Описание программы «Protfile»

1. Аннотация

Настоящий документ содержит описание программной компоненты Protfile и предназначен для пользователей и программистов. Описывается функциональное назначение программы, порядок ее применения и имеющиеся ограничения. Изложена логика работы, описаны составные части – программы Protfile и ScoreMax – и схема их взаимодействия. Описан порядок запуска программ и параметры, передаваемые посредством файла настроек и интерфейса командной строки. Представлены форматы входных и выходных данных, а также сообщения, выдаваемые на консоль оператора.

2. Содержание

3. Основной раздел

3.1. Вводная часть

3.1.1. полное наименование программы

«Программа поиска белка с филогенетическим профилем, наиболее соответствующим профилю пары заданных списков геномов».

3.1.2. обозначение

Protfile

3.1.3. применение программы

Для использования программы требуется операционная система, обеспечивающая компиляцию программы на языке C++ и интерпретацию кода на языке PHP5 командно-строчным (CLI) интерпретатором языка PHP, доступным для многих платформ. Программа исполняется на любой современной операционной системе семейства UNIX/Linux или Windows. Учитывая возможную продолжительность счета, рекомендуется запускать программу на постоянно работающем сервере в качестве фоновой задачи.

На входе программа использует электронное представление геномов, принятое в глобально доступном банке генетических данных «GeneBank» института NCBI; требуемую структуру данных также несложно воспроизвести локально. Дополнительно на входе используется небольшой текстовый файл плюс-минус списка, который строится с помощью текстового редактора или используя специальный режим запуска программы. Результатом работы программы является текстовый файл со списком белков, наиболее соответствующих заданному профилю, с указанием числовой оценки степени соответствия.

3.2. Функциональное назначение

3.2.1. назначение программы

Программа предназначена для поиска белков по заданному филогенетическому профилю, который определяется по двум заданным спискам геномов. Алгоритм определяет белок, наиболее соответствующий этим спискам, т.е. белок, гомологи которого присутствуют во всех геномах из первого списка («*плюс-список*»), но в тоже время лучший гомолог в каждом геноме из второго списка («*минус-список*») имеет меньшее сходство с данным белком, чем лучший гомолог из любого генома, принадлежащего первому списку. Точнее, алгоритм ищет несколько лучших («субоптимальных») белков, удовлетворяющих этому условию. На прак-

тике так могут искаться регуляторные белки по их потенциальным сайтам связывания с ДНК или РНК, когда плюс-список состоит из геномов, содержащих хотя бы один регуляторный сайт рассматриваемого типа, а минус-список состоит из геномов, не содержащих такого сайта. Другим примером является применение этого алгоритма для поиска белков, кодирующих характерные признаки организма (наличие/отсутствие жгутика или фотосистем и т.д.).

3.2.2. общее описание функционирования программы

Алгоритм состоит в специально организованном последовательном просмотре двух входных списков («*плюс*» и «*минус*» списки) геномов с вычислением минимакса. Для этого из плюс-списка выбирается геном, называемый «*первым*», в котором будет производиться поиск целевого белка; этот выбор осуществляется пользователем, причём возможен выбор нескольких геномов в качестве «первых», в этом случае вся нижеописанная процедура однотипно повторяется для каждого из них.

Для каждого генома («текущий геном») из плюс-списка, кроме «первого», и каждого белка («текущий белок») из «первого» генома выполняется следующая процедура. В текущем геноме определяется самый близкий белок к текущему белку. (За меру близости принят индекс FASTA SCORE, вычисление которого описано ниже в этом разделе.) Каждому текущему белку сопоставляется минимакс FASTA SCORE по плюс-списку, где максимум берется по белкам текущего генома, и минимум – по геномам из плюс-списка. Затем для каждого текущего генома из минус-списка и каждого текущего белка из «первого» генома ищется белок из текущего генома, значение меры близости которого с текущим белком превышает минимакс. Если найден такой белок, алгоритм переходит к следующему текущему белку, в противном случае текущий белок с соответствующими ему максимумами качеств относительно всех геномов из плюс и минус списков заносится в итоговый список. Эти максимумы и составляют филогенетический профиль каждого белка из «первого» генома. Из итогового списка отбирается несколько белков с максимальной оценкой сходства их профиля с заданным; список этих белков вместе с упомянутой оценкой и являются результатом работы программы.

Филогенетический профиль белка – функция, сопоставляющая каждому геному из плюс и минус списков (за исключением «первого») численную оценку близости (FASTA SCORE) белка с ближайшим гомологом в геноме. Образцовый филогенетический профиль – функция, принимающая два значения: -1 на геномах из плюс-списка и 1 на геномах из минус-списка. За меру сходства профилей, понимаемых как векторы, принят косинус $\cos\varphi$ угла φ между векторами профилей.

Для численной оценки меры сходства двух белков используется индекс FASTA SCORE, вычисляемый отдельной программой ScoreMax как минимум суммы «штрафов» и «премий» по матрице BLOSUM62 (представленную, например, на странице http://www.uky.edu/Classes/BIO/520/BIO520WWW/blosum62.htm) пар букв в соответственных позициях, который находится по всем вариантам взаимного положения последовательностей.

3.2.3. сведения об ограничениях на применение

Основным фактором, ограничивающим возможность применения программы, является время счёта, растущее как произведение числа «первых» геномов, общего числа геномов, квадрата (среднего) числа белков в геноме и квадрата их (средней) длины. В связи с тем, что время счёта может составлять часы и даже сутки, в программе предусмотрена возможность продолжать вычисления, прерванные из-за сбоя или по инициативе пользователя. Требования к памяти не существенны, т.к. в каждый момент времени в ней хранится лишь пара геномов (объём порядка единиц мегабайт) и статистика максимумов близости белков (несколько байт на пару белок-геном, где белок принадлежит одному из «первых» геномов, а геном пробегает по всем используемым геномам, что составляет на типичных данных единицы мегабайт).

3.3. Описание логики программы

3.3.1. Описание структуры программы и её основных частей

Программная компонента реализована в виде основной программы protfile.php, написанной на интерпретируемом языке высокого уровня PHP5, и вспомогательной программы scoremax, написанной на языке C++, исполняемый модуль которой вызывается программой protfile.php. Обмен информацией между программами осуществляется через файлы. Общий порядок работы таков: интерпретатор php языка PHP5 интерпретирует программу (сценарий) protfile.php; в ходе интерпретации происходит обработка пользовательских параметров, чтение входных файлов, подготовка входных данных для программы scoremax, многократный её запуск с ожиданием завершения, чтение выходных данных программы scoremax, формирование окончательного результата и вывод его в файл результата. Таким образом, пользователь взаимодействует лишь с программой protfile.php, а вся работа с программой scoremax производится полностью автоматически.

3.3.2. Описание функций составных частей и связей между ними

Программа protfile.php реализует основную логику алгоритма, описанную в разделе 3.2.2 выше, а также осуществляет ввод-вывод пользовательских и рабочих данных и интерфейс пользователя (обработка файла настроек, вывод сообщений на консоль оператора и в файл протокола). В ходе своей работы программа protfile.php может использовать программу scoremax для выполнения ресурсоёмких вычислений или (в некоторых случаях) использовать ранее полученные результаты её работы.

Программа scoremax осуществляет подсчёт значений оценки FASTA SCORE для всех пар белков из заданной пары геномов, нахождение для каждого белка из первого генома ближайшего гомолога из второго генома и вывод во временный файл информации о найденных парах гомологов. Эта программа написана в расчёте на работу под управлением программы protfile.php и не рассчитана на запуск пользователем.

Взаимодействие между программами protfile.php и scoremax осуществляется посредством обмена входными/выходными файлами. Программа protfile.php формирует по исходным данным файлы белковых последовательностей, каждый из которых содержит белковые последовательности одного генома (см. приложение 2) и осуществляет запуск программы scoremax для всех пар геномов, один из которых отмечен пользователем в плюс-минус списке как «первый», а другой принадлежит плюс или минус списку. В качестве параметров командной строки программе scoremax передаются имена этих файлов и параметр точности вычисления SCORE, заданный пользователем в файле настроек. Программа scoremax читает файлы белковых последовательностей и формирует файлы максимумов SCORE, содержащие номера пар гомологичных белков и значения FASTA SCORE для этих пар (см. приложение 2). Программа protfile.php производит чтение созданных файлов максимумов SCORE и использует совокупность хранимых в них данных для формирования окончательного результата: списка белков, с профилем, наиболее соответствующим заданному (см. приложение 2).

3.3.3. Сведения о языке программирования

Функции, критичные к быстродействию (перебор пар белков с подсчётом максимумов FASTA SCORE), запрограммированы на языке C++ с соблюдением стандарта ANSI C++ и спецификации ISO/IEC от 1998 г. Тестовая сборка данной подпрограммы производилась компилятором gcc (g++) в среде операционной системы Linux и портами gcc на платформу MS Windows (в средах разработки DJGPP и CygWin). Основная логика программы реализована на языке высокого уровня PHP5 без использования каких-либо его функциональных расширений.

3.3.4. Описание входных и выходных данных

Входными данными программы Protfile являются:

- набор геномов, входящих в плюс и минус списки, организованных в структуру каталогов, аналогично формату представления данных на FTP NCBI в каталоге ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria/;
- текстовый файл плюс-минус списка, указывающий принадлежность геномов тому или иному списку;
- необязательный текстовый файл конфигурации с дополнительными параметрами настройки программы (если файл конфигурации используется, его имя передается в виде параметра при запуске программы Protfile).

Формат входных данных приведен в приложении 1.

Выходными данными программы Protfile являются:

- файл результатов программы, содержащий для каждого «первого» генома список отобранных белков с указанием для каждого из них меры соответствия этого белка заданному плюс-минус списку;
- файл протокола, детально описывающий ход исполнения программы;
- сообщения о ходе работы программы и нештатных ситуациях, выдаваемые на консоль оператора.

Помимо этого, программа Protfile формирует ряд рабочих файлов, используемых для обмена информацией с программой ScoreMax и для возобновления работы программы при ее повторном запуске после прерывания:

- файл списка организмов, где фиксируется набор организмов с их внутренними кодами;
- файлы заголовков белков, содержащие копии информационных заголовков белков из входных faa-файлов одного генома;
- файлы белковых последовательностей, содержащие копии последовательностей из входных faa-файлов одного генома;
- файлы максимумов SCORE с максимальными значениями величины FASTA SCORE для пары геномов.

Детальное описание форматов выходных и рабочих данных программы приведено в приложении 2.

Обмен информацией с вызываемой программой ScoreMax производится через файлы. В качестве входной информации программе ScoreMax в командной строке передаются три параметра: имена двух файлов белковых последовательностей геномов, которые в момент обращения играют роль «первого» и «текущего», и максимально допустимая разность длин по-

следовательностей, при которой они считаются сравнимыми. Результат сравнения геномов (значения максимумов FASTA SCORE и пары номеров белков, на которых они достигаются) передается вызывающей программе через файл максимумов SCORE (в который перенаправляется стандартный вывод программы ScoreMax).

3.3.5. Описание логики составных частей

3.3.5.1. Программа protfile.php

Программа protfile.php написана на языке сценариев PHP и реализована в виде одного файла, содержащего основной текст программы и несколько подпрограмм, выполненных в виде функций языка PHP. Основной текст программы можно разделить на два этапа (здесь и ниже в круглых скобках после знака ':' следуют номера первой и последней строки текста описываемого фрагмента программы):

– ввод и предобработка данных (:4-189);
– анализ профилей и вывод результатов (:191-280).

В основном тексте программы используются следующие функции описанные в тексте программы (:284-593) примерно в том порядке, в котором вызываются и используются:

– load_config( )
– config_list( )
– organisms_list( )
– scan_directory_for_genomes( )
– organisms_codenames( )
– write_organisms_list( )
– prepare_genome( )
– read_scores( )
– load_headers( )
– minimax( )
– maximax( )
– profile_quality( )

Ниже в данном разделе подробно описан каждый из этапов основной программы и каждая используемая в нём функция.

Этап ввода и предобработки данных обеспечивает анализ пользовательских параметров, проверку наличия всех требуемых данных и приведение их к виду, удобному для использования (в том числе программой scoremax) и состоит из следующих основных частей:

– определение параметров конфигурации по умолчанию (:4-16);
– обработка параметров командной строки и файла конфигурации (:26-40);

– проверка наличия директорий геномов (:42-47);
– чтение плюс-минус списка или создание его шаблона (:49-82);
– проверка корректности плюс-минус списка (:84-99);
– определение возможности использования данных предыдущих запусков (:101-127);
– подготовка рабочего пространства и запись списка организмов (:129-155);
– проверка наличия всех требуемых для вычислений геномов (:157-168);
– подготовка геномов (:170-189).

Выполнение программы начинается с определения глобального ассоциативного массива $config – пользовательских параметров конфигурации (подробно описанных в приложении 1) со значениями параметров, принимаемыми по умолчанию, если пользователем не задан файл конфигурации или некоторые параметры в нем опущены (эти значения также указаны в приложении 1).

Если при запуске программы пользователь задал параметры в командной строке, производится анализ первого параметра: если он равен одной из строк “-h”, “--help” или “/?”, на консоль оператора выдаётся сообщение о воспринимаемом формате командной строки и работа программы завершается; в противном случае параметр считается именем файла конфигурации и производится загрузка последнего с помощью функции load_config( ).

Производится проверка наличия директории геномов с именем, заданным параметром конфигурации data_dir. В случае отсутствия директории с заданным именем на консоль оператора выдаётся сообщение об ошибке и работа программы завершается.

Проверяется наличие файла плюс-минус списка с именем, заданным параметром конфигурации list_file_name. В случае его отсутствия производится сканирование директории геномов с помощью функции scan_directory_for_genomes, и если созданный ей список директорий не пуст, на его основе создаётся шаблон плюс-минус списка, выводится в файл с вышеуказанным именем, и на консоль оператора выводится сообщение с просьбой заполнить его, в противном случае (список пуст) на консоль оператора выводится сообщение о том, что в директории данных не найдено геномов. В случае наличия файла плюс-минус списка его содержимое читается посредством вызова функции read_plusminus_list( ), заполняющей на его основе глобальные массивы $organisms, $plus_list, $minus_list и $first_list, ссылки на которые передаются ей в качестве параметров; кроме того, посредством вызова функции organisms_codenames( ) заполняется глобальный массив $org_codes сокращённых наименований организмов. Индексированный массив $organisms содержит в качестве значений имена всех геномов (организмов) из файла плюс-минус списка (даже тех, что не помечены в плюс-минус списке знаком), занумерованные в порядке их следования в файле, начиная с нуля (стандартный метод индексации в языке PHP). Индексированные массивы $plus_list,

$minus_list и $first_list содержат в качестве значений номера (индексы в массиве $organisms) организмов, принадлежащих соответствующим спискам (плюс-списку, минус-списку и списку «первых» геномов). Индексированный массив $org_codes содержит для каждого организма из массива $organisms краткое наименование, используемое при генерации имён временных файлов и выводе сообщений на консоль (соответствующие друг другу элементы двух указанных массивов имеют совпадающие индексы).

Производится проверка непустоты массивов $plus_list, $minus_list и $first_list. Если в каком-то из массивов нет элементов, на консоль оператора выводится соответствующее сообщение и работа программы завершается.

Если включён параметр конфигурации resume_work, производится следующая процедура проверки возможности использования данных предыдущих запусков программы. Проверяется наличие рабочей директории (имя которой задаётся параметром конфигурации work_dir), и в случае её отсутствия или пустоты параметр конфигурации resume_work выключается (т.к. использование данных предыдущих запусков невозможно ввиду их отсутствия); в противном случае проверяется наличие файла списка организмов (его имя задаётся параметром конфигурации orgs_file_name) и (посимвольного) его совпадения с результатом вызова функции organisms_list( ), в случае невыполнения какого-либо из условий параметр конфигурации resume_work выключается (использование данных предыдущих запусков считается в этом случае невозможным из-за изменения списка организмов, препятствующего установлению соответствия между рабочими файлами предыдущих запусков и текущим списком организмов).

Если параметр конфигурации resume_work (к этому моменту) выключён, производится следующая процедура подготовки рабочего пространства. Если рабочая папка отсутствует, она создаётся, иначе из неё удаляются все файлы. Если существует файл списка организмов, он удаляется. Посредством вызова функции write_organisms_list( ) создаётся новый файл списка организмов.

Производится проверка наличия всех необходимых для вычислений геномов: для каждого генома (с индексом $index и именем $organism) из плюс и минус списка (напомним, что геномы из списка «первых» принадлежат по определению плюс-списку) проверяется наличие либо пары файлов с именами $org_codes[$index] и расширениями “.hdr” и “.seq” в рабочей папке (обратим внимание, что в случае, если параметр конфигурации resume_work выключен, отсутствие данных файлов гарантируется предварительной очисткой рабочей директории), либо (в случае их отсутствия) директории с именем $organism в папке геномов. При невыполнении обоих условий на консоль оператора выдаётся сообщение об отсутствии директории генома с данным именем, и работа программы завершается.

В заключение этапа ввода и предобработки данных производится следующая процедура подготовка геномов к работе. Для каждого генома из плюс и минус списка, для которых в рабочей директории отсутствует хотя бы один из вышеупомянутых двух файлов (.hdr и .seq), во временный массив $faa_file_names (посредством вызова функции glob( ) языка PHP) помещаются имена всех файлов с расширением ".faa" из директории генома; в случае их отсутствия (массив пуст) на консоль оператора выдаётся соответствующее сообщение об ошибке, и работа программы завершается, в противном случае требуемые два файла создаются посредством вызова функции prepare_genome( ) с передачей ей в качестве параметров массива $faa_file_names и пары имён создаваемых файлов.

Этап анализ профилей и вывода результатов является реализацией алгоритма, изложенного в разделе 3.2.2 выше и представляет собой цикл по всем «первым» геномам (с индексом) $the_first, перед началом которого открывается, а после завершения – закрывается файл результатов программы. В цикле для удобства определяется массив $plus_list_, отличающийся от массива $plus_list отсутствием в нём генома $the_first. Тело цикла можно разделить на следующие три фазы:

– сравнение геномов (:208-245);
– анализ профилей (:247-255);
– вывод результатов (:257-273).

Сравнение геномов заключается в определении для каждого белка из «первого» генома ближайшего гомолога из каждого генома из $plus_list_ и $minus_list, и значения функционала FASTA SCORE для этой пары гомологов. Для этого в цикле для каждого генома (с номером) $the_second из $plus_list_ и $minus_list, производятся следующие действия. Из массива $org_codes извлекаются значения сокращённых наименований организмов $first_code и $second_code для геномов $the_first и $the_second соответственно; имя файла максимумов SCORE – $score_file_name – полагается равным "$first_code-$second_code.scr"; если файл с таким именем существует в рабочем каталоге, производится попытка считать из него (в глобальный массив $profiles) значения максимумов SCORE посредством вызова функции read_scores($score_file_name, $the_first, $the_second); если чтение прошло успешно (функция read_scores( ) вернула ненулевое значение), цикл по $the_second прерывается (необходимые искомые данные получены); в противном случае производится вызов программы scoremax с передачей ей в качестве аргументов командной строки имён файлов белковых последовательностей, соответствующих геномам $the_first и $the_second и параметра конфигурации scoremax_param и перенаправлением его стандартного выходного потока в файл с именем $score_file_name; после этого производится повторная попытка считать файл максимумов SCORE с именем $score_file_name и в случае его отсутствия или ошибки чтения на

консоль оператора выдаётся сообщение об ошибке, и программа завершает работу (если ошибка будет устранена, возможен повторный запуск программы protfile.php с использованием результатов неудачного запуска).

Анализ профилей заключается в вычислении характеристики их качества относительно плюс-минус списка и отсеве профилей, не удовлетворяющих условию на минимакс. С этой целью для каждого элемента (с индексом $prot_index и значением $profile) массива $profiles[$the_first] проверяется условие minimax($profile, $plus_list_) > maximax($profile, $minus_list), и если оно выполняется, для белка вычисляется значение качества его профиля относительно плюс-минус списка посредством вызова функции profile_quality($profile, $plus_list_, $minus_list), которое сохраняется в глобальный массив $candidates с индексами [$the_first][$prot_index]. После этого производится обратная ассоциативная сортировка массива $candidates[$the_first].

Фаза вывода результатов начинается с загрузки из .hdr-файла информационных заголовков белков генома $the_first посредством вызова функции load_headers( ) для соответствующего файла. После этого в файл результатов выводится для каждого отобранного белка качество его профиля и информационный заголовок; вывод прерывается досрочно, если достигнуто число белков, заданное параметром конфигурации max_results.

Ниже описываются вышеупомянутые функции.

Функция load_config( ) загружает в глобальный массив $config параметры конфигурации из файла с указанным именем. Её единственным аргументом является имя файла конфигурации, из которого загружаются параметры; возвращаемое значение не используется; используется глобальный массив $config. В начале функции открывается файл с заданным именем (если попытка его открыть не удалась, на консоль оператора выдаётся сообщение об ошибке и работа программы завершается). Затем последовательно читаются все строки файла и для каждой из них производится следующая процедура:

– в начале и конце строки удаляются все пробельные символы (пробелы, символы табуляции и разделители строк);
– если после этого строка оказалась пустой, она игнорируется;
– если строка удовлетворяет регулярному выражению “^(#|;|//).*$”, она считается комментарием и игнорируется;
– если строка не удовлетворяет регулярному выражению “^([A-Za-z0-9_]+) *= *\"? *(.*[^]) *\"?;?$”, на консоль оператора выдаётся предупреждение о том, что строка игнорируется, и она игнорируется;
– в противном случае часть выражения в первых круглых скобках считается именем параметра конфигурации, а во вторых – его значением;

– если в массиве $config нет ключа, совпадающего с именем параметра, на консоль оператора выдаётся предупреждение о том, что параметр не известен, и строка игнорируется;
– если соответствующий ключ найден, значение параметра приводится к типу соответствующего ключу значения массива $config и последнее заменяется первым (в случае логического типа дополнительно значения “yes” и “no” заменяются на true и false).

После этого файл закрывается и управление возвращается вызывающей процедуре.

Функция config_list( ) формирует список текущих значений параметров конфигурации в текстовой строке. Параметров функция не имеет; возвращается сформированная строка со списком имён и значений параметров конфигурации; используется глобальная переменная $config. В цикле последовательно обрабатывается каждый элемент глобального массива $config, и для каждого из них формируется строка вида “$key = $value\n”, при этом, если параметр имеет строковый тип, его значение заключается в апострофы (‘). Строка, являющаяся конкатенацией всех таких строк, возвращается вызывающей процедуре.

Функция organisms_list( ) формирует текстовый список всех организмов из глобального массива $organisms или его подмножества, заданного массивом индексов. Функция принимает 0 или 1 аргумент; необязательный аргумент является массивом индексов организмов, которые будут включены в создаваемый список, в случае его отсутствия используются все организмы; возвращается текстовый список организмов по одному на строке; используются глобальные переменные $organisms и $org_codes. В цикле для каждого элемента глобального массива $organisms (индекс которого присутствует в переданном функции массиве, если последний задан) формируется строка вида “$org_codes[$index]: $organism\n”. Строка, являющаяся конкатенацией всех полученных таким образом строк, возвращается вызывающей процедуре.

Функция scan_directory_for_genomes( ) сканирует заданную директорию на наличие в ней поддиректорий, содержащих faa-файлы и формирует их список. Её единственным параметром является имя сканируемой директории; в вызывающую программу возвращается индексированный массив имён найденных поддиректорий, содержащих faa-файлы; глобальные переменные не используются. В начале функции с помощью вызова стандартной функции glob( ) создаётся список имён всех файлов (и в том числе директорий) в заданной директории. Затем для каждого из этих имён $org_dir с помощью glob( ) определяется наличие файлов, удовлетворяющих маске “$org_dir/*.faa”. Если такие файлы существуют, $org_dir добавляется в массив, который по окончании цикла возвращается вызывающей процедуре.

Функция read_plusminus_list( ) читает из заданного файла плюс-минус список и заполняет переданные ей по ссылке массивы плюс-списка, минус-списка, списка «первых» геномов, полного списка организмов и их кодов. Принимаются следующие 4 аргумента:

$filename – имя файла плюс-минус списка, $organisms – ссылка на массив для хранения списка всех организмов, $first_list – ссылка на массив для хранения списка «первых» геномов, $plus_list – ссылка на массив для хранения плюс-списка, $minus_list – ссылка на массив для хранения минус-списка; возвращаемое значение отсутствует; глобальные переменные не используются. Работа функции начинается с попытки открытия файла с именем $filename, если она неудачна, на консоль оператора выводится соответствующее сообщение об ошибке и работа программы завершается. Затем каждая непустая строка входного файла проверяется на соответствие регулярному выражению "^ *\[([*+-])\] *(.*) *$", и в случае положительного результата часть строки, соответствующая выражению во вторых круглых скобках считается именем организма (генома), а часть строки, соответствующая выражению в первых скобках – приписанным геному знаком. В массив $organisms добавляется имя организма. Затем, в зависимости от приписанного знака, номер организма (его индекс в массиве $organisms) добавляется в:

'+' – $plus_list;

'–' – $minus_list;

'*' – $plus_list и $first_list.

После этого файл закрывается и управление возвращается вызывающей процедуре.

Функция organisms_codenames( ) создаёт список сокращённых наименований организмов, используемых для генерации имён рабочих файлов и вывода на консоль. Параметры отсутствуют; возвращаемое значение – ссылка на созданный массив; используются глобальные переменные $config и $organisms. Для каждого организма из глобального массива $config в массив $org_codes помещается запись с таким же индексом и значением, равным порядковому номеру организма (начиная с 1), дополненному нулями слева так, чтобы все сокращённые наименования имели равную длину, и предварённый значением параметра конфигурации org_code_prefix. Ссылка на $org_codes возвращается в вызывающую программу.

Функция write_organisms_list( ) выводит в указанный файл список организмов из глобальной переменной $organisms в том же формате, в котором формирует списки функция organisms_list( ). Параметром является имя используемого файла; возвращаемое значение отсутствует; глобальные переменные не используется. Функция начинается с открытия файла с именем, указанным параметром. В случае неудачи на консоль оператора выводится сообщение об ошибке и работа программы завершается. В противном случае в файл выводится список организмов, получаемый посредством вызова функции organisms_list( ) без параметров. Затем файл закрывается и управление возвращается вызывающей процедуре.

Функция prepare_genome( ) проводит преобразование faa-файлов в директории генома в пару файлов: .hdr (файл заголовков последовательностей) и .seq (файл белковых последова-

тельностей). Функция принимает 3 параметра: список faa-файлов и имена двух выходных файлов; возвращаемое значение отсутствует; глобальные переменные не используются. Выполнение функции начинается с последовательного построчного чтения каждого из faa-файлов в локальные массивы $headers и $sequences (контролируется ситуация ошибки открытия файлов, а также чередование заголовков и тел последовательностей в FASTA-файле). Затем сформированные массивы построчно выводятся в соответствующие файлы заголовков и последовательностей (также с контролем успешности файловых операций). После этого управление возвращается вызывающей процедуре.

Функция read_scores( ) производит считывание максимумов SCORE из файла, созданного программой scoremax. В качестве параметров ей передаётся имя файла и два индекса геномов из глобального массива $organisms; возвращается логический признак успешного считывания всех записей; используется глобальная переменная $profiles. В начале работы функции открывается файл с указанным именем (производится контроль успешности этой операции). Затем открытый файл считывается построчно и для каждой его строки производятся следующие действия.

- в начале и конце строки удаляются все пробельные символы (пробелы, символы табуляции и разделители строк);
- если после этого строка оказалась пустой, она игнорируется;
- если строка совпадает с ключевой строкой “[End]”, устанавливается признак успешного считывания данных, и чтение файла прерывается;
- строка проверяется на соответствие регулярному выражению “ ^([0-9]+)\t(-1|[0-9]+)\t(-?[0-9]+)$” и в случае несоответствия строка игнорируется;
- в противном случае строка считается информационной и её части, соответствующие регулярным выражениям в круглых скобках считаются, соответственно: номером белка в первом геноме ($prot_index), номером белка-гомолога во втором геноме ($nearest_prot) и значением SCORE для данной пары гомологичных белков ($scoremax);
- если номер белка в первом геноме не совпадает с порядковым номером информационной записи в файле, на консоль оператора выдаётся сообщение об ошибке и работа программы завершается;
- в случае совпадения номеров в глобальный массив $profiles добавляется запись с тройным индексом [$the_first][$prot_index][$the_second], значение которой полагается равным массиву из двух элементов: $nearest_prot, $scoremax.

После этого файл закрывается и управление возвращается вызывающей процедуре.

Функция load_headers( ) осуществляет загрузку информационных заголовков белков из ранее созданного файла заголовков (.hdr). В качестве единственного аргумента ей передаётся

имя файла; возвращаемым значением является ссылка на созданный массив заголовков; глобальные переменные не используются. В начале работы функции открывается файл с заданным именем, в случае неудачи на консоль оператора выдаётся соответствующее сообщение об ошибке, и работа программы завершается. Затем файл считывается построчно и каждая непустая (с точностью до пробельных символов) его строка добавляется в массив заголовков; при этом, если строка завершается выражением в квадратных скобках, оно удаляется (это сделано для удобства чтения файла результатов, т.к. в исходных данных в конце информационных заголовков в квадратных скобках обычно помещаются названия организмов, одинаковые для всех белков генома). Затем файл закрывается, и ссылка на полученный массив возвращается в вызывающую процедуру.

Функция minimax( ) вычисляет минимум максимумов SCORE, хранимых в профиле белка по заданному списку организмов. Функции передаётся профиль белка (запись об отдельном белке из глобального массива $profiles) и список номеров организмов, по которому вычисляется минимум (по смыслу задачи это должен быть плюс-список без генома того белка, которому принадлежит профиль); возвращается значение минимума; глобальные переменные не используются.

Функция maximax( ) вычисляет максимум максимумов SCORE, хранимых в профиле белка по заданному списку организмов. Функции передаётся профиль белка (запись об отдельном белке из глобального массива $profiles) и список номеров организмов, по которому вычисляется максимум (по смыслу задачи это должен быть минус-список); возвращается значение минимума; глобальные переменные не используются.

Функция profile_quality( ) вычисляет оценку качества профиля белка относительно заданных плюс и минус списка. Функции передаётся профиль белка (запись об отдельном белке из глобального массива $profiles), список номеров организмов, входящих в плюс-список (без генома того белка, которому принадлежит профиль) и список номеров организмов, входящих в минус-список; возвращается численная оценка качества профиля белка относительно профиля, заданного плюс-минус списком, равная косинусу угла между профилем белка и вектором, координатами которого являются +1 и –1 согласно плюс-минус списку (см. раздел 3.2.2 выше); глобальные переменные не используются.

3.3.5.2. Программа scoremax

Программа scoremax написана на языке C++, её текст оформлен в виде одного файла, содержащего головную функцию main( ) и две подпрограммы (функций языка C++): read_genome( ) и fasta_score( ). Ниже подробно описаны предназначение и логика каждой из этих трёх функций.

Функция main( ) реализует основную логику программы scoremax: обработку параметров командной строки, ввод данных, управление процессом попарного сравнения белков, выбор максимумов и вывод результатов. Выполнение функции начинается с проверки числа аргументов командной строки (строка 30, далее обозначается – :30); если их число меньше трёх – на консоль оператора выдаётся сообщение-подсказка о формате командной строки программы scoremax, и работа программы завершается с кодом возврата 1. Значения трёх первых аргументов командной строки сохраняются (:42-44) в локальные переменные строкового типа *infile1* и *infile2* и глобальную целую переменную *max_len_dif* соответственно; первые две служат именами двух входных файлов, а последняя – регулятором качества/быстродействия подсчёта FASTA SCORE. Производится проверка корректности значения параметра *max_len_dif* (:45-48); если значение параметра выходит за допустимые границы 0-100 – оно полагается равной ближайшей границе этого диапазона. Затем производится загрузка двух геномов из файлов с указанными именами в локальные векторы строк *genome1* и *genome2* соответственно (:50-57); ввод производится посредством вызова функции read_genome( ), и в случае, если она возвращает нулевое значение, работа программы завершается с кодом возврата 1. Для большей наглядности текста число последовательностей в прочитанных геномах сохраняется в локальных константах *numseqs1* и *numseqs2* (:59-60). Далее выполняется основной рабочий цикл программы по белкам генома *genome1* (:64-76). В его теле выполняется вложенный цикл по белкам генома *genome2* (:68-74), в котором посредством вызова функции fasta_score( ) производится вычисление значения FASTA SCORE для пары текущих белков с одновременным вычислением его максимума (*max_score*) для данного (*i*-го) белка из *genome1* и сохранением номера белка (*closest*) из *genome2*, на котором был достигнут этот максимум (если максимум достигается на нескольких белках из *genome2* – *closest* полагается равным номеру первого из них). В конце каждого прохода основного цикла производится запись одной строки в стандартный выходной поток программы (:75), содержащей три числа, разделённые символом табуляции: *i*, *closest* и *max_score* (обе нумерации начинаются с нуля), строки завершаются символом перевода строки, специфическим для операционной системы, в которой производится компиляция программы. Возможен случай, когда (из-за сокращённой процедуры подсчёта FASTA SCORE) не удаётся найти ни одного белка, достаточно близкого к данному (например параметр *max_len_dif* установлен равным 10, а все белки из генома *genome2* имеют длину, отличающуюся от длины *i*-го белка *genome1* более чем на 10%); в этом случае значение *closest* полагается равным –1, а *max_score* – значению глобальной константы *SCORE_MIN* (-1000000). По завершении главного цикла в стандартный выходной поток записывается ещё одна строка (:77), содержащая ключевое слово “[End]”, свидетельствующая, что процесс сравнения геномов был успешно завершён

(т.к. этот процесс может быть довольно длительным, велика вероятность того, что процесс может быть прерван пользователем или из-за системных сбоев). После этого работа программы нормально завершается с кодом возврата 0 (:83).

Функция read_genome( ) обеспечивает чтение из специально сформированного файла белковых последовательностей одного генома (см. приложение 2). В качестве аргументов функция принимает строку (string) *infile_name* с именем файла белковых последовательностей и ссылку на вектор строк (vector<string>) *sequences* для записи прочитанных последовательностей; возвращается число прочитанных из файла последовательностей; глобальные переменные не используются. В начале работы функции производится попытка открыть файл с именем *infile_name*; в случае неудачи на консоль оператора (стандартный поток ошибок stderr) выводится сообщение о невозможности открытия файла, и в вызывающую функцию передаётся значение 0. В противном случае производится построчное считывание файла, и каждая его строка либо игнорируется (если она пуста или начинается с символа ‘>’), либо (в противном случае) добавляется отдельной записью в массив *sequences*. По достижении окончания входного файла, его поток закрывается, и в вызывающую функцию передаётся число элементов массива *sequences*.

Функция fasta_score( ) реализует слегка модифицированный (с целью возможности существенного варьирования скорости счёта за счёт несущественной потери точности) алгоритм вычисления одной из стандартных мер близости белков – FASTA SCORE. Аргументами функции являются две текстовые строки (string), для которых производится подсчёт FASTA SCORE; возвращается вычисленное значение FASTA SCORE или значение глобальной константы *SCORE_MIN*, если при заданном значении глобальной переменной *max_len_dif* данный алгоритм не может вычислить FASTA SCORE; используется глобальная переменная *max_len_dif* и глобальная константа *SCORE_MIN*. В начале функции описан константный массив целых чисел blosum62, заполненный значениями стандартной матрицы близости аминокислот BLOSUM-62 для пар букв, обозначающих аминокислоты и некоторые их специальные комбинации, дополненные нулями для всех неиспользуемых для этой цели букв латинского алфавита. Также описывается целая константа *deletion_penalty* со значением штрафа за каждую букву одной из последовательностей, которой при данном сдвиге не соответствует никакая буква второй. Также определяется статический двумерный массив целых чисел tab (256×256 элементов), который заполняется при первом запуске функции таким образом, что парам ASCII-кодов заглавных и прописных букв, обозначающих аминокислоты, соответствует значение матрицы BLOSUM-62 на этих парах букв, а всем остальным парам символов – нули. Затем следует код, выполняемый при каждом вызове функции. Определяются локальные константы *l1*, *l2*, равные длинам соответствующих сравниваемых последо-

вательностей, *l_max*, *l_min*, равные соответственно максимуму и минимуму этих длин и *max_tail* – максимальную длину «висячих хвостов» последовательностей, при которых производится процедура вычисления FASTA SCORE, равную *max_len_dif* процентов от *l_max*. Далее производится сравнение разницы длин входных последовательностей (*l_max – l_min*), и если это число превосходит *max_tail*, в вызывающую функцию возвращается значение глобальной константы *SCORE_MIN*, что свидетельствует о невозможности подсчёта для этих последовательностей значения FASTA SCORE при заданном значении *max_tail* ввиду слишком большой разницы их длин. Далее производится цикл по «сдвигам» *s* одной последовательности относительно другой, и в случаях, когда длина максимального «висячего хвоста» одной из последовательностей (т.е. её участка, которому при данном сдвиге *s* не соответствуют буквы второй последовательности) не превосходит порога *max_tail*, вычисляется сумма элементов матрицы *tab*, соответствующих всем парам соответствующих (с учётом сдвига *s*) букв входных последовательностей (включая «хвосты», которые дают вклад *deletion_penalty* на каждую позицию), с одновременным подсчётом максимума этих сумм по сдвигам *s*. Последний и является значением FASTA SCORE для данной пары белков; его значение возвращается вызывающей функции.

4. Приложение 1. Формат входных данных программы Protfile

4.1. Набор геномов

Набор геномов представляет собой набор входных файлов, организованных определённым образом в структуру каталогов. В наборе геномов должны быть представлены все геномы из плюс-минус списка (см. 4.2 ниже). Каждый геном представляет собой (по возможности полный) набор белков одного организма. Набор геномов представляется аналогично формату представления данных на FTP NCBI в каталоге ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria/, описанном в файле ftp://ftp.ncbi.nih.gov/genbank/genomes/README. Фактически программа рассчитана на использование локальной копии указанного каталога FTP (в файловой системе сервера, на котором программа исполняется), однако это не является требованием, так как необходимая структура может быть легко воспроизведена самостоятельно по следующему описанию.

- В директории данных программы (задаётся параметром `data_dir`), расположен набор поддиректорий, каждая из которых соответствует одному геному (рис. 1).
- В каждой из этих поддиректорий содержится один или несколько файлов в формате FASTA, имеющих фиксированное расширение ‘.faa’ и содержащих произвольное подмножество белков из данного генома (рис. 2, 3).

К именам директорий, faa-файлов и информационным заголовкам белков внутри файлов не предъявляется каких-либо требований, однако в целях удобства чтения выходных данных рекомендуется использовать в качестве имён каталогов имена или идентификаторы соответствующих организмов, а также обеспечивать уникальность информационных заголовков белков в пределах генома. Данные из вышеупомянутого каталога FTP NCBI удовлетворяют указанным требованиям и рекомендациям.

```
Aster_yellows_witches-broom_phytoplasma_AYWB
Buchnera_aphidicola
Buchnera_aphidicola_Sg
Buchnera_sp
Candidatus_Blochmannia_floridanus
Candidatus_Blochmannia_pennsylvanicus_BPEN
Chlamydia_trachomatis
Mesoplasma_florum_L1
```

Рисунок 1. Пример содержимого директории геномов.

```
NC_002252.faa
NC_002253.faa
NC_002528.faa
```

Рисунок 2. Пример содержимого директории генома.

```
>gi|10957100|ref|NP_057962.1| anthranilate synthase component I
MFLIEKRRKLIQKKANYHSDPTTVFNHLCGSRPATLLLETAEVNKKNNLESIMIVDSAIRVSAVKNSVKI
TALSENGAEILSILKENPHKKIKFFEKNKSINLIFPSLDNNLDEDKKIFSLSVFDSFRFIMKSVNNTKRT
SKAMFFGGLFSYDLISNFESLPNVKKKQKCPDFCFYLAETLLVVDHQKKTCLIQSSLFGRNVDEKNRIKK
RTEEIEKKLEEKLTSIPKNKTTVPVQLTSNISDFQYSSTIKKLQKLIQKGEIFQVVPSRKFFLPCDNSLS
AYQELKKSNPSPYMFFMQDEDFILFGASPESSLKYDEKNRQIELYPIAGTRPRGRKKDGTLDLDLDSRIE
LEMRTNHKELAEHLMLVDLARNDLARICEPGSRYVSDLVKVDKYSHVMHLVSKVVGQLKYGLDALHAYSS
CMNMGTLTGAPKVRAMQLIAEYEGEGRGSYGGAIGYFTDLGNLDTCITIRSAYVESGVATIQAGAGVVFN
SIPEDEVKESLNKAQAVINAIKKAHFTMGSS
>gi|10957101|ref|NP_057963.1| anthranilate synthase small subunit
MANILLLDNIDSFTYNLVEQLRNQKNNVLVYRNTVSIDIIFNSLKKLTHPILMLSPGPSLPKHAGCMLDL
IKKVKGDIPIVGICLGHQAIVEAYGGIIGYAGEIFHGKASLIRHDGLEMFEGVPQPLPVARYHSLICNKI
PEKFVINSYFEKMIMSVRNNCDRVCGFQFHPESILTTHGDQILEKIIHWASLKYITNKKQ
>gi|10957102|ref|NP_057964.1| anthranilate synthase small subunit
MANILLLDNIDSFTYNLVEQLRNQKNNVLVYRNTVSIDIIFNSLKKLTHPILMLSPGPSLPKHAGCMLDL
IKKVKGDIPIVGICLGHQAIVEAYGGIIGYAGEIFHGKASLIRHDGLEMFEGVPQPLPVARYHSLICNKI
PEKFVINSYFEKMIMSVRNNCDRVCGFQFHPESILTTHGDQILEKIIHWASLKYITNKKQ
```

Рисунок 3. Пример faa-файла.

4.2. Файл плюс-минус списка

Входной файл плюс-минус списка имеет текстовый формат и состоит из строк переменной длины. Имя этого файла задается параметром `list_file_name`. Каждая информационная строка содержит имя одного из геномов, которому предшествует пара квадратных скобок, содержащих один из символов “+” (плюс), “–” (минус), “*” (звёздочка) или “ ” (пробел) (рис. 4). Прочие строки считаются комментариями и игнорируются. Геномы, помеченные плюсом или минусом, попадают в плюс- или минус-список соответственно, геномы, помеченные звёздочкой, попадают в плюс-список и используются в качестве «первых», не по-

меченные геномы (пробел в квадратных скобках) индексируются в программе, но в вычислениях никак не используются.

```
[ ] Aster_yellows_witches-broom_phytoplasma_AYWB
[*] Buchnera_aphidicola
[-] Buchnera_aphidicola_Sg
[*] Buchnera_sp
[-] Candidatus_Blochmannia_floridanus
[+] Candidatus_Blochmannia_pennsylvanicus_BPEN
[-] Chlamydia_trachomatis
[ ] Mesoplasma_florum_L1
```

Рисунок 4. Пример файла плюс-минус списка.

4.3. Файл конфигурации программы

Необязательный файл конфигурации содержит дополнительные параметры настройки программы (такие как имена входных/выходных файлов, команда запуска подпрограммы ScoreMax, коэффициент точности подсчёта SCORE и т.п.) и представляет собой текстовый файл со строками переменной длины (рис. 5). Если файл конфигурации используется, его имя передаётся в виде параметра при вызове программы Protfile. Файл конфигурации состоит из строк присваивания, несущих информацию о значении одного из параметров программы, строк комментариев и пустых строк. Строка присваивания имеет вид: “parameter_name = parameter value”, где “parameter_name” – имя одного из параметров программы (см. таблицу 1), “parameter value” – значение, присваиваемое параметру; (пробелы вокруг знака равенства не обязательны и игнорируются в любом числе; значения могут заключаться в кавычки (“), также не обязательные и игнорируемые программой). Строкой комментариев считаются любые строки, первым непустым символом которых является символ “#” (решётка), символ “;” (точка с запятой) либо комбинация символов “//” (две косые черты подряд); такие строки игнорируются программой без выдачи сообщений об ошибке. В файле параметров не обязательно указывать значения всех параметров; параметрам, имя которых не встречается ни в одной из информационных строк, присваивается значение по умолчанию.

```
# input data directory name
data_dir = data

# plus-minus list file name
list_file_name = plusminus.txt

# command to call 'scoremax' program
scoremax_call = scoremax

# maximum number of shown proteins per 'first' genome
max_results = 0

# try to use calculation results from previous run(s)
resume_work = yes
```

Рисунок 5. Пример файла конфигурации.

Каждый параметр может иметь один из трёх типов: строковой (текстовая строка), числовой (целое число или десятичная дробь) или булев (принимает одно из значений, эквивалентных “да” или “нет”). Синтаксис запись параметров различных типов одинаков; значения параметров булева типа могут записываться как “yes”/“no” или “true”/“false”. Все доступные параметры с их типом, значением по умолчанию и кратким объяснением их назначения приведены в таблице 1.

Таблица 1. Параметры программы Protfile.

имя	тип	значение по умолчанию	назначение
data_dir	строка	data	имя каталога с геномами
work_dir	строка	work	имя рабочего каталога для создания временных файлов
list_file_name	строка	plusminus.txt	имя файла плюс-минус списка
orgs_file_name	строка	organisms.txt	имя файла со списком организмов
out_file_name	строка	results.txt	имя выходного файла результатов
log_file_name	строка	protfile.log	имя файла протокола
scoremax_call	строка	scoremax	команда вызова программы ScoreMax
scoremax_param	строка	10	параметры вызова программы ScoreMax
max_results	целое	0	максимальное число выдаваемых программой белков для каждого «первого» генома. 0 – выдавать все белки, прошедшие отбор, без ограничения.
resume_work	булево	true	Включение/выключение режима использования промежуточных данных, полученных при предыдущем запуске программы.
org_code_prefix	строка	ORG	Префикс кода организма, используемого программой для краткого обозначения организмов на консоли и именования временных файлов.
win_style_lf	булево	false	Включение/выключение использования в выходных файлах перевода строки в формате ОС Windows.

5. Приложение 2. Формат выходных и рабочих данных программы Protfile

5.1. Файл результатов программы

Файл результатов (его имя задаётся параметром `out_file_name`) имеет текстовый формат с записями переменной длины и состоит из последовательных блоков, по одному на каждый «первый» геном, разделённых пустой строкой. Каждый блок начинается строкой заголовка вида “`'First' genome –` *`имя генома`*.”, за которой следует список отобранных белков (возможно, пустой). Каждая строка списка содержит численную оценку соответствия белка плюс-минус списку и информационный заголовок белка, разделённые символом табуляции (рис. 6).

```
'First' genome - Buchnera_aphidicola.
0.6991    gi|27904537|ref|NP_777663.1| putative methylase
0.6944    gi|27904986|ref|NP_778112.1| tRNA transferase
0.636     gi|27904658|ref|NP_777784.1| NADH dehydrogenase subunit 2
0.6271    gi|27904629|ref|NP_777755.1| glycyl-tRNA synthetase beta chain
0.5384    gi|27904642|ref|NP_777768.1| isoleucyl-tRNA synthetase

'First' genome - Buchnera_sp.
0.8555    gi|15617160|ref|NP_240373.1| tRNA transferase
0.6854    gi|15616839|ref|NP_240051.1| … ligase
0.6239    gi|15616819|ref|NP_240031.1| penicillin-binding protein 1b
```

Рисунок 6. Пример файла результатов.

5.2. Файл протокола

Файл протокола имеет текстовый формат с записями переменной длины. Каждая запись содержит отметку времени, за которой следует сообщение на английском языке, комментирующее ход выполнения программы и выполненные этапы (рис. 7).

```
[21:19:24] 'Protfile' program started.
[21:19:24] Current configuration parameters:
data_dir = 'data'
work_dir = 'work'
list_file_name = 'plusminus.txt'
<…>

[21:19:24] Reading plus-minus list from file 'plusminus.txt'.
[21:19:24] Read 8 organism names <…>
[21:19:24] Organism names with internal indexes:
ORG1: Aster_yellows_witches-broom_phytoplasma_AYWB
ORG2: Buchnera_aphidicola
ORG3: Buchnera_aphidicola_Sg
<…>

[21:19:24] Looking for data from previous run.
[21:19:24] Using data from previous run.
[21:19:24] Checking existance of all needed genomes.
[21:19:24] All needed genome directories are present.
[21:19:24] Preparing genomes.
[21:19:24] Genomes prepared.
[21:19:24] 'First' genome - Buchnera_aphidicola.
[21:19:24] Comparing genomes.
[21:19:24] Genomes comparison is complete.
[21:19:24] Calculating profiles quality.
<…>
[21:19:25] Sending results to output file.
[21:19:25] All done.
```

Рисунок 7. Пример фрагментов файла протокола.

5.3. Сообщения на консоль оператора

Сообщения на консоль оператора выдаются на английском языке и отображают текущий статус работающей программы (рис. 8).

```
Protfile: Program for Specific Proteins Search
(C) 2006, Laboratory of Mathematical Methods and Models in Bioinformatics
Institute for Information Transmission Problems, Russian Academy of Sci-
ences

Preparing genomes ... done.
'First' genome - ORG2
Analysing profiles ... done.
Loading protein information headers for ORG2 ... done.
Producing results ... done.
'First' genome - ORG4
Analysing profiles ... done.
Loading protein information headers for ORG4 ... done.
Producing results ... done.

Thank you for using the program.
```

Рисунок 8. Пример вывода программы Protfile на консоль.

Кроме того, на консоль могут выдаваться следующие сообщения об ошибках.

Can't find data directory ‘xxx’. Каталог с геномами с именем xxx не существует (вероятно, неверное значение параметра `data_dir`).

No data found in the data directory ‘xxx’. В каталоге с геномами xxx не найдены подкаталоги, содержащие faa-файлы (вероятно, неверное значение параметра `data_dir`).

Plus-minus list template created ('xxx'), please fill it in. Программа не обнаружила файла плюс-минус списка и создала шаблон, содержащий имена всех подкаталогов каталога геномов, содержащих, как минимум, один faa-файл; имя файла шаблона xxx – для продолжения работы необходимо расставить в нём знаки +/-/* и снова запустить программу.

Plus-list must contain at least 1 organism. Stop. Плюс-список пуст; необходимо добавить в него хотя бы один геном.

Minus-list must contain at least 1 organism. Stop. Минус-список пуст; необходимо добавить в него хотя бы один геном.

At least 1 organism must be marked as 'first'. Stop. Ни один из геномов не выбран в качестве «первого»; необходимо пометить звёздочкой хотя бы один геном в файле плюс-минус списка.

Can't find genome directory for xxx. Не удаётся найти подкаталог с именем xxx в каталоге геномов (возможно, опечатка в имени генома в файле плюс-минус списка).

Can't find any .faa files for xxx. В подкаталоге xxx каталога геномов не найдено ни одного файла с расширением ‘.faa’.

Can't open output file 'xxx'. Не удаётся открыть выходной файл с именем xxx; возможно, имя файла недопустимо, или у процесса нет прав на его создание.

Can't load 'scoremax' results from 'xxx'. После запуска программы ScoreMax не удаётся прочитать его выходной файл с именем xxx; возможно, произошла ошибка при запуске ScoreMax или в процессе его работы.

Can't open log file 'xxx'. Не удаётся создать файл протокола с именем xxx; возможно, заданное имя недопустимо, или у процесса нет прав на его создание.

Can't open configuration file 'xxx' for reading. Не удаётся открыть файл конфигурации с именем xxx; возможно, файл с таким именем не существует или у процесса нет прав на его чтение.

Can't open plus-minus list file 'xxx' for reading. Не удаётся открыть файл плюс-минус списка с именем xxx; возможно, файл с таким именем не существует или у процесса нет прав на его чтение.

Can't open organisms list file 'xxx' for writing. Не удаётся создать файл списка организмов с именем xxx; возможно, заданное имя недопустимо, или у процесса нет прав на его создание.

Can't open .faa file 'xxx' for reading. Не удаётся открыть faa-файл с именем xxx; возможно, файл с таким именем не существует или у процесса нет прав на его чтение.

Can't open header file 'xxx' for writing. Не удаётся создать файл заголовков белков с именем xxx; возможно, заданное имя недопустимо, или у процесса нет прав на его создание.

Can't open sequences file 'xxx' for writing. Не удаётся создать файл белковых последовательностей с именем xxx; возможно, заданное имя недопустимо, или у процесса нет прав на его создание.

Can't open scores file 'xxx' for reading. Не удаётся открыть файл максимумов SCORE с именем xxx; возможно, файл с таким именем не существует или у процесса нет прав на его чтение.

Can't open header file 'xxx' for reading. Не удаётся открыть файл заголовков белков с именем xxx; возможно, файл с таким именем не существует или у процесса нет прав на его чтение.

Error in FASTA file 'xxx' on line nnn: headers count doesn't match sequences count. Ошибка в FASTA-файле с именем xxx на строке с номером nnn: число прочитанных заголовков последовательностей не совпадает с числом тел последовательностей.

Error in FASTA file 'xxx': no sequences. В FASTA-файле с именем xxx не содержится ни одной последовательности.

Couldn't open input file 'xxx'. Программе ScoreMax не удалось открыть входной файл с именем xxx.

5.4. Файл списка организмов

Файл списка организмов (его имя задаётся параметром `orgs_file_name`) используется для фиксирования входного набора геномов и привязки к ним внутренних кодов в режиме использования данных предыдущего запуска. Файл может использоваться для выяснения внутреннего кода организма по его имени, но не должен изменяться пользователем. Файл создаётся программой и, если он существует, читается ей при следующем запуске (при условии что включен параметр `resume_work`). Каждая (непустая) строка файла содержит запись вида «код_организма: имя_организма» (рис. 9).

```
ORG1: Aster_yellows_witches-broom_phytoplasma_AYWB
ORG2: Buchnera_aphidicola
ORG3: Buchnera_aphidicola_Sg
ORG4: Buchnera_sp
ORG5: Candidatus_Blochmannia_floridanus
ORG6: Candidatus_Blochmannia_pennsylvanicus_BPEN
ORG7: Chlamydia_trachomatis
ORG8: Mesoplasma_florum_L1
```

Рисунок 9. Пример файла списка организмов.

5.5. Файлы заголовков белков

Каждый файл заголовков белков содержит копии информационных заголовков белков из входных faa-файлов одного генома (без служебного символа ‘>’), по одному заголовку в строке (рис. 10). Файлы создаются в рабочем каталоге (имя которого задаётся параметром `work_dir`) программой Protfile на этапе подготовки данных для всех геномов из плюс и минус списка и читаются ей же (по одному) при формировании файла результатов; удаление файлов не производится, что позволяет корректно продолжить начатые вычисления, даже если исходные данные изменились или стали недоступны. Имена файлов генерируются автоматически посредством добавления расширения ‘.hdr’ к коду соответствующего организма (например, ‘ORG1.hdr’, ‘ORG2.hdr’ и т.п.). Каждая непустая строка ассоциируется с соответствующей по порядковому номеру непустой строкой файла белковых последовательностей.

```
gi|27904514|ref|NP_777640.1| glucose-inhibited division protein A
gi|27904515|ref|NP_777641.1| ATP synthase subunit A
gi|27904516|ref|NP_777642.1| ATP synthase subunit C
gi|27904517|ref|NP_777643.1| ATP synthase B chain
gi|27904518|ref|NP_777644.1| ATP synthase delta chain
…
```

Рисунок 10. Пример файла заголовков белков.

5.6. Файлы белковых последовательностей

Каждый файл белковых последовательностей содержит копии белковых последовательностей из входных faa-файлов одного генома по одной последовательности в строке

(рис. 11). Файлы создаются на этапе подготовки данных программой Protfile в рабочем каталоге (имя которого задаётся параметром `work_dir`) для всех геномов из плюс и минус списка и читаются программой ScoreMax; удаление файлов не производится, что позволяет корректно продолжить начатые вычисления, даже в случае, когда исходные данные изменились или стали недоступны. Имена файлов генерируются автоматически посредством добавления расширения '.seq' к коду соответствующего организма (например, 'ORG1.seq', 'ORG2.seq' и т.п.). Никакой структуры кроме разделения на строки не предполагается, файл целиком состоит из белковых последовательностей, разделённых символом перевода строки. Каждая непустая строка ассоциируется с соответствующей по порядковому номеру непустой строкой файла заголовков белков.

```
MLREQNFDVIIVGGGHAGTEAALACSRMKKKTLLLTQNINTIGALSCNPAIGGIGKSHLVKEVDALGGIM…
MNLEHNFNLPNYINHHLHHLQLNLSNLKFLNFNDNYIRNFWVCNFDSIFLSIFLGLVILISFFKISKTFT…
MDNVSMDMLYIAVAVMIGLAAIGAAVGIGILGSKFLEGVARQPDLTSLLRTQFFVVMGLVDAIPMIAVGL…
…
```

Рисунок 11. Пример файла белковых последовательностей.

5.7. Файлы максимумов SCORE

Каждый файл максимумов SCORE содержит максимумы индекса близости FASTA SCORE для пар белков из пары геномов; для каждого белка из первого генома указан самый близкий белок из второго и соответствующая числовая оценка их близости. Файлы создаются в рабочем каталоге (имя которого задаётся параметром `work_dir`) программой ScoreMax для всех пар (*геном1*,*геном2*), где *геном1* – один из «первых» геномов, а *геном2* – произвольный отличный от него геном из плюс-минус списка, и читаются программой Protfile. Удаление файлов не производится, что позволяет продолжить вычисления после сбоя или остановки программы пользователем, используя результаты проведённых длительных вычислений. Имена файлов генерируются автоматически посредством добавления расширения '.scr' к кодам соответствующих организмов, разделённым знаком '-' (дефис) (например, 'ORG2-ORG3.scr', 'ORG2-ORG4.scr' и т.п.). Каждая непустая строка кроме последней содержит три числа p_1, p_2, s, разделенных знаком табуляции, где p_1 и p_2 – порядковые номера белков первого и второго генома соответственно (начиная с нуля), (белок с номером p_2 является ближайшим гомологом белка p_1 во втором геноме), а s – соответствующее данной паре значение SCORE (рис. 12). Исключение составляет случай, когда для белка из первого генома не найден гомолог во втором, тогда p_2=-1, а s принимает значение большой по модулю отрицательной константы, в остальных случаях p_1 и p_2 натуральные, s – целое число, имеющее по абсолютной величине тот же порядок, что и длины входных последовательностей.

```
0   0     2038
1   1     396
2   2     337
...
501 543   370
502 544   -12
503 545   197
[End]
```

Рисунок 12. Пример файла максимумов SCORE.