Краткое описание и инструкция по использованию программы TwoBox

Оглавление

1. Общие сведения

Программа TwoBox (версия 3.17) предназначена для поиска в наборе исходных последовательностей, составленных из символов фиксированного алфавита, системы наиболее похожих друг на друга сайтов заданной длины, так чтобы из каждой последовательности бралось не более одного сайта. Программа стремится в первую очередь искать по одному сайту в каждой последовательности, но может и исключать некоторые последовательности из поиска, если такое решение оказывается лучшим в смысле используемого функционала качества всей системы (см. раздел 2).

Разыскиваемый сайт может быть представлен одним боксом (т.е. непрерывным участком исходной последовательности), либо состоять из двух боксов, разделенных некоторым числом символов (фиксированным или произвольным в заданном интервале). Длина каждого бокса в случае двухбоксового сайта задается независимо.

Возможен также поиск с учетом априорной информации о всех или некоторых позициях одного или обоих боксов сайта. Эта известная информация задается в форме мотива бокса. Подробное описание входных данных и параметров приводится в разделах 3–5.

Программа TwoBox представляет собой развитие ранее разработанного эвристического алгоритма поиска однобоксового регуляторного сигнала [1-3] методом глобальной оптимизации заданного функционала качества. В результате работы алгоритма находится квазиоптимальное решение, соответствующее максимальному значению функционала по всем локальным экстремумам, достигнутым в ходе поиска, ограничиваемого по ряду внутренних критериев алгоритма, либо по времени или по числу итераций алгоритма. Описание выходной информации программы приведено в разделе 6.

Указанный алгоритм распространен на случай двухбоксового сайта следующим образом. Напомним, что в изначальном алгоритме многократно используется элементарная операция перебора всевозможных позиций сайта-кандидата на полной длине последовательности, т.е. для последовательности из m символов и l-буквенного сайта перебирается $m-l+1$ вариантов начала. Если же искомый сайт состоит из двух боксов с длинами l', l'', между которыми

находится в точности d символов, то аналогичный полный перебор эквивалентен перебору всех боксов длиной $l = l' + l''$ в последовательности с длиной $m - d$, т.е. охватывает $(m-d)-l+1$ вариантов, которые можно трактовать как позиции начала бокса с суммарной длиной в некоторой виртуальной последовательности, получаемой из исходной с помощью неявно заданного отображения. Если длина промежутка между боксами может варьироваться в интервале $[d_{\min}, d_{\max}]$, то общее число подлежащих перебору вариантов равно

$$\sum_{d=d_{\min}}^{d_{\max}} [(m-d)-l+1] = (m-l+1)\cdot D - \bar{d}\cdot D,$$

где $\bar{d} = \frac{1}{2}(d_{\min} + d_{\max})$ – средняя длина промежутка, а $D = d_{\max} - d_{\min} + 1$ – ширина диапазона возможных длин. Отбрасывая второй член, получаем, что неопределенность длины промежутка в интервале шириной D приблизительно эквивалентна увеличению длины последовательности-образа в D раз по сравнению с фиксированной длиной промежутка. Само отображение, результатом которого будет такая последовательность, задается неявно – устройством алгоритма.

Как отмечалось в [1, 3], оценка вычислительной сложности алгоритма поиска однобоксового сайта пропорциональна квадрату числа исходных последовательностей и кубу их средней длины. С учетом рассуждений выше, в случае сайта из двух боксов с фиксированной длиной промежутка вычислительная сложность такая же, а при переменной длине межбоксового промежутка в интервале шириной D – возрастает пропорционально D^3.

Учитывая вычислительную трудоемкость изначального алгоритма (которая вдобавок быстро растет в случае двухбоксового сайта с неизвестной длиной промежутка между боксами), программа TwoBox ориентирована главным образом на параллельные вычислительные установки, в которых межпроцессорный обмен информацией реализуется средствами протокола MPI версии 1.1 или выше [4, 5]. Число процессоров кластера не регламентируется; программа в состоянии задействовать все доступные процессоры; при этом общее время счета снижается за счет распараллеливания приблизительно в $s-1$ раз, где s – число процессоров [2]. Необходимо минимум два логических процессора, так что программа в принципе работоспособна и на обычном двухядерном ПК.

Предлагаемая версия программы (исполняемый модуль архитектуры x86) предназначена для проведения расчетов на кластере, состоящем из одного и более IBM-совместимых ПК с операционной системой Windows, связанных по локальной сети. Среда MPI в этом случае организуется с помощью свободно распространяемого продукта MPICH2 v.1.2 (разработчик – Argonne National Laboratory, http://www.mcs.anl.gov/mpi/mpich2). Для работы с программой этот продукт (или его последующая версия) должен быть установлен на используемых компьютерах.

Программа имеет интерфейс командной строки и рассчитана на запуск в среде командного процессора операционной системы. Язык программирования – C, компилятор Microsoft Visual Studio 2005 Service Pack 1, имя исполняемого модуля – twobox.exe. Целевой процессор – Intel 32-битной архитектуры. Целевые операционные системы – Microsoft Windows XP Service Pack 3, Microsoft Windows Server 2003 Service Pack 2. Использование программы TwoBox на других типах процессоров и операционных систем вполне возможно, но может потребовать дополнительного тестирования и/или перекомпиляции.

Разработчик программы: Л.И. Рубанов, в.н.с. ИППИ РАН им. А.А. Харкевича

Контактный e-mail: rubanov@iitp.ru

2. Функционалы качества системы слов

Алгоритм использует два функционала качества; оба они базируются на близости по Хэммингу, т.е. в качестве меры сходства двух сайтов-кандидатов принимается число позиций, на которых в обоих сайтах стоит одна и та же буква. Пусть заданы n исходных последовательностей, длина каждой не превосходит m, и в каждой (или почти каждой) последовательности необходимо выбрать не более одного сайта, в виде бокса w с длиной l (однобоксовый случай) или двух боксов w', w'' с длинами, соответственно, l', l'' и промежутком между боксами с длиной $d \in [d_{\min}, d_{\max}]$ (двухбоксовый случай) – таким образом, чтобы выбранная система сайтов характеризовалась наибольшим сходством. Ниже излагается реализованный в алгоритме способ вычисления величины такого сходства.

В однобоксовом случае качество сайта w_k, найденного в k-й последовательности, относительно всей построенной системы сайтов вычисляется по формуле

$$q_k = \sum_{\substack{i=1 \\ i \neq k}}^{n} \left(l - H(w_i, w_k) \right), \tag{1}$$

где $H(w_i, w_k)$ - расстояние Хэмминга между боксами w_i, w_k (число совпадений букв на соответственных позициях обоих боксов).

В двухбоксовом случае качество найденного сайта (пары боксов w'_k, w''_k) относительно всей построенной системы таких пар вычисляется по формуле

$$q_k = \sum_{\substack{i=1 \\ i \neq k}}^{n} \left(l' + l'' - H(w'_i, w'_k) - H(w''_i, w''_k) - P(d_i) - P(d_k) \right), \tag{2}$$

где функция $P(d)$ определяет величину штрафа за отклонение расстояния между боксами от предполагаемого «наилучшего» значения в интервале от $d_{\min}$ до $d_{\max}$. Вид этой функции является параметром программы; в частном случае она может быть тождественно равна 0.

Формулы (1) и (2) относятся к случаю, когда нет априорной информации о буквенном составе искомых боксов. Наряду с этим, возможна ситуация, когда заранее известно, что на каких-то позициях боксов во всех последовательностях (или в большинстве их) стоят определенные буквы. Программа позволяет задавать для одного или обоих отыскиваемых боксов «мотив» в форме строки символов той же длины, что и бокс. В каждой позиции мотива должна стоять либо буква алфавита последовательностей (заглавная или строчная, что интерпретируется по-разному), либо фиктивный символ (точка, звездочка или знак вопроса), означающий отсутствие информации о данной позиции.

В однобоксовом случае с указанным мотивом формула качества сайта (1) имеет вид:

$$q_k = \sum_{\substack{i=1 \\ i \neq k}}^{n} \left(l - H(w_i, w_k) + \sum_{j=1}^{l} (\delta_{kj} - \Delta_{kj}) \right), \tag{3}$$

где $\delta_{kj} = \delta$, если в j-й позиции сайта из k-й последовательности стоит *в точности та же* буква, что и в j-й позиции мотива (неважно, строчная или заглавная), и $\delta_{kj} = 0$ в остальных случаях; а $\Delta_{kj} = \Delta$, если в j-й позиции сайта из k-й последовательности стоит *не та* буква, которая указана на j-й позиции мотива *как заглавная*, и $\Delta_{kj} = 0$ в остальных случаях. Другими словами, качество сайта премируется на δ за каждое совпадение позиции последовательности с мотивом и штрафуется на Δ за каждое несовпадение с мотивом на тех позициях, где оно особо желательно (что показывают заглавные буквы в мотиве). Константы δ и Δ являются параметрами программы.

В двухбоксовом случае с указанными мотивами боксов формула (2) имеет вид:

$$q_k = \sum_{\substack{i=1 \\ i \neq k}}^{n} \left(l' + l'' - H(w_i', w_k') - H(w_i'', w_k'') - P(d_i) - P(d_k) + \sum_{j=1}^{l'} (\delta_{kj} - \Delta_{kj}) + \sum_{j=1}^{l''} (\delta_{kj} - \Delta_{kj}) \right), \quad (4)$$

где первая сумма внутри скобок вычисляется по первому боксу, вторая – по второму, а смысл обозначений δ_{kj}, Δ_{kj} тот же, что и в (3). (Если мотив указан только для одного из боксов, то одна из этих сумм принимается равной нулю). Наконец, если вычисленная по формулам (2)–(4) величина качества слова оказывается отрицательной, берется нулевое значение.

После того, как по формулам (1)–(4) вычислено качество всех сайтов системы, первый функционал качества системы в целом задается соотношением

$$Q_1 = \frac{1}{P-1} \sum_{k=1}^{n} q_k \to \max, \quad (5)$$

где P – число выбранных непустых сайтов во всех последовательностях (мощность системы).

Второй функционал качества всей системы имеет вид

$$Q_2 = \frac{1}{P(P-1)} \sum_{k=1}^{n} q_k \to \max. \quad (6)$$

Функционал Q_1 охарактеризует среднее качество сайтов системы, а функционал Q_2 описывает среднюю величину сходства сайтов системы. В предельном случае точного совпадения всех сайтов максимумы обоих функционалов совпадают, а в прочих ситуациях Q_1 используется как основной для оптимизации, а Q_2 учитывается при совпадении значений первого функционала.

Кроме того, как альтернатива мотиву искомых сайтов, в программе предусмотрена возможность использования функционала Q_1 в модифицированной форме, нацеленной на поиск наиболее консервативных сайтов. В однобоксовом случае используется функционал вида

$$Q_1 = \frac{1}{P-1} \sum_{k=1}^{n} q_k + \sum_{j=1}^{l} c_j, \quad (7)$$

а в двухбоксовом применяется формула

$$Q_1 = \frac{1}{P-1} \sum_{k=1}^{n} q_k + \sum_{j=1}^{l'} c_j + \sum_{j=1}^{l''} c_j. \quad (8)$$

Значения c_j в формулах (7),(8) доставляются функцией $c(r)$, монотонно убывающей по мере уменьшения консервативности системы на j-й позиции. Иначе говоря, за каждую абсолютно консервативную позицию в найденных сайтах, когда во всех стоит одна и та же буква на j-й позиции, дается максимальный приз, а если в одном или более сайтах системы встречается другая буква, то величина этого приза снижается. Конкретный вид функции $c(r)$ является параметром программы. В данной версии это ступенчатая функция с тремя значениями: одно значение для абсолютно консервативной позиции, другое – для консервативной позиции, где одна и та же буква стоит всюду, кроме заданного небольшого числа последовательностей, и значение 0 в остальных случаях Подчеркнем, что функционал качества в форме (7) или (8) применяется только в том случае, когда поиск сайтов ведется без использования мотива.

3. Описание входных данных

Входные данные программы передаются через аргументы командной строки, которая должна иметь следующий формат (элементы в квадратных скобках необязательны):

```
TWOBOX [параметры] infile [outfile]
```

- параметры позволяют изменять установленные по умолчанию режим работы программы и характеристики алгоритма; признаком параметра является предшествующий ему символ «дефис» (-) или «наклонная черта» (/);
- аргумент `infile` указывает имя файла исходных данных, содержащего один или несколько наборов последовательностей в формате FASTA; может быть также указан абсолютный или относительный путь к файлу, в противном случае используется текущий каталог;
- аргумент `outfile` указывает имя файла для записи результатов работы программы; если этот аргумент опущен, то будет использоваться имя входного файла с добавлением расширения ".out". Как и в случае входного файла, может быть указан абсолютный или относительный путь к файлу, в противном случае используется текущий каталог.

В соответствии с соглашениями командного процессора Windows, заглавные и строчные буквы в командной строке интерпретируются одинаково. Если имя файла или путь к нему содержат пробелы, соответствующий аргумент должен указываться в кавычках.

Форматы входных данных программы TwoBox приведены в разделах 4 и 5. Краткую подсказку можно получить по команде `TWOBOX -h` или `TWOBOX -?` (эквивалентные формы команды – `TWOBOX /h`, `TWOBOX /?`).

4. Параметры командной строки

Программа TwoBox воспринимает следующие параметры (заглавные и строчные буквы интерпретируются одинаково):

`-a` Если указан этот параметр, выходной файл открывается в режиме дозаписи в конец, что может быть полезно при запуске в пакетном режиме; по умолчанию файл перезаписывается.

`-acid` Если указан этот параметр, в исходных последовательностях используется алфавит аминокислот; по умолчанию используется алфавит нуклеотидов.

`-ba`<число> Значение функции c_j для абсолютно консервативной позиции (см. раздел 2); по умолчанию 3.

`-bc`<число> Значение функции c_j для консервативной позиции с исключениями (см. раздел 2); по умолчанию 1.

`-bx`<число> Максимально допустимое число исключений, т.е. последовательностей, у которых значение на некоторой позиции отличается от преобладающего, при котором эта позиция еще считается консервативной (см. раздел 2); по умолчанию 1.

`-c`<число> Управляет режимом выдачи диагностической информации на консоль (поток *stdout*) из корневой ветви программы:
`-c0` – минимальная выдача, `-c1` – стандартная выдача, `-c2` – максимальная выдача. Параметр может использоваться для наблюдения за ходом работы программы.

`-d`<число> Управляет режимом выдачи диагностической информации в выходной файл каждой ветви. В зависимости от значения параметра `-z`, в конце работы выходные файлы от вторичных ветвей алгоритма либо удаляются, либо присоединяются в конец выходного файла корневой ветви, либо остаются на месте. По умолчанию никакая диагностика не выдается, и в выходной файл попадает только найденное решение (решения). Этот параметр может быть полезен для анализа траектории поиска оптимума и возникающих ошибочных ситуаций.

`-f` Если указан этот параметр, то по окончании работы алгоритма создается файл в формате CSV, содержащий всю последовательность найденных решений, который легко импортируется в электронную таблицу Excel для последующего анализа.

`-g`<режим> Управляет режимом совместного учета (агрегирования) данных о качестве сигнала от предшествующих итераций алгоритма, когда обрабатывались другие расстановки в пределах того же *Q*-списка [2]:
`-g1` (или `-gs`) – значения суммируются (действует по умолчанию),
`-g2` (или `-gm`) – выбирается максимальное значение.

`-h` Вывод подсказки о формате командной строки и параметрах программы.

`-i`<число> Задает максимальное число выполняемых итераций алгоритма, при которых не растет функционал качества (в пределах текущего *Q*-списка). Используется для критерия 1 остановки алгоритма. По умолчанию принимается значение 5, однако для задач размерности n>16 его целесообразно увеличивать (например, до $(0.5\ldots2)\cdot n$).

`-j`<режим> Указывает режим учета новизны сайтов в критерии 1 остановки алгоритма:
`-j0` (или `-ji`) – не учитывать фактор новизны сайтов в критерии 1,
`-j1` (или `-ja`) – учитывать появление новых сайтов при проверке критерия 1,
`-j2` (или `-jr`) – сбрасывать счетчик (его порог задается параметром `-i`) при обнаружении новых сайтов.

`-l`<число> Указывает длину единственного l (или первого l') бокса в искомых сайтах (по умолчанию 18).

`-ll`<число> Указывает длину второго бокса l'' в искомых сайтах (по умолчанию 0, т.е. программа работает в однобоксовом режиме).

`-m'`мотив`'` Указывает мотив единственного (или первого) бокса в искомых сайтах (по умолчанию отсутствует). Мотив задается строкой символов, заключенной в апострофы, длина строки должна совпадать со значением ключа `-l`. В строке могут стоять либо символы алфавита исходных последовательностей (заглавные или строчные), либо фиктивные символы (точка, знак вопроса или звездочка). См. интерпретацию символов в разделе 2.

`-mm'`мотив`'` Указывает мотив второго бокса в искомых сайтах (по умолчанию отсутствует). Мотив задается строкой символов, заключенной в апострофы, длина строки должна совпадать со значением ключа `-ll`. В остальном параметр аналогичен параметру `-m`.

`-mb`<число> Устанавливает величину δ для формул (3), (4); значение по умолчанию 1.

`-mp`<число> Указывает величину Δ для формул (3), (4); значение по умолчанию 5.

`-n`<число> Указывает максимальную длину имени последовательности, выдаваемую на печать. По умолчанию отсечение имен по длине не производится.

`-p`<число> Указывает минимальное число пар последовательностей, которые должны быть взяты в качестве основных ребер на каком-то из уровней дерева сборки. Этот параметр используется в критерии 2 остановки алгоритма и косвенно определяет степень полноты перебора множества расстановок. В данной версии параметр должен указываться только в форме `-p0`, что означает перебор всех $n(n-1)/2$ пар.

`-q`<число> Указывает максимально допустимое число расстановок в *Q*-списке, что позволяет дополнительно ужесточить критерий 1. Значение `-q0` означает отсутствие такого ограничения (принимается по умолчанию).

`-r`<число> Задает число лучших результатов, выдаваемых в выходной файл (по умолчанию 3). Целесообразность этого параметра обусловлена тем, что использованный функционал качества может недостаточно хорошо согласовываться с биологической сущностью задачи, так что по биологическим соображениям искомому сигналу необязательно соответствует абсолютный оптимум функционала (хотя и

близкая по значению точка). Благодаря такой возможности пользователь может самостоятельно выбирать наиболее биологически правильное решение из предлагаемых программой.

`-smin<число>` Минимально допустимое расстояние $d_{\min}$ между двумя боксами в случае поиска сайтов, состоящих из двух боксов. Значение по умолчанию 16.

`-smax<число>` Максимально допустимое расстояние $d_{\max}$ между двумя боксами в случае поиска сайтов, состоящих из двух боксов. Значение по умолчанию 22.

`-spen"строка"` Параметр определяет величину штрафа за расстояние между двумя боксами, т.е. функцию $P(d)$ (подробнее см. раздел 2). Задается взятой в двойные кавычки строкой, где через запятую перечислены значения функции. Первое значение соответствует расстоянию $d = d_{\min}$, последнее – расстоянию $d = d_{\max}$. В строке должно присутствовать в точности $D = d_{\max} - d_{\min} + 1$ значений; по умолчанию используется строка вида "8,0,0,2,4,6,8".

`-t<число>` Указывает используемый в алгоритме нижний порог сходства сайтов, т.е. числа несовпадающих символов на соответственных позициях любой пары боксов. По умолчанию в однобоксовом режиме принимается значение $l/2$, где l – указанная явно параметром `-l` или принятая по умолчанию длина сайтов сигнала. В двухбоксовом режиме по умолчанию принимается значение $(l'+l'')/2$.

`-u<число>` Верхний предел количества итераций алгоритма (т.е. максимальное число проверяемых расстановок из общего их числа $n!$). Значение по умолчанию – 10000.

`-v<число>` Число расстановок Q-списка, порождаемых одним пучком для упрощения выбора траектории и повышения стабильности алгоритма (по умолчанию 3).

`-w<число>` Устанавливает предельное время работы программы (в минутах). Принятое по умолчанию значение `-w0` означает отсутствие ограничения по времени.

`-x<режим>` Способ вычисления сходства сайтов-кандидатов:

`-x0`: вычисление сходства непосредственно каждый раз, когда это требуется (этот режим используется при ограниченном объеме физической оперативной памяти, поскольку он оказывается эффективнее, чем работа с матрицей сходства, части которой размещаются в файле динамической подкачки);

`-x1`: в начале решения задачи вычисляется треугольная матрица попарного сходства всевозможных сайтов-кандидатов, которая затем используется в ходе всего алгоритма (этот принимаемый по умолчанию режим позволяет ускорить время работы алгоритма в 3–4 раза, однако требует для хранения матрицы близости порядка $m^2 \cdot n^2$ (байтов); так, для обработки 30 последовательностей с приведенной длиной 1000 нт (имеется в виду результат неявного отображения, описанного в разделе 1) необходимо не менее 1 Гб физической памяти на каждый процессор).

`-y<число>` Указывает значение параметра критерия 2 – минимальное значение кратности покрытия для каждой пары (при параметре `-p0`). По умолчанию принимается значение $\log_2 n$ (для n, не являющегося степенью 2, значение логарифма берется «сверху», по ближайшей большей степени двойки). Данный параметр определяет длину формируемого по второму критерию P-списка [2].

`-z<режим>` Режим использования выходных файлов, созданных вторичными ветвями, по окончании работы алгоритма:
`-z0` (или `-zr`) – выходные файлы вторичных ветвей удаляются;

`-z1` (или `-za`) – содержимое выходных файлов вторичных ветвей приписывается в конец основного выходного файла (при параметре `-d2`), а сами файлы удаляются;
`-z2` (или `-zs`) – выходные файлы вторичных ветвей не удаляются.

5. Входной файл

Имя файла исходных данных должно быть указано аргументом `infile` в командной строке запуска программы. Файл содержит набор последовательностей и соответствует формату FASTA с рядом ограничений. Каждая последовательность должна быть представлена в точности двумя строками файла (переносить последовательность на следующие строки нельзя, однако длина строки не ограничивается):

- первая строка начинается символом '>', за которым следует произвольное имя, присвоенное последовательности;
- вторая строка содержит саму последовательность в виде набора строчных или прописных букв из применяемого алфавита. По умолчанию это алфавит нуклеотидов {A, C, T, G, U}, в котором буквы T и U считаются эквивалентными[1]. Если при запуске был указан параметр `-acid`, то должен использоваться алфавит аминокислот {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}. При записи последовательностей разрешено использовать символы делеции (минус, подчеркивание или знак равенства), применяемые для выравнивания или удобства записи, однако пробелы не допускаются.

Соблюдение описанного формата проверяется; при его нарушении выдается сообщение об ошибке во входных данных. Наличие строк иного вида (комментарии и др.) в наборе последовательностей не допускается.

Программа TwoBox позволяет за один запуск обработать (с одинаковыми значениями параметров!) несколько наборов последовательностей, для чего в файле исходных данных их необходимо разделить одной (или более) пустой строкой.

6. Описание выходных данные

6.1 Код возврата

Код возврата программы (возвращаемое значение головной функции) может быть произвольной суммой значений, указанных в табл. 1. Проверка кода возврата иногда бывает полезна при составлении командных файлов (скриптов) для пакетной обработки.

Таблица 1. Значения кода возврата программы

Значение	Символика	Смысл возвращаемого значения
0	OK	Нормальное завершение программы
1	ERARG	Ошибка в аргументах командной строки
2	ERDATA	Ошибка в файле исходных данных
4	ERIO	Ошибка ввода-вывода
8	EODATA	Достигнут конец файла исходных данных
16	ERMEM	Недостаточно памяти для работы программы или ошибка в данных
32	HELP	Был запрошен вывод подсказки
64	EDEBUG	(Зарезервировано)
128	ERINT	Внутренняя ошибка программы
256	ERMPI	Ошибка в протоколе MPI

6.2 Выходной файл результатов

В выходной файл программы выдаются таблицы расположения сайтов в исходных последовательностях для заданного параметром `-r` числа лучших найденных сигналов в

[1] Использование расширенного нуклеотидного алфавита в текущей версии программы не предусмотрено.

порядке убывания величины функционала качества (5) или (7) или (8), а при одинаковой величине – функционала (6) среднего сходства сайтов системы. Файл имеет обычный текстовый формат (рис. 1).

```
Best quality result #1 [18/22] (power: 7, quality: 59.33, avg proximity: 8.47)
  1:XX|PH0858[0142] aaaaaatagaaa (52,  8.66)
  2:XX|PH0859[0006] aaaaagtattaa (53,  8.83)
  3:XX|PH1075[0058] aaaatctaccat (42,  7.00)
  4:XX|PH1086[0176] aaaatatataaa (55,  9.16)
  5:XX|PH1087[0062] aaaaattattaa (53,  8.83)
  6:XX|PH1091[0005] aaaagatattcc (46,  7.66)
  7:XX|PH1093[0167] aaaagatataaa (55,  9.16)

Best quality result #2 [19/22] (power: 7, quality: 59.00, avg proximity: 8.42)
  1:XX|PH0858[0142] aaaaaatagaaa (52,  8.66)
  2:XX|PH0859[0006] aaaaagtattaa (52,  8.66)
  3:XX|PH1075[0056] ataaaatctacc (41,  6.83)
  4:XX|PH1086[0176] aaaatatataaa (54,  9.00)
  5:XX|PH1087[0062] aaaaattattaa (52,  8.66)
  6:XX|PH1091[0005] aaaagatattcc (48,  8.00)
  7:XX|PH1093[0167] aaaagatataaa (55,  9.16)

Best quality result #3 [9/22] (power: 7, quality: 58.66, avg proximity: 8.38)
  1:XX|PH0858[0142] aaaaaatagaaa (51,  8.50)
  2:XX|PH0859[0006] aaaaagtattaa (53,  8.83)
  3:XX|PH1075[0144] taaagtttctaa (40,  6.66)
  4:XX|PH1086[0176] aaaatatataaa (53,  8.83)
  5:XX|PH1087[0062] aaaaattattaa (54,  9.00)
  6:XX|PH1091[0005] aaaagatattcc (46,  7.66)
  7:XX|PH1093[0167] aaaagatataaa (55,  9.16)
```

Рис. 1. Пример содержимого выходного файла в режиме однобоксового поиска.

Каждая таблица начинается строкой, описывающей найденное решение в целом (для примера рассмотрим первую строку на рис. 1). Смысл данных в строке следующий:

- «`Best quality result`» – это решение является лучшим по значению функционала качества (5) или (7) или (8). Если указано «`Best avg proximity result`», то решение является лучшим по значению функционала среднего сходства (6);
- `#1` – номер решения в списке лучших (в порядке убывания качества);
- `[18/22]` – порядковый номер итерации алгоритма, на которой было найдено решение, и общее число проделанных итераций (итерации нумеруются с 0);
- `7` – мощность системы (число последовательностей, в которых найдены сайты);
- `59.33` – значение функционала (5), (7) или (8);
- `8.47` – значение функционала (6).

Далее следуют строки таблицы для всех исходных последовательностей. Смысл значений (на примере второй строки рис. 1) следующий:

- `1` – номер исходной последовательности;
- `XX|PH0858` – имя исходной последовательности;
- `[0142]` – позиция исходной последовательности (считая с 1), соответствующая началу найденного сайта, т.е. позиция 5'-конца искомого бокса;
- `aaaaaatagaaa` – бокс, представляющий данный сайт;
- `52` – качество сайта, вычисленное по формуле (2);
- `8.66` – величина сходства этого сайта с остальными сайтами системы, согласно (4).

Если в такой строке присутствует только номер и имя последовательности, то это означает, что в данном решении из этой последовательности сайт не выбран (т.е. мощность найденного сигнала будет $P < n$).

Пример информации, выдаваемой в режиме двухбоксового поиска, показан на рис. 2. Основные отличия по сравнению с вышеописанным форматом состоят в том, что в строке для каждой последовательности в квадратных скобках выдаются два номера позиций 5'-начал двух боксов, и показаны только сами эти боксы, а для промежутка указана лишь его длина (число букв).

Описанная информация выдается во всех случаях. В зависимости от выбранного (параметром `-d` программы) режима выдачи диагностической информации, в выходной файл могут направляться дополнительные сообщения, промежуточные решения, значения параметров, пояснения по ходу процесса оптимизации, матрица близости и т.д.

```
Best quality result #1 [1/54] (power: 12, quality: 189.63, avg proximity: 15.80)
  1:Citrus sinensis psbI         [0007;0031] ttgatg-18-tataaa (164, 14.90)
  2:Aethionema cordifolium psbI  [0007;0030] ttggta-17-tttggt (143, 13.00)
  3:Aethionema grandiflorum psbI [0007;0030] ttggta-17-tttggt (143, 13.00)
  4:Arabidopsis thaliana psbI    [0007;0031] ttggta-18-tataaa (184, 16.72)
  5:Arabis hirsuta psbI          [0007;0031] ttggta-18-tataaa (184, 16.72)
  6:Barbarea verna psbI          [0007;0031] ttggta-18-tataaa (184, 16.72)
  7:Capsella bursa-pastoris psbI [0007;0031] ttggta-18-tataaa (184, 16.72)
  8:Crucihimalaya wallichii psbI [0006;0030] tttgta-18-tataaa (164, 14.90)
  9:Draba nemorosa psbI          [0007;0031] ttggta-18-tataaa (184, 16.72)
 10:Lepidium virginicum psbI     [0007;0031] ttggta-18-tataaa (184, 16.72)
 11:Lobularia maritima psbI      [0007;0031] ttggta-18-tataaa (184, 16.72)
 12:Nasturtium officinale psbI   [0007;0031] ttggta-18-tataaa (184, 16.72)

Best quality result #2 [34/54] (power: 12, quality: 188.00, avg proximity: 15.66)
  1:Citrus sinensis psbI         [0007;0031] ttgatg-18-tataaa (165, 15.00)
  2:Aethionema cordifolium psbI  [0007;0030] ttggta-17-tttggt (142, 12.90)
  3:Aethionema grandiflorum psbI [0007;0030] ttggta-17-tttggt (142, 12.90)
  4:Arabidopsis thaliana psbI    [0007;0031] ttggta-18-tataaa (183, 16.63)
  5:Arabis hirsuta psbI          [0007;0031] ttggta-18-tataaa (183, 16.63)
  6:Barbarea verna psbI          [0007;0031] ttggta-18-tataaa (183, 16.63)
  7:Capsella bursa-pastoris psbI [0007;0031] ttggta-18-tataaa (183, 16.63)
  8:Crucihimalaya wallichii psbI [0007;0030] ttgtat-17-tataaa (155, 14.09)
  9:Draba nemorosa psbI          [0007;0031] ttggta-18-tataaa (183, 16.63)
 10:Lepidium virginicum psbI     [0007;0031] ttggta-18-tataaa (183, 16.63)
 11:Lobularia maritima psbI      [0007;0031] ttggta-18-tataaa (183, 16.63)
 12:Nasturtium officinale psbI   [0007;0031] ttggta-18-tataaa (183, 16.63)
```

Рис. 2. Пример содержимого выходного файла в режиме двухбоксового поиска.

6.3 Информация, выдаваемая на консоль

Протокол, выдаваемый на консоль оператора, содержит сообщения, формируемые по ходу исполнения программы. Пример протокола, выдаваемого по умолчанию, приведен на рис. 3.

В зависимости от выбранного (параметром `-c` программы) режима выдачи диагностической информации на консоль, часть сообщений может не выдаваться, либо могут появляться дополнительные сообщения, промежуточные решения, значения параметров, пояснения по ходу процесса оптимизации и т.д. Ниже приводится описание основных сообщений программы TwoBox.

```
(C) 2009, Laboratory of Mathematical Methods and Models in Bioinformatics
Institute for Information Transmission Problems, Russian Academy of Sciences
Program for specific Two Box Search (multiple CPU, v.3.17)  L.Rubanov, 2009

Number of parallel tasks: 1+4
Job accepted (n=7  m<=200  l=12)
6 s) Signal from [T3] for P2(2) (pow=7  q=55.66  avprx=7.95  new=7)
6 s) Signal from [T1] for P0(0) (pow=7  q=56.00  avprx=8.00  new=3)
6 s) Signal from [T2] for P1(1) (pow=7  q=57.00  avprx=8.14  new=2)
6 s) Signal from [T4] for P3(3) (pow=7  q=56.00  avprx=8.00  new=2)
12 s) Signal from [T1] for Q0,0(5) (pow=7  q=57.33  avprx=8.19  new=5)
12 s) Signal from [T2] for Q1,0(6) (pow=7  q=54.66  avprx=7.80  new=2)
12 s) Signal from [T3] for Q2,0(4) (pow=7  q=57.66  avprx=8.23  new=7)
12 s) Signal from [T4] for Q3,0(7) (pow=7  q=56.00  avprx=8.00  new=3)
18 s) Signal from [T2] for Q1,1(9) (pow=7  q=57.00  avprx=8.14  new=0)
18 s) Signal from [T1] for Q0,1(8) (pow=7  q=57.00  avprx=8.14  new=2)
18 s) Signal from [T4] for Q3,1(11) (pow=7  q=55.66  avprx=7.95  new=1)
18 s) Signal from [T3] for Q2,1(10) (pow=7  q=56.00  avprx=8.00  new=6)
24 s) Signal from [T2] for Q1,2(12) (pow=7  q=54.00  avprx=7.71  new=1)
24 s) Signal from [T1] for Q0,2(13) (pow=7  q=55.33  avprx=7.90  new=1)
24 s) Signal from [T4] for Q3,2(14) (pow=7  q=56.00  avprx=8.00  new=0)
25 s) Signal from [T3] for Q2,2(15) (pow=7  q=58.66  avprx=8.38  new=2)
30 s) Signal from [T2] for P4(16) (pow=7  q=59.00  avprx=8.42  new=7)
31 s) Signal from [T4] for P5(18) (pow=7  q=54.66  avprx=7.80  new=1)
31 s) Signal from [T1] for Q0,3(17) (pow=7  q=57.33  avprx=8.19  new=0)
32 s) Signal from [T3] for Q2,3(19) (pow=7  q=58.66  avprx=8.38  new=0)
36 s) Signal from [T4] for Q5,0(21) (pow=7  q=56.33  avprx=8.04  new=0)
37 s) Signal from [T2] for Q4,0(20) (pow=7  q=59.33  avprx=8.47  new=1)
37 s) Signal from [T1] for Q0,4(22) (pow=7  q=53.66  avprx=7.66  new=7)
38 s) Signal from [T3] for Q2,4(23) (pow=7  q=54.66  avprx=7.80  new=5)
43 s) Signal from [T4] for P6(24) (pow=7  q=59.00  avprx=8.42  new=1)
43 s) Signal from [T2] for Q4,1(25) (pow=7  q=57.00  avprx=8.14  new=2)
44 s) Signal from [T1] for Q0,5(26) (pow=7  q=57.33  avprx=8.19  new=0)
44 s) Signal from [T3] for Q2,5(27) (pow=7  q=59.33  avprx=8.47  new=1)
47 s) Signal from [T4] for Q6,0(28) (pow=7  q=59.33  avprx=8.47  new=2)
47 s) Signal from [T2] for Q4,2(29) (pow=7  q=55.66  avprx=7.95  new=2)
Terminated due to P-list end
Best quality result #1 [20/30] (power: 7, quality: 59.33, avg proximity: 8.47)
Best quality result #2 [16/30] (power: 7, quality: 59.00, avg proximity: 8.42)
Best quality result #3 [24/30] (power: 7, quality: 59.00, avg proximity: 8.42)
RC: OK  (time: 47 sec)

Global RC: OK  (time: 47 sec)

Thank you for using the program.
```

Рис. 3. Пример консольного протокола программы TwoBox.

Number of parallel tasks: 1+kk

Сообщение информирует, что программа выполняется параллельно на kk+1 процессоре кластера (корневая ветвь и kk параллельных вторичных ветвей).

Job accepted (n=xx m<=yy l=zz)

Сообщение-заголовок; выдается после успешного чтения очередной порции исходных данных из входного файла. Здесь xx – число последовательностей в наборе исходных данных; yy – максимальная длина последовательности; zz – длина отыскиваемых сайтов (в двухбоксовом случае – суммарная длина обоих боксов).

ttt s) Signal from [Tkk] for Pii(nn) (pow=xx q=yy avprx=zz new=mm)

Это сообщение выдается после успешного выполнения итерации алгоритма, состоящей в

сборке расстановки из *P*-списка. Здесь ttt – число секунд с начала обработки текущей порции исходных данных; kk – номер параллельной ветви, в которой был найден сигнал; ii – порядковый номер расстановки в *P*-списке; nn – порядковый номер текущей итерации алгоритма, начиная с 0; xx – мощность найденного сигнала; yy – качество сигнала; zz – величина среднего сходства сайтов сигнала; mm – число найденных впервые сайтов.

ttt s) Signal from [Tkk] for Qii,jj(nn) (pow=xx q=yy avprx=zz new=mm)

Это сообщение выдается после успешного выполнения итерации алгоритма, состоящей в сборке расстановки из *Q*-списка. Здесь ttt – число секунд с начала обработки текущей порции исходных данных; kk – номер параллельной ветви, в которой был найден сигнал; ii –- номер *Q*-списка; jj – номер расстановки в *Q*-списке; nn – порядковый номер текущей итерации алгоритма, начиная с 0; xx – мощность найденного сигнала; yy – качество сигнала; zz – величина среднего сходства сайтов сигнала; mm – число найденных впервые сайтов.

Terminated due to P-list end

Остановка алгоритма ввиду завершения обработки всех *Q*-списков и исчерпания *P*-списка.

Terminated due to iteration limit

Остановка алгоритма, так как выполнено заданное максимальное число итераций.

Terminated due to time limit

Остановка алгоритма, так как исчерпано отведенное для решения время.

Terminated due to unknown reason

Остановка алгоритма по прочим причинам, на которые указывает код возврата или другие сообщения программы.

Best quality result #kk [mm/nn] (power: xx, quality: yy, avg proximity: zz)

Общая характеристика одного из окончательно выбранных лучших решений: kk – номер решения в порядке убывания качества сигнала; mm – порядковый номер итерации алгоритма, на которой было найдено это решение; nn – общее число проделанных итераций; xx – мощность найденного сигнала; yy – качество сигнала; zz – величина среднего сходства сайтов сигнала.

Best avg proximity result #kk [mm/nn] (power: xx, quality: yy, avg proximity: zz)

Общая характеристика одного из окончательно выбранных лучших решений: kk – номер решения в порядке убывания среднего сходства сайтов сигнала; mm – порядковый номер итерации алгоритма, на которой было найдено это решение; nn – общее число проделанных итераций; xx – мощность найденного сигнала; yy – качество сигнала; zz – величина среднего сходства сайтов сигнала.

RC: xx (time: yy sec)

Сообщение о результатах обработки очередного набора последовательностей. Здесь xx – символическое обозначение кода возврата (см. табл. 1), yy – время обработки в секундах.

Global RC: xx (time: yy sec)

Сообщение об окончательных результатах обработки всех порций исходных данных. Здесь xx – символическое обозначение кода возврата (см. табл. 1), yy – общее время работы программы в секундах.

P-list consists of xx permutations

Построен *P*-список из xx расстановок на основании критерия 2.

Generate xx permutation(s) for Qyy list

Построена очередь из xx расстановок для *Q*-списка с номером yy на основании критерия 1.

ttt s) Send to task [Tkk] permutation Pii(nn): x1 x2 ... xn

Сообщение выдается при передаче очередной расстановки из *P*-списка на обработку в свободную параллельную ветвь. Здесь ttt – число секунд с начала обработки текущей порции

исходных данных; kk – номер параллельной ветви, в которую передается расстановка; ii – порядковый номер расстановки в *P*-списке; nn – порядковый номер текущей итерации алгоритма, начиная с 0; x1 x2 ... xn - переданная расстановка номеров последовательностей.

ttt s) Send to task [Tkk] permutation Qii,jj(nn): x1 x2 ... xn

Сообщение выдается при передаче очередной расстановки из *Q*-списка на обработку в свободную параллельную ветвь. Здесь ttt – число секунд с начала обработки текущей порции исходных данных; kk – номер параллельной ветви, в которую передается расстановка; ii – номер *Q*-списка; jj – порядковый номер расстановки в *Q*-списке; nn – порядковый номер текущей итерации алгоритма, начиная с 0; x1 x2 ... xn – переданная расстановка номеров последовательностей.

Stop P-list processing; task [Tkk] free

Сообщение о завершении обработки всего *P*-списка, сформированного на основании критерия 2. Параллельная ветвь с номером kk программы становится свободной.

Stop Qxx list (length nn); task [Tkk] free

Сообщение о закрытии *Q*-списка с номером xx после завершения обработки nn его расстановок, поскольку выполнены условия критерия 1, в результате чего параллельная ветвь с номером kk программы становится свободной.

Recurrent permutations rejected: xx

В процессе работы алгоритма были построены повторно (и потому отброшены) xx ранее обработанных расстановок.

The algorithm needs at least 2 tasks!

Для работы программы выделено менее двух процессоров, что не позволяет продолжать работу.

Error xx in a secondary task

При работе одной из параллельных ветвей возникла ошибка с кодом xx (коды ошибок приведены в табл. 1).

Error xx creating result

При выборе лучшего из найденных результатов возникла ошибка с кодом xx (коды ошибок приведены в табл. 1).

Invalid argument "xxx"

Ошибочный аргумент xxx указан в командной строке.

[Tkk]: Cannot open input data file "xxx"

Ошибка в параллельной ветви с номером kk при открытии файла исходных данных xxx (файл отсутствует или заблокирован операционной системой).

[Tkk]: Cannot open output data file "xxx"

Ошибка в параллельной ветви с номером kk при открытии или создании выходного файла xxx для записи результатов работы программы.

Cannot open history file "xxx"

Ошибка при открытии файла последовательности решений с именем xxx.

6.4 Файл последовательности решений

В отличие от выходного файла, в который записывается только заданное аргументом `-r` число лучших найденных сигналов, этот текстовый файл содержит всю последовательность найденных решений в порядке их получения. Файл выдается при запуске программы с параметром `-f` и может быть полезен при исследовании поведения алгоритма. Для удобства анализа файл приспособлен для импорта в электронную таблицу (например, Excel), для чего

он формируется в формате CSV (comma-separated values), где в каждой строке приведены данные по одному сигналу, значения символьных полей указываются в кавычках, и разделителем полей служит символ табуляции. Порядок и содержание полей файла приведены в таблице 2. Первая строка файла есть «шапка», которая мнемонически указывает содержание каждого поля; заголовки полей шапки приведены в кавычках в первой колонке.

Таблица 2.

Тип поля, заголовок	**Содержание**	**Примечание**
Число, "#"	Порядковый номер итерации алгоритма	Итерации нумеруются в порядке генерации расстановок, поэтому при расчетах на кластере хронологический файл по этому полю не упорядочен
Число, "P"	Номер расстановки из *P*-списка или номер *Q*-списка для расстановки из *Q*-списка, в зависимости от следующего поля	Нумерация начинается с 0
Число, "Q"	Номер расстановки в *Q*-списке с номером из предыдущего поля, либо -1 (указывает, что в предыдущем поле номер из *P*-списка)	Нумерация начинается с 0
Число, "Pwr"	Мощность сигнала (число последовательностей, из которых выбраны сайты)	
Число, "New"	Число сайтов, впервые появившихся именно в этом сигнале	
Число, "Qt"	Качество сигнала согласно формуле (5), (7) или (8)	
Число, "QtAvg"	Среднее сходство сайтов сигнала (иначе, системы) согласно формуле (6)	
Число, "W1"	Позиция начальной буквы сайта в первой последовательности (начиная с 1). В двухбоксовом режиме в этом поле выдаются два числа через запятую, указывающие позиции начальной буквы двух боксов	Эта четверка полей повторяется *n* раз (*n* – число последовательностей в исходном наборе). Индекс 1 в заголовках шапки соответственно изменяется до *n* включительно.
Строка, имя посл-ти	Сайт (в качестве заголовка в шапке стоит имя последовательности, откуда он взят). В двухбоксовом режиме выдаются два бокса с промежутком в формате, как на рис. 2	
Число, "Qw1"	Качество данного сайта в этом сигнале согласно формулам (1)–(4)	
Число, "QwA1"	Среднее сходство этого сайта с другими сайтами системы	
Число, "p1"	Значение от 1 до *n*	Эти *n* полей представляют расстановку номеров исходных последовательностей, для которой был найден данный сигнал.
...		
Число, "p*n*"	Значение от 1 до *n*	

Таким образом, каждая строка содержит в точности $5n+7$ полей, где n – число последовательностей в исходном наборе.

7. Комплект поставки программы

Список файлов, содержащихся в комплекте поставки, с их кратким описанием приведен в табл. 3.

Таблица 3

Имя файла	Описание
twoboxXXX_YY.pdf	Настоящий документ на одном или более языках (XXX – номер версии программы, YY – обозначение языка)
twobox.exe	Исполняемый модуль программы TwoBox для платформы x86
twobox.exe.manifest	Системное описание исполняемого модуля программы TwoBox, сопровождающее сам модуль
vcredist_x86.exe	Установочный файл свободно распространяемых динамических библиотек Microsoft Visual Studio 2005 Service Pack 1. (Если на ПК не установлен этот продукт, данный файл необходимо однократно выполнить.)
*.txt	Примеры входных файлов для программы TwoBox (в файлах с именами *_MA.txt исходные последовательности для наглядности представлены в форме множественного выравнивания)
test.bat	Файл сценария для запуска программы TwoBox на примерах
*.out	Выходные файлы, полученные в результате работы программы TwoBox на примерах

На странице загрузки дистрибутива программы могут также присутствовать дополнительные файлы:

mpich2_XXX_x86.zip — Для запуска программы TwoBox на компьютере без установки программного продукта MPICH2 можно использовать содержимое этого файла (подробнее см. раздел 8), однако работоспособность и полная функциональность программы в таком режиме не гарантируется.

source.zip — При наличии архива с указанным именем, он содержит исходные модули программы и конфигурационные файлы для ее компиляции в среде Microsoft Visual Studio 2005 Service Pack 1.

8. Установка и запуск программы

1. На каждом ПК, где предполагается работа с программой TwoBox, создать папку для использования программы (например, d:\twobox\) и скопировать в нее все файлы из комплекта поставки.
2. Если на данном ПК не установлен продукт Microsoft Visual Studio 2005 Service Pack 1, запустить файл vcredist_x86.exe и ответить на задаваемые вопросы программы.
3. Установить программный пакет MPICH2 версии 1.2 (или более поздней), если это не сделано ранее. В качестве альтернативы допускается скопировать в папку с программой содержимое папки с именем mpich2_XXX_x86 из комплекта поставки, если такая имеется. После этого выполнить из этой папки команду `smpd -install`.
4. Запустить командный процессор Windows и перейти в папку программы. Для удобства выполнения этой операции рекомендуется создать на рабочем столе ярлык для запуска командного процессора, в свойствах которого указать соответствующую рабочую папку.

5. Запустить команду `mpiexec -n 2 twobox -h`. В окно командного процессора должен быть выдан текст подсказки о параметрах программы. В зависимости от настроек безопасности Windows и запущенных межсетевых экранов (брандмауэров), могут выдаваться запросы на подтверждение выполнения программы и разрешения ее доступа к сети. Рекомендуется добавить программу TwoBox в список исключений брандмауэра, чтобы в дальнейшем избежать необходимости отвечать на подобные запросы.
6. Запустить файл сценария test.bat, чтобы проверить работу программы на данном ПК в режиме 4 процессоров (даже если ПК имеет меньшее число процессорных ядер, необходимая эмуляция выполнится автоматически). Поскольку при этом имеющиеся эталонные файлы результатов будут перезаписаны, рекомендуется скопировать их в другое место. В зависимости от быстродействия ПК и числа физических процессоров, обработка всего сценария может занять порядка 30–40 мин. и более.
7. По окончании обработки сопоставить полученные файлы результатов (*.out) с эталонными копиями. Точного совпадения может не быть в силу различного быстродействия и случайных факторов, но результаты должны быть сопоставимы.
8. Для реальных расчетов необходимо установить желаемую конфигурацию кластера, руководствуясь документацией пакета MPICH2.

9. Рекомендации по эффективному применению

Необходимость в этом разделе диктуется большой вычислительной сложностью решаемой задачи. Напомним, что даже в однобоксовом случае вычислительная сложность полного перебора составляет порядка $(m\text{-}l)^n$. В двухбоксовом случае вычислительная сложность полного перебора составляет порядка $(m\text{-}l'\text{-}l''\text{-}d_{\max})^n \cdot (d_{\max}–d_{\min})^n$. Это делает невозможным решение задачи полным перебором для биологически интересных размерностей.

Используемый квазиоптимальный алгоритм имеет полиномиальную сложность, время счета в среднем растет пропорционально кубу m и квадрату n в однобоксовом режиме. Трудоемкость поиска в двухбоксовом режиме эквивалентна увеличению m в $(d_{\max}–d_{\min})$ раз, что дает вычислительную сложность $m^3n^2(d_{\max}–d_{\min})^3$. Ограничения по памяти менее критичны, поскольку при ее нехватке имеется возможность не хранить в памяти наиболее объемную структуру – матрицу сходства сайтов (объем которой приблизительно m^2n^2 байт в однобоксовом режиме и $m^2n^2(d_{\max}–d_{\min})^2$ байт в двухбоксовом), а с помощью параметра `-x0` переходить к вычислению сходства сайтов «на ходу», т.е. произвести обмен памяти на быстродействие.

Тем не менее, в текущей версии программы установлены следующие формальные ограничения для размерностей решаемой задачи:

- число последовательностей в исходном наборе $2 \le n \le 256$;
- длина последовательности $m < 4096$;
- длина отыскиваемых сайтов $l < 32$ ($l' + l'' < 32$).

Эти ограничения носят технический характер и при необходимости могут быть легко изменены.

Для интервала возможных расстояний ограничений не установлено, однако рекомендуется использовать как можно более узкий интервал, поскольку трудоемкость расчетов растет пропорционально кубу ширины интервала по сравнению с интервалом ширины 1, т.е. фиксированным промежутком между боксами.

Поскольку при выполнении программы достигается 100% загрузка процессора, число параллельных ветвей, выполняемых на одном ПК, рекомендуется указывать в соответствии с количеством АЛУ, имеющихся в процессоре: по числу ядер – для многоядерных процессоров, 2 – для процессоров Pentium 4, 1 – для более ранних моделей процессоров.

(Следует иметь в виду, что в команде mpiexec при запуске указывается на единицу большее число процессоров, с учетом корневой ветви.) Указание большего числа ветвей не возбраняется, но обычно не дает выигрыша, а при значительном превышении – приводит к проигрышу во времени. При этом также необходимо учитывать имеющийся объем памяти ПК, чтобы каждая параллельная ветвь располагала необходимым количеством оперативной памяти, не прибегая к динамической подкачке.

10. Список литературы

1. Данилова Л.В., Горбунов К.Ю., Гельфанд М.С., Любецкий В.А. Алгоритм выделения регуляторных сигналов в последовательностях ДНК (2) // *Молекулярная биология*, 2001, том 35, № 6, стр. 987-995.
2. С.Н. Истомина, Л.И. Рубанов. Параллельный алгоритм поиска регуляторного сигнала в геномах бактерий // *Информационные процессы*, 2002, т. 2, № 1, с. 85-90. http://www.jip.ru/2002/Isto.pdf
3. L.V. Danilova, V.A. Lyubetsky, M.S. Gelfand. An algorithm for identification of regulatory signals in unaligned DNA sequences, its testing and parallel implementation // *In Silico Biology*, 2003, V. 3, No 1,2, 2003, p. 33-47. http://www.bioinfo.de/isb/2003/03/0004/
4. MPI: A Message-Passing Interface Standard. *Message Passing Interface Forum*, Version 1.1: June 1995. Knoxville, University of Tennessee, 1995.
5. MPI-2: Extensions to the Message-Passing Interface. *Message Passing Interface Forum*, July 18, 1997. Knoxville, University of Tennessee, 1997.