 1.

Шаг 5: Выбор новой подзадачи из списка подзадач и удаление её.

Шаг 6: Выбор «наилучшего» неизвестного элемента.
1) if $a_{ij} = 0$ then
 begin
  $kol_{ij} := 0$ {считаем количество треугольников,
    построенных на неизвестном элементе}
  $mas\_m_{ij} := 0$ {вычисляем максимальную разность}
  for $k := 0$ to $n - 1$ do begin
    if $k \neq i$ and $k \neq j$ and $a_{ki} \neq 0$ and $a_{kj} \neq 0$ then
      begin
        $kol := kol + 1$; $m := |a_{ki} - a_{kj}|$;
        if $m > mas\_m_{ij}$ then $mas\_m_{ij} := m$;
      end;
  end;

2) Определяем наибольшее значение массива kol_{ij}:

$$maxk := \max_{\substack{1 \le i \le n \\ 1 \le j \le n}} kol_{ij}.$$

3) Если $maxk = 0$, то выход 2.

4) Среди элементов, для которых $kol_{ij} = maxk$ находим тот, у которого значение mas_m_{ij} наибольшее.

Шаг 7: Вызов алгоритма 2 для вычисления выбранного элемента.

Шаг 8: Вычисление значения показателя badness для него:
  for $k := 0$ to n do begin
    if $k \neq i$ and $k \neq j$ and $a_{ki} \neq 0$ and $a_{kj} \neq 0$ then
      сортируем по убыванию a_{ki}, a_{kj} и a_{ij}
      и обозначим их соответственно a, b, c;
      $\sigma_k := \frac{a-b}{c}$;
  end;
$\sigma_{ij} := \max\limits_{1 \le k \le n} \sigma_k$.

Шаг 9: Формирование левой и правой подзадач на основе сравнения значения показателя badness для выбранного элемента. В качестве левой выбирается та, для которой это значение меньше.

Шаг 10: Проверяем является ли левая задача допустимой, в противном случае возвращаемся к шагу 6.

Шаг 11: Если значение показателя badness полученной матрицы для левой подзадачи меньше соответствующего значения текущего псевдооптимального решения, то эта подзадача становится текущей и переход к шагу 2. Иначе переход к шагу 5.

Выход 1: Текущее псевдооптимальное решение (если оно есть).

Выход 2: Матрицу A восстановить невозможно. □

IV. Подробное описание примера работы с матрицей малой размерности

В этом разделе мы приведём подробное описание применения метода ветвей и границ для восстановления матрицы малой размерности 5. В нашей предыдущей работе мы в качестве иллюстрации алгоритма брали для примера матрицу размерности 7, но для матрицы такого размера даже для первого прохода возможно 14! (примерно 10^{11}) последовательностей. Поэтому, для наглядности, для подробного описания работы метода ветвей и границ мы ограничились матрицей размерности 5.

Как и раннее, все наши примеры (в этом разделе – для малой размерности 5, а также в следующем, для размерности 28) работают с матрицами, полученными применением алгоритма Нидлмана – Вунша [3]. Этот алгоритм был применён к цепочкам мДНК различных животных, взятых из банка данных NCBI [10]; при этом были взяты секвенированные цепочки мДНК для одного представителя каждого из 28 отрядов млекопитающих, так как МДНК у разных из этих 28 видов меняются только за счет мутации, из-за того, что не подвержены рекомбинации и наследуются только по материнской линии (классификацию млекопитающих выбираем согласно [12], другие варианты классификации не рассматриваем). Полученная матрица, а также все выбранные нами виды животных приведены в приложении [1].

Как и в работе [1], вместо «процента близости» (фактически – результата применения алгоритма Нидлмана – Вунша) мы используем схожую характеристику «относительную удалённость», которая получается вычитанием «процента близости» из 100 и делением на 100. Всюду ниже мы будем использовать именно эти значения, причём приводить 3 значащие десятичные цифры.

В качестве простого примера рассмотрим матрицу 5×5 (таблица 1). В дальнейшем матрицу расстояний

между последовательностями ДНК мы будем приводить полностью, а значения показателя badness – только для элементов верхнего треугольника.

Таблица 1. Исходная матрица порядка 5

0	0,299	0,258	0,270	0,315
0,299	0	0,369	0,298	0,240
0,258	0,369	0	0,292	0,343
0,270	0,298	0,292	0	0,294
0,315	0,240	0,343	0,294	0

Далее. Из этой матрицы мы удаляем 50% элементов (для матрицы малой размерности приемлемое восстановление обычно возможно в том случае, когда это процент не превосходит 60, мы же, для наглядности, остановились на 50) лежащих выше главной диагонали и, вместе с ними, соответствующие элементы ниже главной диагонали. *Возможный* (обрабатываемый нами) вариант удаления приведён в таблице 2.

Таблица 2. Неполностью заполненная матрица порядка 5

0	—	—	0,270	—
—	0	0,369	—	0,240
—	0,369	0	0,292	—
0,270	—	0,292	0	0,294
0,000	0,240	—	0,294	0

Произведём восстановление этой матрицы методом ветвей и границ. Часть дерева поиска для рассматриваемой задачи представлено на рис. 1, где R – всё пространство допустимых решений. Так как в данном примере количество неизвестных элементов равно 5, то дерево поиска для рассматриваемой задачи будет имеет пять уровней, причём элементы четвёртого уровня будут иметь только по одному наследнику.

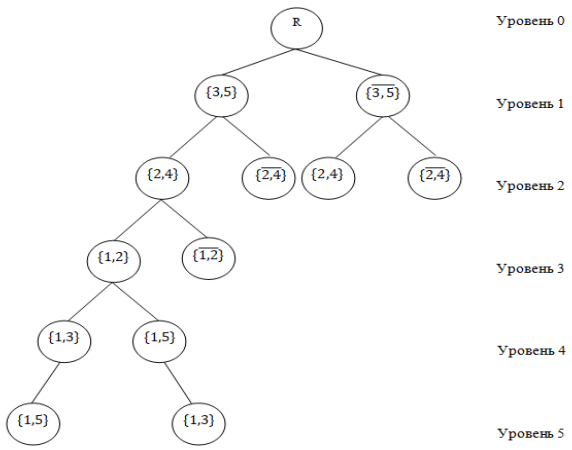

Рис. 1. Часть дерева поиска, необходимая для метода ветвей и границ.

Как отмечалось выше, в качестве верхней границы на начальном этапе мы будем использовать значения показателя badness (таблица 4), полученные для исходной матрицы в результате её восстановления с помощью алгоритма 1 (таблица 3). Если на какой-либо ветви появится значение σ элемента больше, чем в таблице 4, то эту ветвь убираем из рассмотрения. Если же найдётся последовательность восстановления матрицы, для которой значение σ будет меньше, то эта последовательность станет текущей, и в дальнейшем в качестве верхней границы будет использоваться значение её показателя σ.

Таблица 3. Восстановленная алгоритмом 1 матрица порядка 5

0	0,319	0,336	0,270	0,270
0,319	0	0,369	0,369	0,240
0,336	0,369	0	0,292	0,369
0,270	0,369	0,292	0	0,294
0,270	0,240	0,369	0,294	0

Таблица 4. Показатели σ восстановленной алгоритмом 1 матрицы порядка 5

0	0,156	0,130	0,135	0,156
	0	0,090	0,205	0,205
		0	0,205	0,205
			0	0,205
				0

Показатель матрицы $\sigma = 1,69$ и невязка для этой восстановленной матрицы $d = 0,0119$.

Рассматриваем все неизвестные элементы a_{ij} верхнего треугольника и для каждого из них определим количество таких k, для которых элементы a_{ki} и a_{kj} известны. Таким образом:

- для элемента a_{12} получаем $k = 0$;
- для элемента a_{13} получаем $k = 1$;
- для элемента a_{15} получаем $k = 1$;
- для элемента a_{24} получаем $k = 2$;
- для элемента a_{35} получаем $k = 2$.

Итак, у нас есть 4 элемента, которые можно восстановить на первом этапе.

Для выбора разделяющего элемента из этих четырёх применим эвристику, которая была описана в предыдущем разделе: у нас есть два элемента, для которых k принимает наибольшее значение (равное 2), и для этих элементов мы вычисляем *разность между соответствующими известными элементами*; в качестве элемента ветвления выбирем тот, для которого разность между известными элементами *наибольшая*. Таким элементом будет a_{35}.

Таблица 5. Результат восстановления матрица порядка 5 на первом уровне

0	—	—	0,270	—
—	0	0,369	—	0,240
—	0,369	0	0,292	0,331
0,270	—	0,292	0	0,294
0,000	0,240	0,331	0,294	0

Поэтому мы возьмём элемент a_{35} *в качестве разделяющего* и рассмотрим две связанные с ним ветви: $\{3,5\}$ и $\overline{\{3,5\}}$. Таким образом, мы вычислили элемент $a_{35} = 0,331$ и получили матрицу (таблица 5) с характеристиками, представленными в таблице 6.

Таблица 6. Показатели σ матрицы первого уровня

0	—	—	—	—
	0	0,102	—	0,102
		0	0,114	0,114
			0	0,114
				0

Значение показателя badness матрицы $\sigma = 0,546$.

Продолжаем исследование этой ветви. Таким образом:

- для элемента a_{12} получаем $k = 0$;
- для элемента a_{13} получаем $k = 1$;
- для элемента a_{15} получаем $k = 1$;
- для элемента a_{24} получаем $k = 2$.

В качестве следующего элемента ветвления рассматриваем элемент a_{24} и при его вычислении получаем матрицу (таблица 7) с характеристиками, представленными в таблице 8.

Таблица 7. Результат восстановления матрица порядка 5 на втором уровне

0	—	—	0,270	—
—	0	0,369	0,331	0,240
—	0,369	0	0,292	0,331
0,270	0,331	0,292	0	0,294
—	0,240	0,331	0,294	0

Таблица 8. Показатели σ матрицы второго уровня

0	—	—	—	—
	0	0,102	0,114	0,102
		0	0,114	0,114
			0	0,114
				0

Значение показателя badness матрицы $\sigma = 0,672$.

Далее:

- для элемента a_{12} получаем $k = 1$;
- для элемента a_{13} получаем $k = 1$;
- для элемента a_{15} получаем $k = 1$.

Наибольшая разность соответствует элементу a_{12}, поэтому следующее ветвление идёт на нём. При вычислении элемента $a_{12} = 0,331$ имеем

Таблица 9. Результат восстановления матрица порядка 5 на третьем уровне

0	0,331	—	0,270	—
0,331	0	0,369	0,331	0,240
—	0,369	0	0,292	0,331
0,270	0,331	0,292	0	0,294
—	0,240	0,331	0,294	0

Таблица 10. Показатели σ матрицы третьего уровня

0	0,000	—	0,000	—
	0	0,102	0,114	0,102
		0	0,114	0,114
			0	0,114
				0

Значение показателя badness матрицы $\sigma = 0,672$. Затем элемент $a_{13} = 0,331$.

Таблица 11. Результат восстановления матрица порядка 5 на четвёртом уровне

0	0,331	0,331	0,270	—
0,331	0	0,369	0,331	0,240
0,331	0,369	0	0,292	0,331
0,270	0,331	0,292	0	0,294
—	0,240	0,331	0,294	0

Таблица 12. Показатели σ матрицы четвёртого уровня

0	0,102	0,116	0,116	—
	0	0,102	0,114	0,114
		0	0,116	0,114
			0	0,114
				0

Значение показателя badness матрицы $\sigma = 0,672$.

И последний неизвестный элемент – $a_{15} = 0,319$. В результете получаем восстановленную матрицу (таблица 13) с показателем $\sigma = 1,089$ (показатели каждого элемента представлены в таблице 14) и невязкой $d = 0,00867$.

Таблица 13. Результат восстановления матрица порядка 5 на пятом уровне

0	0,331	0,331	0,270	0,319
0,331	0	0,369	0,331	0,240
0,331	0,369	0	0,292	0,331
0,270	0,331	0,292	0	0,294
0,319	0,240	0,331	0,294	0

Для матрицы, восстановленной по этой ветке множества допустимых решений, значение невязки значительно меньше того, который мы взяли за основу сравнения, и также отмечается снижение показателя σ для каждого элемента матрицы. Поэтому в дальнейшем эти значения

становятся текущими.

Таблица 14. Показатели σ матрицы пятого уровня

0	0,102	0,116	0,116	0,079
	0	0,102	0,114	0,114
		0	0,116	0,114
			0	0,114
				0

Проведём анализ других веток, сначала вернёмся назад на один уровень и проанализируем ветвь $\overline{\{1,2\}}$. На этой ветке, при условии вычисления сначала элемента a_{13}, получаем, что $\sigma_{13} = 0,132$, поэтому она уходит из рассмотрения.

Откатываемся ещё на один уровень к элементу $_{24}$ и исследуем ветвь $\overline{\{2,4\}}$. На этой ветке сразу значительно увеличиваются известные значения σ_{45}, σ_{35} и σ_{34}, поэтому дальнейшее рассмотрение этой ветки бессмысленно.

И возвращаемся к элементу a_{35} и ветки $\overline{\{3,5\}}$. Так как сама оптимальная последовательность на ветке a_{24} – a_{12} – a_{13} – a_{15}, то пробуем вычислить элемент a_{35} после a_{13} (все предыдущие элементы имеют номер не связанный с индексами 3 и 5) и вычисление его в любом другом месте не влияет на значения σ.

Следовательно, оптимальное восстановление матрицы получено (таблица 13) и невязка составила $d = 0,00867$.

Проведём сравнение полученных результат восстановления матрицы алгоритмом 1 и методом ветвей и границ. Значение показателя σ для матрицы снизилось на 35%, а показателя невязка – на 27%.

V. Результаты восстановления матрицы ДНК

В этом разделе представлены результаты вычислительного эксперимента большей размерности. Как и в предыдущем разделе, мы выбираем по одному геному из 28 отрядов млекопитающих [12] – но здесь мы рассматриваем представителей всех 28 отрядов.

Как и ранее, мы будем использовать матрицу расстояний, значения в которой изменяются от 0 до 1. При этом в таблицах, приведённых в приложении, данные для удобства обозначены целыми числами, образованными первыми тремя значащими цифрами этих значений (т. е. мы фактически умножаем имеющиеся значения на 1000 и округляем до целых).

Мы взяли ту же неполностью заполненную матрицы расстояний ДНК, которую восстанавливали применением алгоритма 1. При этом из матрицы были удалены примерно 63% пар элементов (осталось примерно 37% пар). Эта матрица приведена в таблице 9 приложения [1]. Для восстановленной матрицы значение показателя badness матрицы равно $\sigma = 92,1$, а значение невязки составило $d = 0,00185$. Полученная матрица приведена в таблице 16 приложения.

Теперь же мы провели восстановление матрицы применением метода ветвей и границ и полученная матрица приведена в таблице 17 приложения. Для восстановленной этим методом матрицы мы получили следующие показатели: $\sigma = 66,6$ и $d = 0,00145$. Таким образом, в результате применения метода ветвей и границ значение показателя badness матрицы уменьшилось на 28%, невязка уменьшилась на 20%.

Таблица 15. Сравнение невязки восстановления одной матрицы с применением метода ветвей и границ и без него

	σ	$max\ d_{ij}$	d
без МВГ	$92,1$	0.141	0.00185
МВГ	$66,6$	0.106	0.00145
улучшение, %	$27,7$	$24,8$	$21,6$

Сравнение результатов применения различных методов для восстановления матрицы расстояний между цепочками ДНК приведено в таблице 15.

Можно сказать, что эта таблица описывает основной результат настоящей статьи.

Важно отметить, что существенное улучшение основных характеристик восстановления матрицы практически не отразилось на увеличении времени работы алгоритма: оно увеличилось лишь примерно на 10%.

VI. Заключение

Итак, мы предлагаем использовать метод ветвей и границ для оптимизации разработанного ранее метода *восстановления* матрицы расстояний между последовательностями ДНК, в основе которого использовался подход, разработанный и применёный на практике для сравнительной оценки других алгоритмов – алгоритмов *расчёта расстояний* между такими последовательностями; упрощая, можно сказать, что мы пытаемся добиться выполнения свойства остроугольной равнобедренности для всех образующихся треугольников. Применение метода ветвей и границ позволило произвести определение неизвестных элементов матрицы в той последовательности, которая минимизирует значение показателя badness матрицы и как следствие невязку.

Как отмечалось в работе [1], применение разработанного подхода на основе применения показателя badness для заполнения матрицы расстояний между последовательностями ДНК позволяет значительно сократить время её заполнения. Применение метода ветвей и границ позволяет произвести вычисление неизвестных элементов в оптимальной последовательности, что минимизирует значение невязки. При этом, время работы алгоритма увеличивается незначительно.

В качестве одного из направлений будущего улучшения описанного нами здесь эвристического алгоритма (прежде всего – уменьшения времени его работы) мы предполагаем использование кластеризации ситуаций, применение которой описано в [12], [13]. При применении этого подхода мы в разных подзадачах (ситуациях), получающихся на различных ветвях дерева перебора метода ветвей и границ, которым (подзадачам) соответствуют некоторые близкие по некоторой «естественной» метрике матрицы, пытаемся выбрать один и тот же разделяющий элемент. Такое применение (если оно возможно) практически не требует времени – при том, что, как мы отмечали в предыдущих публикациях, время, затрачиваемое на выбор элемента, составляет основную часть времени работы МВГ.

Список литературы

[1] Мельников Б. Ф., Тренина М. А. *Об одной задаче восстановления матриц расстояний между цепочками ДНК*. International Journal of Open Information Technologies.ISSN: 2307-8162. Vol. 6. No. 6. 2018. С. 1–13.

[2] Мельников Б. Ф., Пивнева С. В, Трифонов М. А. *Мультиэвристический подход к сравнению качества определяемых метрик на множестве последовательностей ДНК*. Современные информационные технологии и ИТ образование. Т. 13. № 2. 2017. С. 89–96.

[3] Мельников Б. Ф., Тренина М. А., Кочергин А. С. *Подход к улучшению алгоритмов расчёта расстояний между цепочками ДНК (на примере алгоритма Нидлмана – Вунша)*. Известия высших учебных заведений. Поволжский регион. Физико-математические науки. № 1 (45). 2018. (https://izvuz_fmn.pnzgu.ru/fmn4118)

[4] Melnikov B. F., Pivneva S. V., Trifonov M. A. *Comparative analysis of algorithms calculating distances of DNA sequences and some related problems*. Сборник трудов III международной конференции и молодежной школы «Информационные технологии и нанотехнологии (ИТНТ-2017)», Самарский национальный исследовательский университет имени академика С.П. Королева. 2017. С. 1640–1645.

[5] Мельников Б. Ф., Радионов А. Н. *О выборе стратегии в недетермированных антагонистических играх*. Программирование. № 5. 1998. С. 55–62.

[6] Мельников Б. Ф. *Эвристики в программировании недетерминированных игр*. Известия РАН. Программирование. № 5. 2001. С. 63–80.

[7] Гудман С, Хидетниеми С. *Введение в разработку и анализ алгоритмов*. М.: Мир. 1981. 364 с.

[8] Hromkovic J. *Algorithmics for Hard Problems. Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Springer. 2003. 538 p.

[9] Melnikov B. *Multiheuristic approach to discrete optimization problems*. Cybernetics and Systems Analysis. 2006. Vol. 42. No. 3. P. 335–341.

[10] Home – Nucleotide – NCBI [Электрон. ресурс]. – Режим доступа: https://www.ncbi.nlm.nih.gov/nuccore, свободный

[11] Айала Ф, Кайгер Дж. *Современная генетика*. Пер. с англ. Т. 1. – М.: Мир. 1987. 295 с.

[12] Melnikov B., Radionov A., Moseev A., Melnikova E. *Some specific heuristics for situation clustering problems*. ICSOFT, Technologies, Proceedings 1st International Conference on Software and Data Technologies. 2006. P. 272–279.

[13] Мельников Б. Ф., Мельникова Е. А. *Кластеризация ситуаций в алгоритмах реального времени для задач дискретной оптимизации*. Системы управления и информационные технологии. Т. 28. № 2. 2007. С. 16–20.

Приложение 1. Блок-схемы алгоритмов

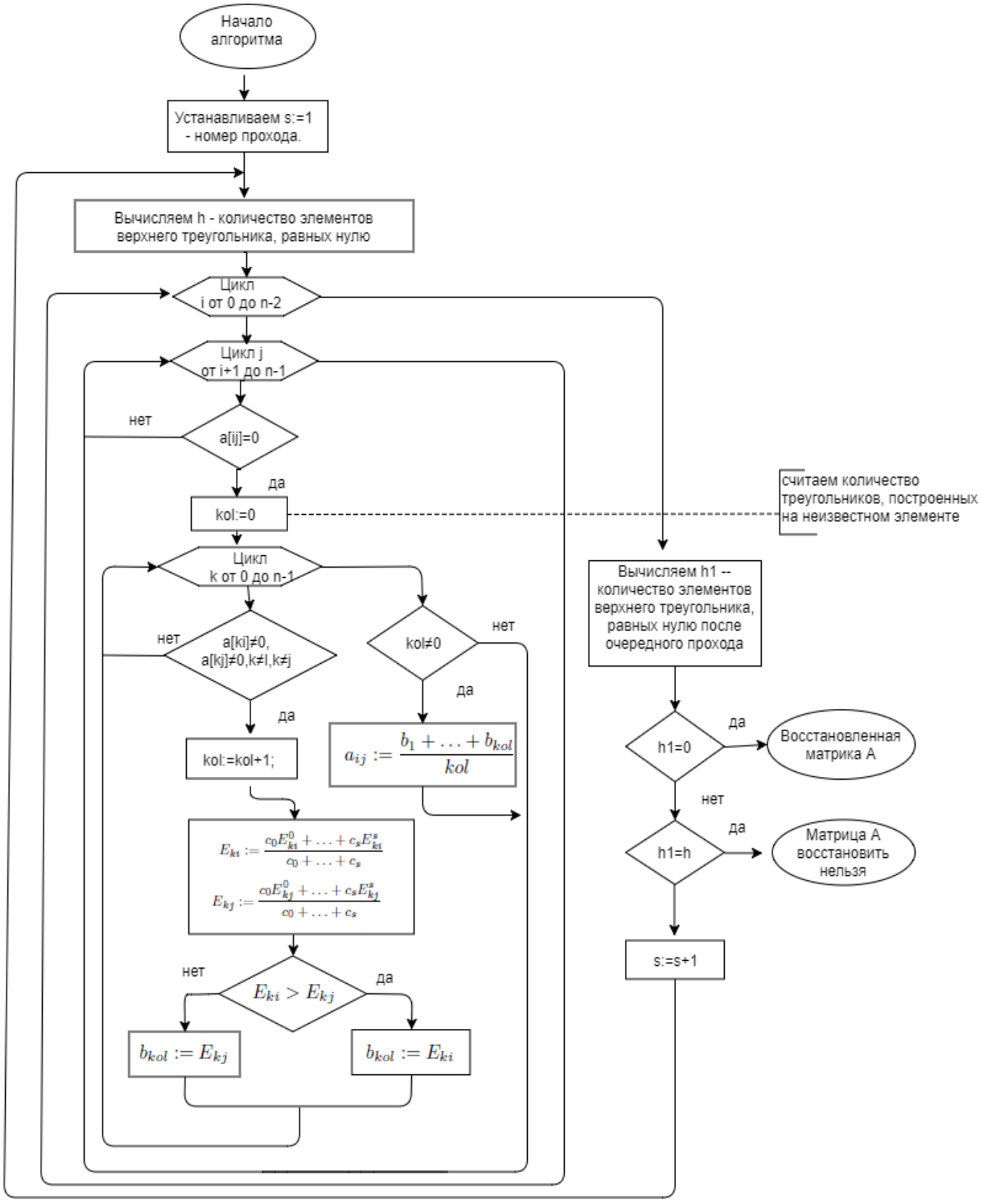

Рис. 2. Блок-схема исходного алгоритма 1.

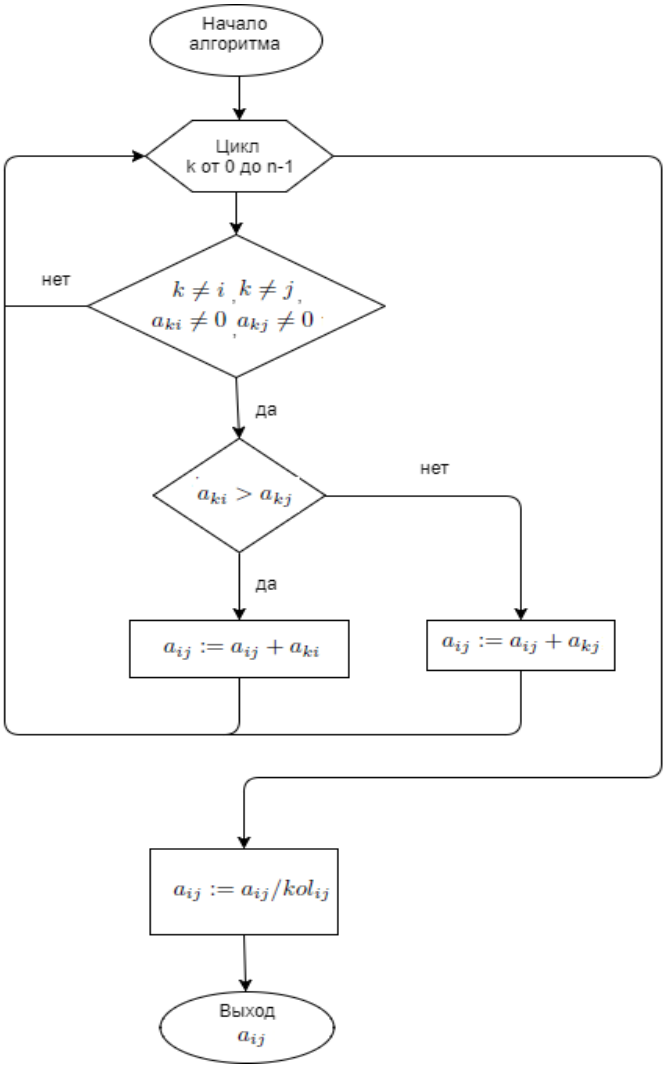

Рис. 3. Блок-схема вспомогательного алгоритма 2 вычисления неизвестного элемента.

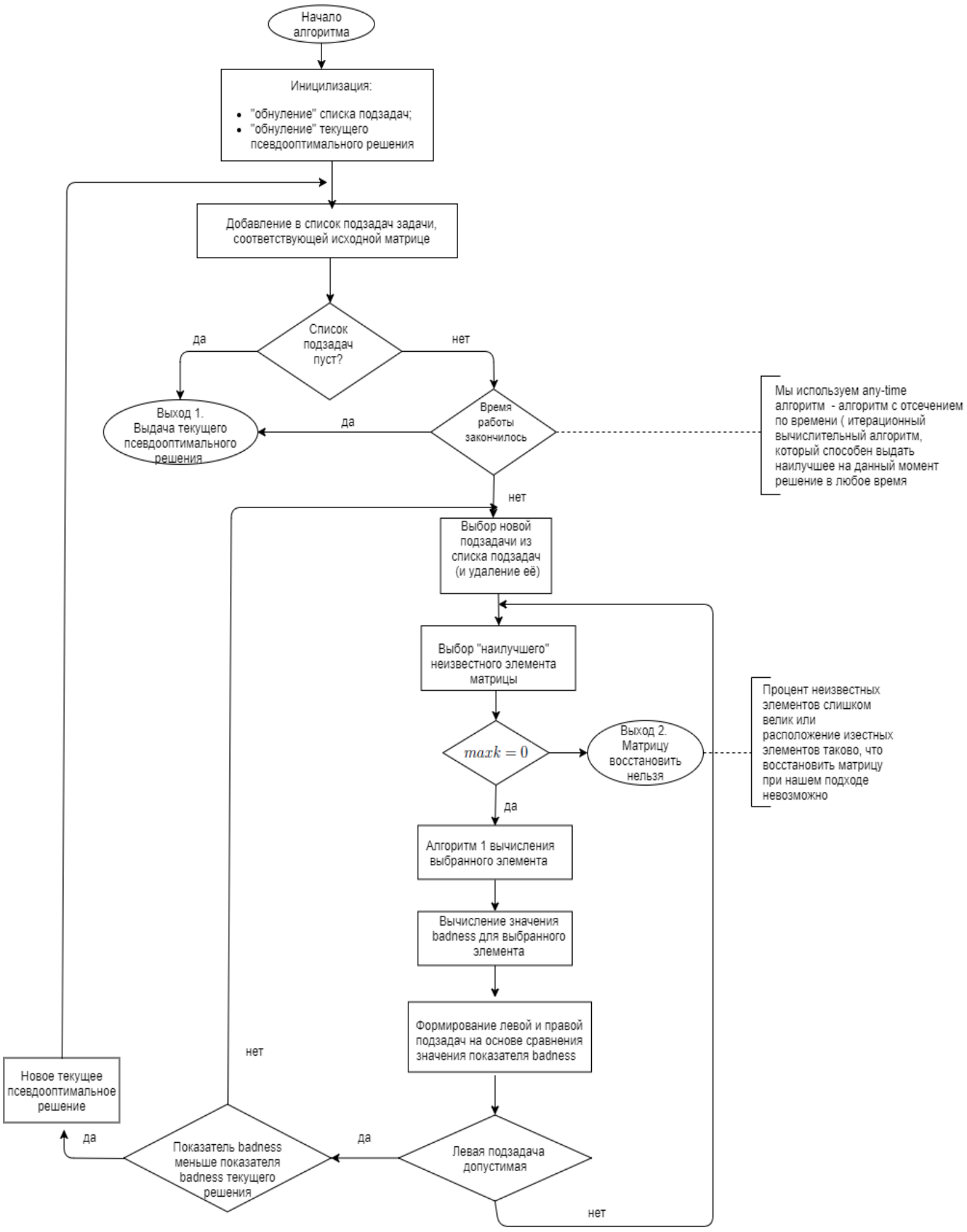

Рис. 4. Блок-схема алгоритма 3 метода ветвей и границ для восстановления неполностью заполненной матрицы расстояний между последовательностями ДНК.

Приложение 2. Результаты восстановления матрицы 28×28 различными способами.

Таблица 16. Восстановленная без метода ветвей и границ матрица размера 28.

0	296	258	262	294	298	275	258	398	294	296	297	287	266	285	288	262	271	269	282	280	996	330	294	303	293	288	307
296	0	267	266	290	309	277	222	398	295	300	298	286	268	289	292	297	274	271	287	292	995	335	269	308	297	292	311
258	267	0	263	343	339	324	291	398	320	308	324	307	293	298	300	295	293	293	294	290	993	337	313	312	273	308	310
262	266	263	0	303	310	292	274	408	299	303	279	296	276	291	293	282	277	279	287	257	996	335	301	308	299	296	309
294	290	343	303	0	307	284	236	395	302	308	305	301	281	295	297	288	275	283	303	286	995	334	306	308	303	301	310
298	309	339	310	307	0	284	290	401	304	351	317	314	300	303	305	298	290	295	296	337	994	345	317	320	315	314	317
275	277	324	292	284	284	0	274	404	318	311	310	307	288	294	297	289	279	288	292	289	994	341	313	314	309	306	313
258	222	291	274	236	290	274	0	406	301	301	305	291	272	287	290	278	273	274	285	282	994	339	298	309	298	294	311
398	398	398	408	395	401	404	406	0	416	394	505	506	414	381	379	410	406	403	400	397	993	435	417	414	412	410	406
294	295	320	299	302	304	318	301	416	0	312	310	302	292	295	297	290	284	290	293	289	994	344	314	317	310	307	313
296	300	308	303	308	351	311	301	394	312	0	320	312	294	300	302	294	289	292	326	288	994	344	332	321	315	313	268
297	298	324	279	305	317	310	305	505	310	320	0	301	295	299	302	295	290	292	315	295	993	352	320	324	317	313	322
287	286	307	296	301	314	307	291	506	302	312	301	0	283	297	300	297	293	287	296	295	994	351	310	323	307	306	321
266	268	293	276	281	300	288	272	414	292	294	295	283	0	297	300	276	272	272	285	264	999	331	307	322	306	302	322
285	289	298	291	295	303	294	287	381	295	300	299	297	297	0	301	313	294	264	270	278	940	391	350	299	341	337	335
288	292	300	293	297	305	297	290	379	297	302	302	300	300	301	0	313	282	320	287	290	987	401	359	256	346	342	341
262	296	295	282	288	298	289	278	410	290	294	295	297	276	313	313	0	279	326	291	289	980	344	306	312	302	299	315
271	274	293	277	275	290	279	273	406	284	289	290	293	272	294	282	279	0	279	287	282	899	287	305	265	301	297	311
269	271	293	279	283	295	288	274	403	290	292	292	287	272	264	320	325	279	0	289	286	873	340	302	312	300	296	314
282	287	294	287	303	296	292	285	400	293	326	315	296	285	270	287	291	287	289	0	303	878	349	312	322	312	310	319
280	292	290	257	286	337	289	282	397	289	288	295	295	264	278	290	289	282	286	303	0	991	32	12	315	308	305	314
996	995	993	996	995	994	994	994	993	994	995	993	994	999	940	987	980	899	873	878	991	0	995	884	873	864	855	847
330	335	337	335	334	345	341	339	435	344	344	352	351	331	391	401	344	287	340	349	342	995	0	335	326	330	325	327
294	269	313	301	306	317	313	298	417	314	332	320	310	307	350	359	306	305	302	312	312	884	335	0	309	297	292	312
303	308	312	308	308	320	314	309	414	317	321	324	323	322	299	256	312	264	312	322	315	873	326	309	0	307	302	308
293	297	273	299	303	315	309	298	412	310	315	317	307	306	341	346	302	301	300	312	308	864	330	297	307	0	244	310
288	292	308	296	301	314	306	294	410	307	313	313	306	302	337	342	299	297	296	310	305	855	325	292	302	244	0	309
307	311	310	309	310	317	313	311	406	313	268	322	321	322	335	341	315	311	314	319	314	847	327	312	308	310	309	0

Таблица 17. Восстановленная с применением метода ветвей и границ матрица размера 28.

0	297	258	269	336	335	303	305	466	313	304	299	299	274	285	280	262	297	320	311	295	990	276	311	275	265	296	306
297	0	293	294	292	305	307	222	467	313	329	307	303	300	280	297	297	303	311	319	292	990	288	269	283	296	303	312
258	293	0	297	343	339	347	350	487	357	353	325	295	294	288	271	309	332	305	354	340	999	321	354	318	273	314	330
269	294	297	0	331	341	292	306	503	315	305	279	272	271	262	255	291	282	302	309	257	989	292	307	287	270	306	303
336	292	343	331	0	286	311	236	467	311	337	313	310	335	282	298	307	308	304	304	332	990	292	325	286	309	336	315
335	305	339	341	286	0	314	304	468	318	351	313	321	343	284	301	315	312	292	334	337	989	317	310	315	313	343	351
303	307	347	292	311	314	0	295	467	318	305	292	296	305	276	267	291	279	299	307	299	990	309	315	301	289	303	306
305	222	350	306	236	304	295	0	466	295	316	287	313	311	253	273	306	282	297	307	305	999	311	299	308	298	314	315
466	467	487	503	467	468	467	466	0	468	466	505	506	466	416	417	467	466	467	468	466	999	433	468	432	466	467	432
313	313	357	315	311	318	318	295	468	0	313	295	302	309	281	276	292	294	306	310	308	990	324	314	318	290	320	316
304	329	353	305	337	351	305	316	466	313	0	320	311	309	293	286	308	300	320	326	304	990	282	332	281	304	315	268
299	307	325	279	313	313	292	287	505	295	302	0	286	307	274	269	283	281	293	315	300	999	281	315	281	282	300	310
299	303	295	272	310	321	296	313	506	302	311	286	0	276	270	265	297	283	309	312	299	999	287	309	283	273	304	309
274	300	294	271	335	343	305	311	466	309	309	307	276	0	269	260	296	303	304	312	264	999	287	311	275	274	303	298
285	280	288	262	282	284	276	253	416	281	293	274	270	269	0	301	321	303	264	295	295	915	298	286	300	294	296	295
280	297	271	255	298	301	267	273	417	276	286	269	265	260	301	0	316	292	321	299	284	924	294	302	256	286	298	285
262	297	309	291	307	315	291	306	467	292	308	283	297	296	321	316	0	277	326	310	297	990	282	311	278	287	303	309
297	303	332	282	308	312	279	282	466	294	300	281	283	303	303	292	277	0	285	307	297	990	287	313	265	272	306	305
320	311	305	302	304	292	299	297	467	306	320	293	309	304	264	321	326	285	0	313	308	999	290	303	289	299	312	319
311	319	354	309	304	334	307	307	468	310	326	315	312	312	295	299	310	307	313	0	303	990	292	312	288	310	311	316
295	292	340	257	332	337	299	305	466	308	304	300	299	264	295	284	297	297	308	303	0	990	281	307	273	292	305	313
990	990	999	989	990	989	990	999	999	990	990	999	999	999	915	924	990	990	999	990	990	0	995	990	990	999	997	990
276	288	321	292	292	317	309	311	433	324	282	281	287	287	298	294	282	287	290	292	281	995	0	292	280	294	282	286
311	269	354	307	325	310	315	299	468	314	332	315	309	311	286	302	311	313	303	312	307	990	292	0	290	309	320	318
275	283	318	287	286	315	301	308	432	318	281	281	283	275	300	256	278	265	289	288	273	990	280	290	0	287	285	274
265	296	273	270	309	313	289	298	466	290	304	282	273	274	294	286	287	272	299	310	292	999	294	309	287	0	244	300
296	303	314	306	336	343	303	314	467	320	315	300	304	303	296	298	303	306	312	311	305	997	282	320	285	244	0	306
306	312	330	303	315	351	306	315	432	316	268	310	309	298	295	285	309	305	319	316	313	990	286	318	274	300	306	0

The application of the branch and bound method in the problem of reconstructing the matrix of distances between DNA strings

Boris Melnikov, Marina Trenina

Abstract—**In this paper, we continue to consider one of the tasks of biocybernetics, i.e. the problem of reconstructing the distance matrix between DNA sequences. In this problem, not all the elements of the matrix under consideration are known at the input of the algorithm (usually 50% or less of the elements). The basis for the development of the algorithm for reconstructing such a matrix is the method of comparative evaluation of algorithms for calculating distances between DNA sequences developed and investigated by us, based on a special analysis of the distance matrix. In this analysis, we applied the badness of each of the triangles of the matrix determined by us before.**

Continuing to improve the algorithms for solving this problem, we consider the use of the branches and bounds method in it. To do this, for some known sequence of unfilled elements, we apply the algorithms we considered in previous publications, but now we choose the sequences ourselves using developed by us version of method of branches and bounds. In our interpretation of this method, all possible sequences of unknown elements of the upper triangular part of the matrix are taken as the set of admissible solutions. In each current subtask, any of the blank elements of the matrix is taken as the separating element, and the sum of the badness values for all triangles that have already been formed by the time this subtask is considered is taken as the boundary. Thus, the definition of elements of an incompletely filled matrix occurs in such a sequence that the final badness indicator for all triangles is selected using greedy heuristics that fits completely into the framework of the classical variants of the description of the branches and bounds method.

As a result of applying such an algorithm, we get results that, from the point of view of the value of badness, are the lowest possible (in the case of a completed version of the branch and boundary method), or close to optimal ones (in the case of its uncompleted version). At the same time, in our computational experiments, the running time of the algorithm practically coincides with the time of the algorithm considered by us in the previous publication (it exceeds it by no more than 10%), and the badness value usually decreases by 20-40% from the initial value. Thus, we are able to quickly and efficiently restore the DNA matrix, often even if it is filled less than 40%.

Keywords—**branches and bounds method, DNA sequences, metric, distance matrix, partially filled matrix, recovery.**