Руководство пользователя программы super3GL (v.1.4.5)

Содержание

1. Назначение

Программа `super3GL` предназначена для решения одной из типовых биоинформатических задач в области филогенетики – построения так называемого *супердерева*, т.е. дерева, в определённом смысле ближайшего к заданному набору деревьев. Исходными данными является набор деревьев, которые отражают эволюцию организмов или их биологических атрибутов для некоторой таксономии. На выходе строится супердерево, листья которого соответствуют таксонам того же или более высокого уровня, так чтобы достигалось его наилучшее (в смысле заданного критерия) согласие с набором исходных деревьев. Например, по набору деревьев генов может быть построено дерево видов, содержащих эти гены, которое имеет минимальную стоимость относительно исходного набора (для заданных значений цены событий потери и/или дупликации гена). Необходимым условием является, чтобы для каждого листа дерева в исходном наборе (т.е. в этом примере гена) было известно, к какому листу супердерева (т.е. виду) он относится. Эта информация представляется непосредственно в наборе исходных деревьев с использованием специального синтаксиса (см. 4.1). Слова «ген» и «вид» не обязательно идентичны одноименным биологическими терминам, а используются в тексте только для определённости.

2. Алгоритм

В программе `super3GL` реализован алгоритм, описанный в **[1]**, который имеет ряд усовершенствований, включая распараллеливание обработки с целью повышения быстродействия программы. Алгоритм состоит из двух фаз: (1) построение набора базисных деревьев, (2) построение супердерева по индукции. Первая фаза должна быть выполнена целиком за один запуск программы (без возможности прерывания работы), её результаты могут быть при желании записаны в виде файла базисных деревьев.

Вторая фаза выполняется либо непосредственно вслед за первой, либо в отдельном запуске программы. Эта работа может быть прервана в любой момент и позже продолжена, начиная с последнего незавершённого шага индукции. Результатом выполнения каждого индуктивного шага (включая последний) является текущее построенное состояние супердерева. Запуск программы для выполнения (или продолжения) расчётов фазы 2 будем ниже называть запуском в *режиме возобновления*.

Исходные деревья генов могут быть не только корневыми бинарными деревьями, но и содержать политомические вершины с любым числом потомков. Тем не менее, предполагается, что деревья правильно укоренены.

3. Особенности реализации и построения

Программа `super3GL` написана на языке C++ и имеет интерфейс командной строки. После соответствующей перекомпиляции она без ограничений может работать под управлением 32- и 64-битных операционных систем Windows, Linux, Unix, MacOS. Программа самостоятельно обнаруживает наличие мультипроцессорной среды, соответствующей стандарту MPI версии 1.2 и позже **[2]**, и в этом случае автоматически задействует реализованные в алгоритме средства параллельной обработки. При наличии высокопроизводительного кластера это позволяет радикально сократить время решения задач большой размерности (см. данные о быстродействии в разделе 7). В некоторых случаях (при наличии оперативной памяти большого объёма) быстродействие программы также может быть повышено за счет изменения структуры рабочих данных (путём обмена времени счёта на память); в такой ситуации может быть полезна 64-битная версия исполняемого модуля для Windows 64 бит, которая способна нормально работать с оперативной памятью объёмом 4 Гб и более.

На веб-странице программы **[5]** для загрузки доступны:

- двоичный исполняемый модуль для Windows 32 бит (без распараллеливания)
- двоичный исполняемый модуль для Windows 64 бит (без распараллеливания)
- исходные тексты программы для компиляции и запуска в среде Linux (с распараллеливанием и без) – предоставляются на условиях лицензии GNU GPL V3.

Компиляция и построение исполняемого модуля для Linux осуществляется с помощью стандартной утилиты `make`. Входящий в дистрибутивный комплект `makefile` подготовлен для построения однопроцессорного модуля. Если в используемой системе установлена какая-либо реализация MPI версии от 1.2, пользователь может построить параллельный вариант программы, для чего он должен внести соответствующие изменения в этот `makefile` (в качестве примера два типичных варианта изменений уже содержатся в нём в закомментированном виде).

Для проверки готового или построенного пользователем исполняемого модуля его можно запустить с опцией `-h` (программа выдаст общую информацию, включая номер версии, и краткую справку по параметрам командной строки, подробнее см. раздел 6.2).

Опции программы – параметры и исходные данные – задаются в файле конфигурации (6.1) и/или командной строке запуска программы (6.2); в последнем случае они имеют приоритет по сравнению с указанными в файле конфигурации. Как частный случай, программа может запускаться вообще без параметров, когда все они указаны в файле конфигурации или принимают значения по умолчанию. Итак, в однопроцессорной среде общий вид командной строки запуска программы из каталога, где она расположена, следующий:

```
super3GL [options]
```

В среде MPI типичная строка запуска программы на 128 процессорах для вычислений в течение не более 1.5 часов может иметь вид:

```
mpirun -np 128 -maxtime 90 super3GL [options]
```

(точный синтаксис зависит от конкретной реализации MPI-среды и описан в ее документации). В квадратных скобках указаны (необязательные) параметры программы `super3GL`.

Примечание: В ОС Windows регистр символов в командной строке не имеет значения, но в других системах, в частности Linux, он важен, поэтому рекомендуется следовать приводимым ниже примерам использования строчных/прописных букв, в том числе в имени программы.

4. Входная информация

При работе программы в обычном режиме (т.е. начиная с фазы 1) в качестве входных данных используется набор исходных деревьев генов, который представлен одним или более файлом входных деревьев (см. 4.1). Каждый такой файл, в свою очередь, содержит одно или более дерево в скобочном формате Newick **[3,4]**. Также имеется возможность записи исходных деревьев прямо в файле конфигурации, без создания отдельного файла.

При запуске программы в режиме возобновления файл деревьев генов и таблица видов не используются. Вместо этого в качестве входной информации выступает промежуточный файл базисных деревьев (5.3), полученный в результате фазы 1. При этом, если доступно корректное текущее состояние результирующего файла супердерева (5.1), то работа программы возобновляется, начиная с этого состояния; в противном случае – начиная с первого шага индукции в фазе 2.

В дополнение к файлу входных или базисных деревьев, в обоих режимах *может* использоваться таблица видов, представленная в отдельном файле (4.2). Она служит троякой цели: а) даёт расшифровку сокращённых обозначений, которые часто используются для указания высокого таксона («вида»); б) указывает максимальное число видов, которые встречаются в задаче; в) позволяет контролировать правильность исходных деревьев в части указываемых имён видов. Если таблица видов не подаётся, то контроля нет, максимальное число видов нужно указывать, а набор видов определяется по совокупности исходных деревьев. В любом случае, программа `super3GL` работает с теми именами видов, которые указаны в листьях исходных деревьев, и выдаёт результаты в той же нотации (при желании, подстановка вместо сокращений полных имён из таблицы может быть выполнена самостоятельной утилитой `uncode`, которую доступна для загрузки на веб-странице программы **[5]**).

К входной информации формально можно отнести файл конфигурации (6.1), в котором пользователь задаёт режимы работы, параметры программы, местоположение исходных

данных и т.п. управляющую информацию. Ввиду особой важности, эти и другие сведения по управлению работой программы выделены в самостоятельный раздел 6.

4.1 Файл входных деревьев

Имя файла деревьев указывается параметром **`-t`** в командной строке или/и параметром **`GeneTreeName`** в файле конфигурации. Имя может включать путь, записанный по правилам используемой ОС. Параметры могут указываться при запуске более одного раза; в таком случае набор исходных деревьев есть объединение деревьев, содержащихся во всех файлах. Указание параметра командной строки в форме **`--t`** отменяет включение *всех* деревьев, указанных в файле конфигурации. Правильность формата записи деревьев проверяется, но за их уникальность и содержание отвечает пользователь.

Файл входных деревьев имеет обычный текстовый формат и содержит одно или более деревьев, записанных в скобочном формате Newick **[3,4]**, включая концевой символ «точка с запятой». Запись дерева может целиком располагаться в одной строке или переноситься на последующие строки по правилам указанного формата. Если файл содержит несколько деревьев, то каждое должно начинаться с новой строки. Дерево может содержать (или не содержать) длины рёбер, метки внутренних вершин и комментарии, всё согласно формату Newick.

Единственная особенность входных деревьев касается меток (имён) терминальных вершин – листьев дерева. Предполагается, что имена всех листьев имеют вид:

```
segm1[_segm2...][stop][...]
```

т.е. состоят из одного или более сегментов, разделённых символом подчёркивания, вслед за которыми может идти стоп-символ, который указывает, что остаток метки должен игнорироваться. По умолчанию стоп-символ не используется, его можно задать в файле конфигурации параметром **`StopLabel`**. Сами сегменты не должны содержать подчёркиваний, пробелов, стоп-символов и других знаков, которые запрещены спецификацией формата.

Также предполагается, что у всех входных деревьях один или более начальных сегментов имени каждого листа (точное число есть параметр программы) в совокупности задаёт имя высокоуровневого таксона («вида»), для которых строится супердерево. Остальные сегменты имени указывают таксоны нижних уровней, наименование организма, макромолекулы и др. информацию (обобщённо, «имя гена»), которая в текущей версии программы не используется и не обязательна.

В идеале, все листья всех деревьев должны иметь имена одинакового формата и притом уникальные в пределах каждого дерева, что на практике имеет место не всегда. Поэтому в программе предусмотрен параметр запуска **`StrictLabelControl`**, определяющий «строгость» контроля исходного формата. При включённом строгом контроле программа проверяет, что помимо требуемого числа начальных сегментов каждое имя обязательно содержит ещё какие-то сегменты до стоп-символа, и что такие полные (до стоп-символа) имена уникальны в пределах дерева. Если строгий контроль выключен, то входные деревья могут иметь имена листьев различного формата, в частности, вообще не содержать имени гена, а уникальность имён будет при необходимости обеспечена средствами программы. А также число начальных сегментов, задающих имя вида, рассматривается как максимальное значение (присутствие имени вида с меньшим числом сегментов не считается ошибкой).

На веб-странице программы **[5]** имеются примеры файлов входных деревьев для обоих описанных случаев. Следует заметить, что программа допускает использование нескольких файлов входных деревьев (в этом случае исходный набор есть объединение деревьев из всех файлов). Однако имена файлов должны указываться явно, не используя метасимволы * и ?, и не по имени каталога, содержащего все нужные файлы (в последнем случае рекомендуется слить все файлы в один).

4.2 Таблица видов

Задаётся файлом в текстовом формате с разделителями, причём используемый в качестве разделителя символ является параметром программы. Может использоваться несколько отдельных таблиц, каждая со своим разделителем. Имя таблицы видов указывается параметром **`-s`** в командной строке или/и параметром **`SpeciesTableName`** в файле конфигурации. Имя может включать путь, записанный по правилам используемой ОС. Параметры могут указываться при запуске более одного раза; в таком случае используется объединение видов из всех файлов (однако пользователь обязан обеспечить уникальность используемых сокращённых обозначений). Указание параметра командной строки в форме **`--s`** отменяет включение *всех* таблиц видов, указанных в файле конфигурации.

Использовать таблицу видов не обязательно, но это позволяет избежать многих ошибок и представить результаты работы программы в более читаемом виде (с помощью отдельной утилиты `uncode`, которую можно загрузить с веб-страницы программы **[5]**).

В начале файла могут присутствовать несколько строк (заголовки полей, описания и т.п.), которые будут пропущены программой (число таких строк задается параметром программы **`SkipLines`**). Последующие строки должны содержать как минимум два поля, разделённых символом-разделителем. Первое поле содержит сокращённое обозначение вида, второе поле – полное наименование. Полное наименование не используется программой `super3GL`, но сокращения должны быть уникальными по всем подаваемым на вход программы таблицам видов. Сокращённые наименования, не встречающиеся во входных деревьях, будут отброшены в процессе работы программы.

Файл таблицы видов удобно создавать, например, в программе Excel с последующим экспортом в формате `CSV`. На веб-странице программы **[5]** имеется пример таблицы бактериальных видов.

5. Выходная информация

Основные выходные данные, являющиеся конечным результатом работы программы `super3GL`, содержатся в файле супердерева (5.1). Как говорилось выше, этот же файл в его текущем состоянии может использоваться и в качестве входной информации при запуске программы в режиме возобновления.

Дополнительно (по желанию) по результатам расчётов выдаётся файл протокола (5.2), который: а) содержит копию всей информации, выдаваемой на консоль (за исключением промежуточных отсчётов); б) может содержать дополнительные данные для анализа процесса решения, которые пользователь запросил в файле конфигурации (6.1).

Кроме того, по окончании фазы 1 программа по желанию пользователя формирует файл базисных деревьев (5.3), который в этом случае естественно считать выходным. Напротив, для режима возобновления этот файл играет роль входного.

5.1 Файл супердерева

Имя файла супердерева указывается параметром **`-o`** в командной строке или параметром **`Super3Name`** в файле конфигурации (приоритет имеет параметр командной строки). Имя может содержать путь, записанный по правилам используемой ОС. Если имя файла супердерева не указано нигде, выдача производится на консоль.

Это текстовый файл, в котором в скобочном формате Newick **[3,4]** записано текущее построенное состояние супердерева. Дерево всегда записывается в одну строку. Во время выполнения фазы 2 алгоритма в это дерево по индукции добавляются новые виды, так что супердерево постепенно растёт. По окончании работы программы файл окончательно содержит построенное алгоритмом полное супердерево.

Такое содержание файла супердерева соответствует стандартной выдаче по умолчанию. Кроме того, по желанию пользователя на каждом индуктивном шаге вместо полной перезаписи файла супердерева может выполняться дозапись в конец, так что файл в каждый момент будет содержать всю последовательность построения супердерева. Для управления выбором служит параметр конфигурации **`Super3Sequence`**.

Как минимум, скобочная запись супердерева содержит метки листьев (имена «видов»). Сверх того, в зависимости от набора параметров, указанных в файле конфигурации, в дерево могут включаться числовые метки внутренних вершин и длины рёбер. «Длина» входящего в вершину ребра (из родительской вершины) – это число от 0 до 1, которое характеризует величину «надёжности» построения данной вершины (большее значение соответствует большей надёжности). В тех редких случаях, когда значение надёжности не может быть вычислено (из-за неопределённости вида *x*/0), используется специальное значение -1 (или иное числовое значение, указанное в файле конфигурации параметром **`Uncertainty`**). Для удобства отображения дерева сторонними программами с учётом масштаба рёбер рекомендуется использовать малое положительное значение. По той же причине нулевую величину надёжности также бывает удобнее представлять малой положительной величиной, которая задаётся параметром **`ZeroValue`**.

Если это запрошено в файле конфигурации (параметр **`Super3Quality`**), перед каждой строкой со скобочной записью дерева в этот файл дополнительно включается еще одна строка с комментарием в формате Newick, следующего вида:

```
ttt N=xxx (yyy added) TotalQ=zzz Q=vvv R=uuu [(ww species skipped)]
```

где `ttt` – время с момента запуска; `xxx` – число листьев (т.е. видов) в супердереве; `yyy` – наименование (кодовое обозначение) вида, добавленного в дерево на последнем шаге индукции; `zzz` – качество полученного дерева (которое максимизирует программа); `vvv` – качество добавленного вида; `uuu` – надёжность добавления указанного листа, выражаемая значением от 0 до 1 (большее значение соответствует большей надёжности, но шкала существенно нелинейная). Запись `ww...` может появляться лишь в том случае, когда параметр конфигурации **`SkipAmbiguos`** установлен в истинное значение, она указывает, что на данном шаге добавления видов в супердерево были отвергнуты `ww` видов, которые, хотя и дают дерево более высокого качества, характеризуются нулевой надёжностью их добавления.

Примеры построенных файлов супердерева приводятся на веб-странице программы **[5]**.

5.2 Файл протокола

По желанию пользователя, программа `super3GL` формирует текстовый файл с протоколом своей работы. Имя файла (может включать путь) задаётся параметром **`-g`** в командной

строке запуска, в противном случае создаётся файл `super3GL.log` в текущем каталоге. Параметр **`--g`** позволяет отменить запись протокола. Если файл с указанным именем уже существует, то при запуске в обычном режиме старое содержимое стирается, а в режиме возобновления производится дозапись в конец существующего файла протокола.

По умолчанию в файл протокола дублируется та информация, которую программа выводит на системную консоль (через стандартный поток `stderr`), в том числе сообщения о встретившихся ошибках. Необходимость в такой выдаче возникает при работе на многих кластерах, где системная консоль недоступна пользователю. Чтобы не перегружать файл избыточной информацией, в него не пишутся служебные отметки (см. **`Milestones`**), которые программа периодически выводит на консоль поверх друг друга, демонстрируя этим успешный ход расчёта.

Примеры файлов протокола приводятся на веб-странице программы **[5]**. В основном, выдаваемая информация самодостаточна и не требует разъяснения. В начале выдачи несколько строк сообщают общие сведения о программе, номер используемой версии, информацию об авторах и условиях лицензии.

Далее, если в командной строке запуска программы указывались какие-либо параметры, печатается строка вида `Arguments: ...` , где вместо многоточия указаны все эти параметры точно в том виде, как они стояли в командной строке.

Следующая строка протокола (`Options: ...`) содержит использованные в данном расчёте важнейшие параметры и режимы работы программы. Ниже даётся полный перечень возможных в этой строке опций (не все они присутствуют в каждом случае):

`Resume` – Означает, что программа запущена в режиме возобновления. Этот режим включается параметром **`-r`** в командной строке или **`Resume`** в файле конфигурации. Если указать в командной строке параметр **`--r`**, то режим возобновления будет отменён, даже если он запрошен в файле конфигурации.

`Prune(p=xxx)` – Означает, что программа использует режим обрезки деревьев, удаляя виды, которые редко встречаются или вообще не используются в наборе исходных деревьев генов. В скобках указано значение порога встречаемости *p*, заданное в командной строке параметром **`-p`** либо в файле конфигурации параметром **`Threshold`** (значение в командной строке имеет приоритет).

`Weight(sf=xx,min=yy)` – Означает, что алгоритм учитывает базисные деревья с весом, отражающим относительную важность того или иного дерева (в противном случае все веса считаются единичными). Этим режимом управляет параметр **`BasisTreeWeight`** в файле конфигурации. В скобках значение `xx` указывает масштабный коэффициент, используемый при вычислении веса, а `yy` – минимальный вес базисного дерева, при котором оно учитывается. Поскольку веса неотрицательные, `min=0` означает, что учитываются все деревья. (Значение `xx` указывается параметром **`-f`** в командной строке или **`ScaleFactor`** в файле конфигурации; значение `yy` задаётся параметром конфигурации **`MinWeight`**).

`Closs=xx` – Указывает значение цены события потери гена, которое алгоритм использует при вычислении качества дерева видов. Задаётся параметром **`-l`** в командной строке или параметром **`CostOfLoss`** в файле конфигурации (приоритет имеет значение в командной строке).

`Cdup=xx` – Указывает значение цены события потери гена, которое алгоритм использует при вычислении качества дерева видов. Задаётся параметром **`-d`** в командной строке или параметром **`CostOfDuplication`** в файле конфигурации (приоритет имеет значение в командной строке).

`ParaPen=xx` – Указывает величину штрафа за паралогичность, которую алгоритм использует при построении оптимального набора базисных деревьев. Задаётся параметром **`-y`** в командной строке или параметром **`ParalogyPenalty`** в файле конфигурации (приоритет имеет значение в командной строке).

`RShift=xx` – Указывает постоянное значение, прибавляемое к значению надёжности добавления вида в фазе 2 алгоритма при нахождении оптимального кандидата. Задаётся параметром **`-z`** в командной строке или параметром **`RShift`** в файле конфигурации (приоритет имеет значение в командной строке).

`SpA=xx` – Указывает способ организации таблицы значений качества топологий тройки видов. Возможные значения `xx`: `0d, 1d, 2d, 3d`. Вариант `3d` имеет максимальное быстродействие, но и наиболее требователен к объёму оперативной памяти, что заставляет для задач с большим числом видов переходить к менее быстрым режимам с представлением `2d` или даже `1d`. Выбор режима осуществляется параметром **`-x`** в командной строке или **`SpecArrayDim`** в файле конфигурации (см. также 8.1).

`MPI-yyy` – Печатается в протоколе, если программа работает в мультипроцессорном режиме (`yyy` – число используемых процессоров). Необходимым условием работы в параллельном режиме является наличие среды MPI версии 1.2 или выше **[2]**, и запуск программы надлежащим способом (см. 3); в этом случае программа super3GL автоматически переходит в мультипроцессорный режим. (Использование параллельного режима можно принудительно запретить параметром **`-nompi`** в командной строке).

При работе в обычном режиме далее с пометками времени выдаются строки протокола, информирующие о количественных и статистических характеристиках исходных данных (числе видов, количестве и размерах исходных деревьев, встречаемости в них видов и т.п.) и объектов, которые алгоритм находит на первой фазе – множество клад *P*, т.н. «хорошие» рёбра или вершины исходных деревьев, базисные множества и, наконец, базисные деревья.

Вторая фаза алгоритма по умолчанию не сопровождается интенсивной выдачей информации на консоль, но в файл протокола выдаются все этапы построения супердерева по индукции, в форме, аналогичной описанной в 5.1. Кроме того, в файле конфигурации предусмотрен ряд параметров протоколирования (их имена начинаются с `Log...`) для выдачи дополнительной информации по желанию пользователя (см. 6.1).

5.3 Файл базисных деревьев

Файл базисных деревьев используется (т.е. создаётся в фазе 1 алгоритма и/или читается в фазе 2), если его имя указано параметром **`-b`** в командной строке или параметром **`BasisName`** в файле конфигурации. Имя файла может включать путь к каталогу.

Файл имеет текстовый формат, каждая строка содержит скобочную запись одного двоичного корневого базисного дерева в формате Newick **[3,4]**. Листьями базисных

деревьев являются виды, не повторяющиеся в пределах одного дерева, и деревья приведены в файле в порядке возрастания числа видов.

Для корня дерева в записи указана метка в виде целого числа и длина. Смысл метки – цена базисного дерева, смысл значения длины – вес данного дерева.

Примеры файла базисных деревьев приводятся на веб-странице программы **[5]**.

6. Управление программой

Основной способ управления программой `super3GL` состоит в изменении значений параметров в файле конфигурации (6.1), что даёт возможность запускать программу вообще без параметров. Тем не менее, по соображениям удобства работы может быть полезно иметь несколько различных конфигураций или модифицировать параметры, не меняя содержимого основного файла конфигурации. В подобных случаях используются параметры, указываемые в командной строке запуска программы (6.2).

6.1 Файл конфигурации

По умолчанию, программа ищет файл конфигурации с именем `super3GL.ini` в текущем каталоге (часто, хотя не обязательно, это тот каталог, где хранится и откуда запускается программа). При желании иное имя и/или местоположение файла конфигурации может быть указано параметром **`-c`** в строке запуска, например:

```
super3GL -c ../example/myconfig.cfg
```

(здесь файл конфигурации с именем `myconfig.cfg` находится в каталоге `example` на уровне текущего каталога). Если программа не может найти файл конфигурации, она устанавливает для всех параметров значение по умолчанию и выдаёт соответствующее сообщение.
Примечание: При записи путей к каталогам необходимо соблюдать правила используемой ОС, в частности, направление наклонной черты, использование кавычек для имён, содержащих пробелы (если они допустимы), и т.п.

Файл конфигурации – это обычный текстовый файл со строками произвольной длины. Символы пробела и табуляции в строках игнорируются. Пустые строки, а также начинающиеся с любого из символов `#  ;  /  *` считаются комментариями и также игнорируются. Кроме того, комментарии могут записываться и в конце смысловых строк, вслед за двумя символами «наклонная черта» ('/'); программа их игнорирует.

Содержательные строки файла имеют общий формат

```
ключ = значение [,значение2...]
```

причём такая конструкция должна целиком располагаться в одной строке (т.е. строки продолжения не предусмотрены). Значения справа от знака равенства могут быть числовыми (целое или дробное число в стандартной нотации, в т.ч. с использованием префикса десятичного порядка `E` или `e`), символьными (если строка символов содержит пробелы или служебные знаки, её необходимо заключить в двойные кавычки), или логическими (истинное значение записывается любым из следующих вариантов: `1, true, yes, enable, on`; ложное значение – любым из вариантов `0, false, no, not, disable, off`). Возможность указывать более одного значения для ключа в данной версии программы пока не используется и воспринимается как ошибка.

Текущая версия программы `super3GL` воспринимает перечисленные ниже имена ключей. При указании ключей в файле конфигурации рекомендуется соблюдать указанный

порядок перечисления (кроме тех случаев, когда группа связанных ключей повторяется). Если имеется значение по умолчанию, оно подчёркнуто.

`MaxValues = 3` Максимально допустимое число значений для ключа (в данной версии программы не используется).

`WorkingDirectory =` *строка_символов* Указывает имя рабочего каталога, в котором программа ищет входные файлы и формирует выходные. Путь к каталогу должен обязательно заканчиваться символом-разделителем, который принят в используемой операционной системе, например, '\' или '/'. Если строка символов содержит пробелы, её необходимо заключить в двойные кавычки. Параметр выполняет ту же функцию, что и параметр **`-w`** в командной строке (указание рабочего каталога в командной строке имеет приоритет).

`Milestones = 6` Параметр управляет периодичностью выдачи на консоль служебных отметок, свидетельствующих о работе программы. Значение 0 отменяет выдачу отметок, прочие значения интерпретируются как длина маски младших бит соответствующего счётчика цикла в программе (например, числа прочитанных деревьев, построенных базисных множеств и т.п.). Если все указанные маской биты значения счётчика нулевые, то отметка выдаётся. Например, значение по умолчанию 6 означает, что отметка будет выдаваться для значений счётчика 0, 64, 128, 192,... Выдача отметок на консоль несколько замедляет работу. При запуске программы в пакетном режиме (на кластере) рекомендуется указывать значение 0. Параметр выполняет ту же функцию, что и параметр **`-m`** в командной строке (указание в командной строке считается более приоритетным).

`Resume = true` Параметр указывает режим запуска программы: значение «ложь» означает обычный режим (фаза 1), значение «истина» – режим возобновления (фаза 2). Логическое значение может указываться многими способами (см. выше). Режим также может выбираться в командной строке параметром **`-r`**, который считается более приоритетным.

`MaxSpecies = 32` Указывает максимальное число видов в наборе исходных деревьев. Параметр обязателен, если таблица видов не подаётся, а значения по умолчанию недостаточно. Значения квантуются с шагом 32 (если указано значение, не кратное 32, программа берёт ближайшее большее).

`CSVdelimiter =` *символ* Символ-разделитель полей таблицы видов (4.2). Непечатаемые символы записываются словами: пробел – `Space`, знак табуляции – `Tab`. Значение по умолчанию – `Tab`.

`SkipLines = 0` Указывает число строк заголовка в начале таблицы видов, которые необходимо пропустить.

`SpeciesTableName =` *строка_символов* Указывает имя файла с таблицей видов. Имя может включать путь к каталогу. При необходимости может использоваться несколько таблиц видов; в этом случае группа ключей **`CSVdelimiter`**, **`SkipLines`**, **`SpeciesTableName`** указывается в файле конфигурации несколько раз (и тогда используется объединение видов из всех таблиц). Кроме того, имеется аналогичный по назначению параметр командной строки **`-s`**. Таблицы видов, переданные через командную строку, объединяются с таблицами из файла

конфигурации; если это нежелательно, в командной строке указывается опция **`--s`**, отменяющая указания всех предложений **`SpeciesTableName`**.

`MaxTrees = 16` Параметр указывает максимальное общее число деревьев генов во всех файлах входных деревьев. Указывать данный параметр не обязательно (при необходимости программа поэтапно увеличивает отводимый объём), но это позволяет несколько ускорить работу и более эффективно управлять имеющейся свободной памятью.

`SpecLabelParts = 1` Указывает для входных деревьев генов число начальных сегментов метки листа, которые представляют имя вида (подробнее см. 4.1).

`StopLabel =` *строка_символов* Параметр устанавливает, какой символ в метках листьев будет рассматриваться как признак конца имени (остаток метки будет проигнорирован). Строка может содержать более одного символа, например, в виде `"@*"`, если в деревьях встречаются разные стоп-символы. По умолчанию, когда этот параметр не указан, стоп-символы не используются, т.е. метка учитывается целиком.

`StrictLabelControl = false` Параметр управляет режимом контроля меток листьев в наборе исходных деревьев. Истинное значение включает режим строгого контроля, ложное выключает (подробнее см. 4.1).

`GeneTreeName =` *строка_символов* Указывает имя файла входных деревьев (4.1), возможно с указанием пути к каталогу. Данный параметр может присутствовать в файле конфигурации более одного раза, если набор исходных деревьев представлен несколькими файлами.

`GeneTree =` *строка_символов* Этот параметр позволяет включить дерево генов в скобочной записи непосредственно в файл конфигурации (вместо ссылки на отдельный файл). Дерево должно быть записано целиком в одной строке в формате Newick. Рекомендуется заключать эту строку символов в двойные кавычки. Предложение может присутствовать в файле конфигурации более одного раза, если необходимо задать несколько исходных деревьев.

`TreeWriteMode = 0` Данный параметр управляет режимом выдачи супердерева в скобочном формате в выходной файл. По умолчанию выдаются только метки терминальных вершин дерева (листьев). Указывается значение параметра, равное сумме значений для желаемого набора дополнительных возможностей вывода:

1: числовые метки в качестве имён внутренних вершин дерева;

2: символьные имена внутренних вершин дерева;

4: длины рёбер, входящих во внутренние вершины (включая корень);

8: длины рёбер, входящих в листья;

16: числовые метки в качестве имён листьев дерева;

32: числовая метка и длина только для корневой вершины.

Например, если указано значение 5, то для каждой внутренней вершины (включая корень дерева) выводится числовая метка и длина входящего ребра. (Напомним, что длина ребра в программе `super3GL` имеет смысл надёжности данной вершины).

`Super3Name` = *строка_символов* Указывает имя файла для записи супердерева (5.1). Может содержать путь к каталогу. Если файл существует, он будет перезаписан новой информацией.

`Super3Sequence` = `true` Параметр управляет выдачей результатов в файл супердерева. Если указано логическое значение «истина», будет выдана последовательность результатов каждого индуктивного шага построения супердерева (5.1), в противном случае – только результат последнего шага, т.е. окончательный вид супердерева, содержащего все виды.

`Super3Quality` = `false` Если указать логическое значение «истина», то перед каждым (или единственным) деревом в файле супердерева будет выведена дополнительная строка комментария, описанная в 5.1.

`TreeLogMode` = `0` Параметр управляет режимом выдачи любых деревьев в скобочном формате в файл протокола (5.2). По умолчанию выдаются только метки терминальных вершин дерева (листьев). Смысл значения тот же, что и для параметра **`TreeWriteMode`**.

`LogActualTrees` = `false` Если указано логическое значение «истина», в файл протокола (5.2) выводится в скобочном формате весь набор исходных деревьев после удаления редко встречающихся видов и неинформативных деревьев (т.е. после «обрезки»).

`LogActualSpecies` = `false` Если указать логическое значение «истина», то в файл протокола (5.2) будет выведено множество видов, оставшихся после обрезки.

`LogEntirePset` = `false` Если указать логическое значение «истина», то в файл протокола (5.2) будет выведен полный набор множеств видов, которые являются элементами множества *P*.

`LogBasisSets` = `false` Если указано логическое значение «истина», в файл протокола (5.2) выводится набор базисных множеств – элементов множества *P*.

`LogBasisTrees` = `false` Если указано логическое значение «истина», в файл протокола (5.2) выводится совокупность построенных базисных деревьев в скобочном формате (по одному дереву в строке протокола). Форма выдачи та же, что и в файл базисных деревьев (5.3).

`LogTopology3` = `false` Если указано логическое значение «истина», в файл протокола (5.2) будет выведена вся таблица значений качества топологии тройки видов (только ненулевые элементы матрицы).

`LogChoice3` = `false` Если указано логическое значение «истина», в файле протокола (5.2) будет дана подробная информация о выборе очередного вида для включения в супердерево на каждом шаге индукции в фазе 2 алгоритма. Чтобы правильно интерпретировать эту информацию, необходимо выдавать в протокол деревья с маркированными внутренними вершинами (см. параметр **`TreeLogMode`**).

`LogSuper3 = true` Если указано логическое значение «истина» (принимается по умолчанию), в файле протокола (5.2) будет представлена вся последовательность построения супердерева по индукции.

`BasisName =` *строка_символов* Указывает имя файла базисных деревьев (5.3); может содержать путь к каталогу. Та же самая информация может указываться параметром **`-b`** в командной строке (в этом случае она имеет приоритет перед указанной в файле конфигурации).

`TreePruningMode = false` Данный параметр управляет режимом обрезки исходных деревьев (значение «истина» включает обрезку, «ложь» – выключает). Этот выбор не окончательный, поскольку зависит от значения порога (параметр **`Threshold`**), которое к тому же может указываться в командной строке параметром **`-p`**. Независимо от значения параметра, из исходного набора будут удалены деревья генов, принадлежащих только одному виду, т.к. они не несут полезной информации для алгоритма.

`Threshold = 1` Параметр дополнительно управляет режимом обрезки исходных деревьев, указывая пороговое значение встречаемости вида. (Это значение учитывается, только если указан параметр `TreePruningMode = true`). Если число встреч вида во всём наборе деревьев меньше порога, то вид удаляется. Значение по умолчанию предписывает удалять только те виды, которые вообще не встречаются в исходном наборе деревьев (а присутствуют только в таблице видов). При нулевом значении порога обрезка не производится. Значение порога также может быть указано в командной строке параметром **`-p`** (это указание имеет приоритет над файлом конфигурации).

`CostOfLoss = 2` Параметр указывает значение цены события потери гена, которое алгоритм использует при вычислении качества дерева видов. Может также задаваться параметром **`-l`** в командной строке (такое значение имеет приоритет).

`CostOfDuplication = 3` Параметр указывает значение цены события дупликации гена, которое алгоритм использует при вычислении качества дерева видов. Может также задаваться параметром **`-d`** в командной строке (такое значение имеет приоритет).

`ExtendPset = false` Параметр управляет составом множества *P*. По умолчанию оно содержит все клады всех исходных деревьев, полный набор видов и все варианты полного набора без одного вида. Если указать истинное значение параметра, то в множество *P* будут дополнительно включены в большом количестве разностные множества. Может также задаваться параметром **`-a`** в командной строке (такое значение имеет приоритет).

`BasisTreeWeight = true` Если указано логическое значение «истина», то базисные деревья учитываются с весом, вычисленным при их построении (вес принимает значения от 0 до величины, на 1 большей значения параметра **`ScaleFactor`**). Если указано логическое значение «ложь», все веса равны 1.

`ScaleFactor = 10` Параметр указывает значение масштабного коэффициента, который используется при вычислении весов базисных деревьев. Значение также может задаваться параметром **`-f`** в командной строке (такое значение имеет приоритет).

`MinWeight = 0` Этот параметр устанавливает пороговое значение веса базисного дерева. Деревья с меньшим весом не учитываются при построении супердерева в фазе 2 алгоритма. Значение по умолчанию 0 означает, что будут использоваться все базисные деревья, независимо от их веса.

`Uncertainty = -1` Этот параметр указывает фиксированное значение надёжности вершины супердерева, которое будет ей приписано в тех случаях, когда расчёт по заданной формуле приводит к неопределённости вида $x/0$. Соответствующее значение будет в таких случаях появляться в скобочной записи супердерева в качестве длины ребра, ведущего в соответствующую внутреннюю вершину. Также может указываться параметром командной строки **`-u`** (что более приоритетно).

`ZeroValue = 0.1` Этот параметр указывает фиксированное значение надёжности вершины супердерева, которое будет ей приписано в тех случаях, когда расчёт по заданной формуле приводит к нулевому значению. Соответствующее значение будет в таких случаях появляться в скобочной записи супердерева в качестве длины ребра, ведущего в соответствующую внутреннюю вершину. Также может указываться параметром командной строки **`-v`** (что более приоритетно).

`EachStepReliability = false` Если указано логическое значение «истина», то на каждом шаге индукции в фазе 2 алгоритма будет вычисляться надёжность добавления очередного вида, т.е. листа, в текущее супердерево. В противном случае эта величина вычисляется только на последнем шаге (для финального супердерева), что даёт небольшую экономию времени. Для той же цели служат параметры **`-e`** / **`--e`** в командной строке, которые имеют приоритет над файлом конфигурации.

`SpecArrayDim = 3` Этот параметр управляет структурой хранения ряда рабочих данных программы, определяя тем самым компромисс между быстродействием и требуемым объёмом оперативной памяти. (Для той же цели служит параметр **`-x`** в командной строке). Режим по умолчанию обеспечивает максимальную скорость работы, однако требует хранить в памяти кубическую матрицу плавающих значений с размерностью, равной числу видов, что не всегда возможно, и поэтому приходится уменьшать значение параметра, снижая тем самым быстродействие. Допустимы значения 3, 2, 1, 0; рекомендации по выбору даются в разделе 8.

`ParalogyPenalty = 1.0` Данный параметр задаёт величину штрафа за паралогичность, используемую при построении набора базисных деревьев. Для той же цели служит параметр **`-y`** в командной строке, который имеет приоритет над указанием в файле конфигурации.

`RShift = 0.5` Данный параметр задаёт величину постоянной добавки к вычисленному значению надёжности и используется при добавлении новых видов в фазе 2 алгоритма. Для той же цели служит параметр **`-z`** в командной строке, который имеет приоритет над указанием в файле конфигурации.

`SpeciesLength = true` Если указано истинное значение, то в супердереве, которое формирует программа, будет указываться надёжность каждой вершины как длина входящего в неё (со стороны корня) ребра.

`SkipAmbiguos = true` Если указано логическое значение «ложь», то на каждом шаге фазы 2 алгоритма в дерево добавляется лучший вид по критерию максимума

величины $Q^*(R + RShift)$, где Q – качество получаемого дерева, R – надёжность добавления этого вида, а *RShift* – значение одноимённого параметра конфигурации. Если параметр имеет логическое значение «истина», то независимо от критерия не будут добавляться виды, для которых $R = 0$, так что в окончательном супердереве могут быть представлены не все виды.

Примеры файла конфигурации со всеми распознаваемыми программой ключами (некоторые из которых закомментированы) содержатся в примерах задач, представленных на веб-странице программы **[5]**.

6.2 Параметры командной строки

Параметры командной строки запуска программы `super3GL` обычно используются, чтобы изменить значение или режим, заданные в файле конфигурации или принятые по умолчанию (параметры в командной строке имеют приоритет). Параметры могут иметь следующий общий вид:

```
-ключ
--ключ
-ключ значение
--ключ значение
```

Значение может быть числом (целым или дробным, в т.ч. с порядком, со знаком или без) или строкой символов. Если строка символов содержит пробелы или служебные символы операционной системы, её необходимо заключать в кавычки. Значение должно отделяться от ключа одним или более пробелом. Краткая справка по параметрам командной строки выдаётся, если указать параметр **`-help`** или просто **`-h`**.

Если не сказано иное, параметры командной строки могут стоять в любом порядке. Для наименований ключей регистр символов не имеет значения, а для символьных строк – в зависимости от операционной системы. Текущая версия программы воспринимает в командной строке следующие ключи (перечислены в алфавитном порядке, в квадратных скобках указаны эквивалентные варианты записи ключа):

`-a` `[-add]`
: Назначение параметра то же, что и у параметра **`ExtendPset`** в файле конфигурации – управление составом множества P. Если параметр указан, то независимо от указаний в файле конфигурации в множество P будут включены дополнительные клады. Если указать параметр в форме `--a`, то добавочные клады не включаются, также вне зависимости от параметра **`ExtendPset`** в файле конфигурации.

`-b` *строка* `[-basis | -bastree | -btree]`
: Смысл параметра тот же, что у параметра **`BasisName`** в файле конфигурации – имя файла базисных деревьев.

`-c` *строка* `[-conf | -config]`
: Этот параметр позволяет указать имя и местоположение файла конфигурации (см. 6.1), если стандартные значения не подходят. Учитывая важность конфигурации, этот параметр необходимо указывать в начале командной строки (ему может предшествовать только параметр **`-w`**).

`-d` *число* `[-cd | -cdup | -dup]`
: Смысл параметра тот же, что у параметра конфигурации **`CostOfDuplication`** – цена события дупликации гена.

`-e` `[-each | -eachstep]`
Назначение этого параметра такое же, как и у параметра конфигурации **`EachStepReliability`**. Если параметр указан, то надёжность добавления очередного листа вычисляется на всех шагах индукции. Если указать параметр в форме `--e`, надёжность вычисляется только на последнем шаге.

`-f` *число* `[-sf | -factor | -scalefactor]`
Смысл параметра тот же, что у параметра конфигурации **`ScaleFactor`** – величина масштабного коэффициента в формуле для вычисления веса базисного дерева.

`-g` *строка* `[-log]`
Параметр указывает имя и местоположение файла протокола (5.2). По умолчанию протокол записывается в файл `super3GL.log` в текущем каталоге. Если указать параметр в форме `--g`, файл протокола не формируется.

`-h` `[-help]`
Если в командной строке присутствует этот параметр, на консоль выдаётся краткая справка по параметрам командной строки (а прочие параметры игнорируются).

`-l` *число* `[-cl | -closs | -loss]`
Смысл параметра тот же, что у параметра конфигурации **`CostOfLoss`** – цена события потери гена.

`-m` *число* `[-milestones | -stamp]`
Смысл параметра тот же, что у параметра конфигурации **`Milestones`** – управление периодичностью выдачи на консоль служебных отметок. Если указано `-m 0`, служебные отметки не выдаются. Если указано отрицательное значение, то используется его абсолютная величина, а выдача отметок производится во всех параллельных ветвях MPI-программы (в противном случае – только на консоль корневой ветви).

`-nompi` `[--mpi]`
Данный параметр позволяет принудительно запретить работу программы в мультипроцессорном режиме (если на конкретной вычислительной установке ошибочно опознаётся наличие среды MPI, либо возникают ошибки при проверке ее наличия).

`-o` *строка* `[-output | -supertree | -super3]`
Назначение параметра то же, что у параметра конфигурации **`Super3Name`** – указать имя и местоположение файла супердерева (5.1).

`-p` *число* `[-threshold | -minocc]`
Смысл параметра тот же, что у параметра конфигурации **`Threshold`** – минимальное пороговое значение встречаемости вида в наборе исходных деревьев (если число встреч меньше порога, то вид удаляется).

`-r` `[-resume | -br | -basisread]`
Назначение параметра то же, что и у параметра конфигурации **`Resume`** – выбирать режим запуска программы. Если параметр указан, программа запускается в режиме возобновления, независимо от указаний в файле конфигурации. Если указать параметр в форме `--r`, то программа будет запущена в обычном режиме, также

независимо от значения в файле конфигурации. В прочих случаях используются указания в файле конфигурации или значение по умолчанию.

`-s` *строка* `[-spec | -species]`

Смысл параметра тот же, что у параметра конфигурации **`SpeciesTableName`** – он указывает имя и местоположение файла с таблицей видов в формате `CSV` (4.2). Параметр может указываться в командной строке несколько раз (если таблица видов состоит из нескольких файлов), но должен стоять раньше параметра, указывающего файл входных деревьев (если есть). Следует иметь в виду, что при указании таблицы видов в командной строке в ней в качестве разделителя полей может использоваться только символ по умолчанию (знак табуляции). Кроме того, если таблица видов указана ещё и в файле конфигурации, то программа будет использовать объединённую таблицу (т.е. объединение множеств видов из командной строки и из конфигурации). Чтобы отменить использование таблиц(ы) видов из файла конфигурации, данный параметр указывается в форме `--s`.

`-t` *строка* `[-tree | -gt | -genetree]`

Смысл параметра тот же, что у параметра конфигурации **`GeneTreeName`** – он указывает имя файла входных деревьев (4.1), возможно вместе с путём к каталогу. При необходимости указать несколько файлов входных деревьев, параметр может присутствовать в командной строке более одного раза. Программа составляет объединённый набор исходных деревьев из всех таких файлов и деревьев, заданных через файл конфигурации. Чтобы отменить использование деревьев из файла конфигурации, данный параметр указывается в форме `--t`.

`-u` *число* `[-uncertain | -uncertainty]`

Смысл параметра тот же, что у параметра конфигурации **`Uncertainty`** – значение надёжности внутренней вершины супердерева, если прямое её вычисление приводит к неопределённости вида *x*/0.

`-v` *число* `[-zerovalue | -value0]`

Смысл параметра тот же, что у параметра конфигурации **`ZeroValue`** – значение надёжности внутренней вершины супердерева, если прямое её вычисление приводит к значению 0.

`-w` *строка* `[-wd | -dir | -wdir]`

Параметр имеет то же назначение, что и параметр **`WorkingDirectory`** в файле конфигурации – указывает каталог для поиска всех используемых программой входных файлов, а также для создания выходных файлов. Если этот параметр указан, он должен предшествовать всем прочим параметрам, поскольку действует для всех файлов, используемых программой. При использовании этого параметра, во избежание ошибок не рекомендуется указывать пути вместе с именами файлов. Последним символом строки должен быть '\' или '/', в зависимости от ОС.

`-x` *число* `[-dim]`

Смысл параметра тот же, что у параметра конфигурации **`SpecArrayDim`** – он управляет структурой представления рабочих данных. В качестве значений допускаются только целые числа от 0 до 3 включительно (чем больше число, тем быстрее работает программа, но тем больше физической памяти ей требуется). См. также раздел 8.

`-y` *число* `[-para]`
: Смысл параметра тот же, что у параметра конфигурации **`ParalogyPenalty`** – он устанавливает величину штрафа за паралогичность при построении базисных деревьев.

`-z` *число* `[-shift]`
: Смысл параметра тот же, что у параметра конфигурации **`RShift`** – он задаёт величину фиксированного сдвига вычисленной надёжности при добавлении новых видов в супердерево.

7. Примеры использования программы

В этом разделе описываются представленные на веб-странице **[5]** программы `super3GL` примеры её использования для решения реальных задач различной сложности. Мы оставляем в стороне биологическую подоплёку этих задач и источник их происхождения, ограничиваясь формальной стороной, относящейся к использованию программы.

Для решения использовались четыре вычислительные установки различной мощности, которые ниже будут символически обозначаться следующим образом:

D1
: Настольная рабочая станция на базе одноядерного процессора Intel Pentium IV с тактовой частотой 3 ГГц и оперативной памятью 2 Гб. Программа запускалась в однопроцессорном режиме (с параметром `-nompi` в командной строке).

D2
: Настольная рабочая станция на базе двухядерного процессора Intel Core 2 Duo с тактовой частотой 2.5 ГГц и оперативной памятью 4 Гб. Программа запускалась в двухпроцессорном режиме.

S32
: Сервер на базе четырех 8-ядерных процессоров Intel Xeon с тактовой частотой 2 ГГц и оперативной памятью 256 Гб. Запуск 64-битной версии программы проводился в мультипроцессорном режиме (вплоть до 32 процессоров).

MVS
: Суперкомпьютер МВС-100К Межведомственного Суперкомпьютерного Центра РАН **[6]**. Расчёты проводились с использованием различного числа процессоров вплоть до 1024; каждый процессор располагал 1 Гб оперативной памяти.

7.1 Пример 1 (40 видов)

Исходный набор деревьев (файл `input_trees.tre`) состоял из 1000 двоичных деревьев генов, относящихся к 40 видам. Основные характеристики исходного набора деревьев представлены во втором столбце Табл. 7.1. Таблица видов (4.2) в этом примере не использовалась.

Использовался файл конфигурации `super3GL.ini` (6.1). Данные о времени выполнения фаз 1 и 2 алгоритма на различных компьютерах приведены в Табл. 7.2. Применительно к компьютеру D2, набор базисных деревьев после первой фазы представлен в файле `basis.tre`, результат решения – в файле `super3.tre`, протокол работы – в файле `super3.log`. Все упомянутые файлы собраны в архив `example040.zip`, который доступен для загрузки на веб-странице программы **[5]**.

Табл. 7.1. Характеристики исходных и промежуточных данные в примерах 1-4.

Характеристика	Пример 1	Пример 2	Пример 3	Пример 4
Число видов	40	108	276	814
Число генов	50932	146545	213370	38022
Число деревьев генов	1000	11184	11516	1509
Число генов в дереве:				
минимум	31	2	7	15
максимум	60	124	164	123
среднее значение	51	13	19	25
стандартное отклонение	6	11	16	14
Число видов в дереве:				
минимум	30	2	3	5
максимум	40	48	88	122
среднее значение	31	7	11	24
стандартное отклонение	2	5	9	13
Общее число встреч видов в наборе исходных деревьев:				
минимум	980	500	25	1
максимум	1461	3140	3140	235
среднее значение	1268	857	527	46
стандартное отклонение	126	495	417	38
Мощность множества *P*	9750	37770	72290	15442
Общее число «хороших» рёбер	73552128	217998302	382098675	8293231
Число хороших рёбер для элемента *P*:				
минимум	980	396	25	1
максимум	22266	32122	43099	5013
среднее значение	7544	5772	5286	537
стандартное отклонение	4095	3891	4510	588
Число хороших рёбер в исходном дереве (с повторениями):				
минимум	47819	1476	2949	1243
максимум	119308	260467	407419	30668
среднее значение	73552	19492	33180	5496
стандартное отклонение	10624	17397	28200	3066
Число базисных множеств	9190	36334	69046	14215
Общее число вариантов разбиения всех базисных множеств	55287	112420	164286	21478
Максимальное число вариантов для одного базисного множества	135	110	74	25
Число базисных деревьев	8784	33239	61995	11397
Максимальное число видов в базисном дереве	40	43	88	122
Вес базисного дерева:				
минимум	0.00	0.00	0.00	0.00
максимум	10.51	2.59	2.02	1.22
среднее значение	1.12	0.37	0.53	0.72
стандартное отклонение	2.19	0.48	0.47	0.42

Табл. 7.2. Время обработки примеров 1-4 на разных компьютерах (в минутах).

Система	Число CPU	Этап расчёта	Пример 1 (40 видов)	Пример 2 (108 видов)	Пример 3 (276 видов)	Пример 4 (814 видов)
D1	1	Фаза 1 Фаза 2 Всего	23 <1 23	243 3 246	646 263 909	99 – –
D2	2	Фаза 1 Фаза 2 Всего	**11** **<1** **11**	90 1 91	229 159 388	9 ≈ 44000 30 сут.
S32	4	Фаза 1 Фаза 2 Всего	6 <1 6	48 <1 48	145 79 224	7 22805 22812
S32	8	Фаза 1 Фаза 2 Всего	3 <1 3	29 1 30	95 37 132	–
S32	16	Фаза 1 Фаза 2 Всего	3 <1 3	17 <1 17	42 13 55	2 4992 4994
S32	32	Фаза 1 Фаза 2 Всего	– – –	**12** **<1** **12**	29 9 38	1 2654 2655
MVS	64	Фаза 1 Фаза 2 Всего	– – –	9 <1 9	21 6 27	1 1721 1722
MVS	128	Фаза 1 Фаза 2 Всего	– – –	7 <1 7	**14** **3** **17**	1 1139 1140
MVS	256	Фаза 1 Фаза 2 Всего	– – –	8 <1 8	13 4 17	2 790 792
MVS	512	Фаза 1 Фаза 2 Всего	– – –	–	22 4 26	**1** **390** **391**
MVS	1024	Фаза 1 Фаза 2 Всего	– – –	–	–	17 630 647
Примечание: Детальные результаты для выделенных клеток таблицы представлены на веб-странице программы **[5]**.						

7.2 Пример 2 (108 видов)

Исходный набор деревьев (файл `new_trees.tre`) состоял из 11516 двоичных деревьев генов, относящихся к 276 видам, но производилась обрезка деревьев по порогу встречаемости видов, равному 500, после чего осталось 11184 деревьев и 108 видов. Основные характеристики исходного набора деревьев представлены в третьем столбце Табл. 7.1. Таблица видов (4.2) в этом примере не использовалась. Использовался файл конфигурации `super3GL.ini` (6.1). Данные о времени выполнения фаз 1 и 2 алгоритма на различных компьютерах представлены в Табл. 7.2. Применительно к компьютеру S32 (для 32 процессоров), набор базисных деревьев после первой фазы представлен в файле `basis.tre`, результат решения – в файле `super3.tre`, протокол работы – в файле `super3.log`. Все упомянутые файлы собраны в архив `example108.zip`, который доступен для загрузки на веб-странице программы **[5]**.

7.3 Пример 3 (276 видов)

Использовался тот же, что и в примере 2, исходный набор деревьев (файл `new_trees.tre`), содержащий 11516 двоичных деревьев генов из 276 видов, но обрезка деревьев не делалась. Основные характеристики исходного набора деревьев представлены в четвёртом столбце Табл. 7.1. Таблица видов (4.2) в этом примере не использовалась. Использовался файл конфигурации `super3GL.ini` (6.1). Данные о времени выполнения фаз 1 и 2 алгоритма на различных компьютерах представлены в Табл. 7.2. Применительно к компьютеру MVS (для 128 процессоров), набор базисных деревьев после первой фазы представлен в файле `basis.tre`, результат решения – в файле `super3.tre`, протокол работы – в файле `super3.log`. Все упомянутые файлы собраны в архив `example276.zip`, который доступен для загрузки на веб-странице программы **[5]**.

7.4 Пример 4 (814 видов)

В этом примере в качестве таблицы видов (4.2) использовался файл `BacNames.csv`, содержащий 820 видов. Исходный набор деревьев (файл `all_trees.tre`) состоял из 1511 двоичных деревьев генов, в которых 6 видов из таблицы не встретились вообще, а два дерева содержали гены только из одного вида. После обрезки по порогу встречаемости видов, равному 1, осталось 1509 деревьев и 814 видов. Основные характеристики исходного набора деревьев представлены во пятом столбце Табл. 7.1. Использовался файл конфигурации `super3GL.ini` (6.1). Данные о времени выполнения фаз 1 и 2 алгоритма на различных компьютерах представлены в Табл. 7.2. Применительно к компьютеру MVS (для 512 процессоров), набор базисных деревьев после первой фазы представлен в файле `basis.tre`, результат решения – в файле `super3.tre`, протокол работы – в файле `super3.log`. Отметим, что в данном примере 82 вида не удалось вставить в супердерево из-за неоднозначности места вставки (нулевое значение надёжности). Затем построенное супердерево (с сокращёнными наименованиями видов) было перекодировано с помощью таблицы видов в дерево `super3n.tre` с полными наименованиями, используя команду `uncode super3.tre BacNames.csv super3n.tre`. Все упомянутые файлы собраны в архив `example814.zip`, который доступен для загрузки на веб-странице программы **[5]**.

8. Рекомендации по применению

Для эффективного использования программы `super3GL` существенную роль играет правильный выбор параметров с учётом размерности и особенностей решаемой задачи. Большинство параметров, описанных в разделе 6, однозначно определяются исходными данными и потребностями пользователя. Исключение составляет параметр **`SpecArrayDim`** (или, что то же самое, ключ **`-x`** в командной строке), выбор которого рассматривается в разделе 8.1. Другой существенный вопрос – выбор числа процессоров для параллельного исполнения программы – обсуждается в разделе 8.2.

Опыт использования программы позволяет дать ряд следующих общих рекомендаций, которые помогут избежать типичных ошибок.

- В качестве текущего каталога рекомендуется использовать тот, где хранятся: файл конфигурации (6.1), файл входных деревьев (4.1) и таблица видов (4.2). Туда же рекомендуется записывать выходные файлы. Исполняемый модуль программы не обязательно помещать в этот каталог – путь к программе можно указать прямо в командной строке (или добавить его в список автопоиска операционной системы). Параметр **`WorkingDirectory`** и ключ **`-w`** в командной строке предназначены для опытных пользователей.

- Рекомендуется указывать необходимые параметры и режимы в файле конфигурации, минимально используя командную строку (главным образом – для модификации значений, указанных в файле конфигурации).

- Не стоит полагаться на значения по умолчанию, поскольку они могут быть изменены в последующих версиях программы. Надёжнее указывать в файле конфигурации все параметры и режимы, даже если они совпадают с принятыми по умолчанию.

- Рекомендуется использовать таблицу видов (4.2), причём содержащую только те виды, которые реально встречаются в наборе исходных деревьев – это поможет избежать непроизводительных затрат оперативной памяти.

8.1 Выбор структуры данных

В зависимости от доступного объёма оперативной памяти, числа процессорных ядер и количества видов в решаемой задаче, имеется возможность с помощью параметра конфигурации **`SpecArrayDim`** (или, что то же самое, ключа **`-x`** в командной строке) управлять обменом памяти на быстродействие. Максимальное быстродействие в фазе 2 алгоритма обеспечивает значение `SpecArrayDim=3`, однако не все практически интересные задачи могут быть решены в этом режиме.

В Табл. 8.1 для разных ситуаций приводится (ориентировочно) максимальное число видов при значении параметра, равном 3. Если оно оказывается недостаточным для конкретной задачи, необходимо указать значение `SpecArrayDim=2`, что может привести к замедлению фазы 2 на 30-50%. Значения 2 достаточно практически для любых задач, но если всё равно проявляется недостаток оперативной памяти, можно использовать значение 1 (при этом скорость работы программы снижается в 3-4 раза и более). Следует учитывать, что в таблице указан объём физической памяти компьютера (без учёта виртуальной памяти и объёма резидентной области ядра ОС), и в предположении, что операционная система одновременно не выполняет другие задачи на тех же процессорах. Точный объём занимаемой памяти также зависит от числа исходных деревьев и их размеров, поэтому фактические пределы могут сдвигаться в меньшую сторону.

Табл. 8.1. Максимальное число видов при значении параметра `SpecArrayDim=3`.

Общий объём памяти (Мб)	Число процессорных ядер, на которых запускается программа							
	1	2	3	4	5	6	7	8
256	360	280	–	–	–	–	–	–
512	460	360	–	–	–	–	–	–
768	520	410	–	–	–	–	–	–
1024	580	460	400	360	–	–	–	–
1536	660	520	460	410	–	–	–	–
2048	730	580	500	460	420	400	380	360
3072	**830**	660	580	520	490	460	430	410
4096	**920**	730	630	580	530	500	480	460
6144	**1050**	**830**	730	660	610	580	550	520
8192	**1160**	**920**	**800**	730	670	630	600	580

Примечание: Для работы в режиме, соответствующем заштрихованным клеткам таблицы, необходимо применять 64-битную операционную систему и процессор с поддержкой 64-битной адресации. Клетки таблицы, выделенные дополнительно жирным шрифтом, требуют также применять 64-битную версию программы.

Следует учитывать, что фаза 2 алгоритма достаточно эффективно распараллелена, так что запуск программы на большем числе процессорных ядер во многих случаях способен дать больший прирост быстродействия, чем уменьшение числа используемых процессоров с целью высвобождения объёма оперативной памяти, достаточного для работы в режиме `SpecArrayDim=3` при необходимом для конкретной задачи числе видов.

8.2 Выбор числа процессоров

На основании данных Табл. 7.2 можно сделать следующие наблюдения:

1) В задачах с небольшим числом видов (менее 250) основная вычислительная нагрузка связана с фазой 1 алгоритма. Напротив, при увеличении числа видов свыше 400 происходит быстрый рост трудоёмкости фазы 2, вынуждающий использовать параллельные вычисления.

2) Трудоёмкость фазы 1 связана прежде всего не с количеством видов, а с количеством генов, т.е. числом листьев в исходном наборе деревьев генов. Напротив, трудоёмкость фазы 2 обусловлена прежде всего количеством видов.

3) Распараллеливание первой фазы алгоритма становится малоэффективным при большом числе процессоров; использовать для неё более 32-64 процессоров, по-видимому, не оправдано.

4) Вторая фаза алгоритма распараллеливается относительно эффективно вплоть до числа используемых процессоров, примерно равного числу видов в задаче, после чего выигрыш снижается.

С учётом этого, для задач большой размерности, в которых фигурирует большой набор исходных деревьев генов, и число видов также велико, целесообразно *разделить фазы решения задачи*:

- Вначале запустить программу в обычном режиме на относительно небольшом числе процессоров, и по окончании построения множества базисных деревьев (с записью их в соответствующий файл – см. 5.3) принудительно прервать выполнение фазы 2. Этот этап работы может выполняться на рабочей станции пользователя или сервере рабочей группы, используя все имеющиеся на них процессоры (ядра), что, однако, требует установить на компьютере среду MPI. Запуск в однопроцессорном режиме также возможен, но время расчёта может измеряться многими часами (для рассмотренных нами задач при использовании 16 процессоров часа машинного времени хватало во всех случаях).
- Затем запускать программу в режиме возобновления на многопроцессорном кластере, возможно в ходе нескольких сеансов счёта, пока не будет построено окончательное супердерево со всеми видами. Рекомендуется начинать расчёты с числа процессоров, близкого к числу видов в задаче (если столько доступно) или меньшего. Предварительно следует выяснить параметры узлов кластера (объём памяти и число процессорных ядер) и спланировать загрузку узлов, возможно, с учётом Табл. 8.1. Запуск должен производиться отрезками времени достаточного размера, чтобы выполнить хотя бы один шаг индукции (оценивается по результатам первых шагов).

8.3 Устранение возможных неисправностей

В ходе первой фазы алгоритма программа `super3GL` может выдавать разнообразные сообщения об ошибках или аварийных ситуациях. Как правило, они связаны с ошибками в

исходных данных и/или параметрах программы. Обычно текст сообщения содержит достаточную информацию, чтобы найти и исправить такие ошибки.

Ошибки в ходе второй фазы алгоритма не всегда сопровождаются ясными пользователю сообщениями, особенно при работе в мультипроцессорном режиме. Иногда возникают ошибки или исключения самой операционной системы или используемой среды MPI. Часто их причиной являются нарушение синхронизации процессов, кратковременные отказы оборудования кластера, сбои и таймауты коммуникационной среды. В подобных случаях рекомендуется пробовать действия в следующем порядке:

1) убедиться, что используется самая свежая версия программы, представленная на веб-странице **[5]** (возможно, эта ошибка уже была исправлена);
2) проверить, не связана ли ошибка с нехваткой оперативной памяти – с помощью средств ОС проверить занятый программой объём памяти и наличие достаточного количества физической памяти; для систем Windows, если этот объём превышает 2 Гб на один процесс – перейти к использованию 64-битной версии программы;
3) перезапустить программу в режиме возобновления, не меняя прочие параметры, и продолжить построение супердерева с последней незаконченной итерации;
4) продолжить расчёты с другим числом процессоров;
5) не трогая файл базисных деревьев (5.3), удалить уже построенный файл супердерева (5.1) и снова запустить программу на другом числе процессоров в режиме возобновления, чтобы начать построение супердерева заново.

Если эти действия не помогают, обращайтесь к разработчику (Рубанов Л.И., rubanov@iitp.ru), с указанием номера версии и типа (32/64 бит) применяемой программы, приложив исходные данные, файл конфигурации и командную строку запуска, выходные файлы, протокол работы программы и текст сообщения системы, если есть.

9. Литература

1. Lyubetsky V.A., Rubanov L.I., Rusin L.Yu., Gorbunov K.Yu. “Cubic time algorithms of amalgamating gene trees and building evolutionary scenarios”. *Biology Direct,* 2012, 7:48.
2. MPI: A message-passing interface standard [http://www.mpi-forum.org/docs/mpi21-report.pdf]
3. Newick format - Wikipedia [http://en.wikipedia.org/wiki/Newick_format]
4. Gary Olsen's Interpretation of the "Newick's 8:45" Tree Format Standard [http://evolution.genetics.washington.edu/phylip/newick_doc.html]
5. Программа построения супердерева [http://lab6.iitp.ru/ru/super3gl/]
6. Межведомственный Суперкомпьютерный Центр Российской Академии Наук [http://www.jscc.ru/]