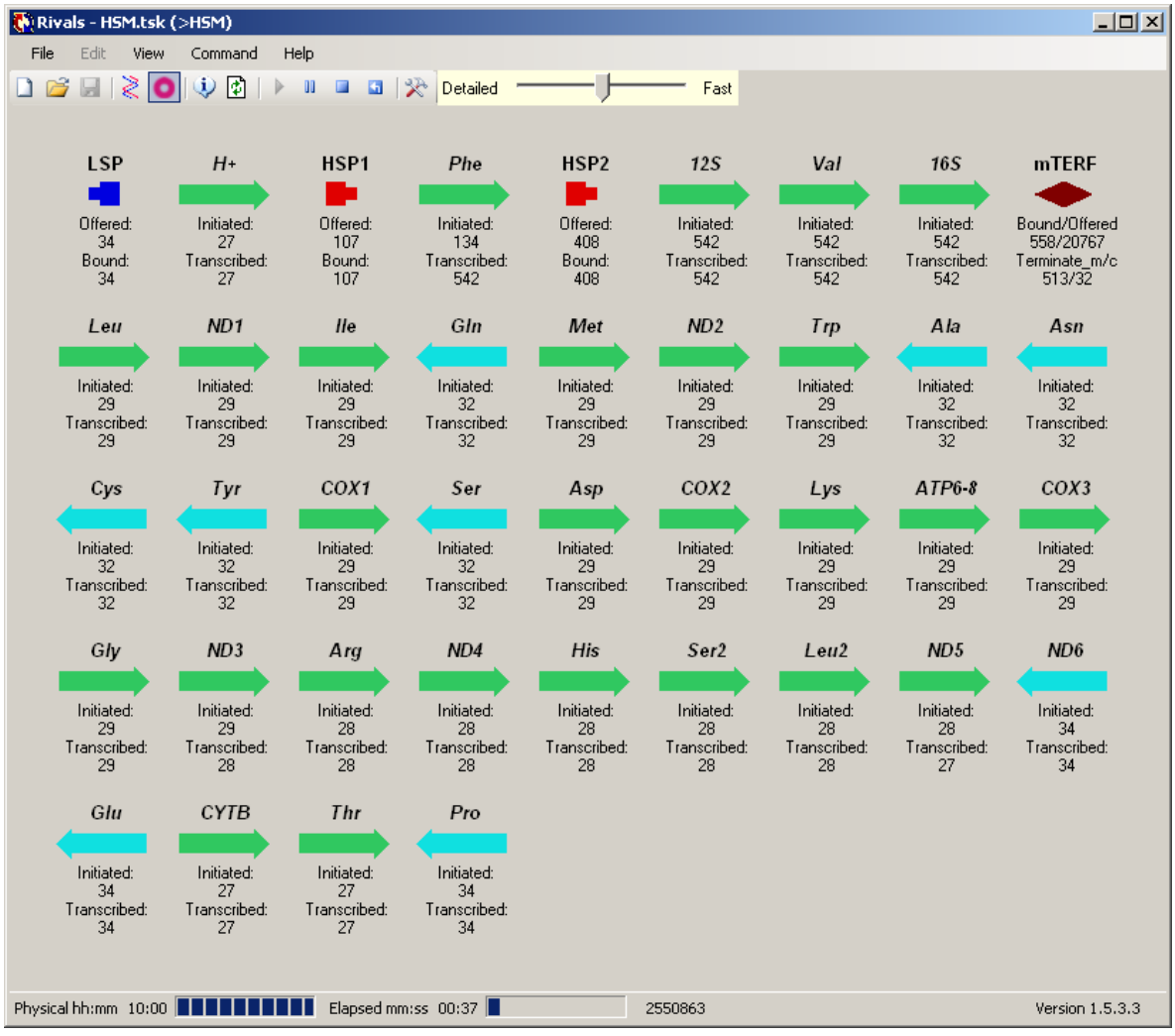
Rivals: Программа моделирования конкуренции РНК-полимераз

Описание и руководство пользователя



Версия программы: 1.5.3.3 (23.11.2012)

Версия руководства: 2.0 (23.11.2012)

Содержание

1. Назначение программы и особенности реализации 3
1.1. Общие сведения 3
1.2. Особенности программной реализации 4
1.3. Объектная модель программы 5
1.3.1. Последовательность ДНК 5
1.3.2. Ген 5
1.3.3. PEP-промотор 7
1.3.4. Холофермент 8
1.3.5. Кор-фермент 9
1.3.6. Внешний источник кор-ферментов 10
1.3.7. NEP-промотор 11
1.3.8. Полимераза фагового типа 12
1.3.9. Внешний источник полимераз фагового типа 12
1.3.10. Репрессор 13
1.3.11. Терминатор 15
1.4. Алгоритм работы модели 16
1.4.1. Правила разрешения коллизий 19
1.4.2. Динамическое изменение параметров модели 22

2. Программа Rivals (приложение Windows) 24
2.1. Минимальные требования, установка и тестирование 24
2.2. Интерфейс и параметры командной строки 24
2.2.1. Функции главного меню 25
2.2.2. Панель инструментов 31
2.2.3. Рабочая область окна 31
2.2.4. Строка состояния 33
2.2.5. Параметры командной строки 35
2.3. Работа с программой в интерактивном режиме 36
2.3.1. Настройка конфигурации 36
2.3.2. Установка и изменение параметров объектов 44
2.3.3. Типовые сценарии использования программы 56
2.4. Обработка заданий в пакетном режиме 58

3. Программа cpRivals (утилита командной строки) 62
3.1. Компиляция и тестирование 62
3.2. Параметры командной строки 63
3.3. Приоритет учета параметров модели 66
3.4. Использование в однопроцессорном режиме 67
3.5. Использование в параллельном режиме 69

4. Форматы файлов 73
4.1. Файл конфигурации 73
4.2. Файл задачи 77
4.3. Файл заданий 81
4.4. Файл протокола 87
4.5. Файл результатов 89

5. Кое-что на закуску 90
5.1. О быстродействии программы 90
5.2. Об интерпретации результатов 92
5.3. Об ограничениях текущей версии программы 93
5.4. О представлении неспецифичных PEP-промоторов 93
5.5. О работе с кольцевой ДНК 95
5.6. О регистрации типов файлов в Windows 96
5.7. Об использовании программы Rivals в Linux 97

6. Литература 98

1. Назначение программы и особенности реализации

1.1. Общие сведения

Программа Rivals служит для моделирования регуляции экспрессии генов в условиях конкуренции РНК-полимераз, транскрибирующих один и тот же локус [1, 2]. Это явление, в частности, наблюдается в геномах пластид многих видов растений, а также в митохондриальном геноме хордовых, включая человека.

Программа реализована и предлагается в двух вариантах:

Rivals – исполняемый модуль rivals.exe с графическим интерфейсом пользователя (GUI) для работы в 32-разрядной среде Microsoft Windows, а также в среде Linux с эмулятором Wine (подробнее об этом см. раздел 5.7);

cpRivals – переносимый вариант с интерфейсом командной строки для работы в среде DOS, Windows или Linux. На условиях лицензии GNU General Public License поставляются исходные тексты на языке C++, которые могут быть откомпилированы для используемой 32- или 64-битной ОС. Этот вариант программы предусматривает также работу на вычислительном кластере в среде MPI версии 1.2 и выше, с любым доступным числом процессоров.

За исключением интерфейса и возможности распараллеливания вычислений, оба варианта программы функционально идентичны; имеющиеся отличия оговорены в тексте особо.

Программа Rivals (с GUI) позволяет моделировать процесс транскрипции локуса в двух режимах – пакетном и интерактивном:

1. В **интерактивном режиме** пользователь с помощью GUI может:
   - настраивать режимы работы программы и устанавливать значения параметров модели, которые принимаются по умолчанию (совокупность этих настроек будем называть «конфигурация»);
   - готовить набор исходных данных для моделирования («задача»);
   - запускать задачу и наблюдать за динамикой моделирования;
   - вносить изменения в задачу и/или конфигурацию и повторно запускать модель с измененными данными.
2. В **пакетном режиме** пользователь может автоматически многократно запускать модель с заранее подготовленными наборами значений параметров, но при неизменных конфигурации и задаче, которые должны быть подготовлены заранее. Сценарий работы в пакетном режиме имеет вид файла «задания». При работе программы Rivals в пакетном режиме GUI обеспечивает ограниченный набор функций, допуская наблюдение за ходом моделирования, приостановку и возобновление работы.

Программа cpRivals (с интерфейсом командной строки) обеспечивает работу только в пакетном режиме, аналогичном пакетному режиму программы Rivals. В этом случае наблюдение за ходом моделирования ведется по протоколу, выдаваемому на консоль (за исключением работы в параллельной среде), и нельзя приостановить/возобновить работу.

Оба варианта программы используют одинаковые файлы конфигурации, задачи и заданий, формат которых описан в главе 4.

Во всех режимах работы программа ведет протокол моделирования в текстовом формате и также выдает полученные результаты в формате CSV (значения с разделителями), например, для последующего импорта в электронную таблицу Excel (см. главу 4).

Модель разработана в Лаборатории математических моделей и методов в биоинформатике Института проблем передачи информации им. А.А. Харкевича РАН в течение 2008 – 2012 гг.

Авторы модели: зав. лаб., д.ф.-м.н., проф. Любецкий В.А.; с.н.с., к.ф.-м.н. Селиверстов А.В.
Алгоритмы, программная реализация и документация: в.н.с., к.т.н. Рубанов Л.И.
Тестирование программ, проведение расчетов и обработка результатов: м.н.с. Зверков О.А.

1.2. Особенности программной реализации

Программа Rivals реализована на языке Visual C++ for .NET 2.0 как приложение Windows и откомпилирована в среде Visual Studio .NET 2008 Service Pack 1. Исполняемый модуль rivals.exe работоспособен под управлением 32-битных ОС Windows и Linux (с эмулятором wine). Программа cpRivals написана на языке C++ в форме утилиты командной строки с переносимым кодом и использует программный интерфейс MPI 1.2 для распараллеливания алгоритма. Компиляция и работа этого варианта программы проверялась с компиляторами Microsoft Visual C++ 2005/2008 в среде ОС Windows XP, Windows Server 2003, Vista (все 32- и 64-бит) и свободно распространяемым (http://www.mpich.org/downloads/) пакетом MPICH2 v.1.2 и выше, а в среде Linux с компиляторами gcc v.4.x.x, icc v.10.1.x и разнообразными реализациями MPI. Массовое моделирование с применением программы cpRivals велось на высокопроизводительном кластере МВС-100К в Межведомственном Суперкомпьютерном Центре РАН (http://www.jscc.ru), с привлечением до 2048 процессоров, а также на собственном лабораторном кластере с 64 процессорами.

Программа изначально предназначена для моделирования конкуренции полимераз в крупных локусах, которые могут содержать десятки генов и других интересующих объектов (промоторов, сайтов репрессии и пр.), находящихся на длинных (десятки тысяч оснований) геномных последовательностях. Учитывая это, в основе алгоритмической реализации лежит система с многими агентами, взаимодействие которых моделируется методом Монте-Карло. Поведение каждого агента в конкретный момент времени описывается некоторым детерминированным или случайным процессом с заданным распределением вероятностей, например, пуассоновским. Эти процессы протекают асинхронно, поэтому фиксированное квантование времени со сколь угодно мелким шагом не обеспечивает точного определения очередности событий. Чтобы гарантировать желаемую временную точность и сохранить при этом приемлемое быстродействие, в модели используется неравномерное дискретное время, отсчеты которого соответствуют моментам времени наступления очередного события, которые организованы в хронологически упорядоченную очередь. В этом смысле архитектура модели аналогична операционной системе, управляемой событиями (в противоположность ОС с жестким квантованием времени). Такое построение позволило для средней решаемой задачи получить скорость работы, т.е. изменения физического времени в модели, на 1-2 порядка быстрее хода процессорного времени счета.

Каждый прогон модели соответствует единичной реализации внутриклеточного процесса, тогда как в биологических экспериментах участвует культура, состоящая из миллионов клеток, так что результаты усредняются естественным образом. В экспериментах *in silico* такого усреднения можно добиться только путем многократного запуска модели. Поэтому, несмотря на довольно высокую скорость отдельного моделирования, получение результатов сопоставимой с биологическими экспериментами точности требует значительного времени, сократить которое можно за счет распараллеливания расчетов на вычислительном кластере, для чего используется вариант программы cpRivals с интерфейсом командной строки. Вариант программы Rivals с GUI удобно использовать при подготовке исходных данных, начальном планировании расчетов и детальном анализе динамики процесса. При отсутствии кластера этот вариант программы может использоваться и для расчетов в пакетном режиме, с распараллеливанием средствами ОС Windows (за счет многопоточности и посредством запуска нескольких копий программы), хотя, разумеется, быстродействие этого варианта меньше, чем у программы cpRivals с интерфейсом командной строки.

Принципиальное отличие описываемой версии программы от предыдущих версий состоит в возможности моделирования не только линейно организованного локуса, искусственно вычлененного из макромолекулы ДНК, но и с полной макромолекулой, в том числе замкнутой в виде кольца. Такое увеличение размерности исходных данных потребовало радикально повысить производительность программы. В ходе профилирования работы программы на реальных данных были определены её узкие места (главным образом – обслуживание очереди событий) и соответственно переработаны структура промежуточных данных и программный код. **В результате на типичных исходных данных работа графического варианта программы ускорилась в среднем в 2 раза, а варианта с командной строкой – в 3.7 раза по сравнению с предыдущей выпущенной версией.**

1.3. Объектная модель программы

Эффективное использование программы невозможно без хотя бы поверхностного представления об агентах, действующих в модели. Каждый из них более или менее точно соответствует реальному биологическому объекту или явлению, однако ниже дается лишь формальное описание, в котором биологический аспект изложен в упрощенной форме; за детальным описанием следует обращаться к биологической литературе. Описание некоторых объектов, свойств и методов, предназначенных исключительно для поддержки функционирования программы (например, для динамического изменения параметров модели по ходу траектории моделирования, для усреднения результатов по пучку траекторий и т.п.), не приводится.

1.3.1. Последовательность ДНК

Это единственный статический объект в модели, который не меняется на протяжении всего моделирования. Содержательно это строка символов, обозначающих последовательность нуклеотидов изучаемого участка ДНК (только одной цепи, которую будем считать основной; комплементарная цепь восстанавливается автоматически). Используется расширенный алфавит IUPAC-IUB, регистр букв безразличен. Символы пробела, табуляции, перевода строки игнорируются; также пропускаются применяемые символы вставки при выравнивании: дефис, подчеркивание, знак равенства и звездочка. С помощью функций GUI последовательность ДНК можно импортировать в файл задачи из обычного текстового файла, а также непосредственно из файла ГенБанка NCBI (.gbk).

Отображение в GUI: Прямо не отображается. Условно представляет собой ось, на которой располагаются отображаемые объекты (с переносом на последующие строки, если ширина главного окна программы недостаточна для отображения всех объектов в одну строку). В случае кольцевой последовательности подразумевается, что её конец соединён с началом. Направление оси вправо соответствует ориентации 5'→3' основной цепи ДНК, комплементарная цепь имеет противоположное направление.

Свойства объекта:

1) Length – длина последовательности.
2) Origin – начало отсчета; в текущей версии всегда Origin=1.
3) Circular – для кольцевой последовательности значение совпадает с её длиной (Length), для линейной последовательности (в т.ч. отрезка кольца) значение равно 0.

Методы:

1) Import – получение последовательности ДНК из текстового файла .txt или .gbk.

Примечание: В текущей версии программы нуклеотидный состав ДНК-последовательности не учитывается; она используется только для привязки прочих объектов к конкретным позициям последовательности. В будущих версиях программы это может быть изменено.

1.3.2. Ген

Неподвижный объект. Фактически это отмеченный участок ДНК-цепи, которому приписаны несколько счетчиков.

Отображение в GUI:

Ген на основной цепи ДНК

Ген на комплементарной цепи ДНК

(Здесь и далее показаны цвета, которые программа Rivals присваивает по умолчанию. Пользователь может по желанию менять цвет заливки, однако начертание символов остается неизменным).

Свойства:

1) Name – имя (идентификатор) гена. (Здесь и далее для возможности работы с обоими вариантами программы длина имени не должна превышать 31 символа; для полного отображения имени в GUI рекомендуется выбирать более короткие имена. При работе в программе в пакетном режиме действуют дополнительные ограничения, описанные ниже в разделе 2.3.2).
2) Strand = {true|false} – признак цепи ДНК, на которой находится ген; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – позиция в основной цепи самого левого нуклеотида гена (если ген на основной цепи, то это его 5'-начало, если на комплементарной – 3'-конец).
4) Right (алиас End) – позиция в основной цепи самого правого нуклеотида гена (если ген на основной цепи, то это его 3'-конец, если на комплементарной – 5'-начало).
5) InitP – счетчик инициаций транскрипции гена PEP-полимеразами.
6) InitN – счетчик инициаций транскрипции гена NEP-полимеразами.
7) Initiated – итоговый счетчик инициированных транскрипций гена.
8) TransP – счетчик завершенных транскрипций гена PEP-полимеразами.
9) TransN – счетчик завершенных транскрипций гена NEP-полимеразами.
10) Transcribed – итоговый счетчик завершенных транскрипций гена.

Методы:

1) Reset – сброс всех счетчиков в 0 (в начале каждой траектории).
2) Enter – увеличивается на 1 значение счетчика Initiated. Этот метод применяется, когда любая РНК-полимераза впервые входит в область, занимаемую данным геном, и тем самым инициирует его транскрипцию.
   Примечание: Если промотор располагается внутри или в непосредственной близости от начала области гена, то значение этого счётчика следует интерпретировать с осторожностью: в первом случае оно будет занижено, т.к. полимераза не пересекает начало области, во втором – может быть завышено из-за неоднократного входа в область гена в ходе абортивного процесса. В таких ситуациях рекомендуется искусственно обрезать область гена, чтобы избежать пространственного наложения процессов связывания полимеразы с промотором и начала транскрипции гена.
3) Leave – метод применяется, когда любая полимераза впервые покидает область, занимаемую геном, и тем самым его транскрипция закончена. В зависимости от выбранного режима программы, ведется либо общий подсчет числа транскрипций (в этом случае увеличивается на 1 значение счетчика Transcribed), либо раздельно подсчитывается число транскрипций гена PEP- и NEP-полимеразами (тогда метод состоит в увеличении на 1 значения счетчика, соответственно, TransP или TransN).

Примечания:

а) Поскольку РНК-полимераза занимает на цепи ДНК более одного нуклеотида, в программе предусмотрен выбор между двумя способами учета входа/выхода полимеразы в/из область гена: либо момент входа фиксируется, когда полимераза целиком оказывается в области гена (а выхода – когда полимераза полностью покидает область гена), либо момент входа/выхода определяется одной фиксированной позицией полимеразы – её центром транскрипции (режим по умолчанию). Выбранный режим хранится в конфигурации и одинаково применяется для всех однотипных полимераз и всех генов.

б) Метод Leave считается успешно применённым лишь в том случае, если сдвиг полимеразы за пределы области гена реально происходит. Если же при сдвиге возникает коллизия, в результате которой полимераза уничтожается, то факт окончания транскрипции гена не будет зафиксирован. Поэтому если терминатор или репрессор расположен вплотную к области гена, рекомендуется укоротить ген, чтобы их разделяла хотя бы одна свободная позиция.

1.3.3. PEP-промотор

Неподвижный объект, расположенный на том участке цепи ДНК, который занимает холофермент (комплекс из сигма-субъединицы и кор-фермента) в случае успешного связывания с ДНК. Связывание возможно только в том случае, когда весь этот участок свободен от прочих динамических объектов (в частности, холоферментов, полимераз и репрессоров). Попытки связывания холофермента происходят с некоторой частотой, зависящей от качества промотора и других факторов. В модели принимается, что это пуассоновский процесс с заданной постоянной интенсивностью λ, которая есть параметр данного промотора. Этот процесс моделируется независимо от других процессов и от успеха связывания, начиная с момента запуска модели. В случае успешного связывания образуется новый объект – холофермент (см. п. 1.3.4).

Отображение в GUI:

PEP-промотор на основной цепи ДНК

PEP-промотор на комплементарной цепи ДНК

Свойства:

1) Name – имя (идентификатор) промотора.
2) Strand = {true|false} – признак цепи ДНК, на которой находится промотор; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – позиция в основной цепи самого левого нуклеотида холофермента сразу после посадки (если промотор на основной цепи, то это его 5'-начало, если на комплементарной – 3'-конец).
4) Right (алиас End) – позиция в основной цепи самого правого нуклеотида холофермента (если промотор на основной цепи, то это его 3'-конец, если на комплементарной – 5'-начало).
5) Tcenter (алиас TrPoint) – позиция в основной цепи центра транскрипции холофермента (совпадает с центром транскрипции его кора) сразу после посадки.

Примечание: GUI позволяет задавать положение центра транскрипции PEP-промотора либо в виде абсолютного номера позиции от начала отсчета последовательности, либо как величину смещения относительно 5'-начала ближайшего гена на той же цепи, где находится промотор. Положение краёв промотора задаётся косвенно с помощью длин полимеразы и сигма-субъединицы. При любом способе указания позиций в конечном счёте они преобразуются в номера нуклеотидов последовательности.

6) Sigma – тип сигма-субъединицы холоферментов, связывающихся с данным PEP-промотором. Поскольку предусмотрено только одно значение, то для реальных промоторов, зависящих сразу от нескольких сигма-субъединиц, в модель на одной и той же или близких позициях вводится соответствующее число отдельных PEP-промоторов с разными субъединицами, каждый со своим значением интенсивности; общая интенсивность промотора равна сумме этих отдельных значений (они могут быть не обязательно одинаковыми, отражая различия концентрации разных сигма-субъединиц). См. также п. 5.4.
7) Lambda – значение λ интенсивности связывания для данного промотора. Поскольку для пуассоновского процесса она равна математическому ожиданию частоты событий, то ее размерность в данной модели [c^{-1}].

8) TouchTime – момент времени следующей попытки связывания холофермента. (Здесь и далее модельное (т.н. «физическое») время отсчитывается от момента запуска модели). Поскольку процесс считается пуассоновским, то интервал времени между событиями моделируется по формуле $\Delta t = -(1/\lambda)\ln\xi$, где ξ – случайная величина, распределенная равномерно на отрезке (0, 1).
9) Offered – счетчик попыток связывания холофермента с этим промотором.
10) Bound – счетчик успешных попыток связывания холофермента (при которых сайт посадки был свободен).

Методы:

1) Reset – сброс всех счетчиков в 0 (в начале каждой траектории).
2) Touch – вычисление нового значения TouchTime.
3) Offer – увеличивается на 1 значение счетчика Offered. Метод применяется при каждой попытке связывания холофермента, т.е. в момент TouchTime.
4) Bind – увеличивается на 1 значение счетчика Bound. Метод применяется при успешном связывании холофермента, т.е. когда в момент времени TouchTime на участке ДНК с позициями от Left до Right (на обеих цепях!) нет других объектов.

1.3.4. Холофермент

Это объект с неподвижным 5'-концом, и ограниченно подвижным 3'-концом. Холофермент возникает на ДНК в результате успешного связывания с PEP-промотором (п. 1.3.3) и состоит из двух объектов – сигма-субъединицы (на 5'-конце холофермента) и кор-фермента (на 3'-конце). Сигма-субъединица неподвижна все время, пока существует холофермент, а кор совершает т.н. абортивные движения: отходит от сигма-субъединицы, пока промежуток между ними не достигнет заданного числа нуклеотидов *r*, после чего или мгновенным скачком возвращается в начальное положение, или же продолжает движение, окончательно отрываясь от сигма-субъединицы. В последнем случае он превращается в новый объект модели – кор (см. п. 1.3.5), а холофермент прекращает существовать, освобождая занятое сигма-субъединицей место на позициях PEP-промотора. Амплитуда *r* абортивных процессов считается постоянной, а число повторений (перед окончательным отделением кора) – случайной величиной, имеющей пуассоновское распределение с заданной интенсивностью μ. Скорость элонгации кор-фермента *R* во время абортивного процесса принимается той же, что и при транскрипции гена.

Холофермент существует либо до окончания абортивного процесса (после чего уничтожается, порождая кор), либо до столкновения с другим подвижным объектом (коллизии), при котором остается или уничтожается согласно заданным правилам разрешения коллизий (см. п. 1.4.1).

Отображение в GUI: не отображается.

Свойства:

1) Name – имя (идентификатор) холофермента, которое имеет вид H<pepname>#<num>, где <pepname> – имя PEP-промотора, <num> – порядковый номер успешной посадки холофермента на этот промотор.
2) Strand = {true|false} – признак цепи ДНК, на которой находится холофермент; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – текущая позиция в основной цепи самого «левого» нуклеотида холофермента (здесь и далее *левым* называется 5'-край объекта на основной цепи или 3'-край объекта на комплементарной цепи).
4) Right (алиас End) – текущая позиция в основной цепи самого «правого» нуклеотида холофермента (здесь и далее *правым* называется 3'-край объекта на основной цепи или 5'-край объекта на комплементарной цепи).
5) Tcenter (алиас TrPoint) – текущая позиция в основной цепи центра транскрипции холофермента (совпадает с центром транскрипции его кора).
6) SigLength – длина сигма-субъединицы холофермента.

7) Shift – текущая величина промежутка между кором и сигма-субъединицей в ходе абортивного процесса (изменяется от 0 до r).
8) Aborting – параметр μ пуассоновского процесса – среднее число абортивных движений до освобождения кора.
9) AbortRange – амплитуда абортивных движений (параметр r).
10) Countback – счётчик оставшихся абортивных движений холофермента. Начальное значение счётчика устанавливается применением метода SetCount. После каждого абортивного движения значение счетчика уменьшается на 1, и когда он обнуляется, абортивный процесс прекращается.
11) Rate – скорость прямого движения кора в абортивном процессе (параметр R), [нт/с].
12) TouchTime – момент времени перехода кор-фермента на следующий нуклеотид. Этот процесс детерминированный, поэтому продолжительность интервала времени между последовательными переходами вычисляется как $\Delta t = 1/R$.

Методы:

1) Reset – сброс абортивного процесса в начальное состояние (после посадки).
2) SetCount – вычисление (розыгрыш) числа повторений абортивного процесса для заданной интенсивности μ. Находится как минимальное значение k, при котором $\prod_{i=1}^{k} \xi_i \le e^{-\mu}$, где ξ_i – независимые последовательные значения случайной величины, равномерно распределенной на интервале (0, 1). Метод применяется однократно при посадке, после чего абортивный процесс становится целиком детерминированным.
3) Touch – вычисление нового значения TouchTime, как было указано при описании этого свойства.
4) ShiftKor – сдвиг кора на 1 нт в направлении 3' цепи ДНК, в пределах амплитуды r абортивных движений.
5) ResetKor – возврат кора в начальное положение, т.е. вплотную к сигма-субъединице. Метод применяется при отходе кор-фермента от сигма-субъединицы на расстояние r нт, пока не будет сделано найденное с помощью метода SetCount число повторений.

1.3.5. Кор-фермент

Это подвижный объект, представляющий собой РНК-полимеразу эубактериального типа. В момент запуска модели кор-ферменты отсутствуют; они порождаются либо холоферментами по окончании абортивных движений (см. п. 1.3.4), либо внешними источниками кор-ферментов (см. п. 1.3.6). Кор-фермент уничтожается при выходе за границы рассматриваемого локуса (кроме случая, когда моделируется вся кольцевая макромолекула ДНК) или в результате столкновения с другим подвижным объектом (коллизии), при котором остается или уничтожается согласно заданным правилам разрешения коллизий (см. п. 1.4.1). Движение кора по цепи ДНК описывается детерминированным процессом.

Отображение в GUI: не отображается.

Свойства:

1) Name – имя (идентификатор) кора. Если кор порожден холоферментом, имя имеет вид K<holoname>, где <holoname> – имя холофермента. Если кор порожден внешним источником, ему присваивается имя вида K<ekorname>#<num>, где <ekorname> – имя внешнего источника, <num> – порядковый номер успешной генерации кор-фермента этим источником (см. п. 1.3.6).
2) Strand = {true|false} – признак цепи ДНК, по которой движется кор-фермент; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – текущая позиция в основной цепи самого левого нуклеотида кора.
4) Right (алиас End) – текущая позиция в основной цепи самого правого нуклеотида кора.
5) Tcenter (алиас TrPoint) – текущая позиция в основной цепи центра транскрипции кора.
6) Rate – скорость движения кора по цепи ДНК (параметр R), [нт/с]. Подчеркнем, что

движение всех объектов по цепям ДНК в данной модели предполагается скачкообразным, т.е. время перехода с одного нуклеотида на другой бесконечно малое.

7) TouchTime – момент времени перехода кор-фермента на следующий нуклеотид. Этот процесс детерминированный, поэтому продолжительность интервала времени между последовательными переходами вычисляется по формуле $\Delta t = 1/R$ [c^{-1}].

Методы:

1) Reset – инициализация кора.
2) Touch – вычисление нового значения TouchTime – момента следующего сдвига кора.
3) Shift – сдвиг кора на 1 нт в направлении 3' цепи ДНК (вправо для основной цепи, влево для комплементарной).

1.3.6. Внешний источник кор-ферментов

Это неподвижный объект, который не соответствует никакому реальному биологическому объекту, но может вводиться в модель для компенсации искусственного вычленения локуса из всей ДНК-последовательности. Если есть основания полагать, что транскрипция генов по соседству с локусом сопровождается заметным потоком полимераз, достигающих данного локуса, то это явление можно адекватно представить в модели с помощью такого внешнего источника кор-ферментов, помещаемого вне локуса на одной или обеих цепях ДНК. Поведение внешнего источника в нашей модели описывается пуассоновским процессом с интенсивностью λ (параметр источника). В соответствии с этим процессом, в случайные моменты времени (со средней частотой λ) делается попытка посадки кора на сайт внешнего источника. Если сайт в это время не занят другими подвижными объектами, образуется новый объект – кор-фермент (п. 1.3.5), который в дальнейшем функционирует по своему закону. Если же сайт внешнего источника в этот момент частично или полностью перекрыт другими РНК-полимеразами на любой цепи, то не происходит ничего (неудачная попытка посадки).

Отображение в GUI:

Внешний источник кор-ферментов на основной цепи ДНК

Внешний источник кор-ферментов на комплементарной цепи ДНК

Свойства:

1) Name – имя (идентификатор) источника.
2) Strand = {true|false} – признак цепи ДНК, на которой находится источник; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – позиция в основной цепи самого левого нуклеотида кор-фермента сразу после посадки.
4) Right (алиас End) – позиция в основной цепи самого правого нуклеотида кор-фермента сразу после посадки.
5) Tcenter (алиас TrPoint) – позиция в основной цепи центра транскрипции кор-фермента сразу после посадки. (Способы задания положения внешнего источника те же, что и для PEP-промотора, см. п. 1.3.3).
6) Lambda – значение λ интенсивности связывания для данного источника. Поскольку для пуассоновского процесса она равна математическому ожиданию частоты событий, то ее размерность в данной модели [c^{-1}].
7) TouchTime – момент времени следующей попытки посадки кора. Процесс считается пуассоновским, поэтому интервал времени между попытками моделируется по формуле $\Delta t = -(1/\lambda)\ln\xi$, где ξ – случайная величина, равномерно распределенная на (0, 1).
8) Offered – счетчик попыток посадки кор-фермента от этого источника.
9) Bound – счетчик успешных попыток связывания кор-фермента (при которых сайт посадки был свободен).

Методы:

1) Reset – сброс всех счетчиков в 0 (в начале каждой траектории).
2) Touch – вычисление нового значения TouchTime, как указано выше.
3) Offer – увеличивается на 1 значение счетчика Offered. Метод применяется при каждой попытке посадки кор-фермента, т.е. в момент TouchTime.
4) Bind – увеличивается на 1 значение счетчика Bound. Метод применяется при успешной посадке кор-фермента, т.е. когда в момент времени TouchTime на участке ДНК с позициями от Left до Right (на обеих цепях!) нет других объектов.

1.3.7. NEP-промотор

Неподвижный объект, расположенный на том участке цепи ДНК, который занимает РНК-полимераза фагового типа в случае успешного связывания с ДНК. Связывание возможно только в том случае, когда обе цепи ДНК в пределах всего этого участка свободны от прочих динамических объектов (в частности, холоферментов, полимераз и репрессоров). Попытки связывания полимеразы происходят с некоторой частотой, зависящей от качества промотора и других факторов. В модели принимается, что это пуассоновский процесс с заданной постоянной интенсивностью λ, которая есть параметр данного промотора. Этот процесс моделируется независимо от других процессов и от успеха связывания, начиная с момента запуска модели. В случае успешного связывания образуется новый подвижный объект – полимераза фагового типа (см. п. 1.3.8).

Отображение в GUI:

NEP-промотор на основной цепи ДНК

NEP-промотор на комплементарной цепи ДНК

Свойства:

1) Name – имя (идентификатор) промотора.
2) Strand = {true|false} – признак цепи ДНК, на которой находится промотор; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – позиция в основной цепи самого левого нуклеотида полимеразы сразу после посадки.
4) Right (алиас End) – позиция в основной цепи самого правого нуклеотида полимеразы сразу после посадки.
5) Tcenter (алиас TrPoint) – позиция в основной цепи центра транскрипции полимеразы сразу после посадки. (Способы задания положения NEP-промотора те же, что и для PEP-промотора, см. п. 1.3.3).
6) Polym – вид полимераз фагового типа, порождаемых данным NEP-промотором. Поскольку предусмотрено только одно значение, то для реальных промоторов, с которыми могут связываться полимеразы сразу нескольких видов, в модель вводится соответствующее число отдельных NEP-промоторов на одних и тех же (или близких) позициях, каждый со своим значением интенсивности; общая интенсивность промотора равна сумме этих отдельных значений (они могут быть не обязательно одинаковыми, отражая различия концентрации разных полимераз фагового типа).
7) Lambda – значение λ интенсивности связывания для данного промотора. Поскольку для пуассоновского процесса она равна матожиданию частоты событий, то ее размерность в данной модели [c^{-1}].
8) TouchTime – момент времени следующей попытки связывания полимеразы с промотором. Процесс считается пуассоновским, поэтому интервал времени между событиями моделируется по формуле $\Delta t = -(1/\lambda)\ln\xi$, где ξ – случайная величина, распределенная равномерно на отрезке (0, 1).
9) Offered – счетчик попыток связывания полимеразы с этим промотором.
10) Bound – счетчик успешных попыток связывания полимеразы (при которых сайт посадки был свободен).

Методы:

1) Reset – сброс всех счетчиков в 0 (в начале каждой траектории).
2) Touch – вычисление нового значения TouchTime.
3) Offer – увеличивается на 1 значение счетчика Offered. Метод применяется при каждой попытке связывания полимеразы, т.е. в момент TouchTime.
4) Bind – увеличивается на 1 значение счетчика Bound. Метод применяется при успешном связывании полимеразы, т.е. когда в момент времени TouchTime на участке ДНК с позициями от Left до Right (на обеих цепях!) нет других объектов.

1.3.8. Полимераза фагового типа

Это подвижный объект, представляющий собой РНК-полимеразу фагового типа. В момент запуска модели полимеразы отсутствуют; они порождаются либо NEP-промоторами (см. п. 1.3.7), либо внешними источниками полимераз фагового типа (см. п. 1.3.9). Полимераза уничтожается при выходе за границы рассматриваемого локуса (кроме случая моделирования полной кольцевой макромолекулы ДНК) или в результате столкновения с другим подвижным объектом (коллизии), при котором остается или уничтожается согласно заданным правилам разрешения коллизий (см. п. 1.4.1). Движение полимеразы по цепи ДНК описывается детерминированным процессом.

Отображение в GUI: не отображается.

Свойства:

1) Name – имя (идентификатор) полимеразы фагового типа. Если она порождена NEP-промотором, имя имеет вид R<nepname>#<num>, где <nepname> – имя промотора, а <num> – порядковый номер успешного связывания полимеразы с этим промотором. Если полимераза порождена внешним источником, ей присваивается имя вида R<erponame>#<num>, где <erponame> – имя внешнего источника, <num> – порядковый номер успешной генерации полимеразы этим источником (см. п. 1.3.9).
2) Strand = {true|false} – признак цепи ДНК, по которой движется полимераза; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – текущая позиция в основной цепи самого левого нуклеотида полимеразы фагового типа.
4) Right (алиас End) – текущая позиция в основной цепи самого правого нуклеотида полимеразы фагового типа.
5) Tcenter (алиас TrPoint) – текущая позиция в основной цепи центра транскрипции полимеразы.
6) Rate – скорость движения полимеразы по цепи ДНК (параметр R), [нт/с].
7) TouchTime – момент времени перехода полимеразы на следующий нуклеотид. Этот процесс детерминированный, поэтому продолжительность интервала времени между последовательными переходами вычисляется по формуле $\Delta t = 1/R$ [$с^{-1}$].

Методы:

1) Reset – инициализация полимеразы фагового типа.
2) Touch – вычисление нового значения TouchTime – момента следующего сдвига полимеразы.
3) Shift – сдвиг полимеразы на 1 нт в направлении 3' цепи ДНК (вправо для основной цепи, влево для комплементарной).

1.3.9. Внешний источник полимераз фагового типа

Это неподвижный объект, который не соответствует никакому реальному биологическому объекту, но может вводиться в модель для компенсации искусственного вычленения локуса из всей ДНК-последовательности. Если есть основания полагать, что транскрипция генов по соседству с локусом сопровождается заметным потоком полимераз фагового типа, достигающих данного локуса, то это явление можно адекватно представить в модели с помощью такого внешнего источника полимераз, помещаемого вне локуса на одной или

обеих цепях ДНК. Поведение внешнего источника в нашей модели описывается пуассоновским процессом с интенсивностью λ (параметр источника). В соответствии с этим процессом, в случайные моменты времени (со средней частотой λ) делается попытка посадки полимеразы фагового типа на сайт внешнего источника. Если сайт в это время не занят другими подвижными объектами, образуется новый объект – полимераза фагового типа (п. 1.3.8), который в дальнейшем функционирует по своему закону. Если же сайт внешнего источника в этот момент частично или полностью перекрыт другими полимеразами на любой цепи, то не происходит ничего (неудачная попытка посадки).

Отображение в GUI:

Внешний источник полимераз фагового типа на основной цепи ДНК

Внешний источник полимераз фагового типа на комплементарной цепи ДНК

Свойства:

1) Name – имя (идентификатор) источника.
2) Strand = {true|false} – признак цепи ДНК, на которой находится источник; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – позиция в основной цепи самого левого нуклеотида полимеразы сразу после посадки.
4) Right (алиас End) – позиция в основной цепи самого правого нуклеотида полимеразы сразу после посадки.
5) Tcenter (алиас TrPoint) – позиция в основной цепи центра транскрипции полимеразы сразу после посадки. (Способы задания положения внешнего источника те же, что и для PEP-промотора, см. п. 1.3.3).
6) Lambda – значение λ интенсивности связывания для данного источника. Поскольку для пуассоновского процесса она равна математическому ожиданию частоты событий, то ее размерность в данной модели [c^{-1}].
7) TouchTime – момент времени следующей попытки посадки полимеразы. Процесс считается пуассоновским, поэтому интервал времени между попытками моделируется по формуле $\Delta t = -(1/\lambda)\ln\xi$, где ξ – случайная величина, распределенная равномерно на отрезке (0, 1).
8) Offered – счетчик попыток посадки полимеразы фагового типа для этого источника.
9) Bound – счетчик успешных попыток посадки полимеразы (при которых сайт посадки был свободен).

Методы:

1) Reset – сброс всех счетчиков в 0 (в начале каждой траектории).
2) Touch – вычисление нового значения TouchTime, как указано выше.
3) Offer – увеличивается на 1 значение счетчика Offered. Метод применяется при каждой попытке посадки полимеразы, т.е. в момент TouchTime.
4) Bind – увеличивается на 1 значение счетчика Bound. Метод применяется при успешной посадке полимеразы, т.е. когда в момент времени TouchTime на участке ДНК с позициями от Left до Right (на обеих цепях) нет других объектов.

1.3.10. Репрессор

Это неподвижный объект, который с заданной интенсивностью λ возникает на фиксированном участке ДНК (сайт репрессии). Данный объект может моделировать связывание с ДНК белка-репрессора, блокирующего посадку полимераз или транскрипцию гена, либо играющего роль терминатора транскрипции. Связывание репрессора происходит только в том случае, если сайт посадки свободен от любых подвижных объектов, а также от ранее связавшегося репрессора. Действие репрессора распространяется на обе цепи ДНК. Если сайт репрессии частично перекрывает какой-либо промотор, то после возникновения репрессора полностью прекращается связывание любых полимераз с этим промотором. Даже

если репрессор не пересекается с промоторами, он сбрасывает любые полимеразы, когда они достигают сайта репрессии (и тем самым препятствует транскрипции генов или осуществляет её терминацию). Сам репрессор при этом также уничтожается, что соответствует отделению белка-репрессора от ДНК (подробнее о правилах разрешения коллизий см. п. 1.4.1).

Примечание: Начиная с версии 1.5.x.x, функциональность репрессора усовершенствована, так что он допускает и другую биологическую интерпретацию, а именно, в виде терминатора (скажем, шпильки) с возможностью частичного протекания полимераз. Когда полимераза встречает такой объект, возможны два сценария взаимодействия: (а) полимераза прекращает движение и срывается, а репрессор остаётся или уничтожается в соответствии с правилами разрешения коллизий; и (б) элонгация продолжается, а репрессор уничтожается (шпилька «расплетается»), т.е. имеет место протекание через терминатор. Это в равной мере относится к движению полимераз по обеим цепям ДНК, но относительная вероятность сценария (б) устанавливается независимо для каждой цепи, что позволяет моделировать поляризацию такого терминатора.

Отображение в GUI:

 Репрессор/терминатор на обеих цепях ДНК

Свойства:

1) Name – имя (идентификатор) репрессора.
2) Left (алиас Begin) – позиция в основной цепи самого левого нуклеотида репрессора.
3) Right (алиас End) – позиция в основной цепи самого правого нуклеотида репрессора.
4) Lambda – значение λ интенсивности появления данного репрессора. Поскольку для пуассоновского процесса она равна математическому ожиданию частоты событий, то ее размерность в данной модели [c^{-1}].
5) TouchTime – момент времени следующей попытки появления репрессора. Процесс считается пуассоновским, поэтому интервал времени между событиями моделируется по формуле $\Delta t = -(1/\lambda)\ln\xi$, где ξ – случайная величина, распределенная равномерно на отрезке (0, 1).
6) Offered – счетчик попыток связывания репрессора.
7) Bound – счетчик успешных попыток связывания репрессора.
8) Qmain – вероятность протекания терминатора по основной цепи ДНК.
9) Qcomp – вероятность протекания терминатора по комплементарной цепи.

Примечание. Значения Qmain, Qcomp определяют для каждой цепи долю случаев, когда взаимодействие идёт по описанному выше сценарию (б). Если обе эти вероятности нулевые, то репрессор всегда действует по сценарию (а), т.е. как в прежних версиях программы.

10) Mterminated – счётчик полимераз, которые терминировали, двигаясь по основной цепи.
11) Cterminated – счётчик полимераз, которые терминировали на комплементарной цепи.

Методы:

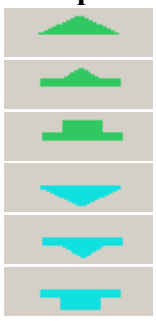
1) Reset – сброс всех счетчиков в 0 (в начале каждой траектории).
2) Touch – вычисление нового значения TouchTime.
3) Offer – увеличивается на 1 значение счетчика Offered. Метод применяется при каждой попытке появления репрессора, т.е. в момент TouchTime.
4) Bind – увеличивается на 1 значение счетчика Bound. Метод применяется при успешном появлении репрессора, т.е. когда в момент времени TouchTime на участке ДНК с позициями от Left до Right (на обеих цепях!) нет других объектов.
5) Pass – реализуется случайный (в соответствии с величиной Qmain/Qcomp) выбор одного из вышеупомянутых сценариев взаимодействия РНК-полимеразы и репрессора в момент их столкновения.

1.3.11. Терминатор

Это неподвижный объект, который с заданной интенсивностью λ возникает на фиксированном участке ДНК. Объект моделирует процесс срыва полимеразы с ДНК по окончании транскрипции гена. Этот объект во многом аналогичен репрессору, но его действие распространяется только на полимеразы, движущиеся по той же самой цепи ДНК. Поскольку возможны различные молекулярно-биологические механизмы терминации, и этот процесс еще не полностью изучен, в модели предусмотрены терминаторы различного действия, в том числе влияющие избирательно на различные типы полимераз. За отсутствием детальных данных, в модели предполагается, что терминатор возникает в соответствии с пуассоновским процессом, после чего сбрасывает все полимеразы соответствующего типа, которые находятся в этом сайте терминации (или достигают его). Сам терминатор при этом остается или тоже уничтожается, в зависимости от заданных правил разрешения коллизий (подробнее см. п. 1.4.1).

Подчеркнем, что в отличие от ранее описанного репрессора, терминатор может возникать и в том случае, когда на его сайте располагаются подвижные объекты (например, полимеразы, которые при этом будут сброшены, если они находятся на той же цепи и терминатор чувствителен к их типу). Однако повторное возникновение терминатора (если он уже существует) запрещено, чтобы можно было подсчитать число реальных событий терминации. Поэтому не следует пытаться моделировать симметричный терминатор для двух цепей с помощью пары терминаторов с одним и тем же сайтом; для этой цели предназначен другой объект – репрессор (см. п. 1.3.10).

Отображение в GUI:



Терминатор для всех полимераз на основной цепи ДНК

Терминатор для кор-ферментов на основной цепи ДНК

Терминатор для полимераз фагового типа на основной цепи ДНК

Терминатор для всех полимераз на комплементарной цепи ДНК

Терминатор для кор-ферментов на комплементарной цепи ДНК

Терминатор для полимераз фагового типа на комплементарной цепи ДНК

Свойства:

1) Name – имя (идентификатор) терминатора.
2) Strand = {true|false} – признак цепи ДНК, на которой находится терминатор; значение true соответствует основной цепи, false – комплементарной.
3) Left (алиас Begin) – позиция в основной цепи самого левого нуклеотида терминатора.
4) Right (алиас End) – позиция в основной цепи самого правого нуклеотида терминатора.
5) Lambda – значение λ интенсивности появления данного терминатора. Поскольку для пуассоновского процесса она равна математическому ожиданию частоты событий, то ее размерность в данной модели [c^{-1}].
6) TouchTime – момент времени следующей попытки появления терминатора. Процесс считается пуассоновским, поэтому интервал времени между событиями моделируется по формуле $\Delta t = -(1/\lambda)\ln\xi$, где ξ – случайная величина, распределенная равномерно на отрезке (0, 1).
7) SenseKor = {true|false} – признак того, что терминатор действует на кор-ферменты (если значение равно true).
8) SenseRpo = {true|false} – признак того, что терминатор действует на полимеразы фагового типа (если значение равно true).
9) Offered – счетчик попыток появления терминатора.
10) Bound – счетчик успешных попыток появления терминатора (когда сайт посадки не занят ранее образовавшимся терминатором).

Методы:

1) Reset – сброс всех счетчиков в 0 (в начале каждой траектории).
2) Touch – вычисление нового значения TouchTime.
3) Offer – увеличивается на 1 значение счетчика Offered. Метод применяется при каждой попытке появления терминатора, т.е. в момент TouchTime.
4) Bind – увеличивается на 1 значение счетчика Bound. Метод применяется при успешном появлении терминатора, т.е. когда в момент времени TouchTime на этом участке нет терминатора. (Заметим, что если на этом участке данной цепи ДНК находятся полимеразы того типа, к которым чувствителен данный терминатор, то это не препятствует посадке – все полимеразы будут уничтожены, после чего вновь образованный терминатор останется или будет удален, как того требуют заданные правила разрешения коллизий; см. п. 1.4.1).

1.4. Алгоритм работы модели

До начала собственно моделирования должны быть подготовлены файлы конфигурации и задачи, а для моделирования в пакетном режиме – еще и файл задания. Форматы этих файлов описаны в главе 4, а возможности их создания и/или коррекции внутри самой программы – в разделе 2.3.

Работа модели начинается с инициализации датчика псевдослучайной последовательности и пуска внутренних часов, отсчитывающих моделируемое физическое время (напомним, что это неравномерное дискретное время, отсчёты которого соответствуют событиям в модели). В этот момент никаких подвижных объектов еще нет, и, следовательно, коллизии невозможны. Затем к каждому имеющемуся в задаче динамическому объекту применяется метод Reset (начальный сброс), а затем Touch (см. раздел 1.3), чтобы определить момент времени TouchTime, когда этот объект должен измениться (например, для промотора – когда произойдет первая попытка связывания РНК-полимеразы). При этом разыгрывается соответствующий случайный процесс, который управляет действием данного объекта. Найденные моменты времени TouchTime упорядочиваются по возрастанию в единую очередь событий, основанную на алгоритме двоичной кучи, и дальше работа модели складывается из типовых итераций. Поскольку время вычисляется аналитически и представляется непрерывной плавающей величиной с двойной точностью, вероятность совпадения двух моментов времени для случайных процессов (типа пуассоновского) пренебрежимо мала, и ее можно не принимать в расчет. Даже если на каком-то этапе процесс становится детерминированным (например, движение полимераз с постоянной скоростью по цепи ДНК), это не меняет картины, поскольку начальный момент каждого такого детерминированного процесса является случайным.

На каждой итерации из очереди выбирается первый (т.е. ближайший) момент времени и реализуется намеченное в этот момент событие для соответствующего объекта, в соответствии с реализованной в нём внутренней логикой и правилами разрешения коллизий (когда подвижный объект должен перейти на позицию, уже занятую другим объектом). Опишем эти действия подробнее для каждого класса объектов.

РЕР-промотор: Если сайт промотора свободен от других объектов на обеих цепях ДНК, то на нем размещается новый объект – **холофермент**, который инициализируется последовательным применением методов Reset, SetCount и Touch, после чего помещается в очередь согласно значению свойства TouchTime. Затем (как и в случае, когда сайт промотора занят) к промотору применяется метод Touch и он передвигается в очереди на новое место в соответствии со своим значением TouchTime.

Холофермент: Если кор холофермента еще не отошел от сигма-субъединицы на величину максимальной амплитуды абортивного процесса *r*, то применением метода ShiftKor он сдвигается еще на один нуклеотид в направлении 3'. Если максимальный сдвиг Shift уже достигнут, то проверяется, какое это по счету повторение абортивного процесса. Если еще не все запланированные (с помощью метода SetCount) повторения абортивного процесса проделаны, т.е. свойство Countback имеет ненулевое значение, то применением метода ResetKor кор-фермент возвращается на исходную позицию – вплотную к сигма-субъединице. Во всех вышеперечисленных случаях после этого к холоферменту применяется метод Touch, и он передвигается в очереди на новое место в соответствии со своим значением TouchTime. По завершении запланированного числа повторений абортивного процесса холофермент уничтожается: сигма-субъединица отделяется от ДНК, а кор становится новым самостоятельным объектом – **кор-ферментом** на достигнутой к этому моменту позиции, и инициализируется последовательным применением методов Reset и Touch, после чего помещается в очередь согласно значению его свойства TouchTime. Во время существования холофермента с ним могут происходить коллизии, когда кор передвигается на позицию, уже занятую другим объектом на любой цепи ДНК, или же когда во время паузы в движении кора на занятую холоферментом позицию пытается передвинуться другой подвижный объект. Правила разрешения подобных коллизий рассматриваются в разделе 1.4.1; они, в частности, могут предусматривать преждевременное отделение (т.е. уничтожение) холофермента.

Кор-фермент: Если соседняя с кор-ферментом позиция в направлении 3' цепи ДНК свободна, то к нему применяется метод Shift для сдвига на 1 нт и затем Touch, после чего он передвигается в очереди на новое место в соответствии со своим значением TouchTime. Если соседняя позиция занята, т.е. имеет место коллизия, то действия выполняются по правилам, описанным в разделе 1.4.1. Коллизии могут происходить и во время паузы в движении кора, когда на занятую им позицию пытается передвинуться другой подвижный объект. В зависимости от заданных правил, в случае коллизии кор может уничтожить конкурирующий объект, или/и самоуничтожиться, или/и остановиться, или же продолжать движение, как в отсутствие конкуренции. В процессе движения каждого кор-фермента фиксируются моменты, когда он входит в (или покидает) область, занимаемую геном на той же цепи ДНК; в таких случаях к этому гену применяется, соответственно, метод Enter или Leave для модификации соответствующих счетчиков.

Внешний источник кор-ферментов: Если сайт источника свободен от других динамических объектов на обеих цепях ДНК, то на нем размещается новый объект – **кор-фермент**, который инициализируется последовательным применением методов Reset и Touch, после чего помещается в очередь согласно значению свойства TouchTime. Затем (как и в случае, когда сайт внешнего источника был занят) к источнику применяется метод Touch и он передвигается в очереди на новое место в соответствии со своим значением TouchTime.

NEP-промотор: Если сайт промотора свободен от других объектов на обеих цепях ДНК, то на нем размещается новый объект – **полимераза фагового типа**, который инициализируется последовательным применением методов Reset и Touch, после чего помещается в очередь согласно значению свойства TouchTime. Затем (как и в случае, когда сайт промотора был занят) к промотору применяется метод Touch и он передвигается в очереди на новое место в соответствии со своим значением TouchTime.

Полимераза фагового типа: Если соседняя с полимеразой позиция в направлении 3' цепи ДНК свободна, то к этому объекту применяется метод Shift для сдвига на 1 нт и затем Touch, после чего она передвигается в очереди на новое место в соответствии со своим значением TouchTime. Если соседняя позиция занята, т.е. имеет место коллизия, то действия

выполняются по правилам, описанным в разделе 1.4.1. Коллизии могут происходить и во время паузы в движении полимеразы фагового типа, когда на занятую ей позицию пытается передвинуться другой подвижный объект. В зависимости от заданных правил, в случае коллизии полимераза может уничтожить конкурирующий объект, или/и самоуничтожиться, или/и остановиться, или же продолжать движение, как в отсутствие конкуренции. В процессе движения каждой полимеразы фагового типа фиксируются моменты, когда она входит в (или покидает) область, занимаемую геном на той же цепи ДНК; в таких случаях к этому гену применяется, соответственно, метод Enter или Leave для модификации соответствующих счетчиков.

Внешний источник полимераз фагового типа: Если сайт источника свободен от других динамических объектов на обеих цепях ДНК, то на нем размещается новый объект – **полимераза фагового типа**, которая инициализируется последовательным применением методов Reset и Touch, после чего помещается в очередь согласно значению свойства TouchTime. Затем (как и в случае, когда сайт внешнего источника занят) к источнику применяется метод Touch и он передвигается в очереди на новое место в соответствии со своим значением TouchTime.

Репрессор: При запуске модели все репрессоры инициализируются последовательным применением методов Reset и Touch, после чего помещаются в очередь в соответствии со значением свойства TouchTime. Если в очередной момент времени TouchTime сайт репрессии свободен от других динамических объектов на обеих цепях ДНК (в том числе и от ранее активизировавшегося репрессора), то он активизируется, т.е. считается занятым (белком-репрессором или терминирующей структурой). Затем (как и в случае, когда сайт репрессии был занят) к репрессору применяется метод Touch и он передвигается в очереди на новое место в соответствии с новым значением TouchTime. Таким образом, эффект от активизации репрессора состоит в том что все позиции сайта репрессии становятся занятыми на обеих цепях ДНК, и при попытке перехода на них других объектов возникает коллизия, которая разрешается по общим правилам, описанным в разделе 1.4.1, но с учётом новой специфики репрессора, допускающей протекание с заданной независимо для каждой цепи вероятностью (подробнее об этом см. п. 1.3.10.

Терминатор: При запуске модели все терминаторы инициализируются последовательным применением методов Reset и Touch, после чего помещаются в очередь в соответствии со значением свойства TouchTime. Если в очередной момент времени TouchTime сайт терминации свободен от ранее активизировавшегося терминатора на той же самой цепи ДНК, то он активизируется, т.е. становится занятым, и в дальнейшем вход полимеразы на сайт терминации вызывает коллизию. Побочный эффект активизации терминатора состоит в том, что все РНК-полимеразы, полностью или частично уже находящиеся в пределах сайта терминации на той же цепи, срываются с нее, прекращая тем самым дальнейшую транскрипцию. Если терминатор чувствителен к какому-то одному типу полимераз, т.е. кор-ферменту или полимеразе фагового типа, то вышесказанное относится только к таким полимеразам. Сам терминатор после коллизии остается в активном состоянии или уничтожается в соответствии с заданными правилами разрешения коллизий (см. п. 1.4.1). Если при первоначальной активизации терминатор не был уничтожен, к нему применяется метод Touch и он передвигается в очереди на новое место в соответствии с новым значением TouchTime.

В программе предусмотрена возможность не учитывать начальный этап работы модели, пока она входит в установившийся (в некотором смысле) режим. В этом случае, при достижении заданной продолжительности физического времени с момента запуска, счетчики всех объектов (в частности, счетчики транскрипций генов) обнуляются, однако все подвижные

объекты сохраняются и модель далее работает, как описано выше. Тем самым можно ослабить влияние того факта, что все процессы в модели запускаются в некоторый единый момент времени («начало жизни»), а до того не протекали, что по-видимому не вполне адекватно биологии клетки. (Побочным эффектом от использования такой возможности может быть небольшое занижение общего числа инициированных транскрипций, поскольку в нем не будут учтены те РНК-полимеразы, которые к моменту окончания «разгона» модели уже находятся на участках, занятых генами).

Начиная с версии 1.5.x.x, в описываемую программу внесено существенное изменение – возможность работы с полной кольцевой последовательностью макромолекулы ДНК, а не только с линейной цепочкой, отвечающей отдельному локусу этой последовательности. В этом режиме моделирования, с формальной точки зрения, имеются всего два отличия:
1) номер позиции левого края объекта может быть больше номера позиции правого края; и
2) подвижные объекты, выходящие за любой из краёв последовательности, не уничтожаются, как в случае линейного локуса, а появляются на другом краю последовательности.
Разумеется, увеличение длины цепи и числа динамических объектов в таком режиме неизбежно приводит к замедлению моделирования, однако опыт показывает, что модель может с успехом применяться для моделирования на кольцевых последовательностях длиной в десятки тысяч нуклеотидов, таких, например, как митохондриальный геном человека [2].

1.4.1. Правила разрешения коллизий

Когда подвижный объект (которому соответствует реальная молекула) занимает некоторый участок цепи ДНК, все нуклеотиды этого участка оказываются вовлечены в сложные физико-химические связи с этой молекулой и, как правило, не могут участвовать в других подобных связях. Более того, размеры таких макромолекул (в частности, РНК-полимераз) часто настолько велики, что они блокируют и соседнюю цепь ДНК, не позволяя связываться другим объектам с ее нуклеотидами. Поэтому в нашей модели ситуация, когда два подвижных объекта претендуют на связь с одной и той же позицией ДНК, рассматривается как потенциальная коллизия, подлежащая разрешению. Детали конкурентных физико-химических процессов, происходящих при таких коллизиях, еще недостаточно хорошо изучены, поэтому правила действий не заложены жёстко в программу, а являются важным параметром ее конфигурации, который при необходимости можно изменить. Для настройки файла конфигурации рекомендуется использовать версию с GUI (приложение Rivals); форма, в которой в ней представлены правила разрешения коллизий, описывается ниже.

Как уже говорилось, модельное (физическое) время протекает неравномерно и дискретно, от очередного события к следующему, причем одновременные события крайне маловероятны. Отсюда следует, что при каждой коллизии без ограничения общности можно считать, что в ней взаимодействуют только два объекта, из которых один неподвижный («пассивный»), а второй («активный») претендует занять позицию, уже занятую первым объектом. Это позволяет описать всевозможные случаи в виде компактной таблицы, однако ее представление в файле конфигурации не слишком наглядно, почему и рекомендуется вносить любые изменения через GUI (используя в меню приложения пункты **Edit > Options**, закладка **General**). Таблица занимает нижнюю часть этой закладки и по умолчанию имеет следующий вид (Рис. 1).

Опишем подробнее содержание таблицы. В верхней части (три строки, не считая строку с названиями столбцов) устанавливаются правила для случаев коллизии двух подвижных объектов, каковыми являются холофермент, кор-фермент и полимераза фагового типа. Каждая из трех строк относится к своему типу *активного* объекта (в том же порядке перечисления); а столбцы относятся к *пассивному* объекту: левые три столбца – к пассивным объектам на той же цепи ДНК (попутное столкновение), правые три – к пассивным объектам

на комплементарной цепи (встречное столкновение). Порядок перечисления типов объектов тот же, что и выше. В каждой клетке таблицы имеются три окошечка для пометок, какие действия надлежит выполнить при данной коллизии. Возможных действий три; о порядке пометок напоминает надпись над таблицей: остановить активный объект (т.е. не сдвигать его на следующий нуклеотид), удалить активный объект с цепи ДНК (и из модели), удалить пассивный объект. Может быть запрошено сразу несколько действий, но не менее одного (в противном случае программа будет работать неправильно, т.к. в ней не предусмотрено возможности «наезда» подвижных объектов друг на друга).

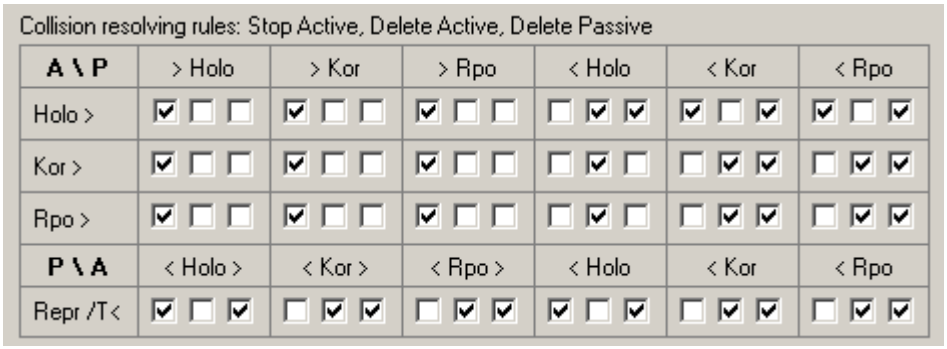


Рис. 1. Правила разрешения коллизий.

В нижней части таблицы, состоящей из одной строки, устанавливаются правила для случая коллизии подвижного объекта с неподвижным (неподвижных объектов два типа – репрессор и терминатор). Естественно, в такой коллизии неподвижный объект всегда является пассивным. Клетки левых трех столбцов относятся к пассивному объекту – репрессору, правых трех столбцов – к терминатору. В каждом случае трех столбцов достаточно, поскольку репрессор взаимодействует с обеими цепями ДНК, а терминатор, наоборот, только со своей цепью. Столбцы соответствуют активным объектам в прежнем порядке (холофермент, кор-фермент и полимераза фагового типа), содержание каждой клетки аналогично верхней части таблицы. Следует учитывать, что указанные для репрессора правила относятся к сценарию терминации (а не протекания, см. п. 1.3.10), а к терминатору правила применяются только в том случае, если он чувствителен к соответствующему типу РНК-полимеразы (эубактериальному или фаговому), подробнее см. п. 1.3.11.

В заключение сформулируем текстуально всю совокупность правил разрешения коллизий, установленных по умолчанию в текущей версии программы (порядок перечисления соответствует обходу таблицы на Рис. 1 по строкам слева направо). На первый взгляд, некоторые правила могут показаться избыточными или неочевидными, однако они внесены для сохранения логической целостности программы. Внося изменения в эти правила, необходимо помнить, что программа не проверяет логической непротиворечивости и целостности заданных правил, так что допущенные на этом этапе ошибки легко могут привести к неправильным результатам или даже аварийному завершению программы.

1. Если холофермент в процессе абортивного движения догоняет другой холофермент на той же цепи, то этот шаг абортивного движения не производится.
2. Если холофермент в процессе абортивного движения догоняет кор-фермент на той же цепи, то этот шаг абортивного движения не производится.
3. Если холофермент в процессе абортивного движения догоняет полимеразу фагового типа на той же цепи, то этот шаг абортивного движения не производится.
4. Если холофермент в процессе абортивного движения встречается с другим холоферментом на комплементарной цепи, то оба холофермента срываются с ДНК.

5. Если холофермент в процессе абортивного движения встречается с кор-ферментом на комплементарной цепи, то этот шаг абортивного движения не производится, а кор срывается с ДНК.
6. Если холофермент в процессе абортивного движения встречается с полимеразой фагового типа на комплементарной цепи, то этот шаг абортивного движения не производится, а полимераза срывается с ДНК.
7. Если кор-фермент догоняет холофермент на той же цепи, то этот шаг кора не выполняется.
8. Если кор-фермент догоняет другой кор-фермент на той же цепи, то этот шаг первого кора не выполняется.
9. Если кор-фермент догоняет полимеразу фагового типа на той же цепи, то этот шаг кора не выполняется.
10. Если кор-фермент встречается с холоферментом на комплементарной цепи, то кор срывается с ДНК.
11. Если кор-фермент встречается с другим кор-ферментом на комплементарной цепи, то оба кора срываются с ДНК.
12. Если кор-фермент встречается с полимеразой фагового типа на комплементарной цепи, то кор и полимераза срываются с ДНК.
13. Если полимераза фагового типа догоняет холофермент на той же цепи, то этот шаг полимеразы не выполняется.
14. Если полимераза фагового типа догоняет кор-фермент на той же цепи, то этот шаг полимеразы не выполняется.
15. Если полимераза фагового типа догоняет другую полимеразу фагового типа на той же цепи, то этот шаг первой полимеразы не выполняется.
16. Если полимераза фагового типа встречается с холоферментом на комплементарной цепи, то полимераза срывается с ДНК.
17. Если полимераза фагового типа встречается с кор-ферментом на комплементарной цепи, то кор и полимераза срываются с ДНК.
18. Если полимераза фагового типа встречается с другой полимеразой фагового типа на комплементарной цепи, то обе полимеразы срываются с ДНК.
19. Если в процессе абортивного движения холофермент встречает репрессор в активном состоянии, то репрессор делается неактивным, а этот абортивный шаг холофермента не выполняется.
20. Если кор-фермент встречает репрессор в активном состоянии, то репрессор делается неактивным, а кор-фермент срывается с ДНК.
21. Если полимераза фагового типа встречает репрессор в активном состоянии, то репрессор делается неактивным, а полимераза срывается с ДНК.

Примечание: Показанные на рисунке и описанные в п.п. 20, 21 правила для репрессора лучше отвечают его прежней функциональности (без протекания), поскольку даже при терминации полимеразы (сценарий (а), раздел 1.3.10) сам репрессор уничтожается. Для репрессора с протеканием эти правила более естественно изменить так, чтобы при срыве полимераз репрессор оставался активным. Что касается уничтожения репрессора в сценарии (б), то оно происходит безусловно и не подчиняется обсуждаемым правилам.

22. Если в процессе абортивного движения холофермент догоняет терминатор в активном состоянии, то этот шаг холофермента не выполняется и терминатор делается неактивным.
23. Если кор-фермент догоняет терминатор в активном состоянии, то кор срывается с ДНК и терминатор делается неактивным.

24. Если полимераза фагового типа догоняет терминатор в активном состоянии, то полимераза срывается с ДНК и терминатор делается неактивным.

1.4.2. Динамическое изменение параметров модели

До сих пор предполагалось, что на протяжении одной траектории моделирования (от запуска модели до истечения заданного интервала физического времени) параметры модели, в частности, её объектов, неизменны. Для того, чтобы на модели можно было воспроизводить эксперименты, в ходе которых изменяются значения ряда параметров модели (например, при варьировании окружающей температуры в экспериментах по тепловому шоку [1]), предусмотрен особый тип объекта – «динамическое изменение параметра заданного агента».

Этот неподвижный объект создаётся в начале каждой траектории для каждого шага динамически изменяемого параметра. Ему приписаны координаты того агента, чей параметр изменяется, и значение физического времени в тот момент, когда это изменение надлежит выполнить. В соответствии с этим временем объект помещается в общую очередь событий. При наступлении указанного времени объект выбирается из очереди, значение параметра изменяется, после чего объект уничтожается. Данный объект не отображается в GUI и не учитывается при анализе правил разрешения коллизий.

Свойства:

1) Begin – позиция самого левого нуклеотида модифицируемого объекта (если объект на основной цепи, то это его 5'-начало, если на комплементарной – 3'-конец).
2) End – позиция самого правого нуклеотида модифицируемого объекта (если объект на основной цепи, то это его 3'-конец, если на комплементарной – 5'-начало).
3) TouchTime – момент времени, когда надлежит изменить значение параметра.
4) agent – дескриптор модифицируемого объекта (указатель).
5) parnum – указатель параметра, значение которого будет изменено (параметры однозначно идентифицируются буквенным кодом в первом столбце Табл. 1).
6) val1 – первое значение (обязательное).
7) val2 – второе значение (необязательное).
8) preset – базовое значение параметра (в начале траектории).

Указанные три значения суть действительные числа, ниже объясняется их смысл.

9) operation – код выполняемой операции; в текущей версии программы предусмотрено две операции – Set и Linear. Операция Set присваивает заданному параметру значение val1. Операция Linear присваивает заданному параметру значение val1 * preset + val2. Иными словами, первая операция используется для установки абсолютного значения, а вторая – для установки относительного значения, которое линейно зависит от фиксированной базы.

Методы:

1) Touch – устанавливает значение TouchTime, вызывается только при создании объекта.
2) Set – устанавливает значение заданного параметра согласно одной из операций выше, вызывается при достижении момента времени TouchTime.

Помимо этого, для обеспечения динамического изменения параметров для всех классов агентов (см. раздел 1.3) реализованы методы: Modify, GetValue, Store, Restore. Первые два из них позволяют изменить текущее или получить исходное значение выбранного динамического параметра. Другие два метода обеспечивают сохранение и восстановление начальных значений сразу всех динамических параметров объекта.

Поскольку в модели используется дискретное время, любые изменения предполагаются ступенчатыми и мгновенными. На траектории модели можно задать ограниченное число (в текущей версии программы не более 16) моментов физического времени, в которые

изменяется значение параметра агента. Общее число агентов задачи, параметры которых можно менять, также ограничено (не более 64 в текущей версии). В Табл. 1 перечислены допустимые типы агентов и их параметров. Подвижные объекты, прежде всего кор-фермент и полимераза фагового типа, самостоятельно в таблице не фигурируют, поскольку они порождаются соответствующими неподвижными объектами, наследуя текущие значения их параметров. Имена параметров (т.е. свойств объектов) были приведены в разделе 1.3. Подчеркнём, что геометрия модели (положение и размеры объектов) динамически меняться не может, за исключением нескольких параметров (положение центра транскрипции, вид и длина сигма-субъединицы, вид полимеразы фагового типа). Появление и пропадание объектов отдельно не предусмотрено, поскольку его можно эффективно моделировать с помощью других свойств, таких как интенсивность.

Табл. 1. Параметры объектов модели, которые могут изменяться динамически.

Код	Объект / Параметр	Ген	PEP-промотор (включая холо- и кор-фермент)	NEP-промотор (включая полимеразы фагового типа)	Внешний источник кор-ферментов (и они сами)	Внешний источник полимераз фагового типа (и они сами)	Репрессор	Терминатор
I	Initiated	✓						
E	InitP	✓						
F	InitN	✓						
T	Transcribed	✓						
P	TransP	✓						
N	TransN	✓						
L	Lambda		✓	✓	✓	✓	✓	✓
S	Sigma		✓					
C	Tcenter		✓	✓	✓	✓		
H	SigLength		✓					
A	Aborting		✓					
G	AbortRange		✓					
R	Rate		✓	✓	✓	✓		
O	Offered		✓	✓	✓	✓	✓	✓
B	Bound		✓	✓	✓	✓	✓	✓
Y	Polym			✓		✓		
X	SenseKor							✓
Z	SenseRpo							✓
M	Qmain						✓	
Q	Qcomp						✓	
D	Mterminated						✓	
K	Cterminated						✓	

В заключение отметим, что динамическое изменение параметров удобно применять не только по ходу траектории, но и в нулевой момент времени, что позволяет устанавливать желаемые значения некоторых параметров для всего задания в целом, вместо того, чтобы указывать их в каждой строке файла заданий (см. раздел 4.3).

2. Программа Rivals (приложение Windows)

2.1. Минимальные требования, установка и тестирование

Программа представляет собой обычное приложение Windows с GUI, и не имеет каких-либо особенностей. Минимальные программные требования следующие:

- Операционная система Microsoft Windows 2000 / XP / Server 2003 / Vista / Windows 7 (все 32-битной или 64-битной версии), локализация не требуется, но рекомендуется установить в системе все позднейшие пакеты обновлений.
- Среда .NET 2.0 Framework Service Pack 2.0 (или более поздняя версия); если еще не установлена, то доступна для загрузки с сайта Microsoft.
- Свободно распространяемые библиотеки времени исполнения для Microsoft Visual C++ 2008 Service Pack1 (если этот продукт не установлен в системе), доступны для загрузки с сайта Microsoft или со страницы программы.

Специальных аппаратных требований не предъявляется, они зависят от размеров моделируемых локусов и числа одновременно запускаемых копий программы. Ориентировочно, для типичных задач достаточно иметь объем оперативной памяти 512 Мб и процессор с тактовой частотой 1600 МГц, однако настоятельно рекомендуется применять б**о**льшие значения и многоядерные процессоры, что позволит ускорить моделирование, в том числе за счет распределения работы между несколькими копиями программы. Следует учитывать, что программа обеспечивает 100% загрузку процессорного ядра, и моделирование может продолжаться часами или сутками, что предъявляет повышенные требования к надежности и качеству применяемого оборудования.

Программа не имеет установщика и ничего не записывает в системный реестр. Для установки достаточно распаковать архив с нужной версией программы в рабочую папку и поместить туда же файлы конфигурации и задачи (а также файл задания, если используется). Примеры файлов входят в комплект поставки. Можно также скопировать программу в другую папку, например, одну из папок в общем пути автоматического поиска, а местоположение рабочих файлов указать в соответствующих параметрах командной строки запуска (или в ярлыке программы).

Для удаления программы с компьютера достаточно удалить исполняемый файл rivals.exe и все относящиеся к нему файлы данных.

В комплект поставки программы входят файлы, с помощью которых можно быстро провести проверку ее работоспособности. Для этого из папки, куда были распакованы файлы, нужно запустить командный файл **test.bat**. В зависимости от быстродействия компьютера, тестирование занимает до 5 минут, в результате чего будут получены файлы результатов HSM.csv, HSM.log. Следует сравнить их содержание с аналогичными файлами HSM0.csv, HSM0.log из комплекта поставки (из-за разного быстродействия компьютеров данные о времени счета могут отличаться), но количество записей в файлах и численные значения транскрипции генов должны совпадать.

2.2. Интерфейс и параметры командной строки

Общий вид главного окна программы в процессе моделирования задачи представлен на Рис. 2 (здесь и далее на рисунках может представляться и более ранняя версия программы, если нет существенных внешних отличий между версиями). Графический интерфейс пользователя включает в себя заголовок окна, главное меню, панель инструментов, рабочую область и строку состояния.

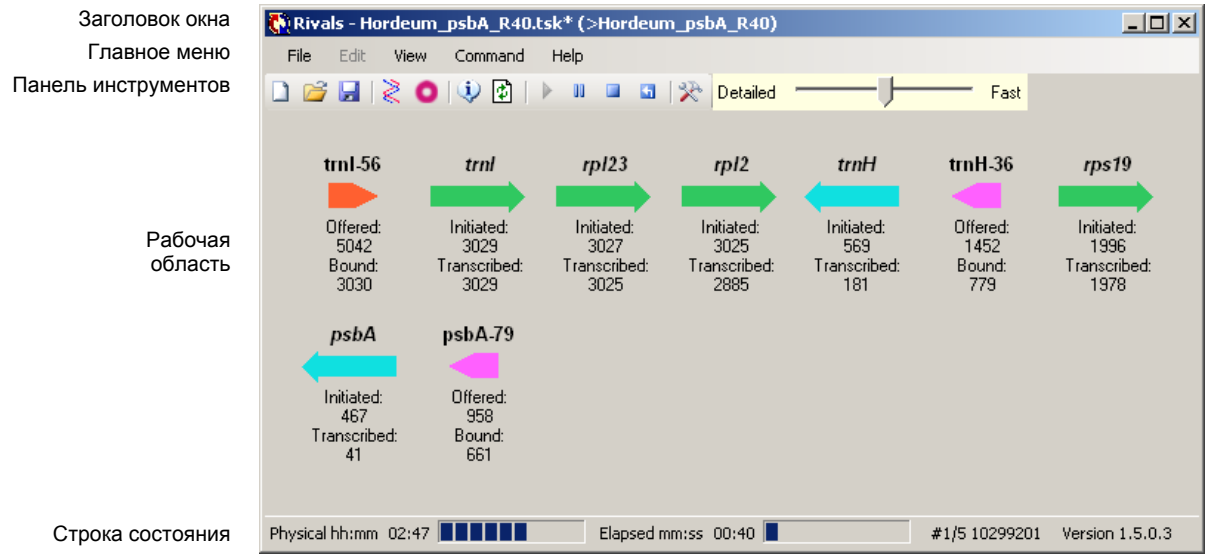


Рис. 2. Элементы главного окна программы.

Заголовок окна обеспечивает все стандартные функции окна Windows (развертывание, свертывание, перемещение и изменение размеров окна, а также выход из приложения) и, сверх того, информирует о режиме работы программы следующими способами.

- После запуска программы, пока не загружена никакая задача, в заголовке стоит только имя приложения, Rivals.
- При загруженной задаче в заголовке стоит имя приложения, затем имя задачи и в скобках символ > и имя выходного файла, например: Rivals - AT1_psbB.tsk (>AT1_result).

  Примечание 1. Если создается новая задача, то ей по умолчанию присваивается имя noname.tsk, до тех пор, пока она не будет сохранена как файл задачи с другим именем.

  Примечание 2. Имя выходного файла указано без расширения, поскольку в общем случае формируется как минимум два выходных файла, с расширениями .csv и .log. По умолчанию, имя выходного файла совпадает с именем задачи.

  Примечание 3. Если в задачу были внесены изменения, которые еще не сохранены в файле задачи, то сразу после имени задачи ставится звездочка.
- При работе программы в пакетном режиме в заголовке перед символом > в скобках указывается имя выполняемого файла задания.

2.2.1. Функции главного меню

В этом разделе последовательно описываются функции иерархического главного меню программы Rivals. Для ряда функций меню имеются акселераторы («горячие» клавиши), которые указаны справа от соответствующего пункта меню и здесь не обсуждаются. Некоторые пункты меню доступны только в определенных режимах или в соответствующие моменты работы.

2.2.1.1. Меню File

New Task Создание новой (пустой) задачи. Ранее загруженная в память задача стирается; если в нее были внесены изменения, программа предлагает их сохранить.

Load Task Загрузка задачи в память из существующего файла. Ранее загруженная в память задача стирается; если в нее были внесены изменения, программа предлагает их сохранить. При выборе этого пункта появляется стандартное окно открытия файла Windows. По умолчанию файлы задачи имеют системное расширение .tsk. Во время загрузки файла задачи проверяется его корректность, и при отсутствии ошибок в рабочей области отображается диаграмма задачи, аналогичная показанной на Рис. 2. Схема представляет фигурирующие в данной задаче объекты (см. раздел 1.3), которые размещены на воображаемой оси, представляющей последовательность ДНК. Программа автоматически выбирает способ отображения диаграммы, исходя из текущих размеров окна, и при недостаточных размерах размещает ось ДНК в несколько «этажей» и/или выводит линейки прокрутки. Порядок объектов отражает их очередность с привязкой к основной цепи ДНК, но если объекты полностью или частично перекрываются, то очередность определяет программа. Точные координаты всех объектов можно посмотреть в любой момент, нажав кнопку **Location** на панели инструментов (см. ниже).

Save Task Сохранение измененной задачи в файле с прежним именем.

Save As task Сохранение измененной задачи в файле с новым именем. При выборе этого пункта появляется стандартное окно сохранения файла Windows. По умолчанию файлы задачи имеют системное расширение .tsk.

Close Task Стирание загруженной в память задачи (без загрузки новой). Если в задачу были внесены изменения, программа предлагает их сохранить.

Import DNA Sequence Загрузка в файл задачи последовательности ДНК из текстового файла (расширение по умолчанию .txt) или файла ГенБанка NCBI (.gbk). При выборе этого пункта появляется стандартное окно открытия файла Windows. Текстовый файл не должен содержать никакой другой информации, кроме нуклеотидной последовательности, иначе будет выдано сообщение об ошибке. Последовательность задается в расширенном алфавите IUPAC-IUB, регистр букв безразличен. Символы пробела, табуляции, перевода строки игнорируются; также пропускаются применяемые символы вставки при выравнивании: дефис, подчеркивание, знак равенства и звездочка.

Get from Clipboard Загрузка в файл задачи последовательности ДНК из буфера обмена Windows. Требования к содержимому буфера обмена те же, что и к текстовому файлу в предыдущем пункте меню. Удобно использовать в тех случаях, когда в имеющемся файле с последовательностью содержится посторонняя информация (в файлах .gbk эта информация игнорируется автоматически).

Circular sequence Позволяет указать программе, что обрабатывается кольцевая последовательность ДНК (в этом случае данный пункт меню отмечается галочкой). Поскольку информация о том, является ли последовательность кольцевой или линейной, содержится в файле задачи, то этот пункт меню может быть уже отмечен при загрузке такой задачи. Для отмены режима (и перехода к линейной последовательности) следует выбрать этот пункт повторно, после чего отметка исчезнет. *Примечание:* Невозможно отменить кольцевой режим, если в составе задачи имеются объекты, переходящие через границу последовательности, поскольку в линейном режиме левый край любого объекта должен располагаться в позиции с совпадающим или меньшим номером, чем правый край).

Output to file Позволяет указать или изменить имя выходных файлов программы. Если у имени не указан путь, то файлы будут созданы в папке с программой. Имя должно указываться без расширения; если расширение присутствует, оно игнорируется.

Run job Загрузка файла задания для автоматического исполнения в пакетном режиме. При выборе этого пункта появляется стандартное окно открытия файла Windows. Файлы задания по умолчанию имеют расширения .rjb. Сразу после выбора файла программа начнет работать автоматически, прекратить выполнение можно только закрыв приложение.

Exit Прекращение работы программы и закрытие главного окна. Если в задачу были внесены изменения, программа предлагает их сохранить.

2.2.1.2. Меню Edit

Add Добавление в задачу нового объекта. Эта функция используется при составлении новой задачи или при корректировке задачи, которая в настоящий момент загружена в память. Выбор данного пункта меню приводит к появлению подменю со следующими пунктами:

- **Gene** Переход в диалоговое окно редактирования параметров гена, описанное в п. 2.3.2.1.
- **PEP promoter** Переход в диалоговое окно редактирования параметров PEP-промотора (п. 2.3.2.2).
- **NEP promoter** Переход в диалоговое окно редактирования параметров NEP-промотора (п. 2.3.2.3).
- **Repression site** Переход в диалоговое окно редактирования параметров репрессора (п. 2.3.2.4).
- **External Kor** Переход в диалоговое окно редактирования параметров внешнего источника кор-ферментов (п. 2.3.2.5).
- **External Rpo** Переход в диалоговое окно редактирования параметров внешнего источника полимераз фагового типа (п. 2.3.2.6).
- **Terminator** Переход в диалоговое окно редактирования параметров терминатора (п. 2.3.2.7).

Object Просмотр и/или корректировка параметров интересующего объекта задачи. При выборе этого пункта появляется такое же подменю, как в пункте **Add**, но для каждого пункта этого подменю выдается меню следующего уровня в виде списка объектов данного класса (например, PEP-промоторы), которые присутствуют в задаче. Этот список включает и скрытые объекты, которые не видны в рабочей области, так что это единственно возможный способ изменить параметры такого объекта, в частности, снова его отображать. При выборе объекта из списка появляется одно из диалоговых окон редактирования объекта (см. п. 2.3.2), заполненное текущими значениями параметров, которые можно править. Единственный параметр, который нельзя изменить – имя объекта; для этой цели следует использовать функцию переименования объектов (см. п. 2.2.3.2).

Options Просмотр и/или настройка параметров конфигурации программы, как подробно описано в разделе 2.3.1.

2.2.1.3. Меню View

Normal Обычный режим, который применяется при показе информации об объектах задачи в рабочей области главного окна, принимаемый по умолчанию (см. пример на Рис. 2). В этом режиме для каждого отображаемого динамического объекта показаны наименования и значения двух счетчиков, например, для промотора – общее число попыток посадки РНК-полимеразы и число успешных связываний. Альтернативой является режим отображения, включаемый следующим пунктом меню.

Compact Компактный режим показа информации об объектах задачи в рабочей области окна (Рис. 3). В этом режиме для каждого динамического объекта показано значение только основного счетчика (законченных транскрипций – для генов, успешных посадок РНК-полимераз – для промоторов и т.п.), и наименования счетчиков опущены, что позволяет сэкономить место в рабочей области окна для задач с большим числом объектов. Переключаться из обычного режима просмотра в компактный и обратно можно в любой момент, даже по ходу моделирования.

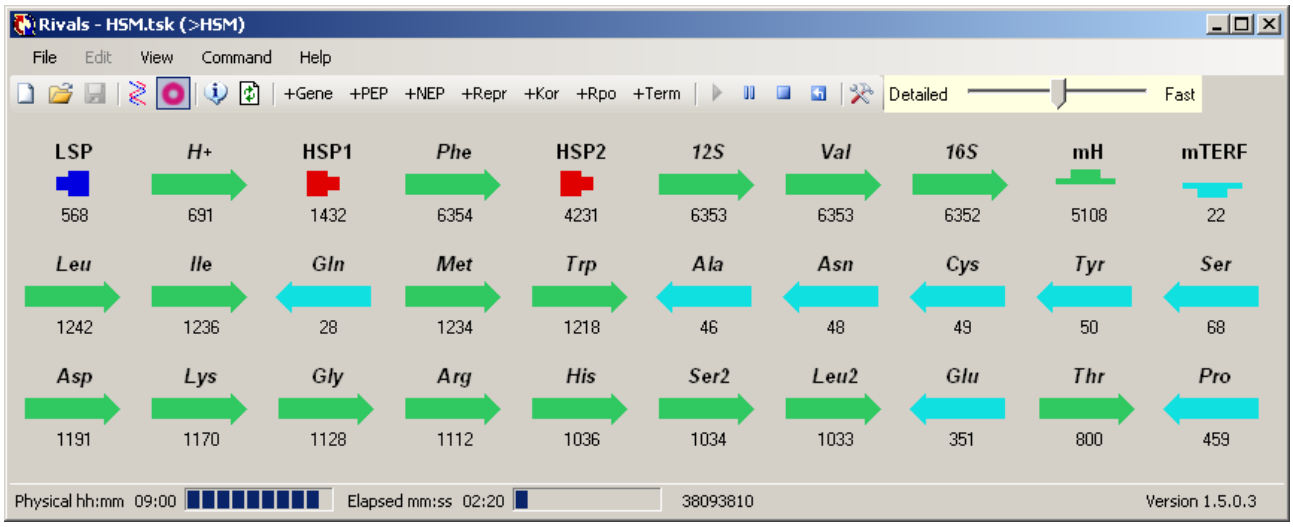


Рис. 3. Компактный режим показа рабочей области окна.

Plus buttons Показ на панели инструментов кнопок добавления объектов, для большего удобства при подготовке новых задач (см. Рис. 3 – Рис. 5). Каждая кнопка заменяет один из пунктов подменю **Edit > Add**, например, нажатие кнопки **+Gene** эквивалентно выбору в главном меню функции **Edit > Add > Gene**. По умолчанию данный режим выключен.

Joint transcription Включенный по умолчанию режим суммарного учета числа инициированных и завершенных транскрипций гена, независимо от типа РНК-полимеразы (отображение информации в этом режиме представлено на Рис. 2 и Рис. 3). Если выключить этот режим в данном пункте меню, то число транскрипций генов подсчитывается раздельно по кор-ферментам и полимеразам фагового типа (см. пример на Рис. 4, где буква **p** стоит перед числом транскрипций с PEP-промотора, а **n** – с NEP-промотора). Данный режим управляет только отображением в рабочей области и не влияет на информацию в выходных файлах. Включать и выключать режим можно в любое время.

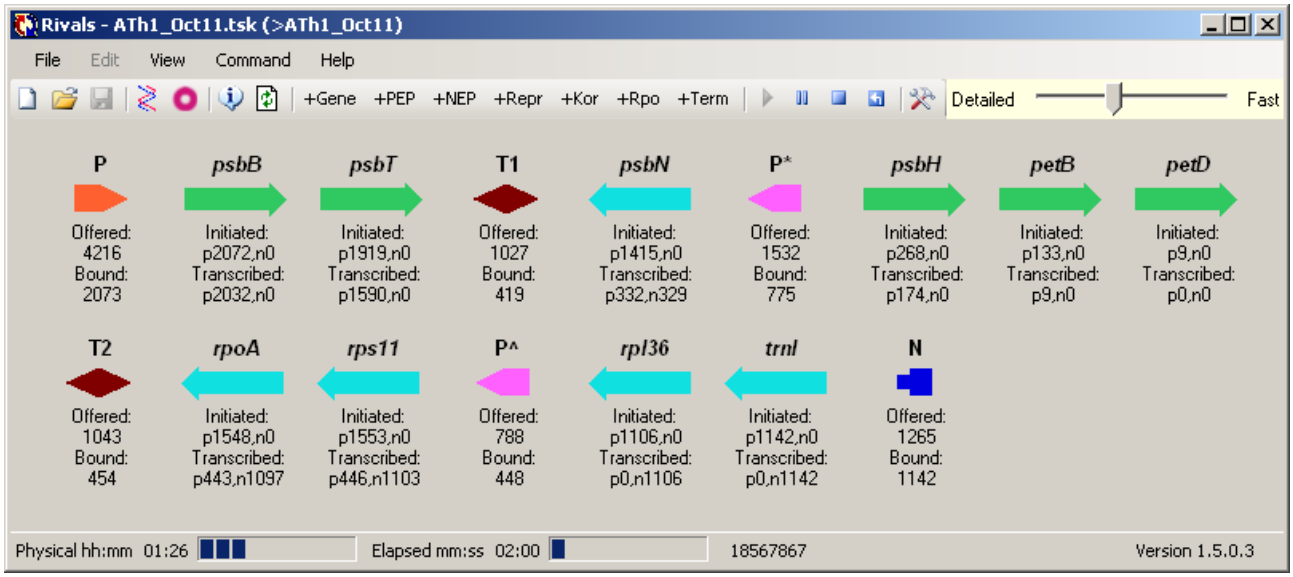


Рис. 4. Раздельный подсчет числа транскрипций по типам РНК-полимераз.

Locations Информационная функция, обеспечивающая показ местоположения всех объектов задачи на последовательности ДНК. В обычном режиме отображения будут показаны: комментарии к объекту (если есть), позиция старта транскрипции – для промоторов и внешних источников РНК-полимераз, длина объекта и позиции левого и правого концов (Рис. 5). В компактном режиме отображаются только позиции краев объекта. Показанная информация сохраняется до следующего обновления окна, выполненного вручную или автоматически.

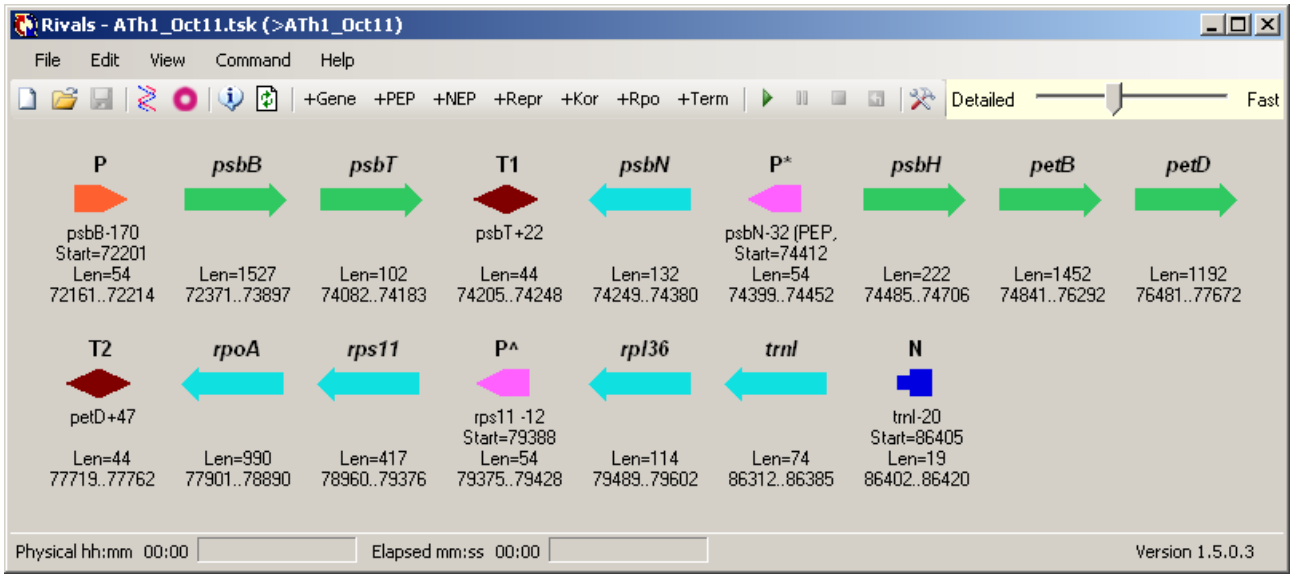


Рис. 5. Показ местоположения всех объектов задачи.

Refresh Обновление рабочей области главного окна. В большинстве случаев, когда требуется обновить окно, это делается автоматически; при необходимости обновление также можно выполнить вручную. Например, по ходу моделирования значения счетчиков могут обновляться не сразу, а периодически. Данная функция меню позволяет увидеть текущие значения всех счетчиков, если они отображаются с заданным шагом.

2.2.1.4. Меню Command

Это меню активизируется, когда в память загружена готовая к запуску задача. Меню содержит следующие пункты:

Run Запуск очередной траектории моделирования.

Pause Временная приостановка траектории, например, для того, чтобы сопоставить текущие значения счетчиков или освободить процессорные ресурсы для параллельно выполняемых задач. Повторный выбор этой функции возобновляет моделирование.

Stop Остановка траектории с получением финальных значений счетчиков. После остановки продолжить моделирование по этой траектории невозможно.

Reset Сброс траектории (независимо от того, ведётся или остановлено моделирование) с обнулением значений всех счетчиков. Повторный запуск модели возможен только после сброса текущей траектории.

Run job Этот пункт меню идентичен одноименной функции в меню **File** (п. 2.2.1.1).

2.2.1.5. Меню Help

Command line arguments Краткая справка по параметрам командной строки запуска программы (Рис. 6). Описание параметров командной строки см в п. 2.2.5.

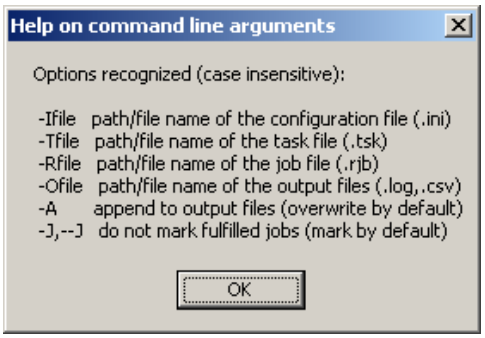


Рис. 6. Памятка о параметрах командной строки.

About Общие сведения о программе и контактная информация (Рис. 7).

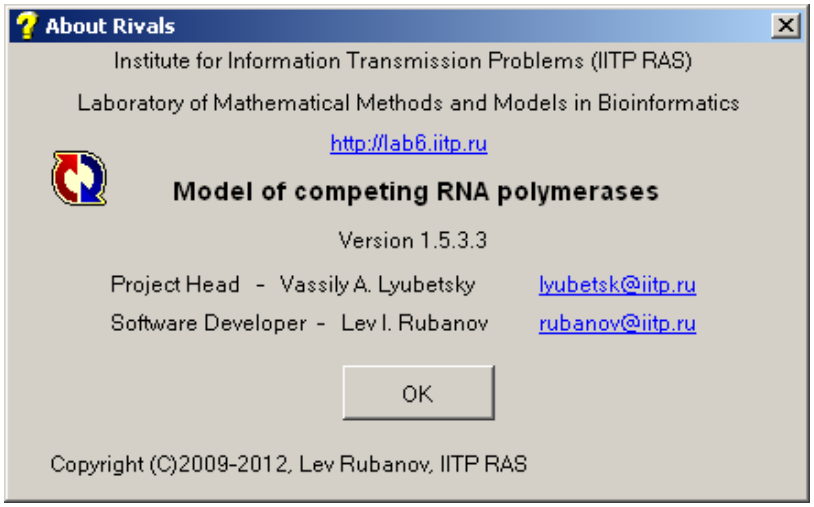


Рис. 7. Общие сведения о программе.

2.2.2. Панель инструментов

Для быстрого обращения к некоторым пунктам меню в составе GUI имеется панель инструментов (пример на Рис. 2), в которой при желании отображается ряд дополнительных кнопок, как описано в п. 2.2.1.3. Функции постоянно присутствующих основных кнопок эквивалентны выбору следующих пунктов меню:


File > New task – создание новой задачи


File > Load task – загрузка существующей задачи из файла


File > Save task / Save As task – запись измененной задачи в файл. Если файл задачи уже существует, то используется функция **Save**, в противном случае – **Save As** (поэтому при желании сохранить существующую задачу с новым именем следует использовать меню, а не эту кнопку)


File > Import DNA sequence – загрузка последовательности ДНК из текстового файла или файла ГенБанка


File > Circular sequence – режим кольцевой последовательности ДНК. Если режим включён, т.е. соответствующий пункт меню отмечен, то эта кнопка показана в рамке, в противном случае – без рамки. Нажатие кнопки включает/выключает режим кольцевой ДНК.


View > Locations – показ местоположения объектов задачи.


View > Refresh – обновление рабочей области окна


Command > Run – запуск траектории моделирования (эта и следующие три кнопки активны только при загруженной задачи, как и соответствующее меню)


Command > Pause – пауза


Command > Stop – остановка моделирования


Command > Reset – сброс результатов перед новым моделированием


Edit > Options – настройка конфигурации программы (см. п. 2.3.1)

В правой части панели инструментов имеется движок, позволяющий установить желаемое соотношение между скоростью работы программы и оперативностью вывода текущей информации (т.е. значений счётчиков) в рабочей области окна. В среднем положении движка (по умолчанию) значения счётчиков меняются с шагом 10, т.е. изменения в пределах десятка не видны. В крайнем левом положении движка шаг отображения равен 1, т.е. видны все изменения счётчиков в реальном времени (однако быстродействие программы при этом несколько уменьшается). Наоборот, в крайнем правом положении движка шаг обновления счётчиков равен 100; при этом влияние на быстродействие незначительное.

Примечания: 1) При запуске программы в пакетном режиме шаг обновления значений счетчиков определяется параметром **Granularity** в файле задания, однако при изменении положения указанного движка в ходе счета программа перейдет на новую настройку.
2) Чтобы вывести точные текущие значения всех счетчиков, можно в любой момент нажать кнопку **Refresh** (Обновить)**.**

2.2.3. Рабочая область окна

В рабочей области окна (см. Рис. 2) отображаются символические обозначения объектов задачи и некоторые их параметры. Система обозначений объектов приведена в разделе 1.3. Над каждым символом объекта указано его имя (длинные имена могут быть выведены не полностью). Объекты располагаются в одну или несколько строк; порядок показа приблизительно соответствует очередности расположения объектов на ДНК (относительно основной цепи), но это может нарушаться при пересечении или наложении объектов.

Более точная информация об объекте выдается в форме всплывающей подсказки, если на некоторое время задержать указатель мыши на объекте. Первая строка подсказки имеет следующий формат:

<*class*> <*name*> : [complement] <*begin*>..<*end*>

где <*class*> – сокращенное обозначение типа объекта задачи (см. Табл. 2), <*name*> – имя объекта, <*begin*> и <*end*> – позиции соответственно левого и правого краев объекта на основной цепи ДНК. Слово complement появляется, если объект находится на комплементарной цепи ДНК.

Табл. 2. Сокращенные обозначения типов объектов.

Обозначение	**Тип объекта и раздел, где он описан**
Gene	Ген (см. п. 1.3.2)
PEP	PEP-промотор (см. п. 1.3.3)
NEP	NEP-промотор (см. п. 1.3.7)
Repr	Репрессор (см. п. 1.3.10)
EKor	Внешний источник кор-ферментов (см. п. 1.3.6)
ERpo	Внешний источник полимераз фагового типа (см. п. 1.3.9)
Term	Терминатор (см. п. 1.3.11)

Если объект характеризуется параметром интенсивности λ, то его значение приводится во второй строке подсказки. Сверх того, в последующих строках подсказки воспроизводятся комментарии к объекту, если они имеются в файле задачи.

Заметим, что всплывающая подсказка входит в набор стандартных элементов Windows и, как показывает опыт, иногда перестает появляться или пропадать с экрана. В подобных ситуациях рекомендуется использовать функцию **Refresh** программы (горячая клавиша **F5**). Иногда более удобно получить информацию об объекте из режима редактирования его параметров (функция меню **Edit > Object** или контекстное меню объекта, описанное ниже), но это возможно только до начала или после прекращения моделирования вдоль траектории.

При щелчке правой кнопкой мыши по объекту в рабочей области окна появляется контекстное меню этого объекта, вид и функции которого зависят от того, идет ли в это время моделирование вдоль траектории (включая и паузы в процессе счета) или нет.

2.2.3.1. Контекстное меню во время моделирования

В процессе моделирования, даже если счет в данный момент приостановлен, изменения в модели или задаче недопустимы, поэтому контекстное меню всех объектов одинаково и содержит только функции **Run, Pause, Stop, Reset,** идентичные описанным в разделе 2.2.1.4.

2.2.3.2. Контекстное меню объекта вне времени моделирования

Контекстное меню объекта содержит следующие функции:

Disable Отключить объект. Используется, чтобы временно исключить объект из моделирования, например, промотор при моделировании нокаута по сигма-субъединице, от которой зависит этот промотор. При выборе этой функции объект в рабочей области отображается пригашенным. Если снова вызвать контекстное меню того же объекта, то эта функция заменяется на функцию **Enable**, которая снова включает ранее отключенный объект.

Hide Скрыть объект. Позволяет не показывать в области задачи какие-то объекты, существенные для моделирования, но не требующие постоянного наблюдения за ними. Работа модели не зависит от того, все ли объекты отображаются или часть их скрыта. Функция может быть полезна для задач с большим числом объектов. Поскольку скрытый объект не отображается, и, следовательно, его контекстное меню недоступно, для отмены этой функции используется меню **Edit > Object** (см. п. 2.2.1.2), в подменю которого присутствуют все объекты, включая и скрытые.

Rename Переименовать объект. При выборе этой функции появляется диалоговое окно **Rename Object** с текущим именем объекта, которое можно изменить и нажать кнопку **OK**, либо закрыть при отказе от переименования. Имя объекта должно быть уникальным в пределах задачи и может состоять из любых символов, включая пробелы, в количестве не более 31. В зависимости от текущих настроек Windows, отведенного в рабочей области места может оказаться недостаточным для показа имени полностью, поэтому не рекомендуется злоупотреблять длинными именами объектов. Примечание: Для обработки в пакетном режиме имена объектов не должны содержать пробелы, знаки табуляции, и символы . , = : % ! ?.

Delete Удалить объект. При выборе этой функции программа запрашивает подтверждение на удаление объекта из задачи. Операция удаления необратима, поэтому единственный способ восстановить удаленный объект – это заново загрузить задачу (в предположении, что файл задачи до удаления сохранялся). При этом будет выдано окно с предупреждением, что результаты изменения не сохранены (Рис. 8). Разумеется, в данной ситуации правильным будет ответ **Yes**.

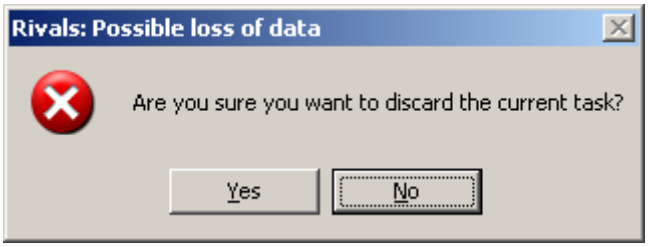


Рис. 8. Предупреждение о несохраненной задаче.

Properties Показать и/или изменить параметры объекта. Редактирование свойств всех типов объектов подробно описывается в разделе 2.3.2.

2.2.4. Строка состояния

Строка состояния программы Rivals позволяет наблюдать за ходом моделирования, а также быстро понять, в каком режиме используется программа. Иногда только по изменениям в строке состояния можно судить о том, идет ли счёт. Строка состояния отображается в одном из четырех возможных вариантов, в зависимости от того, как была запущена программа.

2.2.4.1. Строка состояния в режиме одиночной траектории

Это самый примитивный режим запуска программы, когда загруженная задача моделируется только на одной траектории. Вид строки состояния в этом случае показан на Рис. 9.

Рис. 9. Вид строки состояния в режиме одиночной траектории.

В строке указана следующая информация (слева направо):

- физическое (моделируемое) время в часах и минутах, на рисунке – 44 мин.;
- индикатор прогресса по отношению к заданному физическому времени моделирования;
- время счета программы в минутах и секундах, на рисунке – 1 мин. 5 сек.;
- индикатор прогресса по отношению к заданному максимальному времени счёта;
- порядковый номер текущей итерации модели (см. раздел 1.4), на рисунке – 11303354;
- номер версии программы, на рисунке – 1.3.7.1.

Предельные значения моделируемого физического времени и компьютерного времени счёта, которые отвечают 100% заполнению индикаторов прогресса, устанавливаются в составе конфигурации программы (см. раздел 2.3.1). Моделирование продолжается, пока не будет достигнуто любое из этих ограничений. Программа позволяет указывать нулевое значение в качестве одного или обоих пределов; это трактуется как отсутствие данного ограничения. Чтобы и в этом случае сохранить какую-то информативность индикаторов прогресса, для индикаторов прогресса используются «большие» значения, имеющие смысл бесконечности: 30 суток для моделируемого физического времени и 72 часа для компьютерного времени.

2.2.4.2. Строка состояний в режиме пучка траекторий

Чтобы получить более достоверные данные о транскрипции генов при фиксированных значениях параметров задачи, моделирование методом Монте-Карло обычно проводится на пучке траекторий. Каждая траектория стохастического моделирования дает свои значения счётчиков числа транскрипций генов, связывания промоторов и т.д., которые в конце усредняются по всему пучку траекторий. В этом режиме строка состояния имеет вид, показанный на Рис. 10.

Рис. 10. Вид строки состояния в режиме пучка траекторий.

Выводится в основном та же информация, что и в режиме одиночной траектории (она относится к той траектории пучка, которая моделируется в настоящий момент), но перед номером текущей итерации выводится добавочный элемент данных вида *#m/n*, где *m* – номер текущей траектории пучка, *n* – число траекторий в пучке (так, на Рис. 10 пучок состоит из 10 траекторий, а показанная информация относится к третьей траектории).

2.2.4.3. Строка состояния в пакетном режиме с одиночными траекториями

Этот относительно редко применяемый режим позволяет промоделировать одну траекторию для заранее сформированной совокупности наборов параметров. Каждый набор параметров называется заданием, а вся их совокупность – пакетом заданий (который записан в файл заданий, см. п. 4.3). Таким образом, в этом режиме проводится расчёт по одной траектории для каждого задания. Строка состояния имеет следующий вид (Рис. 11).

Рис. 11. Вид строки состояния в пакетном режиме с одиночными траекториями.

Выводится следующая информация (слева направо):

- моделируемое физическое время в часах и минутах, на рисунке – 48 мин.;
- индикатор прогресса по отношению к заданному физическому времени моделирования;

- порядковый номер обрабатываемого задания и общее число заданий в пакете, на рисунке – 7 из 18;
- индикатор прогресса по отношению к объему пакета заданий;
- номер текущей итерации модели (см. раздел 1.4), на рисунке – 9756201;
- номер версии программы, на рисунке – 1.3.7.1.

Таким образом, левый индикатор прогресса показывает состояние текущего задания, а правый – всего пакета заданий в целом.

2.2.4.4. Строка состояния в пакетном режиме с пучком траекторий

Для массовых расчетов наиболее естественно использовать этот режим, в котором каждое задание из пакета моделируется на пучке траекторий. Строка состояния (Рис. 12) в основном содержит ту же информацию, что и в пакетном режиме с одиночными траекториями, но перед номером текущей итерации выводится добавочный элемент данных вида *#m/n*, где *m* – номер текущей траектории пучка, *n* – число траекторий в пучке (так, на Рис. 12 пучок состоит из 100 траекторий, а информация относится к седьмой траектории второго задания из 12).

Рис. 12. Вид строки состояния в пакетном режиме с пучком траекторий.

2.2.5. Параметры командной строки

Чтобы исключить выполнение вручную некоторых часто повторяемых операций и упростить составление сценариев (командных файлов) операционной системы, в программе Rivals предусмотрена возможность указания ряда необязательных параметров в командной строке запуска. Параметры отделяются друг от друга одним или более пробелом, поэтому если значение параметра содержит пробелы, его необходимо указывать в двойных кавычках. Регистр букв и очередность параметров не имеют значения. Первым символом любого параметра должен быть знак '-' (дефис) или '/' (косая черта). Описываемая версия программы воспринимает в командной строке следующие параметры.

-i<*filename*> Указывает местоположение и имя файла конфигурации. Вместо <*filename*> указывается имя файла конфигурации, которое может содержать составленный по правилам Windows путь – относительный из папки, где находится программа, или абсолютный, с указанием буквы диска. По умолчанию файл конфигурации находится в папке с программой и имеет имя **rivals.ini**.

-t<*filename*> Указывает местоположение и имя файла задачи. Вместо <*filename*> указывается имя файла задачи, которое может содержать составленный по правилам Windows путь – относительный из папки, где находится программа, или абсолютный, с указанием буквы диска. Даже если этот параметр указан, его значение впоследствии может быть изменено в файле задания или интерактивно, используя функции GUI.

-o<*filename*> Указывает местоположение и имя выходных файлов. Вместо <*filename*> указывается имя файла, которое может содержать составленный по правилам Windows путь – относительный из папки, где находится программа, или абсолютный, с указанием буквы диска. Расширение имени файла несущественно и игнорируется, если указано: программа записывает один или два выходных файла с расширениями **.log** и **.csv** (второй файл

формируется, если это запрошено в конфигурации программы; при запуске в пакетном режиме также формируется второй файл **.csv** с результатами незаконченных заданий). Даже если этот параметр указан, его значение впоследствии может быть изменено в файле задания или интерактивно, используя функции GUI.

-r<*filename*> Указывает местоположение и имя файла заданий (п. 4.3), который содержит пакет заданий для моделирования. Вместо <*filename*> указывается имя файла заданий, которое может содержать составленный по правилам Windows путь – относительный из папки, где находится программа, или абсолютный, с указанием буквы диска. Если этот параметр указан, то программа сразу начинает счет, а по исчерпании пакета заданий – автоматически закрывается. *Примечание:* Для расчётов в пакетном режиме эффективнее применять версию программы с интерфейсом командной строки (глава 3).

-a Включает режим дозаписи в выходные файлы. По умолчанию старое содержимое файлов стирается. Эта настройка впоследствии может быть изменена в файле заданий.

-j Если указан этот параметр, то в процессе обработки файла заданий в нём не будут делаться никакие пометки. По умолчанию строки с выполненными заданиями превращаются в комментарии, что позволяет продолжать работу с того задания, на котором она была прервана. Для совместимости с вариантом программы для командной строки этот параметр может также указываться в форме --j.

Краткую подсказку с перечнем и синтаксисом воспринимаемых параметров командной строки можно получить с помощью функции меню **Help > Command line arguments** (Рис. 6).

2.3. Работа с программой в интерактивном режиме

В этом разделе описываются важные вопросы интерактивного использования программы, которые не были освещены выше при описании интерфейса, и приводятся типовые сценарии работы с программой в интерактивном режиме.

2.3.1. Настройка конфигурации

Запуск программы Rivals без файла конфигурации невозможен; в комплект поставки входит файл конфигурации с настройками по умолчанию. В дальнейшем эти настройки могут быть изменены по желанию пользователя, для чего используется функция **Edit > Options** основного меню или соответствующая кнопка панели инструментов (п. 2.2.2). Изменение конфигурации возможно только до запуска собственно моделирования или после его прекращения. Сделанные настройки можно сохранить в первоначальном файле конфигурации или в виде нового конфигурационного файла, в противном случае они действуют только до выхода из программы.

При переходе в режим настройки конфигурации появляется диалоговое окно настроек с рядом закладок; вначале открывается закладка **General** с общими настройками (Рис. 13). (Состояние всех диалоговых окон этого раздела определяется текущей конфигурацией и может отличаться от показанного на рисунках). В нижней части этой закладки находится таблица правил разрешения коллизий, которая описывалась в разделе 1.4.1 и потому здесь не обсуждается. Смысл прочих настроек описывается ниже.

2.3.1.1. Закладка General

Physical time Указывает максимальное физическое время моделирования траектории, в секундах. Шаг изменения кнопками «меньше/больше» – 1 час. Значение 0 означает, что физическое время моделирования не ограничивается, и модель будет работать до исчерпания отведенного компьютерного времени или остановки вручную. Обычно для лучшей сопоставимости результатов в расчётах используется фиксированное значение физического времени, такое же, как в соответствующем биологическом эксперименте. Максимально допустимое значение физического времени $2 \cdot 10^9$ с (свыше 60 лет). Значение этого параметра может быть изменено в файле заданий.

Startup duration Указывает продолжительность (в секундах) начального этапа, в ходе которого модель «разгоняется». Шаг изменения кнопками «меньше/больше» – 1 час. Когда физическое время в модели достигает заданного значения, сбрасываются все накопленные к этому моменту значения счетчиков для всех динамических объектов, так что все события, случившиеся на этапе разгона, в частности, выполненные транскрипции гена, не включаются в конечные результаты моделирования. Нулевое значение означает, что этап разгона не используется. Значение этого параметра может быть изменено в файле задания.

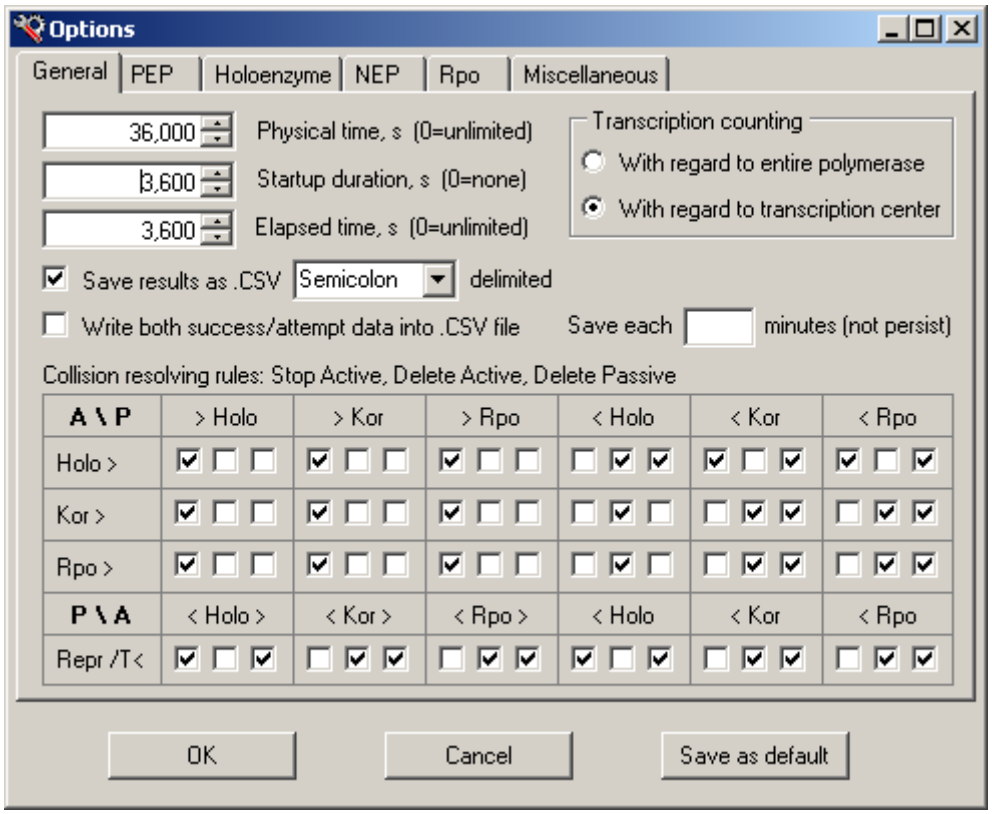


Рис. 13. Общие параметры конфигурации.

Elapsed time Указывает максимальную продолжительность в секундах компьютерного счета, после которой траектория будет принудительно остановлена. Шаг изменения кнопками «меньше/больше» – 100 с. Значение 0 означает, что компьютерное время счета не ограничивается, и модель будет работать до

достижения заданного физического времени или остановки вручную. Используется в пакетном режиме в качестве защитной меры против зацикливания и других ошибочных ситуаций. Максимально допустимое значение компьютерного времени 100000 с (более суток).

Save result as .CSV Если в этом пункте сделана пометка, то в дополнение к стандартному выходному файлу протокола с расширением **.log** результаты моделирования будут сформированы в виде файла с расширением **.csv**, содержащего значения с символами-разделителями, а также строку с заголовками полей данных. В дальнейшем такой файл может быть проимпортирован в электронную таблицу Excel или иным образом использован для автоматической обработки.

... delimited Указывает символ, который будет использован в качестве разделителя в файле формата **.csv**. В выпадающем списке имеются следующие варианты:

Comma	запятая (следует использовать с осторожностью, т.к. в случае российской локализации может затруднить импорт файла)
Semicolon	точка с запятой
Tab	символ табуляции
Blank	пробел

Write both ... Если в этом пункте сделана пометка, то в выдаваемом по умолчанию файле формата **.csv** для каждого объекта задачи будут приведены значения двух счетчиков (для гена – число начатых и завершенных транскрипций, для промотора – число попыток связывания полимеразы и число успешных посадок, и т.д.), иначе – только основного счетчика (успешных событий). Эта настройка касается только формата по умолчанию, тогда как в файле заданий можно заказать другой конкретный формат по желанию пользователя.

Save each ... minutes Позволяет выдавать результаты в файл формата **.csv** не только в конце траектории, но и в промежуточных точках с заданным шагом физического времени в минутах. Эта настройка не сохраняется в файле конфигурации и, кроме того, может быть изменена в файле задания.

Transcription counting Указывает, как фиксируется начало/конец транскрипции гена (см. примечания в п. 1.3.2). Если отмечен верхний вариант, то транскрипция фиксируется по полной длине РНК-полимеразы, если нижний – только по центру транскрипции.

В нижней части формы имеются три кнопки, которые можно нажать в любой момент настройки конфигурации, т.е. на любой закладке. Кнопки имеют следующие функции:

OK Принять сделанные изменения и выйти из режима настройки конфигурации. Сделанные изменения НЕ сохраняются в файле конфигурации и действуют только до выхода из программы.

Cancel Отменить все изменения, сделанные после последней записи конфигурации в файл (не только на текущей, но и на всех закладках!) и выйти из режима настройки конфигурации.

Save as default Сохранить сделанные изменения в первоначальном файле конфигурации. Если в будущем программа снова будет запущена с тем же файлом конфигурации, то будут использоваться новые значения (это не касается тех настроек, про которые в явном виде сказано, что они не сохраняются в файле конфигурации.

2.3.1.2. Закладка PEP

На этой закладке (Рис. 14) задаются принимаемые по умолчанию значения параметров PEP-промоторов. Эти значения используются только при добавлении к задаче нового PEP-промотора и не влияют на уже существующие промоторы.

Параметры устанавливаются раздельно для разных сигма-субъединиц. В текущей версии предусмотрено шесть типов сигма-субъединицы, Sig1 – Sig6, и только один параметр – интенсивность связывания холофермента с этим промотором (Binding), которая в нашей модели есть параметр λ пуассоновского процесса (подробнее см. раздел 1.3.3).

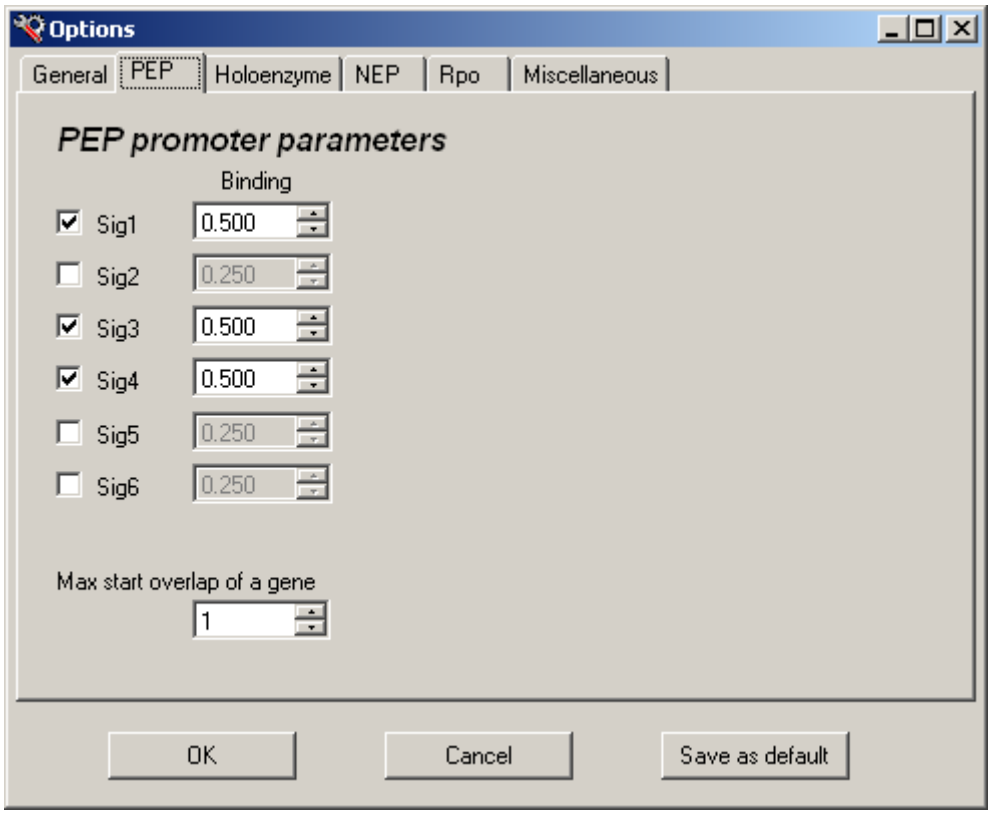


Рис. 14. Начальные параметры PEP-промотора.

В задачах реально могут участвовать только те типы сигма-субъединицы, перед которыми на этой закладке стоит пометка. В диалоговом окне редактирования параметров PEP-промотора (п. 2.3.2.2) для выбора будут предлагаться только такие сигма-субъединицы. Значения интенсивности связывания могут быть в диапазоне от 0.0 до 10.0, шаг изменения кнопками «больше/меньше» равен 0.01.

В нижней части окна указывается максимально допустимая величина перекрытия гена центром транскрипции PEP-промотора (значение 1 соответствует 5'-краю гена). Минимально допустимая величина (отрицательная) не лимитируется. Эта настройка не имеет отношения собственно к моделированию, а используется только для управления диалоговыми окнами установки параметров объектов задачи.

2.3.1.3. Закладка Holoenzyme

Эта закладка (Рис. 15) фактически является продолжением предыдущей; на ней устанавливаются принимаемые по умолчанию значения параметров холофермента, который связывается с PEP-промотором. Эти значения используются только при добавлении к задаче нового PEP-промотора и не влияют на уже существующие промоторы.

Значения параметров устанавливаются раздельно по типам сигма-субъединицы (и только для тех типов, которые были отмечены на закладке PEP). Указывается длина сигма-субъединицы Length (т.е. число нуклеотидов на цепи ДНК, которые она связывает) и интенсивность абортивного процесса Aborting (т.е. параметр μ соответствующего пуассоновского процесса, подробнее см. раздел 1.3.4).

В правой части закладки указываются принимаемые по умолчанию значения следующих параметров кор-фермента:

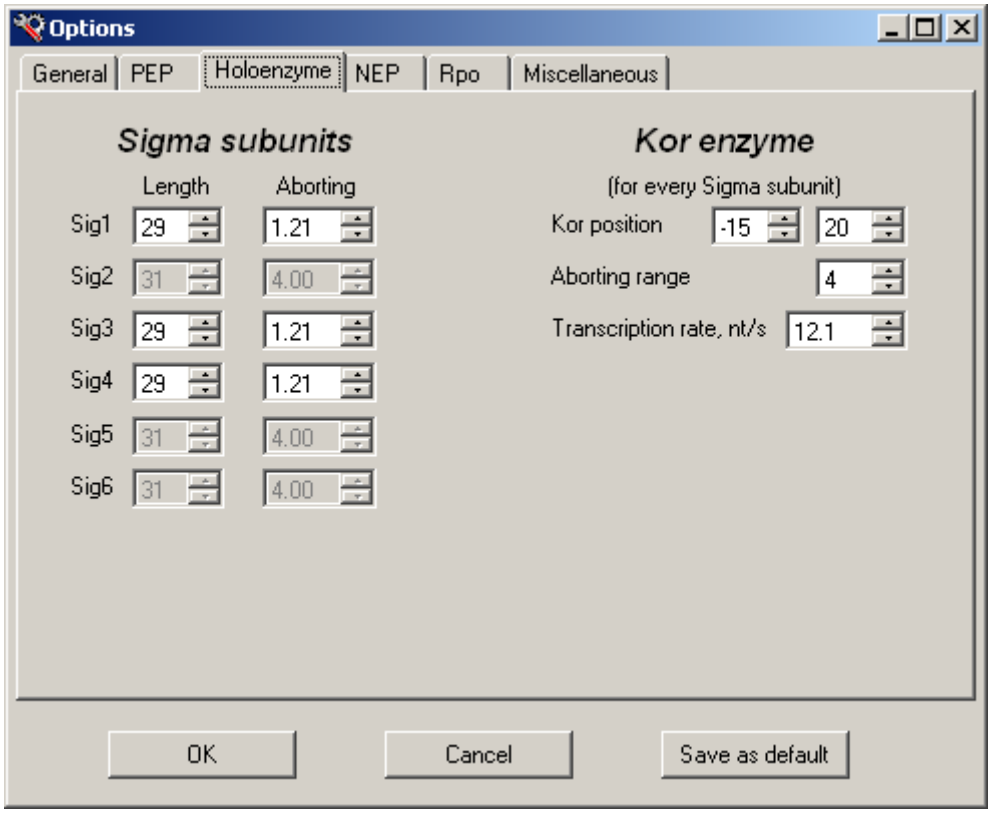


Рис. 15. Начальные параметры холофермента.

Kor position Положение 5'-начала (левое окошко формы) и 3'-конца (правое окошко) кора относительно центра транскрипции. Здесь следует помнить, что центру транскрипции приписана позиция +1, предыдущий (в направлении 5') нуклеотид имеет номер позиции –1, а нулевой номер позиции не используется. Этот параметр также неявно определяет длину кора.

Aborting range Амплитуда *r* абортивного процесса (см. раздел 1.3.4).

Transcription rate Скорость *R* движения кора по цепи ДНК, [нт/с]. В нашей модели предполагается, что эта скорость постоянная и одинакова как в ходе абортивных процессов, так и при последующей транскрипции. Тем не менее, имеется возможность изменять скорость элонгации по ходу траектории, моделируя желаемую температурную зависимость. Подробнее о динамическом изменении параметров модели говорится в разделе 1.4.2.

2.3.1.4. Закладка NEP

На этой закладке (Рис. 16) задаются принимаемые по умолчанию значения параметров NEP-промоторов. Эти значения используются только при добавлении к задаче нового NEP-

промотора и не влияют на уже существующие промоторы. Параметры устанавливаются раздельно для различных полимераз фагового типа. В текущей версии предусмотрено пять типов фаговых полимераз и только один параметр – интенсивность связывания полимеразы с промотором (Binding), которая в нашей модели есть параметр λ пуассоновского процесса (подробнее см. раздел 1.3.7).

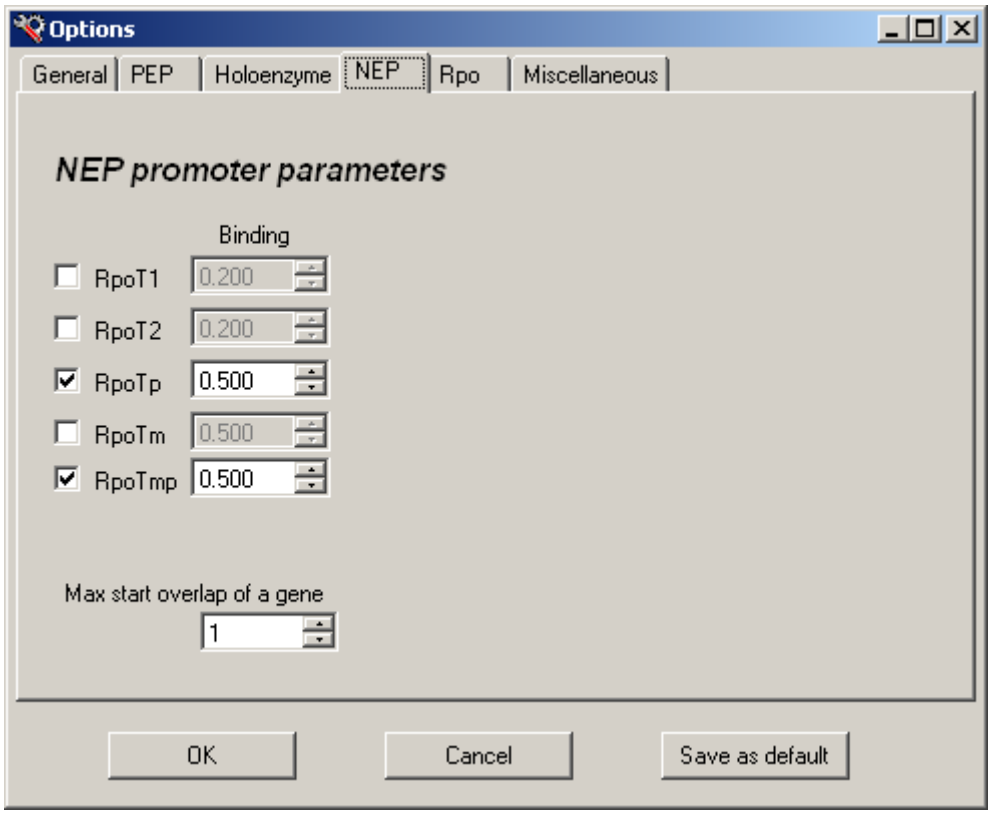


Рис. 16. Начальные параметры NEP-промотора.

В задачах реально могут участвовать только те типы фаговых полимераз, перед которыми на этой закладке стоит пометка. В диалоговом окне редактирования параметров NEP-промотора (п. 2.3.2.3) для выбора будут предлагаться только такие полимеразы фагового типа. Значения интенсивности связывания могут быть в диапазоне от 0.0 до 10.0, шаг изменения кнопками «больше/меньше» равен 0.01.

В нижней части окна указывается максимально допустимая величина перекрытия гена центром транскрипции NEP-промотора (значение 1 соответствует 5'-краю гена). Минимально допустимая величина (отрицательная) не лимитируется. Эта установка не имеет отношения собственно к моделированию, а используется только для управления диалоговыми окнами установки параметров объектов задачи.

2.3.1.5. Закладка Rpo

Эта закладка (Рис. 17) является продолжением предыдущей; на ней устанавливаются принимаемые по умолчанию значения параметров полимеразы фагового типа, которая связывается с NEP-промотором. Эти значения используются только при добавлении к задаче нового NEP-промотора и не влияют на уже существующие промоторы.

Значения параметров устанавливаются раздельно для разных полимераз фагового типа (и только для тех типов, которые были отмечены на предыдущей закладке). В настоящей версии программы имеются следующие три параметра:

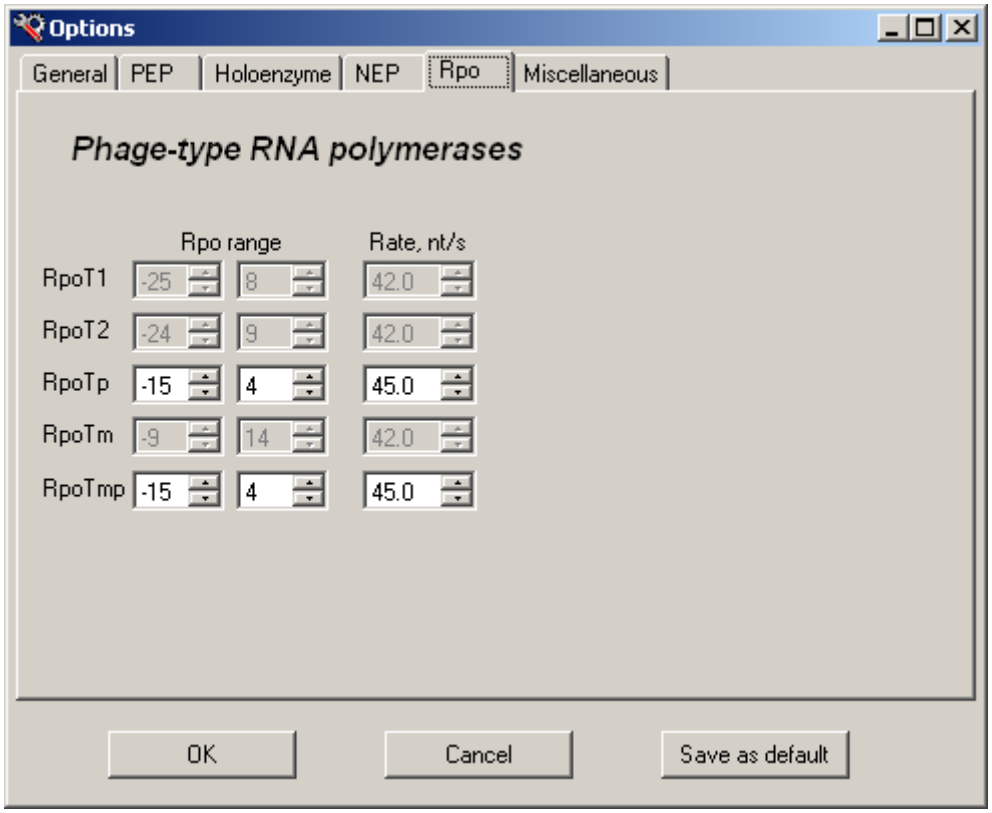


Рис. 17. Начальные параметры полимеразы фагового типа.

Rpo range (левое поле). Положение 5'-начала полимеразы фагового типа относительно центра транскрипции.

(правое поле). Положение 3'-конца полимеразы фагового типа относительно центра транскрипции.

Здесь следует помнить, что центру транскрипции приписана позиция +1, предыдущий (в направлении 5') нуклеотид имеет номер позиции –1, а нулевой номер позиции не используется. Эти два параметра также неявно определяют длину полимеразы фагового типа.

Rate Скорость движения полимеразы фагового типа по цепи ДНК, [нт/с]. В нашей модели предполагается, что эта скорость постоянная. (См. также о возможностях динамического изменения параметров модели в разделе 1.4.2).

2.3.1.6. Закладка Miscellaneous

На этой закладке (Рис. 18) собраны различные вспомогательные параметры, которые не сохраняются в файле конфигурации. Они обычно устанавливаются для моделирования только одной траектории или серии расчетов.

Random generator Устанавливает режим работы датчика псевдослучайных чисел, который по существу определяет весь ход моделирования вдоль траектории. В первой позиции (**Time dependent**) для инициализации датчика используется значение компьютерных часов на момент запуска траектории (и это стартовое значение датчика заносится в окошко против третьей позиции данного органа управления). Во второй позиции (**Last seed value**) будет использовано то же значение, что и для последней запущенной траектории, что позволяет воспроизвести предыдущий расчет в случае каких-то сбоев.

Если отметить третью позицию этого переключателя, то для инициализации датчика псевдослучайной последовательности будет применяться то значение, которое указано в окошке, и повторные траектории будут давать те же самые результаты, что может быть полезно при отладке.

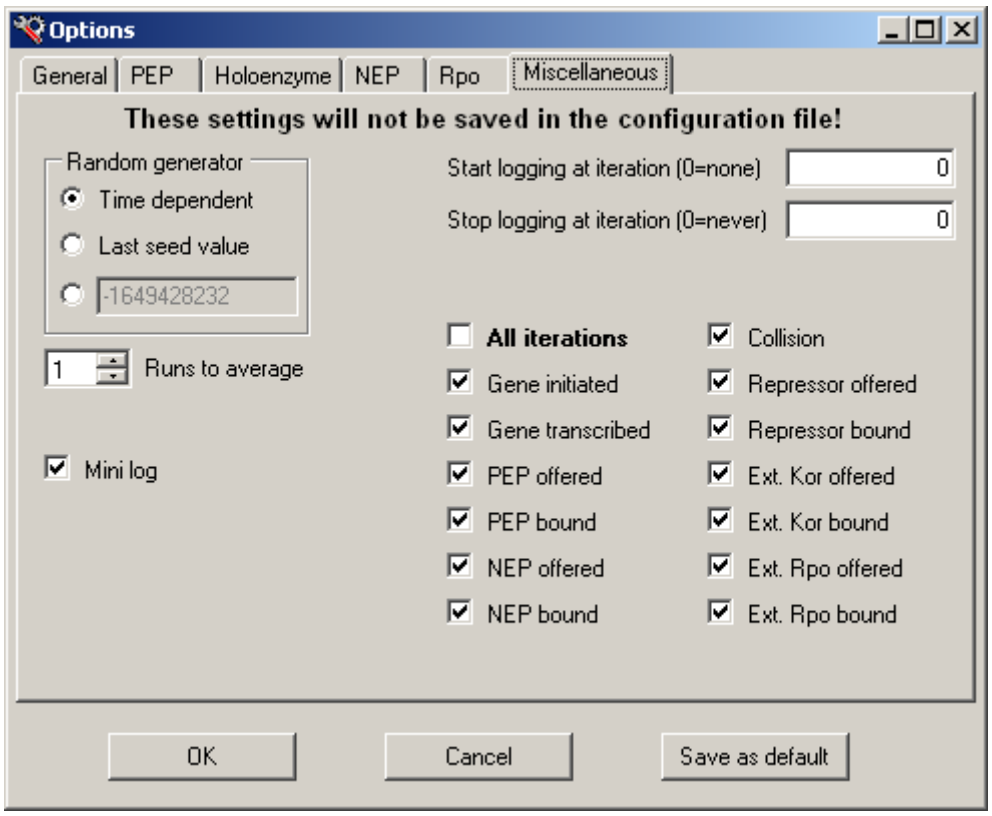


Рис. 18. Прочие параметры конфигурации.

Runs to average Указывает число траекторий моделирования в пучке (см. п. 2.2.4.2). По умолчанию размер пучка равен 1, т.е. моделирование ведется в режиме одиночной траектории, без усреднения. Значение этого параметра может быть изменено в файле заданий.

Mini log В этом режиме (который включен по умолчанию) в файл протокола выдается только информация о завершении каждого задания в составе пакета. При моделировании только одной траектории (в интерактивном режиме) можно получить в протоколе обширную дополнительную информацию для анализа и отладки. Состав выдачи в файл протокола определяют остальные параметры, представленные на этой закладке; они кратко описываются ниже.

Start logging ... Номер итерации, с которой начинается протоколирование (начиная с 1).

Stop logging ... Номер итерации, на которой заканчивается протоколирование (если указано 0, то протокол продолжается до конца траектории).

Следует учитывать, что протоколирование значительно замедляет работу программы; кроме того, полный протокол траектории может состоять из десятков миллионов записей (т.е. строк) – на каждой итерации формируется запись протокола. Поскольку номер итерации выдается в строке состояния, то один из способов ограничить выдачу – это начать протоколирование не с начала траектории, а с того момента, когда «что-то пошло не так». Другой способ основан на возможности фильтрации протоколируемых событий: в протокол попадают не все итерации, а только интересующего вида. Например, можно не выдавать в

протокол все итерации, где событие состоит просто в сдвиге РНК-полимеразы на следующий нуклеотид и не возникает коллизии с другими объектами.

Чтобы выводить в протокол все итерации, нужно включить пометку **All iterations**. Если эта пометка снята, то прочие пометки включают или выключают протоколирование следующих видов итераций модели:
Gene initiated – начало транскрипции какого-либо гена
Gene transcribed – окончание транскрипции какого-либо гена
PEP offered – любая попытка посадки холофермента на PEP-промотор
PEP bound – успешная посадка холофермента на PEP-промотор
NEP offered – любая попытка посадки полимеразы фагового типа на NEP-промотор
NEP bound – успешная посадка полимеразы фагового типа на NEP-промотор
Collision – любое столкновение движущегося объекта с (м.б. временно) неподвижным
Repressor offered – любая попытка активизации репрессора
Repressor bound – успешная активизация репрессора (сел белок, сложилась шпилька...)
Ext. Kor offered – любая попытка генерации кора внешним источником кор-ферментов
Ext. Kor bound – успешное связывание кор-фермента от внешнего источника
Ext. Rpo offered – попытка генерации полимеразы фагового типа внешним источником
Ext. Rpo bound – успешное связывание полимеразы фагового типа от внешнего источника

2.3.2. Установка и изменение параметров объектов

Устанавливать параметры объектов требуется при создании новой задачи или добавлении нового объекта в существующую задачу. Кроме того, при корректировке существующей задачи иногда возникает необходимость изменить параметры её объектов. Указанные операции выполняются с помощью диалогового окна, специфичного для конкретного типа объекта. Режимы установки и изменения принципиально не отличаются; разница в том, что во первом случае поля диалогового окна пустые, а во втором случае они уже заполнены текущими значениями. Кроме того, при изменении параметров существующего объекта его имя изменить нельзя (соответствующее поле притушено); для этой цели следует использовать функцию **Rename** контекстного меню объекта (см. п. 2.2.3.2).

2.3.2.1. Диалоговое окно установки параметров гена

Внешний вид окна в режиме установки параметров нового гена, после того как введено его имя, представлен на Рис. 19.

Примечание: В процессе ввода данных проводится их контроль, поэтому в качестве общего правила следует заполнять поля диалогового окна сверху вниз (насколько возможно, это обеспечивается тем, что элементы форм становятся активными только после заполнения предыдущих элементов). В частности, пока не указано имя объекта, все остальные параметры недоступны для редактирования.

Gene name В этом поле требуется указать имя, которое будет присвоено объекту. Обычно используется общепринятое обозначение гена. Все имена объектов в пределах одной задачи должны быть уникальными. Заглавные и строчные буквы различаются. Максимальная длина имени 31 символ, но для работы удобнее ограничиться 7–8 символами, т.к. длинные имена будут обрезаны при отображении объектов в рабочей области окна. При работе в интерактивном режиме имена могут составляться из любых символов, включая пробел.
Примечание: Для обработки задачи в пакетном режиме имена объектов не должны содержать пробелы, знаки табуляции, и символы . , = : % ! ?.

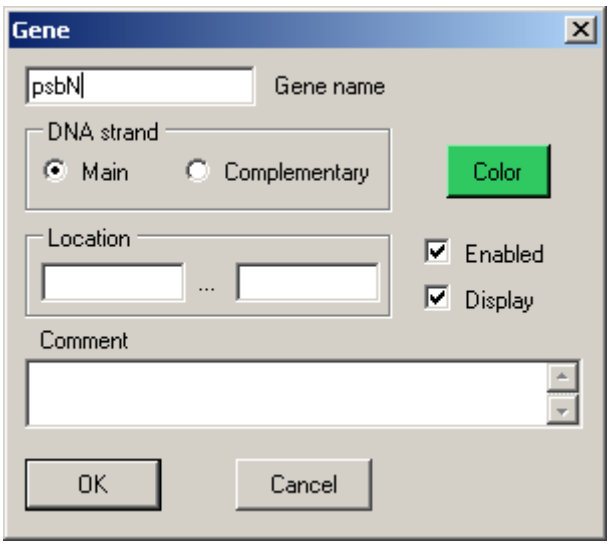


Рис. 19. Диалоговое окно параметров гена.

DNA strand В этой группе необходимо указать, на какой цепи ДНК находится ген, основной (**Main**) или комплементарной (**Complementary**).

Color Цвет кнопки совпадает с цветом заливки символа гена в рабочей области окна программы. При желании, нажатием этой кнопки можно указать любой желаемый цвет (выдается стандартный диалог Windows).

Location В этой группе необходимо указать позиции левого (в левом окошке) и правого (в правом окошке) краев гена на основной цепи ДНК. Например, сначала можно указать левый край (в зависимости от цепи, на которой находится ген, это будет его 5'-начало или 3'-конец), а потом правый край. Отсчет позиций идет с 1 (и в пределах длины предварительно загруженной последовательности). Допускается в левом окошке одновременно указывать позиции левого и правого краев гена в той форме, как это делается в файлах ГенБанка, например, **44827..44913** (в качестве разделителя также можно использовать любое число пробелов или символов из набора . , : ; - | / \ ). Заметим, что в режиме кольцевых последовательностей позиция левого края может быть и больше позиции правого края.

Enabled По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет принимать участие в моделировании (он отображается притушенным в рабочей области окна). См. также п. 2.2.3.2.

Display По умолчанию эта пометка стоит; если убрать ее, то данный объект не будет отображаться в рабочей области окна, но будет участвовать в моделировании (т.н. «скрытый» объект). См. также п. 2.2.3.2.

Comment В этом поле можно указать произвольный текст комментария, который будет появляться с прочими данными во всплывающей подсказке к объекту (см. п. 2.2.3) или вместе с данными о местоположении объектов (Рис. 5). Комментарий может занимать несколько строк, но при необходимости будет обрезаться при выводе.

В нижней части диалога имеются две кнопки:

OK Сохранить параметры и закрыть диалоговое окно. Если при контроле будут обнаружены ошибки, выдается соответствующее сообщение и программа снова вернётся в то же диалоговое окно, чтобы исправить эти ошибки.

Cancel Отменить сделанные изменения и закрыть диалоговое окно.

В режиме корректировки параметров существующего объекта появляется еще третья кнопка:

Delete Удалить этот объект из задачи и закрыть диалоговое окно.

2.3.2.2. Диалоговое окно установки параметров PEP-промотора

Внешний вид окна в режиме корректировки параметров существующего PEP-промотора представлен на Рис. 20.

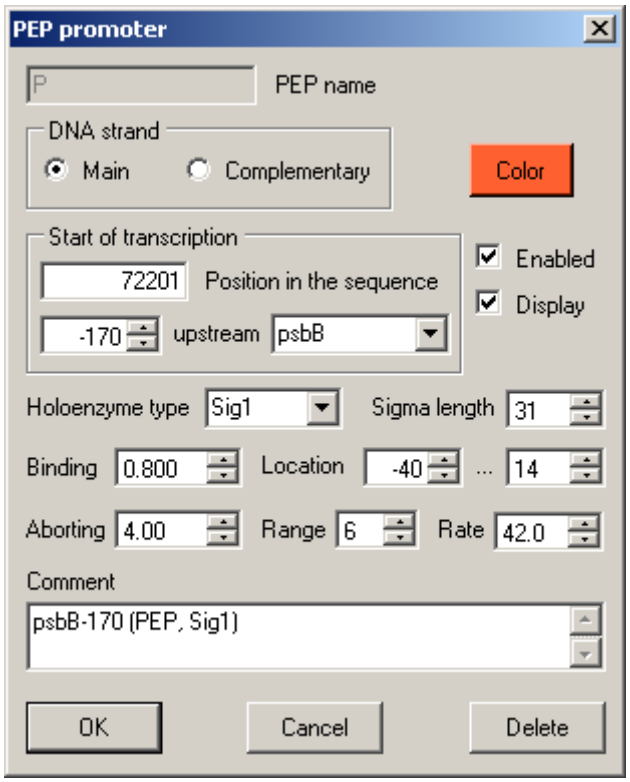


Рис. 20. Диалоговое окно параметров PEP-промотора.

Для нового промотора параметры заносятся в следующем порядке.

PEP name В этом поле требуется указать имя, которое будет присвоено объекту (например, для промоторов иногда используют имя гена и смещение, например, psbB-170). Все имена объектов в пределах одной задачи должны быть уникальными. Заглавные и строчные буквы различаются. Максимальная длина имени 31 символ, но для работы удобнее ограничиться 7–8 символами, т.к. длинные имена обрезаются при отображении объектов в рабочей области окна. В интерактивном режиме имена могут составляться из любых символов, включая пробел.
Примечание: Для обработки в пакетном режиме имена объектов не должны содержать пробелы, знаки табуляции, и символы . , = : % ! ?.

DNA strand В этой группе необходимо указать, на какой цепи ДНК находится промотор, основной (**Main**) или комплементарной (**Complementary**).

Color Цвет кнопки совпадает с цветом заливки символа PEP-промотора в рабочей области окна программы. При желании, нажатием этой кнопки можно указать любой желаемый цвет (выдается стандартный диалог Windows).

Start of transcription В этой группе параметров необходимо одним из двух способов задать

положение старта транскрипции: можно либо указать абсолютную позицию старта транскрипции в поле **Position in the sequence**, либо выбрать ген в выпадающем списке **Gene name** (будут показаны гены только той цепи ДНК, которая выбрана для промотора) и затем в поле слева от слова **upstream**, где до того стояло неиспользуемое значение 0, указать позицию центра транскрипции относительно 5'-начала гена (обычно это отрицательное значение, максимальное положительное значение хранится в конфигурации, см. п. 2.3.1.2). При выборе одного из способов соответствующие данные для другого способа вносятся автоматически. Аналогично, при корректировке абсолютной позиции старта для существующего промотора будет изменена относительная позиция (и наоборот).

Enabled По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет принимать участие в моделировании (он отображается притушенным в рабочей области окна). См. также п. 2.2.3.2.

Display По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет отображаться в рабочей области окна, но будет участвовать в моделировании (т.н. «скрытый» объект). См. также п. 2.2.3.2.

Holoenzyme type В этом выпадающем списке необходимо выбрать тип холофермента, т.е. сигма-субъединицу, от которой зависит данный промотор. В списке будут предлагаться только те сигма-субъединицы, которые были выбраны для работы в конфигурации программы (см. п. 2.3.1.2). После этого остальные параметры заполнятся теми значениями по умолчанию для выбранной сигма-субъединицы, которые были указаны в конфигурации программы (см. пп. 2.3.1.2, 2.3.1.3). По желанию, можно оставить эти значения или изменить.

Sigma length Длина сигма-субъединицы, [нт].

Binding Интенсивность связывания промотора, λ.

Location Положение 5'- и 3'-концов холофермента относительно начала транскрипции.

Aborting Интенсивность абортивного процесса, μ.

Range Амплитуда абортивного процесса, r [нт].

Rate Скорость движения кор-фермента, R [нт/с].

Comment В этом поле можно указать произвольный текст комментария, который будет появляться с прочими данными во всплывающей подсказке к объекту (см. п. 2.2.3) или вместе с данными о местоположении объектов (Рис. 5). Комментарий может занимать несколько строк, но при необходимости будет обрезан при выводе.

В нижней части диалога имеются две кнопки:

OK Сохранить параметры и закрыть диалоговое окно. Если при контроле будут обнаружены ошибки, выдается соответствующее сообщение и программа снова вернётся в диалоговое окно, чтобы исправить эти ошибки.

Cancel Отменить сделанные изменения и закрыть диалоговое окно.

В режиме корректировки параметров существующего объекта появляется еще третья кнопка:

Delete Удалить этот объект из задачи и закрыть диалоговое окно.

2.3.2.3. Диалоговое окно установки параметров NEP-промотора

Внешний вид окна в режиме корректировки параметров существующего NEP-промотора представлен на Рис. 21.

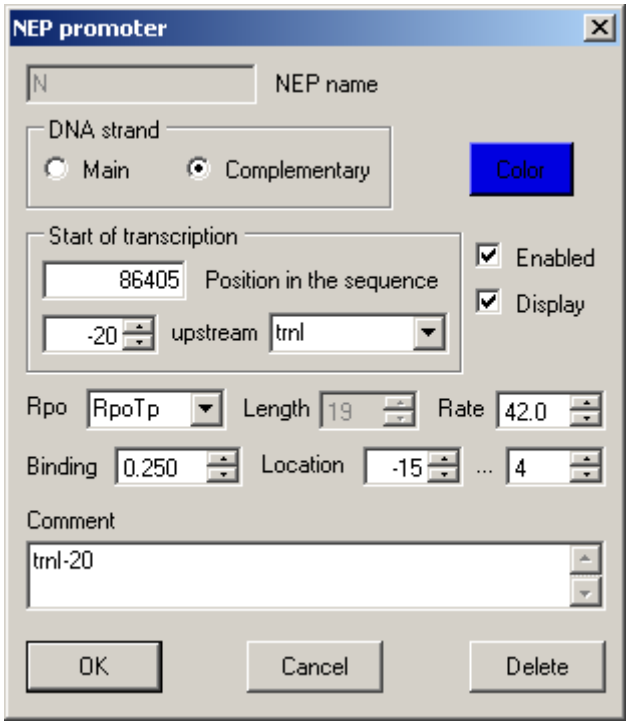


Рис. 21. Диалоговое окно параметров NEP-промотора.

Для нового промотора параметры заносятся в следующем порядке.

NEP name В этом поле требуется указать имя, которое будет присвоено объекту (часто используют имя гена и смещение, например, ycf1-39). Все имена объектов в пределах одной задачи должны быть уникальными. Заглавные и строчные буквы различаются. Максимальная длина имени 31 символ, но для работы удобнее ограничиться 7–8 символами, т.к. длинные имена обрезаются при отображении объектов в рабочей области окна. В интерактивном режиме имена могут составляться из любых символов, включая пробел.
Примечание: Для обработки в пакетном режиме имена объектов не должны содержать пробелы, знаки табуляции, и символы . , = : % ! ?.

DNA strand В этой группе необходимо указать, на какой цепи ДНК находится промотор, основной (**Main**) или комплементарной (**Complementary**).

Color Цвет кнопки совпадает с цветом заливки символа NEP-промотора в рабочей области окна программы. При желании, нажатием этой кнопки можно указать любой желаемый цвет (выдается стандартный диалог Windows).

Start of transcription В этой группе одним из двух способов необходимо задать положение старта транскрипции: можно либо указать абсолютную позицию старта транскрипции в поле **Position in the sequence**, либо выбрать ген в выпадающем списке **Gene name** (будут показаны гены только той цепи ДНК, которая выбрана для промотора) и затем в поле слева от слова **upstream**, где до того стояло неиспользуемое значение 0, указать позицию центра транскрипции относительно 5'-начала гена (обычно это отрицательное значение, максимальное положительное значение хранится в конфигурации, см. п. 2.3.1.4). При выборе одного из способов соответствующие данные для

другого способа вносятся автоматически. Аналогично, при корректировке абсолютной позиции старта для существующего промотора будет изменена относительная позиция (и наоборот).

Enabled По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет принимать участие в моделировании (он отображается притушенным в рабочей области окна). См. также п. 2.2.3.2.

Display По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет отображаться в рабочей области окна, но будет участвовать в моделировании (т.н. «скрытый» объект). См. также п. 2.2.3.2.

Rpo В этом выпадающем списке необходимо выбрать полимеразу фагового типа, от которой зависит данный промотор. В списке будут предлагаться только те полимеразы, которые были выбраны для работы в конфигурации программы (п. 2.3.1.4). После этого остальные параметры заполнятся теми значениями по умолчанию для выбранной полимеразы фагового типа, которые были указаны в конфигурации программы (см. пп. 2.3.1.4, 2.3.1.5). По желанию, можно оставить предложенные значения или изменить их.

Length Длина полимеразы фагового типа, [нт]. Этот параметр не указывается и напрямую не может быть изменён, его значение определяется параметрами Location.

Rate Скорость движения полимеразы фагового типа, [нт/с].

Binding Интенсивность связывания промотора, λ.

Location Положение 5'- и 3'-концов полимеразы относительно начала транскрипции.

Comment В этом поле можно указать произвольный текст комментария, который будет появляться с прочими данными во всплывающей подсказке к объекту (см. п. 2.2.3) или вместе с данными о местоположении объектов (Рис. 5). Комментарий может занимать несколько строк, но при необходимости будет обрезан при выводе.

В нижней части диалога имеются две кнопки:

OK Сохранить параметры и закрыть диалоговое окно. Если при контроле будут обнаружены ошибки, выдается соответствующее сообщение и программа снова вернётся в то же диалоговое окно, чтобы исправить эти ошибки.

Cancel Отменить сделанные изменения и закрыть диалоговое окно.

В режиме корректировки параметров существующего объекта появляется еще третья кнопка:

Delete Удалить этот объект из задачи и закрыть диалоговое окно.

2.3.2.4. Диалоговое окно установки параметров репрессора

Внешний вид окна в режиме установки параметров существующего репрессора представлен на Рис. 22. Напомним, что название «репрессор» условное, т.к. данный объект можно рассматривать и как терминатор, причем с регулируемым протеканием РНК-полимераз отдельно в каждую сторону (подробности изложены в п. 1.3.10). В отличие от других объектов, сайт репрессии логически не сопоставлен какой-то цепи ДНК, а относится к обеим цепям. По этой причине в диалоговом окне отсутствует переключатель для выбора между основной и комплементарной цепью ДНК, а в выпадающем списке генов для указания относительного положения сайта репрессии присутствуют гены обеих цепей. Кроме того, при указании относительного положения сайта пользователь может выбрать, находится ли сайт перед (upstream) или после (downstream) выбранного гена, с учётом направления цепи,

на которой этот ген расположен. Переключение производится щелчком мыши по слову “upstream” (или “downstream”), которое ведёт себя как обычная ссылка.

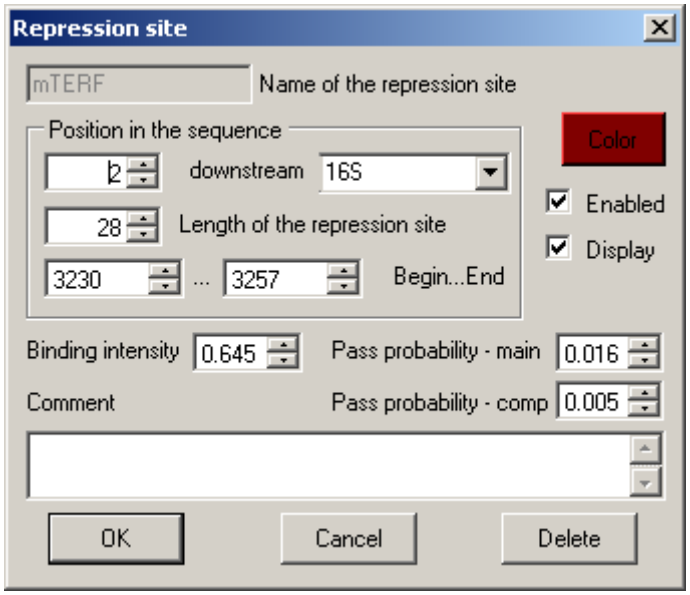


Рис. 22. Диалоговое окно параметров сайта репрессии.

Name of the repression site В этом поле требуется указать имя, которое будет присвоено объекту. Все имена объектов в пределах одной задачи должны быть уникальными. Заглавные и строчные буквы различаются. Максимальная длина имени 31 символ, но для работы удобнее ограничиться 7–8 символами, т.к. длинные имена обрезаются при отображении объектов в рабочей области окна. В интерактивном режиме имена могут составляться из любых символов, включая пробел.
Примечание: Для обработки в пакетном режиме имена объектов не должны содержать пробелы, знаки табуляции, и символы . , = : % ! ?.

Position in the sequence В этой группе указывается положение сайта репрессии одним из двух способов: можно либо (1) в выпадающем списке **Gene name** выбрать ген, вблизи начала или конца которого находится сайт репрессии (в списке будут показаны гены на обеих цепях ДНК), и его относительное расположение – перед (**upstream**) или после (**downstream**) этого гена, затем в поле слева указать смещение центра транскрипции соответственно от 5'-начала гена (отрицательное значение) или от 3'-конца гена (положительное значение), и после этого в поле **Length of the repression site** указать длину сайта; либо (2) в полях **Begin ... End** указать абсолютные позиции левого и правого краев сайта в основной цепи ДНК. При выборе одного способа соответствующие данные для другого способа будут внесены автоматически. Аналогично, при корректировке одного из параметров для существующего репрессора будут изменяться другие параметры.

Color Цвет кнопки совпадает с цветом заливки символа репрессора в рабочей области окна программы. При желании, нажатием этой кнопки можно указать любой желаемый цвет (выдается стандартный диалог Windows).

Enabled По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет принимать участие в моделировании (он отображается притушенным в рабочей области окна). См. также п. 2.2.3.2.

Display По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет отображаться в рабочей области окна, но будет участвовать в моделировании (т.н. «скрытый» объект). См. также п. 2.2.3.2.

Binding Интенсивность связывания с сайтом белка-репрессора или возникновения на сайте терминатора, λ.

Pass probability - main Вероятность прохождения РНК-полимеразы через сайт по основной цепи, при условии что репрессор/терминатор активен, т.е. связан с сайтом (иначе эта вероятность равна 1). Логика взаимодействия полимеразы с репрессором подробно описана в п. 1.3.10.

Pass probability - comp Вероятность прохождения РНК-полимеразы через сайт по комплементарной цепи, при условии что репрессор/терминатор активен, т.е. связан с сайтом (иначе эта вероятность равна 1). Логика взаимодействия полимеразы с репрессором подробно описана в п. 1.3.10.

Comment В этом поле можно указать произвольный текст комментария, который будет появляться с прочими данными во всплывающей подсказке к объекту (см. п. 2.2.3) или вместе с данными о местоположении объектов (Рис. 5). Комментарий может занимать несколько строк, но при необходимости будет обрезан при выводе.

В нижней части диалога имеются две кнопки:

OK Сохранить параметры и закрыть диалоговое окно. Если при контроле будут обнаружены ошибки, выдается соответствующее сообщение и программа снова вернётся в диалоговое окно, чтобы исправить эти ошибки.

Cancel Отменить сделанные изменения и закрыть диалоговое окно.

В режиме корректировки параметров существующего объекта появляется еще третья кнопка:

Delete Удалить этот объект из задачи и закрыть диалоговое окно.

2.3.2.5. Диалоговое окно установки параметров внешнего источника кор-ферментов

Внешний вид окна в режиме установки параметров нового внешнего источника кор-ферментов после указания его имени представлен на Рис. 23.

Параметры заносятся в следующем порядке.

Name of ... В этом поле требуется указать имя, которое будет присвоено объекту. Все имена объектов в пределах одной задачи должны быть уникальными. Заглавные и строчные буквы различаются. Максимальная длина имени 31 символ, но для работы удобнее ограничиться 7–8 символами, т.к. длинные имена обрезаются при отображении объектов в рабочей области окна. В интерактивном режиме имена могут составляться из любых символов, включая пробел.
Примечание: Для обработки в пакетном режиме имена объектов не должны содержать пробелы, знаки табуляции, и символы . , = : % ! ?.

DNA strand В этой группе необходимо указать, на какой цепи ДНК находится внешний источник, основной (**Main**) или комплементарной (**Complementary**).

Color Цвет кнопки совпадает с цветом заливки символа внешнего источника кор-ферментов в рабочей области окна программы. При желании, нажатием этой кнопки можно указать любой желаемый цвет (выдается стандартный диалог Windows).

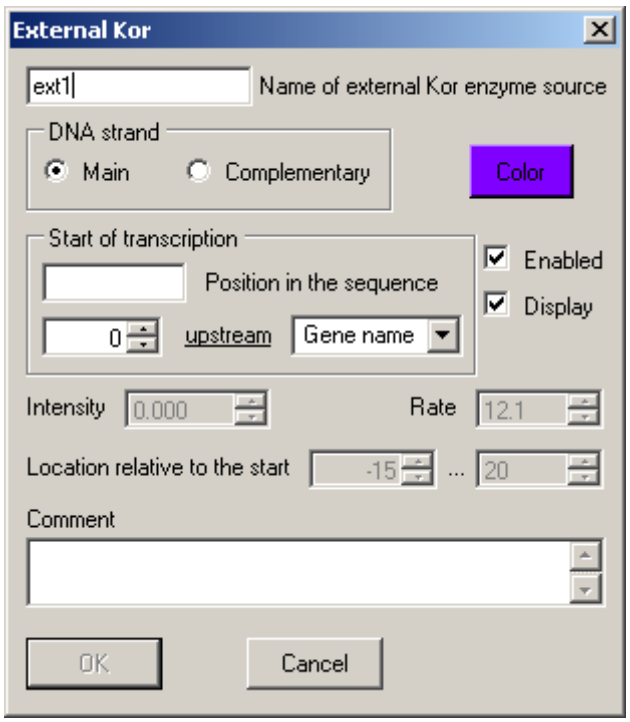


Рис. 23. Диалоговое окно параметров внешнего источника кор-ферментов.

Start of transcription В этой группе необходимо одним из двух способов задать положение старта транскрипции: можно либо указать абсолютную позицию центра транскрипции кор-фермента в поле **Position in the sequence**, либо выбрать ген в выпадающем списке **Gene name** (обычно первый или последний ген локуса), выбрать направление (**upstream** или **downstream**) и затем в поле слева от него указать положение центра транскрипции относительно ближайшего края выбранного гена (как правило, отрицательное значение для upstream и положительное – для downstream). При выборе одного из способов соответствующие данные для другого способа вносятся автоматически. То же происходит и при изменении параметров существующего внешнего источника кор-ферментов. Выбор направления производится щелчком мыши по соответствующему слову, которое ведёт себя как ссылка.

Enabled По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет принимать участие в моделировании (он отображается притушенным в рабочей области окна). См. также п. 2.2.3.2.

Display По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет отображаться в рабочей области окна, но будет участвовать в моделировании (т.н. «скрытый» объект). См. также п. 2.2.3.2.

После того, как указано положение внешнего источника, остальные параметры кор-фермента примут те значения по умолчанию, которые были указаны в конфигурации программы (см. п. 2.3.1.3). По желанию, можно оставить эти значения или изменить.

Intensity Интенсивность связывания кор-фермента, λ.

Rate Скорость движения кор-фермента, R [нт/с].

Location ... Положение 5'- и 3'-концов кор-фермента относительно начала транскрипции.

Comment В этом поле можно указать произвольный текст комментария, который будет появляться с прочими данными во всплывающей подсказке к объекту (см.

п. 2.2.3) или вместе с данными о местоположении объектов (Рис. 5). Комментарий может занимать несколько строк, но при необходимости будет обрезан при выводе.

В нижней части диалога имеются две кнопки:

OK Сохранить параметры и закрыть диалоговое окно. Если при контроле будут обнаружены ошибки, выдается соответствующее сообщение и программа снова вернётся в диалоговое окно, чтобы исправить эти ошибки.

Cancel Отменить сделанные изменения и закрыть диалоговое окно.

В режиме корректировки параметров существующего объекта появляется еще третья кнопка:

Delete Удалить этот объект из задачи и закрыть диалоговое окно.

2.3.2.6. Диалоговое окно установки параметров внешнего источника полимераз фагового типа

Внешний вид окна в режиме установки параметров нового внешнего источника полимераз фагового типа после указания его имени представлен на Рис. 24.

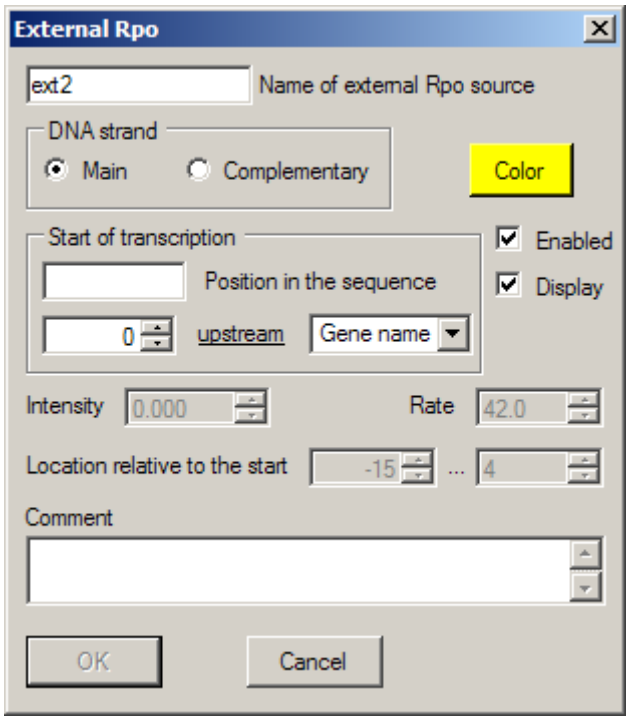


Рис. 24. Диалоговое окно параметров внешнего источника полимераз фагового типа.

Параметры заносятся в следующем порядке.

Name of ... В этом поле требуется указать имя, которое будет присвоено объекту. Все имена объектов в пределах одной задачи должны быть уникальными. Заглавные и строчные буквы различаются. Максимальная длина имени 31 символ, но для работы удобнее ограничиться 7–8 символами, т.к. длинные имена обрезаются при отображении объектов в рабочей области окна. В интерактивном режиме имена могут составляться из любых символов, включая пробел.

Примечание: Для обработки в пакетном режиме имена объектов не должны содержать пробелы, знаки табуляции, и символы . , = : % ! ?.

DNA strand В этой группе необходимо указать, на какой цепи ДНК находится внешний источник, основной (**Main**) или комплементарной (**Complementary**).

Color Цвет кнопки совпадает с цветом заливки символа внешнего источника полимераз фагового типа в рабочей области окна программы. При желании, нажатием этой кнопки можно указать любой желаемый цвет (выдается стандартный диалог Windows).

Start of transcription В этой группе необходимо одним из двух способов задать положение старта транскрипции: можно либо указать абсолютную позицию центра транскрипции полимеразы фагового типа в поле **Position in the sequence**, либо выбрать ген в выпадающем списке **Gene name** (обычно это первый или последний ген локуса), выбрать направление (**upstream** или **downstream**) и затем в поле слева от него указать положение центра транскрипции относительно ближайшего края выбранного гена (как правило, отрицательное значение для upstream и положительное – для downstream). При выборе одного из способов соответствующие данные для другого способа вносятся автоматически. То же происходит и при изменении параметров существующего внешнего источника полимераз фагового типа. Выбор направления производится щелчком мыши по соответствующему слову, которое ведёт себя как ссылка.

Enabled По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет принимать участие в моделировании (он отображается притушенным в рабочей области окна). См. также п. 2.2.3.2.

Display По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет отображаться в рабочей области окна, но будет участвовать в моделировании (т.н. «скрытый» объект). См. также п. 2.2.3.2.

После того, как указано положение внешнего источника, остальные параметры полимеразы фагового типа примут те значения по умолчанию, которые были указаны в конфигурации программы (см. п. 2.3.1.5). По желанию, можно оставить эти значения или изменить.

Intensity Интенсивность связывания полимеразы фагового типа, λ.

Rate Скорость движения полимеразы фагового типа, [нт/с].

Location ... Положение 5'- и 3'-концов полимеразы фагового типа относительно начала транскрипции.

Comment В этом поле можно указать произвольный текст комментария, который будет появляться с прочими данными во всплывающей подсказке к объекту (см. п. 2.2.3) или вместе с данными о местоположении объектов (Рис. 5). Комментарий может занимать несколько строк, но при необходимости будет обрезан при выводе.

В нижней части диалога имеются две кнопки:

OK Сохранить параметры и закрыть диалоговое окно. Если при контроле будут обнаружены ошибки, выдается соответствующее сообщение и программа снова вернётся в диалоговое окно, чтобы исправить эти ошибки.

Cancel Отменить сделанные изменения и закрыть диалоговое окно.

В режиме корректировки параметров существующего объекта появляется еще третья кнопка:

Delete Удалить этот объект из задачи и закрыть диалоговое окно.

2.3.2.7. Диалоговое окно установки параметров терминатора

Внешний вид окна в режиме корректировки параметров существующего терминатора представлен на Рис. 25.

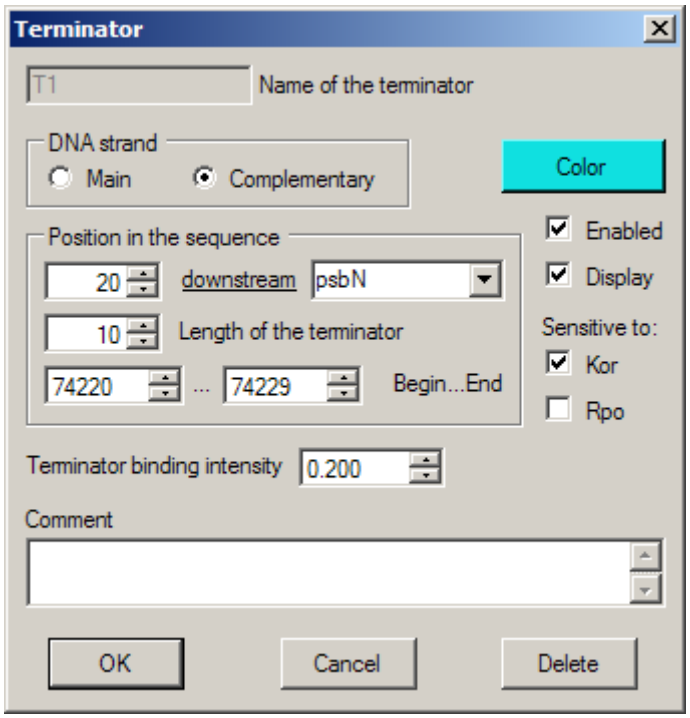


Рис. 25. Диалоговое окно параметров терминатора.

При создании нового терминатора его параметры заносятся в следующем порядке.

Name of the terminator В этом поле требуется указать имя, которое будет присвоено объекту. Все имена объектов в пределах одной задачи должны быть уникальными. Заглавные и строчные буквы различаются. Максимальная длина имени 31 символ, но для работы удобнее ограничиться 7–8 символами, т.к. длинные имена обрезаются при отображении объектов в рабочей области окна. В интерактивном режиме имена могут составляться из любых символов, включая пробел.
Примечание: Для обработки в пакетном режиме имена объектов не должны содержать пробелы, знаки табуляции, и символы . , = : % ! ?.

DNA strand В этой группе необходимо указать, на какой цепи ДНК находится терминатор, основной (**Main**) или комплементарной (**Complementary**).

Color Цвет кнопки совпадает с цветом заливки символа терминатора в рабочей области окна программы. При желании, нажатием этой кнопки можно указать любой желаемый цвет (выдается стандартный диалог Windows).

Position in the sequence В этой группе указывается положение терминатора одним из двух способов: либо (1) в выпадающем списке **Gene name** выбрать ген, вблизи начала или конца которого находится терминатор (на любой цепи ДНК), и его относительное расположение – перед (**upstream**) или после (**downstream**) этого гена, затем в поле слева указать смещение центра транскрипции соответственно от 5'-начала гена (отрицательное значение) или от 3'-конца гена (положительное значение), и после этого в поле **Length of the terminator** указать длину терминатора; либо (2) в полях **Begin ... End** указать абсолютные позиции левого и правого концов терминатора в основной цепи ДНК. При выборе одного из способов соответствующие

данные для другого способа будут внесены автоматически. Аналогично, при коррекции одного из параметров существующего репрессора будут меняться другие параметры.

Enabled По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет принимать участие в моделировании (он отображается притушенным в рабочей области окна). См. также п. 2.2.3.2.

Display По умолчанию эта пометка стоит, если убрать ее, то данный объект не будет отображаться в рабочей области окна, но будет участвовать в моделировании (т.н. «скрытый» объект). См. также п. 2.2.3.2.

Kor Если эта пометка стоит, то терминатор чувствителен к кор-ферментам (т.е. сбрасывает их со своей цепи ДНК). По умолчанию включено.

Rpo Если эта пометка стоит, то терминатор чувствителен к полимеразам фагового типа (т.е. сбрасывает их со своей цепи ДНК). По умолчанию включено.

Могут быть включены оба типа РНК-полимераз или только один (но не оба выключены). Подробнее см. раздел 1.3.11.

Terminator binding intensity Интенсивность появления терминатора, λ.

Comment В этом поле можно указать произвольный текст комментария, который будет появляться с прочими данными во всплывающей подсказке к объекту (см. п. 2.2.3) или вместе с данными о местоположении объектов (Рис. 5). Комментарий может занимать несколько строк, но при необходимости будет обрезан при выводе.

В нижней части диалога имеются две кнопки:

OK Сохранить параметры и закрыть диалоговое окно. Если при контроле будут обнаружены ошибки, выдается соответствующее сообщение и программа снова вернётся в диалоговое окно, чтобы исправить эти ошибки.

Cancel Отменить сделанные изменения и закрыть диалоговое окно.

В режиме корректировки параметров существующего объекта появляется еще третья кнопка:

Delete Удалить этот объект из задачи и закрыть диалоговое окно.

2.3.3. Типовые сценарии использования программы

Можно выделить несколько типовых приемов использования программы, в которых важно соблюдать правильную очередность действий. В этом разделе кратко излагаются некоторые такие сценарии использования. Предполагается, что заранее выполнены все необходимые настройки конфигурации или используется конфигурация по умолчанию из комплекта поставки.

2.3.3.1. Составление новой задачи с использованием файла ГенБанка

1. Запустить программу.
2. Нажать кнопку (**New task**) на панели инструментов.
3. Нажать кнопку (**Import DNA**) на панели инструментов. Открывается стандартное окно выбора файла Windows. Выделить сохранённый на компьютере файл ГенБанка (.gbk) с интересующим геномом и нажать кнопку **Open** (Открыть).
4. Выбрать в меню программы функцию **View > Plus buttons** для показа дополнительных кнопок в панели инструментов.
5. Не закрывая программу, открыть выбранный файл ГенБанка в текстовом редакторе,

например, **Notepad** (Блокнот).

6. Поочередно добавить к задаче все гены того локуса, который предполагается моделировать, выполняя шаги 7...14. Очередность добавления генов не имеет значения. Не следует добавлять в задачу не интересующие гены, т.к. это замедляет моделирование.
7. Нажать кнопку **+Gene (Add gene)** на панели инструментов. Откроется диалоговое окно параметров гена (Рис. 19).
8. В открытом файле ГенБанка найти раздел нужного гена и скопировать его имя в буфер обмена.
9. Вставить содержимое буфера обмена в поле **Gene name** диалогового окна. При необходимости сократить имя и/или убрать недопустимые символы.
10. Если в файле ГенБанка перед координатами гена указано **complement**, то отметить в диалоговом окне, что ген находится на комплементарной цепи (по умолчанию отмечено **Main,** т.е. ген на основной цепи).
11. В открытом файле ГенБанка выделить целиком координаты гена (например, **117804..118109**) и сохранить их в буфере обмена.
12. В диалоговом окне вставить содержимое буфера обмена в левое поле группы **Location.**
13. Если требуется, внести комментарии в поле **Comment** диалогового окна.
14. Нажать кнопку **OK** и убедиться, что новый ген появился в рабочей области окна. При желании правильность координат генов можно проверить, нажав кнопку .
15. Если последовательность ДНК кольцевая, и будет моделироваться целиком, выбрать режим кольцевой ДНК нажатием кнопки . (Если рассматривается только сравнительно короткий локус длинной кольцевой ДНК, то не следует переходить к более медленному кольцевому режиму, а работать в обычном режиме линейной ДНК. См. также раздел 5.5).
16. Нажать кнопку (**Save task**) на панели инструментов. Открывается стандартное окно сохранения файла Windows. Выбрать нужную папку и указать имя файла с новой задачей (.tsk), после чего нажать **Save** (Сохранить). Рекомендуется не откладывать этот пункт до конца составления новой задачи, а делать это на промежуточных этапах, начиная с п. 4, чтобы не потерять уже сделанную работу.
17. Поочередно добавить к задаче все остальные объекты, руководствуясь описаниями соответствующих диалогов в разделе 2.3.2. Порядок добавления объектов не имеет значения. Какие объекты добавлять – решает пользователь. В качестве примера укажем возможную последовательность операций при добавлении PEP-промотора.
18. Нажать кнопку **+PEP (Add PEP promoter)** на панели инструментов. Откроется диалоговое окно параметров PEP-промотора (Рис. 20).
19. Занести имя промотора в поле **PEP name**.
20. В группе **DNA strand** указать, на какой цепи ДНК находится промотор (по умолчанию на основной).
21. В группе **Start of transcription** выбрать из выпадающего списка **Gene name** имя гена, в апстримной области которого располагается промотор, затем слева от списка указать величину смещения центра транскрипции от начала гена.
22. Из выпадающего списка **Holoenzyme type** выбрать тип сигма-субъединицы, от которой зависит данный промотор (из отмеченных в п. 2.3.1.3).
23. В поле **Binding** указать предполагаемую интенсивность связывания промотора.
24. Если требуется – изменить другие параметры промотора.
25. Нажать кнопку **OK** и убедиться, что новый PEP-промотор появился в рабочей области окна.
26. Окончательно сохранить задачу нажатием кнопки на панели инструментов.

2.3.3.2. Внесение изменений в существующую задачу

1. Запустить программу.
2. Нажать кнопку (**Load task**) на панели инструментов и указать нужный файл задачи. Схема задачи отображается в рабочей области окна. Далее перечисляются различные виды операций, из которых складывается любое изменение задачи.
3. Добавление нового объекта. Выбрать в основном меню функцию **Edit > Add > *тип объекта***. В зависимости от выбранного типа добавляемого объекта, появится одно из описанных в разделе 2.3.2 диалоговых окон. Процедура добавления объекта та же, что и при создании новой задачи (см. п. 2.3.3.1).
4. Удаление существующего объекта. Правой кнопкой мыши щелкнуть по нужному объекту в рабочей области окна и выбрать в контекстном меню пункт **Delete**. Откроется окно с запросом на подтверждение; если нажать в этом окне кнопку **Yes**, объект будет удален.
5. Переименование существующего объекта. Правой кнопкой мыши щелкнуть по нужному объекту в рабочей области окна и выбрать в контекстном меню пункт **Rename**. Откроется окно с текущим именем объекта; после внесения необходимых изменений нажать **OK**.
6. Изменение параметров существующего объекта. Правой кнопкой мыши щелкнуть по нужному объекту в рабочей области окна и выбрать в контекстном меню пункт **Properties**. Появится одно из описанных в разделе 2.3.2 диалоговых окон в режиме редактирования, с текущими значениями параметров. После внесения необходимых изменений нажать **OK**. Для отмены изменений нажать **Cancel**.
7. Для сохранения задачи с прежним именем нажать кнопку на панели инструментов.
8. Для сохранения задачи с новым именем выбрать в основном меню программы функцию **File > Save As task**, выбрать нужную папку, ввести новое имя файла и нажать кнопку **Save (Сохранить)**.

2.3.3.3. Запуск задачи на выполнение в интерактивном режиме

1. Запустить программу.
2. Нажать кнопку (**Load task**) на панели инструментов и указать нужный файл задачи. Схема задачи отображается в рабочей области окна.
3. По умолчанию задача будет запущена в режиме одиночной траектории. Чтобы запустить задачу в режиме усреднения по пучку траекторий, нажать кнопку (**Options**) на панели инструментов. Откроется окно настройки конфигурации программы.
4. Перейти на закладку **Miscellaneous** и указать нужное число траекторий в поле **Runs to average**. Нажать кнопку **OK**, чтобы закрыть окно настройки конфигурации.
5. Нажать кнопку (**Start simulation**) на панели инструментов, чтобы запустить задачу.

2.4. Обработка заданий в пакетном режиме

Как уже говорилось выше, для массовых расчетов, в том числе на высокопроизводительных кластерах, в основном предназначен вариант cpRivals программы (см. главу 3). Тем не менее, при отсутствии кластера и для предварительной проверки правильности составленного файла заданий возможен запуск в пакетном режиме и программы Rivals. Следует иметь в виду, что, в отличие от варианта с командной строкой, программа Rivals не записывает в выходной файл **.csv** вычисленные для пучка траекторий средние значения и стандартные отклонения (однако эта информация может выводиться в файл протокола).

Для работы в пакетном режиме необходим заранее подготовленный файл заданий (см. раздел 4.3). Удобнее всего, если исполняемый файл программы (rivals.exe) и все относящиеся к ней файлы (файл конфигурации, файл задачи, файл заданий и файлы результатов) находятся в одной папке, хотя квалифицированный пользователь может свободно размещать их по своему желанию, указывая нужные пути в параметрах командной строки (см. п. 2.2.5). Ниже в примерах предполагается, что все файлы располагаются в папке с программой.

В пределах одного запуска для обработки заданий в пакетном режиме (а иногда и нескольких запусков) программа всегда использует одну и ту же конфигурацию, моделирует одну и ту же задачу и записывает результаты обработки в одни и те же файлы (по выбору пользователя прежнее содержимое этих файлов можно стирать или новая информация будет добавляться в конец существующего файла). Поэтому во многих случаях будет естественным указать имена файлов задачи и выходных файлов непосредственно в файле заданий, как описано в разделе 4.3.

Таким образом, различия между заданиями одного пакета могут касаться только тех параметров задачи и конфигурации, варьирование которых возможно посредством файла задания. В первую очередь это относится к не вполне точно известным параметрам объектов, например, интенсивностям связывания с ДНК для промоторов и репрессоров, абортивных процессов, скоростям транскрипции для различных типов полимераз и т.п. (полный перечень параметров приведен в п. 4.3). Варьирование набора объектов задачи, их длин и занимаемых позиций на цепях ДНК в описываемой версии программы невозможно и не планируется.

Начиная с версии 1.5.x.x, программа позволяет динамически изменять большинство упомянутых параметров не только перед началом, но и на протяжении траектории моделирования. Это позволяет воспроизводить более сложные экспериментальные сценарии, когда в процессе эксперимента меняется окружающая температура (и, следовательно, скорость элонгации полимераз, интенсивность связывания промоторов и т.п.). Динамические изменения поддерживаются только в пакетном режиме и применяются однотипно ко всем траекториям в пределах конкретного файла заданий. Подробнее об этом говорится в разделе 1.4.2.

Если, как это обычно делается, в файле заданий уже указаны имена файла задачи и выходных файлов, то запуск в пакетном режиме осуществляется любым из трех нижеизложенных способов:

Запуск из GUI:

1. Запустить программу.
2. Выбрать из главного меню функцию **File > Run job**. Откроется стандартное окно выбора файла Windows. Указать нужный файл задания и нажать кнопку **Open (Открыть)**.
3. Начнется автоматическое выполнение задания, прекратить которое можно только закрыв приложение.

Запуск из командной строки:

1. В главном меню Windows выбрать функцию **Start > Run (Пуск > Выполнить**). Появится окно с командной строкой (в котором может стоять последняя запускавшаяся команда).
2. Нажать кнопку **Browse (Обзор)**; в появившемся окне просмотра перейти в папку с программой, выбрать из списка файл rivals.exe и нажать кнопку **Open (Открыть).**
3. В окно командной строки перенесется полный путь к исполняемому файлу программы. Добавить к нему пробел, после которого указать параметр -r*jobfile* (вместо *jobfile* указать имя нужного файла заданий).

4. Нажать кнопку **OK**. Программа запустится и сразу начнется автоматическое выполнение задания, прекратить которое можно только закрыв приложение.

Запуск из командного процессора Windows:

1. Запустить командный процессор Windows и перейти в папку программы.
2. Ввести команду **rivals -r***jobfile* (вместо *jobfile* указать имя нужного файла заданий) и нажать клавишу Enter.
3. Программа запустится и сразу начнется автоматическое выполнение задания, прекратить которое можно только закрыв приложение.

Если в файле заданий не указано имя задачи и/или имя выходных файлов, то описанные выше процедуры модифицируются следующим образом:

Запуск из GUI:

1. Запустить программу.
2. Нажать кнопку (**Load task**) на панели инструментов и загрузить нужную задачу.
3. Если необходимо, выбрать из главного меню функцию **File > Output to file** и указать желаемое имя выходных файлов (см. п. 2.2.1.1).
4. Выбрать из главного меню функцию **File > Run job**. Откроется стандартное окно выбора файла Windows. Указать нужный файл задания и нажать кнопку **Open (Открыть)**.
5. Начнется автоматическое выполнение задания, прекратить которое можно только закрыв приложение.

Запуск из командной строки:

1. В главном меню Windows выбрать функцию **Start > Run (Пуск > Выполнить**). Появится окно с командной строкой (в котором уже стоит последняя запускавшаяся команда).
2. Нажать кнопку **Browse (Обзор)**; в появившемся окне просмотра перейти в папку с программой, выбрать из списка файл rivals.exe и нажать кнопку **Open (Открыть).**
3. В окно командной строки перенесется полный путь к исполняемому файлу программы. Добавить к нему пробел, после которого указать параметр -r*jobfile* (вместо *jobfile* указать имя нужного файла заданий) и, при необходимости, еще пробел, после которого указать параметр -t*taskfile* (вместо *taskfile* указать имя файла задачи) и/или -o*outfile* (вместо *outfile* указать желаемое имя файлов результатов).
4. Нажать кнопку **OK**. Программа запустится и сразу начнется автоматическое выполнение задания, прекратить которое можно только закрыв приложение.

Запуск из командного процессора Windows:

1. Запустить командный процессор Windows и перейти в папку программы.
2. Ввести команду **rivals -t***taskfile* **-r***jobfile* **-o***outfile* (вместо *taskfile* указать имя файла задачи, вместо *jobfile* указать нужного файла заданий, вместо *outfile* –имя файлов результатов) и нажать клавишу Enter. (Для большего удобства соответствующая команда может быть заранее заготовлена и сохранена в виде командного файла Windows с расширением **.cmd** или **.bat**).
3. Программа запустится и сразу начнется автоматическое выполнение задания, прекратить которое можно только закрыв приложение.

Примечание: В программе Rivals содержимое файла заданий имеет приоритет, поэтому описанным способом нельзя изменить имя файла задачи и/или выходных файлов, если они уже указаны в файле заданий.

Обработка заданий в пакетном режиме протекает следующим образом. Вначале программа прочитывает файл конфигурации, преамбулу файла заданий и файл задачи, затем делает все необходимые начальные настройки и открывает следующие выходные файлы (в режиме перезаписи/создания или дозаписи в конец):

- *outfile***.log** – файл протокола (см. п. 4.4);
- *outfile***.csv** – файл для записи результатов успешно выполненных заданий (см. п. 4.5), в том числе промежуточных данных вдоль траекторий, если этот режим включен;
- *outfile***_rejected.csv** – файл для записи неполных результатов заданий, которые аварийно завершились или были прерваны по указанному в файле заданий критерию фильтрации (см. п. 4.3). Формат файла тот же, что у файла результатов (п. 4.5).

Здесь *outfile* – имя выходных файлов, указанное в командной строке (п. 2.2.5), файле заданий (п. 4.3) или по умолчанию принятое совпадающим с именем задачи. Если имя выходных файлов не содержит пути, все файлы создаются в папке программы.

Затем по очереди для каждого задания в пакете программа:

- создает временный файл *outfile***_current.csv** для записи результатов текущего задания;
- устанавливает указанные в задании значения переменных параметров задачи;
- выполняет моделирование на одной траектории или пучке траекторий указанного в файле заданий размера (п. 4.3)
- записывает финальные (и промежуточные, если запрошено) результаты всех траекторий пучка в созданный временный файл;
- при работе с пучком траекторий – проверяет критерий фильтрации (если задан) и при его выполнении прерывает обработку данного задания в пакете.
- по окончании обработки задания результаты из временного файла переносятся в конец файла *outfile***.csv** (если задание выполнено успешно) или файла *outfile***_rejected.csv** (если задание было отфильтровано или вызвало ошибки), и временный файл уничтожается.
- в файле протокола *outfile***.log** делается запись об окончании обработки задания.
- обработанная строка файла заданий помечается символом **';'** в первой позиции, что превращает ее в комментарий (однако это действие можно отменить, указав в командной строке запуска программы параметр **-j** или **--j**).

Таким образом, если обработка задания была прервана вручную или по внешней причине (отказ оборудования, перебой электропитания и т.п.), имеется возможность продолжить работу с тем же самым файлом заданий: ранее выполненные задания будут пропущены (как любые строки комментариев), и обработка начнется с последнего невыполненного задания. Перед тем, как перезапускать задание, следует проверить, что включен режим дозаписи выходных файлов, иначе прежние результаты будут потеряны.

Несмотря на описанную возможность, определенные потери в случае прерывания и возобновления пакетной обработки будут иметь место. Например, если выполнение каждого задания связано с моделированием на пучке траекторий, то результаты от траекторий, которые уже выполнены к моменту прерывания, будут потеряны, и моделирование всего пучка начнется заново. В варианте программы cpRivals (глава 3) этот недостаток устранен, и максимально возможные потери состоят из результатов вдоль одной траектории моделирования (в мультипроцессорном режиме это относится к незавершенной траектории, моделируемой каждым процессором).

3. Программа cpRivals (утилита командной строки)

Необходимо отчетливо понимать, что хотя оба варианта программы реализуют одну и ту же модель, это совершенно различные программы, вплоть до того, что они написаны на разных языках программирования – Visual C++ .NET 2008 (rivals.exe) и ANSI C++ (cprivals.exe). Эти программы используют по-разному реализованные контейнеры и разные датчики псевдослучайной последовательности, с различными статистическими характеристиками и периодом. Поэтому моделирование с помощью каждой программы может, вообще говоря, приводить к отличающимся результатам. Автор приложил максимум усилий, чтобы поведение программ было одинаковым, но не может дать гарантий на этот счет. Во всяком случае, оба варианта программы совместимы по файлам конфигурации, задачи и заданий.

Кроме того, программа cpRivals поставляется в виде исходных текстов на условиях AS IS («как есть»; полный текст лицензионного соглашения содержится в текстовом файле в комплекте поставки). Автор оставляет за собой право не вступать в переписку в связи с ошибками при работе программы вследствие любых изменений в исходных текстах или использования ее на конкретных платформах, отличных от тех, которые использовал автор.

В целом использование программы cpRivals, тем более в параллельной среде MPI, требует более высокой квалификации, чем у массового пользователя ПК. Поэтому изложение в этой главе ведется в основном без описания примитивных приемов и общеизвестных вопросов, которые, без сомнения, знакомы разработчикам и квалифицированным пользователям.

3.1. Компиляция и тестирование

Исходные тексты программы сделаны по мере возможности переносимыми между платформами Windows и Linux, хотя полностью исключить ошибки компиляции и различные предупреждения во всем разнообразии диалектов языка, компиляторов и вариантов операционных систем вряд ли возможно. Даже при успешной компиляции следует внимательно изучить выдаваемые конкретным компилятором сообщения, т.к. в них могут крыться причины последующей ошибочной работы программы.

Перед компиляцией программы необходимо уточнить наличие и ознакомиться с документацией к имеющемуся на конкретной платформе варианту реализации протокола MPI. В исходных текстах программы подключается заголовочный файл **mpi.h** и предполагается наличие привязки основных функций спецификации MPI 1.2 для языка C++. Такую привязку и доступность всех исходных и объектных библиотек для применяемых компилятора и сборщика должна обеспечивать команда **mpiCC**, которая вызывается в make-файле из комплекта поставки. При соблюдении этих условий сборку программы в среде Linux обеспечивает обычная команда **make**, очистку перед повторной сборкой – команда **make clean**. Если же аналогичная команда имеет другой синтаксис (например, в некоторых реализациях MPI используется команда **mpicxx**), то необходимо сначала соответственно отредактировать файл **make** из комплекта поставки.

В среде Windows рекомендуется использовать компилятор Microsoft Visual C++ 2008, а в качестве среды MPI – установочный пакет MPICH2 v.1.4.1 (или более поздней версии, см. сайт разработчиков http://www.mpich.org/downloads/). В самостоятельно составляемом проекте для Visual Studio необходимо, как минимум:

- для этапа компиляции C++ указать местоположение дополнительных исходных файлов MPI, например, C:\Program Files\MPICH2\include
- для этапа сборки указать местоположение дополнительных объектных библиотек MPI, например, C:\Program Files\MPICH2\lib; в качестве добавочных подключаемых файлов – mpi.lib и cxx.lib (или другие, в зависимости от реализации MPI).

Примечание: Для удобства в комплект поставки программы Rivals включён пример исполняемого модуля cprivals.exe (версии x86 32bit), собранный для системы Windows с установленным пакетом MPICH2 v.1.4.1, и командный файл **cptest.bat** для его тестирования в однопрограммном режиме на той же задаче, на которой тестировалась программа rivals.exe (см. раздел 2.1). Выходные файлы, однако, будут отличаться от прежних (эталонные варианты для сравнения поименованы HSM1.csv, HSM1.log в комплекте поставки).

Тестирование самостоятельно откомпилированной программы удобно проводить на примере, входящем в состав поставляемых исходных текстов программы cpRivals (вложенный каталог **example**). Его следует начать с запуска командной строки **cprivals -h** (в среде Linux – **./cprivals -h**) и проверки, что программа выдает подсказку по параметрам командной строки.

Далее проверяется работа в однопроцессорном режиме. Если при компиляции использовался make-файл из комплекта поставки, то достаточно ввести команду **make test**. В противном случае рекомендуется скопировать исполняемый модуль в каталог **example** и уже оттуда выполнить команду **cprivals -nompi -rjob3 --j** . В зависимости от скорости компьютера, тестирование занимает около минуты, в результате чего будут сформированы выходные файлы AT1.csv, AT1.log. Следует сравнить их с эталонными файлами AT1_chk.csv, AT_chk.log из комплекта поставки – во втором файле могут наблюдаться небольшие расхождения во времени счёта из-за разного быстродействия компьютеров.

Далее можно переходить к тестированию программы в многопроцессорном режиме, для чего может потребоваться предварительно загрузить диспетчер используемой среды MPI, как описано в ее документации. После этого, если при компиляции использовался make-файл из комплекта поставки, то достаточно ввести команду **make testmpi** (расчёт будет проведён на 8 проуессорах). В противном случае из каталога **example** для запуска программы на 8 процессорах следует выполнить команду:

mpirun -np 8 cprivals -rjob3.rjb --j -seed0 -step0 -runs10

Здесь вместо **mpirun** может стоять другое имя, принятое в конкретной реализации MPI (например, MPICH2 в версии для Windows использует имя программы **mpiexec**). По сравнению с тестированием в однопроцессорном варианте выше, здесь размер пучка траекторий увеличен с 3 до 10 параметром **-runs** и полученные результаты будут отличаться соответственно. Успешная работа программы во всех вышеперечисленных тестах является достаточным подтверждением ее работоспособности.

3.2. Параметры командной строки

В этом разделе описываются только параметры командной строки самой программы cpRivals. Описание параметров команд mpirun/mpiexec содержится в документации к используемой среде MPI. Параметры могут указываться в командной строке в произвольном порядке. В системе Windows регистр используемых символов безразличен; в системе Linux заглавные и строчные буквы в именах файлов различаются, а в наименованиях ключей программы – нет. Кроме того, при использовании в системе Windows файлов с именами, содержащими пробелы, такие имена необходимо заключать в кавычки при указании их в командной строке программы, например, **-i"My configuration.ini"**.

Параметры в командной строке отделяются друг от друга одним или более пробелом. Каждый параметр начинается символом **-** (дефис) или **/** (наклонная черта), после которого

идет ключ, идентифицирующий данный параметр, и далее (в большинстве случаев, но не всегда) *без пробела* числовое или символьное значение параметра. Ниже перечисляются параметры командной строки, воспринимаемые текущей версией программы. Полный список параметров и краткая подсказка выдаются по команде **cprivals -h** .

-i<*filename*> Указывает местоположение и имя файла конфигурации. Вместо <*filename*> указывается имя файла конфигурации, которое может включать составленный по правилам операционной системы путь – относительный из каталога, где находится программа, или абсолютный. По умолчанию файл конфигурации находится в каталоге с программой и имеет имя **rivals.ini**.

-t<*filename*> Указывает местоположение и имя файла задачи. Вместо <*filename*> указывается имя файла задачи, которое может включать составленный по правилам операционной системы путь – относительный из каталога программы или абсолютный.

-r<*filename*> Указывает местоположение и имя файла заданий, который содержит пакет заданий для моделирования с различными наборами параметров задачи. Вместо <*filename*> указывается имя файла заданий, которое может включать составленный по правилам операционной системы путь – относительный из каталога программы или абсолютный.

-o<*filename*> Указывает местоположение и имя выходных файлов. Вместо <*filename*> указывается имя файла, которое может включать составленный по правилам операционной системы путь – относительный из каталога программы или абсолютный. Расширение имени файла несущественно и игнорируется, если оно указано: программа записывает два (или более) выходных файла с расширениями **.log** и **.csv** соответственно.

-a Включает режим дозаписи в выходные файлы. По умолчанию старое содержимое файлов стирается (такой же эффект дает параметр **--a**).

-c Включает режим сбора консольного вывода от всех задач. По умолчанию в мультипроцессорном режиме на консоль поступает только стандартный вывод от корневой (управляющей) задачи; тот же эффект дает параметр **--c**. Если режим включен, то на консоль передается также стандартный вывод от всех параллельно выполняемых дочерних задач.

-g<*number*> Устанавливает периодичность выдачи на консоль оперативной информации о моделируемой траектории (только в однопроцессорном режиме, -nompi). Вместо <*number*> указывается шаг (в виде числа итераций), с которым будут выдаваться отметки о выполненных вдоль траектории итерациях модели. Значение по умолчанию 10000, значение 0 отменяет эту выдачу.

-h Выдача подсказки по параметрам командной строки.

-l<*number*> Управляет выдачей отладочной информации о межзадачном обмене информацией в многопроцессорном режиме (не описывается).

-j Включает стандартный режим работы с файлом заданий: после того, как задание выполнено целиком, соответствующая строка файла заданий помечается символом **;** (точка с запятой) в первой позиции, что превращает ее в комментарий. Это позволяет в случае сбоя или принудительного завершения работы программы продолжить ее работу после нового запуска с тем же файлом заданий, и выполнение начнется с последнего неоконченного задания (более того, с учетом всех ранее проделанных траекторий этого задания). Для выключения данного режима следует указать параметр **--j**.

-s<*number*> Устанавливает начальное значение для инициализации псевдослучайного датчика (в мультипроцессорном режиме датчик в каждой ветви будет проинициализирован своим значением, которое определяется указанным числом). Если в качестве <*number*> указан 0 (такой режим действует по умолчанию), то датчик инициализируется с использованием текущих показаний компьютерных часов, так что повторные запуски программы с одними и теми же значениями параметров модели будут приводить к различным результатам (и это используется при моделировании на пучке траекторий). Если же вместо <*number*> стоит ненулевое целое значение, то оно используется при инициализации датчика. В этом случае повторные запуски программы будут давать одни и те же результаты, но только в однопроцессорном режиме; гарантировать воспроизводимость результатов на кластере невозможно ввиду индивидуальных различий быстродействия узлов и прочих неуправляемых факторов.

-u<*number*> Устанавливает продолжительность (в минутах) фазы разгона модели, на протяжении которой не ведется подсчет событий (таких, как инициация и окончание транскрипции гена, связывание РНК-полимеразы с промотором, срыв полимеразы на сайте терминации и т.п.), но в остальном модель функционирует обычным образом. По умолчанию действует значение, указанное в файле конфигурации (если иное не указано в файле заданий). Значение 0 интерпретируется как отсутствие фазы разгона.

-dcsn<*number*> Этот параметр используется совместно с параметром **-mrat** для фильтрации заведомо неперспективных заданий из общего пакета (подробнее см. раздел 4.3). Значение <*number*> устанавливает минимальное число траекторий пучка, после выполнения которых начинает проверяться заданный критерий. По умолчанию действует значение **0**, которое интерпретируется как отсутствие фильтрации (если иное не указано в файле заданий).

-mrat<*number*> Этот параметр используется совместно с параметром **-dcsn** для фильтрации заведомо неперспективных заданий из общего пакета (подробнее см. раздел 4.3). Действительное значение <*number*> устанавливает нижний предел для величины отношения числа транскрипций любого гена к суммарному числу транскрипций всех генов. Как только фактическая величина этого отношения, вычисленная на уже проделанных траекториях пучка (числом не менее заданного параметром **-dcsn**), оказывается меньше указанного порога, дальнейшая обработка пучка траекторий прекращается. По умолчанию принимается значение **-mrat0.001** (оно действует, если указан ненулевой параметр **-dcsn** и не указано иное значение в файле заданий).

-gran<*number*> То же, что и **-g**<*number*>.

-nompi Переводит программу в однопроцессорный режим, в котором подсистема MPI не задействована. По умолчанию выключен, и программа работает в многопроцессорном режиме, для чего она должна (средствами MPI) запускаться минимум на двух процессорах. Следует учитывать, что один из выделенных для программы процессоров всегда используется для управляющих целей (корневая ветвь), на остальных ведется моделирование траекторий. Каждая траектория от начала до конца обрабатывается только на одном процессоре в одном вычислительном потоке, и обеспечивает загрузку ядра, близкую к 100%.

-runs<*number*> Задает число траекторий в пучке. По умолчанию используется значение 1, что соответствует режиму одиночных траекторий.

-seed<*number*> То же, что и **-s**<*number*>.

-step<*number*> Управляет выдачей промежуточных результатов по ходу траектории. По умолчанию, в файл результатов выдается одна запись по окончании каждой траектории моделирования (или пучка траекторий, если применяется усреднение). Имеется возможность дополнительно выводить промежуточные результаты по истечении числа минут физического времени, заданного значением <*number*>. Например, если установлена длительность физического моделирования сутки (параметр **-time1440**), то с параметром **-step60** для каждой траектории (или пучка) в файл **.csv** будет выведено 24 записи. Если значение <*number*> равно 0, то промежуточная выдача не выполняется (режим по умолчанию).

-stup<*number*> То же, что и **-u**<*number*>.

-time<*number*> Задает продолжительность моделирования одной траектории (моделируемое физическое время) в минутах.

3.3. Приоритет учета параметров модели

Многие важные для моделирования параметры могут задаваться в различных местах. Например, физическое время моделирования может быть задано в конфигурации программы, в файле задания и в командной строке запуска. Поэтому важно представлять, какое из нескольких значений будет использоваться в конкретном расчёте. В этом отношении логика программы cpRivals для автоматизированного счета отличается от логики интерактивной программы Rivals с GUI, о чем следует помнить, используя одни и те же файлы для работы с обоими вариантами программы.

В программе cpRivals используется следующая общая очередность (в порядке убывания приоритета): командная строка запуска, файл заданий, файл задачи, файл конфигурации. В строках Табл. 3 перечислены основные параметры модели, которыми может управлять пользователь. Колонки соответствуют возможным местам указания параметра и перечислены в порядке убывания приоритета слева направо. Заполненные клетки отвечают допустимым способам задания значения параметра и содержат имя параметра (ключ в командной строке, предложение в файле и т.п.).

Табл. 3. Параметры программы, места их задания и очередность применения.

Параметр	**Командная строка**	**Файл заданий (.rjb)**	**Файл задачи (.tsk)**	**Файл конфигурации (.ini)**
Имя файла конфигурации	**-i**			
Имя файла заданий	**-r**			
Имя файла задачи	**-t**	**Task**		
Имя выходных файлов результатов	**-o**	**Result**		
Дозапись или перезапись файлов результатов	**-a** **--a**	**Append**		
Выдача в файл .csv				**CSV**
Разделитель в файле .csv				**Delimiter**
По умолчанию вывод в .csv всех счётчиков или только успешных		**BothCSV**		**Both**

Параметр	Командная строка	Файл заданий (.rjb)	Файл задачи (.tsk)	Файл конфигурации (.ini)
Заказная структура вывода в .csv		**Record**		
Физическое время моделирования	**-time** (в минутах)	**Pminutes** (в минутах)		**PhysicalTime** (в секундах)
Продолжительность фазы разгона модели	**-u**, **-stup** (в минутах)	**Startup** (в минутах)		**StartupTime** (в секундах)
Периодичность выдачи промежуточных результатов	**-step** (в минутах)	**Interval** (в минутах)		
Число траекторий в пучке	**-runs**	**Repeat**		
Число траекторий до начала фильтрации	**-dcsn**	**Decision**		
Минимальная доля транскриптов гена (в общей сумме по генам)	**-mrat**	**MinRatio**		
Зародыш для датчика псевдослучайных чисел	**-s** **-seed**	**Seed**		
Периодичность оперативной выдачи на консоль (только вне MPI)	**-g** **-gran** (итераций)	**Granularity**		
По умолчанию общий учет транскрипций или раздельно PEP и NEP		**Joint**		
Режим учета: от всей полимеразы или только от центра транскрипции				**CountCenter**
Последовательность ДНК			**[DNA ... Length Circular**	
Неподвижные объекты задачи и их параметры		**Names** (только параметры λ, μ, r, R ...)	**[Gene] [PEP] [NEP] [Repr] [EKor] [ERpo] [Term]**	
Правила разрешения коллизий				**SAP, DAP, DPP, SAC, DAC, DPC, SAR, DAR, DPR, SAT, DAT, DPT**
Каталог для временных файлов		**Scratch**		**Scratch**

Незаполненные клетки таблицы означают, что соответствующей возможности в текущей версии программы нет.

3.4. Использование в однопроцессорном режиме

Программа cpRivals используется в однопроцессорном режиме при отсутствии кластера, а также для предварительной проверки правильности сформированных заданий перед их запуском на кластере. Переход в однопроцессорный режим обеспечивается указанием в командной строке параметра **-nompi**.

В основном, работа в этом режиме аналогична использованию программы Rivals в пакетном режиме (см. раздел 2.4), но есть ряд отличий:

1. Быстродействие программы cpRivals с теми же исходными данными выше.
2. Программа cpRivals запускает только один вычислительный поток (thread) (программа Rivals в зависимости от версии ОС и прочих условий обычно инициирует от 8 до 12 потоков). При этом в обоих случаях загрузка процессорного ядра достигает 100%, но накладные расходы для программы cpRivals меньше, кэш-память используется более эффективно, и планирование вычислений в мультипроцессорной системе упрощается.
3. Поскольку работа с программой cpRivals обычно предполагает многочасовые расчеты, предусмотрены дополнительные средства, которые позволяют свести к минимуму непроизводительные потери при внезапном прерывании счета (из-за отказа оборудования, перебоя электропитания, плановых профилактических работ и т.д.) и легко продолжить вычисления после возобновления работоспособности компьютера. Эти средства подробнее описываются ниже.

Запуск задания в обработку выполняется путем указания в командной строке операционной системы имени программы cprivals и необходимых параметров командной строки (раздел 3.2), с учетом действующей системы приоритетов значений параметров (раздел 3.3).

Обработка задания в однопроцессорном режиме выполняется следующим образом. Вначале программа прочитывает файл конфигурации, преамбулу файла заданий и файл задачи, затем делает все необходимые начальные настройки и открывает следующие выходные файлы (в режиме перезаписи/создания или дозаписи в конец):

- *result***.log** – файл протокола (см. п. 4.4);
- *result***.csv** – файл для записи результатов успешно выполненных заданий (см. п. 4.5), в том числе промежуточных данных вдоль траекторий, если этот режим включен;
- *result*_**rejected.csv** – файл для записи неполных результатов заданий, которые аварийно завершились или были прерваны по указанному в файле заданий критерию фильтрации (см. п. 4.3). Формат файла тот же, что у файла результатов (п. 4.5).

Здесь *result* – имя выходных файлов, указанное в командной строке (п. 3.2), файле заданий (п. 4.3) или принятое по умолчанию совпадающим с именем задачи. Если имя выходных файлов не содержит пути, все файлы создаются в папке программы.

Затем для каждого очередного задания в пакете программа:

- открывает файл *result*_**current.csv** для результатов текущего задания в режиме дозаписи (при отсутствии файла он создается), поэтому если задание было прервано во время расчёта на пучке траекторий, то при возобновлении работы уже обсчитанные траектории не будут моделироваться заново;
- устанавливает указанные в задании значения переменных параметров задачи;
- выполняет моделирование на одной траектории или пучке траекторий указанного в файле заданий размера (п. 4.3), с учетом ранее просчитанных траекторий;
- записывает финальные (и промежуточные, если запрошено) результаты всех траекторий пучка в файл *result*_**current.csv**;
- при работе с пучком траекторий – проверяет критерий фильтрации (если задан) и при его выполнении прерывает обработку данного задания в пакете.
- по окончании обработки задания результаты из файла *result*_**current.csv** переносятся в конец файла *result***.csv** (если задание выполнено успешно) или файла *result*_**rejected.csv** (если задание было отфильтровано или привело к ошибке), а сам файл уничтожается.
- в файле протокола *result***.log** делается запись об окончании обработки задания.

- обработанная строка файла заданий помечается символом **';'** в первой позиции, что превращает ее в комментарий.

Таким образом, если обработка задания была прервана вручную или по внешней причине (отказ оборудования, перебой электропитания и т.п.), имеется возможность продолжить работу с тем же самым файлом заданий. При этом ранее выполненные задания будут пропущены как любые строки комментариев, и обработка начнется с очередной (по счету внутри пучка) траектории последнего невыполненного задания. Перед тем, как перезапускать задание, следует проверить, что включен режим дозаписи выходных файлов, иначе прежние результаты будут потеряны.

Примечание: Следует иметь в виду, что если программа настроена для работы с выдачей промежуточных результатов по ходу траектории, то в случае прерывания задания часть таких результатов для последней незавершенной траектории останется в файле *result*_**current.csv** и может помешать последующей автоматической обработке результатов. По этой причине перед тем, как перезапускать прерванное задание, рекомендуется вручную отредактировать файл *result*_**current.csv** подходящим текстовым редактором, удалив из этого файла всю информацию, оставшуюся от последней незаконченной траектории. При работе программы cpRivals в мультипроцессорном режиме на кластере (п. 3.5) такой проблемы нормально не возникает – все необходимые действия будут выполнены автоматически.

3.5. Использование в параллельном режиме

При использовании программы для расчетов на высокопроизводительном кластере возникают трудности двоякого рода. Во-первых, необходимы организационные усилия для получения доступа к мощной технике (зачастую уникальной), освоение регламентов и процедур ее использования, получение кодов доступа, установка требуемых клиентских программ, изучение особенностей применяемых компиляторов, реализации MPI-среды и синтаксиса ее управляющих команд, а также команд диспетчера очереди заданий. Все эти вопросы находятся за пределами данного руководства. Предполагается, что такая технология уже отработана, а сама программа успешно откомпилирована и оттестирована в конкретной мультипроцессорной среде.

Второй аспект возникающих трудностей – это эффективное планирование вычислительных работ. Машинное время высокопроизводительного кластера – дорогостоящий ресурс, который желательно расходовать с наибольшей эффективностью. От объёма запрашиваемых ресурсов (таких как число процессоров и максимальное время счета) сильно зависит время ожидания задач в очереди, а значит и сроки получения конечных результатов. Чтобы правильно планировать вычисления, нужно хорошо представлять себе заложенный в программу cpRivals алгоритм распараллеливания вычислений и фактические пределы его масштабируемости. Указанные вопросы и освещаются ниже в этом разделе.

Обработка заданий в мультипроцессорном режиме выполняется следующим образом. Вначале корневая ветвь программы прочитывает файл конфигурации, преамбулу файла заданий и файл задачи, затем делает все необходимые начальные настройки и открывает следующие выходные файлы (в режиме перезаписи/создания или дозаписи в конец):

- *result*.**log** – файл протокола (см. п. 4.4);
- *result*.**csv** – файл для записи результатов успешно выполненных заданий (см. п. 4.5), в том числе промежуточных данных вдоль траекторий, если этот режим включен;
- *result*_**rejected.csv** – файл для записи неполных результатов заданий, которые аварийно завершились или были прерваны по указанному в файле заданий критерию фильтрации (см. п. 4.3). Формат файла тот же, что у файла результатов (п. 4.5).

Здесь *result* – имя выходных файлов, указанное в командной строке (п. 3.2), файле заданий (п. 4.3) или принятое по умолчанию совпадающим с именем задачи. Если имя выходных файлов не содержит пути, все файлы создаются в папке программы.

Затем корневая ветвь программы:

- выбирает очередное незавершенное задание из пакета и определяет число оставшихся для выполнения траекторий пучка;
- если это не было сделано раньше, открывает файл *result*_**xxx.csv** для результатов задания из строки xxx файла заданий в режиме дозаписи (при отсутствии файла он создается), поэтому если задание было прервано в процессе расчета на пучке траекторий, то при возобновлении работы уже обсчитанные траектории не будут моделироваться заново. Если имя выходных файлов не указывает другого пути, файл открывается в папке программы;
- передает свободным от расчетов дочерним ветвям указанные в задании значения переменных параметров задачи. Если фильтрация не применяется, то при наличии достаточного числа свободных ветвей это делается вплоть до исчерпания пучка. Если используется фильтрация вырожденных заданий, то для каждого нового задания его траектории поручаются дочерним ветвям только до достижения начального порога принятия решения (см. параметры **-dcsn** в п. 3.2, **Decision** в п. 4.3), а затем программа переходит к следующему заданию, откладывая продолжение текущего до принятия решения, т.е. когда минимально требуемое число траекторий будут просчитаны.

По мере получения результатов завершенных траекторий от дочерних ветвей, корневая ветвь программы:

- записывает результаты траекторий каждого задания (т.е. пучка) в свой файл *result*_**xxx.csv**;
- при работе с фильтрацией – проверяет заданный критерий и при его выполнении прерывает обработку данного задания в пакете, сообщив об этом тем дочерним ветвям, которые еще продолжают моделирование для данного задания;
- если обработку задания решено продолжать, корневая ветвь передает свободным от расчетов дочерним ветвям значения переменных параметров вплоть до конца пучка;
- по окончании обработки задания результаты из файла *result*_**xxx.csv** переносятся в конец файла *result***.csv** (если задание выполнено успешно) или файла *result*_**rejected.csv** (если задание было отфильтровано или привело к ошибке), а сам файл уничтожается.
- в файле протокола *result***.log** делается запись об окончании обработки задания.
- обработанная строка файла заданий помечается символом **';'** в первой позиции, что превращает ее в комментарий.

Каждая из дочерних ветвей программы:

- принимает от корневой ветви значения переменных параметров задачи;
- выполняет моделирование на одной траектории, записывая результаты во временный файл *result*_**current_yyy.csv** (где yyy – номер ветви);
- сообщает корневой ветви об окончании моделирования вдоль траектории;
- по запросу корневой ветви – передает ей содержимое временного файла и удаляет его.
- по требованию корневой ветви – прекращает моделирование вдоль траектории, удаляет все временные данные и сообщает корневой ветви о готовности принять новые значения параметров задачи.

Временные файлы по умолчанию создаются в файловой системе того процессора, на котором выполняется эта ветвь, в папке программы или по другому пути, указанному в имени *result*. В зависимости от архитектуры конкретного кластера, это может означать

совершенно разные разделы внешней памяти. В текущей версии программы предусмотрена возможность указать параметром **Scratch** (в файле конфигурации или файле задания, см. главу 4) место для размещения временных файлов, но этот каталог обязан существовать.

Примечание: Параметр **Scratch** и путь в имени выходных файлов *result* интерпретируются одинаково всеми ветвями программы, включая корневую.

Каждая из параллельных дочерних ветвей работает на своем процессоре и потому использует свой генератор псевдослучайной последовательности, однако в программе приняты специальные меры, чтобы эти последовательности у любых двух ветвей были некоррелированными. Поэтому параллельное моделирование пучка траекторий на разных процессорах не приводит к ухудшению статистических характеристик, даже при использовании фиксированного значения «зародыша» псевдослучайного датчика.

Если обработка заданий была прервана вручную оператором или по внешней причине (отказ оборудования, перебой электропитания и т.п.), имеется возможность продолжить работу с тем же самым файлом заданий. При этом ранее выполненные задания будут пропущены как любые строки комментариев, и обработка начнется с очередной (по счету внутри пучка) траектории последнего невыполненного задания. Перед тем, как перезапускать задание, следует проверить, что включен режим дозаписи выходных файлов, иначе прежние результаты будут потеряны.

Планирование обширных вычислений, которые в сумме могут занять дни, недели и даже месяцы, рекомендуется производить в последовательности, приводимой ниже.

Планирование расчетов на высокопроизводительном кластере

1. Подготовить файлы конфигурации и задачи и проверить их правильность с помощью интерактивной версии программы Rivals.
2. Определить необходимое физическое время моделирования, состав переменных параметров задачи, желаемые интервал и шаг их варьирования, а также число m траекторий в пучке. Составить единый пакет заданий (совокупность наборов параметров) и прототип будущего файла заданий.
3. С помощью пробных запусков программы cpRivals на процессорах, аналогичных процессорам кластера (или иных доступных, применяя соответствующий масштабный пересчет), получить оценку компьютерного времени моделирования одной траектории. Общая закономерность быстродействия программы такова: чем больше объектов в задаче и чем выше значения интенсивностей связывания промоторов и внешних источников, тем длиннее в среднем будет очередь событий, и тем медленнее работает программа. Поэтому для таких пробных запусков рекомендуется выбирать точки с б**о**льшими значениями параметров интенсивности. Следует также учитывать возможный стохастический разброс длительности моделирования траектории даже при одних и тех же значениях параметров, который в наших экспериментах достигал 20%. Так или иначе, требуется получить надежную *верхнюю* оценку, которую обозначим t [минут].
4. Если пакет состоит из n заданий, то общая оценка сверху суммарного процессорного времени кластера составляет $T_\Sigma = n \cdot m \cdot t$, и с учетом привлечения k процессоров необходимое машинное время кластера составит $T = T_\Sigma / (k-1)$. Если найденные значения T, k оказываются приемлемыми для одного сеанса, допускаются по действующему регламенту и не приведут к излишне долгому ожиданию задания в очереди, то построенный пакет вполне можно запустить одним заданием, запросив для него у диспетчера кластера k процессоров на T минут. Однако иногда практичнее разделить весь

пакет хотя бы на два файла заданий с половинными требованиями в части ресурсов, чтобы можно было предварительно обработать результаты первого задания, пока второе считается.

5. Более типична, однако, ситуация, когда приемлемого сочетания *T, k* достичь не удается, и пакет заданий неизбежно приходится разделять на несколько файлов заданий, при том что каждый из них тоже может выполняться в несколько приемов. Трудно дать для такого случая универсальные рекомендации, поэтому далее мы просто перечислим ряд соображений, которые необходимо принимать в расчет.

6. Запрашиваемый квант машинного времени кластера должен быть никак не меньше *t*, и лучше, если он будет составлять 5*t* и более, иначе непроизводительные затраты машинного времени в результате принудительного прерывания работы диспетчером кластера окажутся недопустимо высокими.

7. В большинстве современных кластеров процессоры не независимы, а собраны в узлы по 2–8 процессоров с общим полем оперативной памяти, и такой узел целиком выделяется пользовательскому заданию. Поэтому *k* фактически может меняться только с заданным шагом. Все рассуждения выше относятся к случаю, когда объема памяти узла достаточно для обработки задачи сразу всеми его процессорами. Если программа начнёт завершаться аварийно с выдачей сообщения о нехватке памяти (а иногда и без), то приходится средствами диспетчера MPI уменьшать число параллельных ветвей, запускаемых на каждом узле, что с позиций диспетчера будет равносильно большему числу запрашиваемых процессоров.

8. При распараллеливании задачи на $k = 1024$ процессоров и более начинает сказываться торможение из-за перегрузки корневой ветви программы. Поэтому мы не рекомендуем увеличивать число запрашиваемых процессоров сверх 2048.

9. Все вышеприведенные соображения относились к случаю, когда не используется никакая фильтрация вырожденных заданий. Фильтрация является эффективным средством сокращения перебора, поскольку позволяет уменьшить объем ненужных расчетов с такими наборами параметров, при которых нарушается правильное функционирование локуса, скажем, полностью прекращается транскрипция какого-то гена. Описанная выше логика распределения траекторий пучка между параллельными дочерними ветвями программы предполагает, что критерий вырожденности задания (заданный параметрами **Decision** и **MinRatio** в преамбуле файла заданий – см. п. 4.3, или эквивалентными параметрами в командной строке) должен либо сработать после того числа траекторий, которое определено для принятия решения, либо после этого не срабатывать до конца расчетов на всем пучке. В противном случае возрастает доля непроизводительных расчетов из-за того, что многие задания будут отброшены уже после того, как обработана (или почти обработана) б**о**льшая числа траекторий всего пучка, так что в этом случае целесообразнее было бы вообще не применять фильтрацию. Отсюда следует, что фильтр по возможности должен быть «тонкий» и отсеивать лишь те задания, которые заведомо окажутся вырожденными, что проявляется уже на небольшом числе первых траекторий.

10. При работе с большими файлами заданий следует заранее оценить объем будущих файлов результатов на предмет действующих системных или административных ограничений, а также возможности средств последующей автоматизированной обработки (например, ограничение нынешних версий электронной таблицы Excel – не более 65535 строк в одном рабочем листе). Не следует без необходимости злоупотреблять выводом промежуточных результатов вдоль траектории моделирования.

11. Настоятельно рекомендуется перед запуском большого задания на кластере предварительно проверить его правильность в однопроцессорном режиме на более доступной вычислительной установке или на том же кластере, но запрашивая меньший

объём ресурсов. Снижение быстродействия при этом можно частично скомпенсировать за счет уменьшения размера пучка (скажем, с 100–1000 до 2–3, но с учётом критерия фильтрации, если он используется) и/или физического времени моделирования одной траектории.

12. Обработку заданий на кластере лучше начинать, запрашивая небольшое число процессоров (8–32) на время порядка 5*t* минут, и только в случае успеха обработки переходить к запросу большего объема ресурсов.
13. Если это позволяет регламент, одновременно на кластере могут выполняться или находиться в очереди несколько пользовательских заданий. Для использования программы cpRivals в таком режиме необходимы отдельные файлы заданий для каждой запущенной копии программы.

Описанная в этом разделе технология и принципы успешно применялись в наших расчетах [1, 2] с использованием высокопроизводительного кластера МВС-100К в Межведомственном Суперкомпьютерном Центре РАН (www.jscc.ru), во время которых параллельная обработка заданий проводилась на 512–2048 процессорах сеансами по несколько часов (вплоть до 24).

4. Форматы файлов

В этой главе для справок приводится описание входных и выходных файлов программы. Оба варианта программы используют одни и те же файлы, отличия в интерпретации содержимого файлов особо оговорены в тексте.

Все файлы имеют текстовый формат и используют алфавит ASCII, однако следует помнить о различиях в оформлении конца строки в ОС Linux и Windows. При использовании в среде одной ОС файлов, подготовленных для другой ОС, могут возникать различные ошибки. Точно так же рекомендуется воздержаться от использования символов кириллицы в именах файлов и объектов задачи.

Общий формат файлов следующий. Строки, начинающиеся с символов ';' (точка с запятой) и '/' (наклонная черта), а также пустые строки рассматриваются как комментарии. Файл может состоять из разделов; каждый раздел начинается со строки, содержащей имя раздела в квадратных скобках. Раздел состоит из информационных строк; каждая строка имеет формат *параметр=значение* (перед и после знака равенства может стоять любое число пробелов и/или символов табуляции, вместо знака равенства также может использоваться двоеточие). Заглавные и строчные буквы в именах и значениях параметров эквивалентны, кроме тех случаев, когда значением параметра является имя файла или путь, а файловая система ОС чувствительна к регистру символов.

Описание файлов проводится на конкретных примерах типичных (и, по мере возможности, представительных) файлов из нашей практики. Строки комментариев не описываются.

4.1. Файл конфигурации

Ниже приводится содержимое типичного файла конфигурации. Как уже говорилось, редактировать файл вручную не рекомендуется; для этих целей служит меню **Options** программы Rivals (см. раздел 2.3.1).

```
; Rivals configuration file: rivals.ini
[General]
Version=1.5.3.3
Scratch=/scratch/rubanov/
```

```
PhysicalTime=18000
StartupTime=0
ElapsedTime=3600
CountCenter=True
CSV=True
Both=False
Delimiter=59
SAP=111111111
DAP=000000000
DPP=000000000
SAC=011000000
DAC=100111111
DPC=111011011
SAR=100
DAR=011
DPR=111
SAT=100
DAT=011
DPT=111
[PEP]
PEPSig1=True
BindSig1=0.5
PEPSig2=False
BindSig2=0.25
PEPSig3=True
BindSig3=0.5
PEPSig4=True
BindSig4=0.5
PEPSig5=False
BindSig5=0.25
PEPSig6=False
BindSig6=0.25
[Holoenzyme]
LengthSig1=29
AbortSig1=1.21
LengthSig2=31
AbortSig2=4
LengthSig3=29
AbortSig3=1.21
LengthSig4=29
AbortSig4=1.21
LengthSig5=31
AbortSig5=4
LengthSig6=31
AbortSig6=4
[Kor enzyme]
KorLeft=-15
KorRight=20
AbortRange=4
KorRate=12.1
[NEP]
NEPRpoT1=False
BindRpoT1=0.2
NEPRpoT2=False
BindRpoT2=0.2
NEPRpoTp=True
BindRpoTp=0.5
NEPRpoTm=False
BindRpoTm=0.5
NEPRpoTmp=True
BindRpoTmp=0.5
[Rpo]
LeftRpoT1=-25
RightRpoT1=8
RateRpoT1=42
LeftRpoT2=-24
RightRpoT2=9
RateRpoT2=42
LeftRpoTp=-15
RightRpoTp=4
RateRpoTp=45
```

```
LeftRpoTm=-9
RightRpoTm=14
RateRpoTm=42
LeftRpoTmp=-15
RightRpoTmp=4
RateRpoTmp=45
PEPover=1
NEPover=1
; End of configuration file
```

В приведённом файле содержится следующая информация:

[General] Раздел общих параметров конфигурации.

Version Номер версии программы, которой был записан данный файл. Оба варианта программы в момент запуска проверяют соответствие версии программы этому параметру конфигурации, выдавая предупреждение при несовпадении. (Программа Rivals при этом предлагает перезаписать файл конфигурации используемой версией программы).

Scratch Существующий каталог для временных файлов, (с символом / или \ на конце, в зависимости от ОС). Параметр используется только программой cpRivals, программа Rivals его игнорирует и не записывает (т.е. теряет) при сохранении конфигурации из диалогового окна **Options**).

PhysicalTime Физическое время моделирования в секундах, принимаемое по умолчанию. По достижении указанной величины траектория заканчивается. Значение 0 интерпретируется как отсутствие заданного ограничения физической длительности траектории. Максимально допустимое значение составляет $2^{31}-1 \approx 2\,000\,000\,000$ секунд (свыше 60 лет).

ElapsedTime Максимальная продолжительность компьютерного счета одной траектории, по достижении которой моделирование прекращается. Значение 0 интерпретируется как неограниченная продолжительность моделирования. Максимально допустимое значение составляет 100000 секунд (более суток). Программа cpRivals игнорирует данный параметр, продолжительность обработки одной траектории никак не ограничена (но обычно ограничивается при запуске общее время расчётов, после чего программу принудительно завершает диспетчер кластера).

CountCenter Возможные значения: **true**, **false**. Способ учета транскрипций гена: если указано **true**, начало транскрипции фиксируется, когда центр транскрипции кор-фермента или полимеразы фагового типа впервые оказывается в области, занимаемой геном, а завершение транскрипции фиксируется, когда центр транскрипции впервые выходит из области, занимаемой геном. Если указано **false**, то началом транскрипции считается, когда все нуклеотиды РНК-полимеразы оказываются в области гена, а концом – когда все они оказываются вне области гена.

CSV Возможные значения: **true**, **false**. Если указано **true**, то результаты моделирования будут записаны в формате .csv, иначе будет выдан только файл протокола (в котором по умолчанию результаты не присутствуют).

Both Возможные значения: **true**, **false**. Если указано **true**, то в формируемые по умолчанию записи результатов траектории в формате .csv будут включены значения обоих основных счетчиков для всех объектов задачи (для генов это счетчики начатых и завершенных транскрипций, для промоторов – попыток связывания РНК-полимеразы и успешных посадок ее на ДНК, и т.п.). В

противном случае в запись по умолчанию для каждого объекта включается значение только одного, наиболее важного счетчика (завершенных транскрипций, успешных посадок и т.п.).

Примечание: При работе с программой cpRivals или при запуске программы Rivals в пакетном режиме имеется возможность указать в файле заданий точный состав записи результатов (см. описание параметра **Record** в разделе 4.3); в этом случае данный параметр игнорируется.

Delimiter Указывает десятичное значение символа, который должен использоваться в качестве разделителя в файле формата .csv. Например: 9 – символ табуляции, 32 – пробел, 44 – запятая, 59 – точка с запятой.

SAP, DAP, DPP, SAC, DAC, DPC, SAR, DAR, DPR, SAT, DAT, DPT
Этими параметрами закодирована матрица правил разрешения коллизий (см. раздел 1.4.1). Параметры не предназначены для корректировки вручную, и поэтому детали использованного представления не описываются.

[PEP] Раздел общих параметров PEP-промотора, принимаемых по умолчанию. Далее в именах параметров вместо символа * стоит цифра от 1 до 6 в соответствии с обслуживаемым набором сигма-субъединиц, Sig1 – Sig6. Программа cpRivals не использует весь этот раздел.

PEPSig* Возможные значения: **true**, **false**. Если указано **true**, то данный тип сигма-субъединицы обслуживается программой (в частности, предлагается в интерактивном режиме при добавлении нового PEP-промотора), в противном случае этот тип сигма-субъединицы не обслуживается.

BindSig* Действительное значение, указывающее принимаемую по умолчанию интенсивность связывания λ с PEP-промоторами данного типа сигма-субъединицы.

[Holoenzyme] Раздел общих параметров холофермента, принимаемых по умолчанию. Далее в именах параметров вместо символа * стоит цифра от 1 до 6 в соответствии с обслуживаемым набором сигма-субъединиц, Sig1 – Sig6. Программа cpRivals не использует весь этот раздел.

LengthSig* Длина сигма-субъединицы данного типа по умолчанию (число нуклеотидов).

AbortSig* Действительное значение, указывающее принимаемую по умолчанию интенсивность абортивного процесса μ для холофермента с данным типом сигма-субъединицы.

[Kor enzyme] Раздел общих параметров кор-фермента, принимаемых по умолчанию. Программа cpRivals не использует весь этот раздел.

KorLeft Смещение 5'-конца кор-фермента относительно центра транскрипции (который имеет координату +1, нулевая координата не используется).

KorRight Смещение 3'-конца кор-фермента относительно центра транскрипции (который имеет координату +1, нулевая координата не используется).

AbortRange Максимально достигаемая величина r промежутка от сигма-субъединицы до кор-фермента в ходе абортивных процессов (число нуклеотидов).

KorRate Скорость сдвига R кор-фермента во время абортивных процессов и при транскрипции (нуклеотидов в секунду).

[NEP] Раздел общих параметров NEP-промотора, принимаемых по умолчанию. Далее в именах параметров вместо символа # стоит одно из обозначений полимеразы фагового типа (T1, T2, Tp, Tm, Tmp) в соответствии с

обслуживаемым набором полимераз фагового типа: RpoT1, RpoT2, RpoTp, RpoTm, RpoTmp. Программа cpRivals не использует весь этот раздел.

NEPRpo# Возможные значения: **true**, **false**. Если указано **true**, то данная полимераза фагового типа обслуживается программой (в частности, предлагается в интерактивном режиме при добавлении нового NEP-промотора), в противном случае эта полимераза фагового типа не обслуживается.

BindRpo# Действительное значение, указывающее принимаемую по умолчанию интенсивность связывания λ с NEP-промоторами данной полимеразы фагового типа.

[Rpo] Раздел общих параметров кор-фермента, принимаемых по умолчанию. Далее в именах параметров вместо символа # стоит одно из обозначений полимеразы фагового типа (T1, T2, Tp, Tm, Tmp). Программа cpRivals не использует весь этот раздел.

LeftRpo# Смещение 5'-конца полимеразы фагового типа относительно центра транскрипции (который имеет координату +1, нулевая координата не используется).

RightRpo# Смещение 3'-конца полимеразы фагового типа относительно центра транскрипции (который имеет координату +1, нулевая координата не используется).

RateRpo# Скорость сдвига R полимеразы фагового типа во время транскрипции (нуклеотидов в секунду).

PEPover Максимально допустимое перекрытие гена центром транскрипции PEP-промотора или внешнего источника кор-ферментов (параметр используется только графическим приложением Rivals).

NEPover Максимально допустимое перекрытие гена центром транскрипции NEP-промотора или внешнего источника полимераз фагового типа (параметр используется только графическим приложением Rivals).

4.2. Файл задачи

Ниже приводятся фрагменты содержимого типичного файла задачи (сокращен раздел с ДНК-последовательностью и опущены объекты повторяющихся типов). Раздел [DNA...] должен идти первым, очередность последующих разделов не имеет значения. Редактировать файл вручную не рекомендуется; для этого лучше использовать разнообразные средства GUI программы Rivals, описанные в главе 2.

```
; Rivals task file: D:\Work\Rivals\Linux\HSM\HSM.tsk
Version=1.5.3.3
[DNA from D:\Work\Rivals\Linux\HSM\Homo sapiens mitochondrion.gbk]
Length=16569
Circular=True
       1 gatcacaggt ctatcaccct attaaccact cacgggagct ctccatgcat ttggtatttt
      61 cgtctggggg gtatgcacgc gatagcattg cgagacgctg gagccggagc accctatgtc
     121 gcagtatctg tctttgattc ctgcctcatc ctattattta tcgcacctac gttcaatatt
     181 acaggcgaac atacttacta aagtgtgtta attaattaat gcttgtagga cataataata
...
   16441 actctcctcg ctccgggccc ataacacttg ggggtagcta aagtgaactg tatccgacat
   16501 ctggttccta cttcagggtc ataaagccta aatagcccac acgttcccct taaataagac
   16561 atcacgatg

...
```

```
[Term]  ***************************************
Name=CSB-L
Strand=False
Display=False
Enabled=False
Begin=282
End=300
Color=FF10E0E0
Lambda=0.2
SenseKor=True
SenseRpo=True

[NEP]  ***************************************
Name=LSP
Strand=False
Display=True
Enabled=True
Begin=407
End=422
Color=FF0000E0
Polym=2      // RpoTp
Lambda=0.001085
Tcenter=407
Rate=500

[Gene]  ***************************************
Name=Phe
Strand=True
Display=True
Enabled=True
Begin=577
End=647
Color=FF30C860

[Repr]  ***************************************
Name=mTERF
Display=True
Enabled=True
Begin=3230
End=3257
Color=FF800000
Lambda=0.6456
Qmain=0.0164
Qcomp=0.0056

...
; End of task file
```

Version Номер версии программы, которой был записан данный файл. Оба варианта программы в момент загрузки задачи проверяют соответствие версии программы этому параметру, выдавая предупреждение при несовпадении. (Программа Rivals при этом предлагает перезаписать данный файл задачи используемой версией программы).

[DNA ...] Раздел файла, содержащий ДНК-последовательность. В скобках может быть указан источник, например, путь к файлу, имя организма и т.п. (Оба варианта программы игнорируют эту информацию).

Length Длина последовательности ДНК, приведенной в этом разделе (используется для контроля правильности ввода).

Circular Если стоит значение **true**, эта последовательность будет обрабатываться как кольцевая (с указанной полной длиной). В противном случае последовательность считается линейной и имеет длину от начала самого левого до конца самого правого объекта. Если интересующий линейный

участок кольцевой последовательности содержит точку разреза кольца, его можно обрабатывать в линейном режиме, как описано в разделе 5.5.

В последующих строках раздела приводится сама последовательность в формате, близком к используемому ГенБанком NCBI: группы по 10 букв, 6 групп в строке, первое поле строки содержит порядковый номер первой буквы последовательности, стоящей в данной строке, разделитель полей – пробел.

[Term] Раздел параметров терминатора. В файле задачи должно содержаться по одному подобному разделу для каждого участвующего терминатора. Названия параметров идентичны свойствам терминатора (см. п. 1.3.11), порядок параметров несуществен.

Name Имя (идентификатор) терминатора.

Strand Признак цепи ДНК, на которой находится терминатор. Возможные значения – true (основная цепь) или false (комплементарная цепь).

Display Признак отображения терминатора в GUI. Возможные значения – true (отображать терминатор в рабочей области окна) или false (не отображать, как в приведённом примере). Значение не влияет на ход моделирования.

Enabled Признак использования терминатора. Возможные значения – true (учитывать терминатор при моделировании) или false (не учитывать, как в этом примере, где данный терминатор не моделировался).

Begin Позиция в основной цепи самого левого нуклеотида терминатора (если терминатор на основной цепи, то это его 5'-начало, если на комплементарной – 3'-конец). Может использоваться альтернативное название параметра, **Left**.

End Позиция в основной цепи самого правого нуклеотида терминатора (если терминатор на основной цепи, то это его 3'-конец, если на комплементарной – 5'-начало). Может использоваться альтернативное название параметра, **Right**.

Color Цвет заливки символа терминатора в GUI. (Здесь и далее цвет задается в форме четырехбайтового шестнадцатеричного значения в формате ARGB, значение каждой из цветовых компонент R,G,B занимает один байт, A соответствует т.н. альфа-каналу – однобайтовому показателю прозрачности, у которого FF означает полную непрозрачность, а 00 – полную прозрачность).

Lambda Значение λ интенсивности (частоты возникновения) данного терминатора.

SenseKor Признак действия терминатора на кор-фермент. Возможные значения – true (терминатор действует на кор-ферменты) или false (не действует).

SenseRpo Признак действия терминатора на полимеразы фагового типа. Возможные значения – true (терминатор действует на полимеразы фагового типа) или false (не действует).

[NEP] Раздел параметров NEP-промотора. В файле задачи должно содержаться по одному подобному разделу для каждого участвующего NEP-промотора. Названия параметров идентичны свойствам NEP-промотора (см. п. 1.3.7), порядок параметров несуществен.

Name Имя (идентификатор) NEP-промотора.

Strand Признак цепи ДНК, на которой находится NEP-промотор. Возможные значения – true (основная цепь) или false (комплементарная цепь).

Display Признак отображения NEP-промотора в GUI. Возможные значения – true (отображать промотор в рабочей области окна) или false (не отображать). Значение не влияет на ход моделирования.

Enabled Признак использования NEP-промотора. Возможные значения – true (учитывать промотор при моделировании) или false (не учитывать).

Begin Позиция в основной цепи самого левого нуклеотида NEP-промотора (если промотор на основной цепи, то это его 5'-начало, если на комплементарной – 3'-конец). Может использоваться альтернативное название параметра, **Left**.

End Позиция в основной цепи самого правого нуклеотида NEP-промотора (если промотор на основной цепи, то это его 3'-конец, если на комплементарной – 5'-начало). Может использоваться альтернативное название параметра, **Right**.

Color Цвет заливки символа NEP-промотора в GUI.

Polym Целое значение от 0 до 4, которое указывает вариант РНК-полимеразы фагового типа, соответственно, RpoT1, RpoT2, RpoTp, RpoTm, RpoTmp.

Lambda Значение λ интенсивности связывания для данного промотора.

Tcenter Позиция в основной цепи центра транскрипции полимеразы фагового типа сразу после ее посадки на данный NEP-промотор. Может использоваться альтернативное название параметра, **TrPoint**.

Rate Скорость движения полимеразы фагового типа R [нт/с].

[Gene] Раздел параметров гена. В файле задачи должно содержаться по одному подобному разделу для каждого гена рассматриваемого локуса. Названия параметров идентичны свойствам гена (см. п. 1.3.2), порядок несуществен.

Name Имя (идентификатор) гена. (Здесь и далее длина имени не должна превышать 31 символа; для удобства работы с задачей в GUI рекомендуется выбирать имена не длиннее 7–8 символов. Для обработки задачи в пакетном режиме имя не должно содержать пробелов, знаков табуляции и символов . , = : % ! ?)

Strand Признак цепи ДНК, на которой находится ген. Возможные значения – true (основная цепь) или false (комплементарная цепь).

Display Признак отображения гена в GUI. Возможные значения – true (отображать ген в рабочей области окна) или false (не отображать). Значение не влияет на ход моделирования.

Enabled Признак использования гена. Возможные значения – true (учитывать ген при моделировании) или false (не учитывать).

Begin Позиция в основной цепи самого левого нуклеотида гена (если ген на основной цепи, то это его 5'-начало, если на комплементарной – 3'-конец). Может использоваться альтернативное название параметра, **Left**.

End Позиция в основной цепи самого правого нуклеотида гена (если ген на основной цепи, то это его 3'-конец, если на комплементарной – 5'-начало). Может использоваться альтернативное название параметра, **Right**.

Color Цвет заливки символа гена в GUI.

[Repr] Раздел параметров сайта репрессии (или терминации). В файле задачи должно содержаться по одному подобному разделу для каждого участвующего репрессора/терминатора. Названия параметров идентичны свойствам репрессора (см. п. 1.3.10), порядок параметров несуществен.

Name Имя (идентификатор) репрессора.

Strand Признак цепи ДНК, на которой находится репрессор. Возможные значения – true (основная цепь) или false (комплементарная цепь).

Display Признак отображения репрессора в GUI. Возможные значения – true (отображать репрессор в рабочей области окна) или false (не отображать). Значение не влияет на ход моделирования.

Enabled Признак использования репрессора. Возможные значения – true (учитывать репрессор при моделировании) или false (не учитывать).

Begin Позиция в основной цепи самого левого нуклеотида сайта репрессии (если сайт на основной цепи, то это его 5'-начало, если на комплементарной – 3'-конец). Может использоваться альтернативное название параметра, **Left**.

End Позиция в основной цепи самого правого нуклеотида сайта репрессии (если сайт на основной цепи, то это его 3'-конец, если на комплементарной – 5'-начало). Может использоваться альтернативное название параметра, **Right**.

Color Цвет заливки символа репрессора в GUI.

Lambda Значение λ интенсивности связывания репрессора (или возникновения терминатора).

Qmain Вероятность протекания терминатора по основной цепи. Возможные значения – от 0 (терминатор сбрасывает все РНК-полимеразы) до 1 (терминатор не действует).

Qcomp Вероятность протекания терминатора по комплементарной цепи. Возможные значения – от 0 (терминатор сбрасывает все РНК-полимеразы) до 1 (терминатор не действует).

В этом примере не содержатся некоторые классы неподвижных объектов задачи, такие как PEP-промоторы и внешние источники кор-ферментов или полимераз фагового типа, но для них синтаксис соответствующих разделов файла задачи (соответственно, [PEP], [EKor] и [ERpo]) аналогичен приведенному выше, а перечень свойств и их наименования даны в разделе 1.3.

4.3. Файл заданий

В отличие от файлов конфигурации и задачи, которые обычно готовятся и корректируются средствами GUI программы Rivals, файл заданий полностью составляется пользователем с помощью подходящего текстового редактора или даже командного скрипта операционной системы, если формирование наборов параметров требует автоматизации. Ниже представлен фрагмент файла заданий, который демонстрирует все возможные виды управляющих строк; в реальных ситуациях не все они обязательны. Файл состоит из преамбулы и собственно пакета заданий, который начинается после строки заголовка раздела **[Job to run]**. При составлении и запуске файла заданий следует помнить, что некоторые указанные в нем параметры могут устанавливаться и в других местах (например, в командной строке запуска программы), причем в двух вариантах программы реализована разная очередность использования значений, заданных в нескольких местах (подробнее см. раздел 3.3). Как обычно, строки, начинающиеся символом ';' или '/', и пустые строки рассматриваются как

комментарии и не интерпретируются. Переносить длинные строки не разрешается, в примере ниже это сделано для удобства показа.

```
; Rivals job file
; Preamble lines follow
; Task file name (should be before Names line)
Task=HSM.tsk
; Name of output file(s)
Result=HSM
; Append to the results (default overwrite)
Append=false
; In process data refresh granularity
Granularity=10000
; Number of repetitions for each job line
Repeat=100
; Comma/semicolon/blank/tab separated names of variable parameters.
;  Each parameter has the format as follows: agent_name.parameter_name,
;  where agent_name comes from the task and parameter_name can be:
;  L = Lambda (binding parameter),
;  R = Polymerase rate,
;  A = Aborting parameter of PEP,
;  G = Aborting range parameter of PEP
;  M = Repr pass probability (main strand)
;  Q = Repr pass probability (complement strand)
Names= LSP.L HSP1.L HSP2.L mTERF.L
; scratch directory (must include terminating slash if nonempty!)
Scratch=/scratch/rubanov/
; Physical time in minutes
Pminutes=600
; Startup time in minutes
Startup=60
; Intermediate data output interval in minutes
Interval=0
; Seed value for random number generator
Seed=12345
; Number of runs to decide whether the point is acceptable
Decision=10
; Smallest acceptable Min / Sum transcription ratio
Minratio=0.001
; Join or split gene transcription counters by polymerase types (default join)
Joint=false
; Write both successful/attempt data into .CSV file
BothCSV=true
; CSV record structure (in addition to the variable parameter values)
;  The format is similar to the Names statement, additional options are:
;  # = serial number of the run (in multiple run case),
;  I = initiated transcription count (for genes),
;  E = initiated transcription count by Kors (from PEP),
;  F = initiated transcription count by Rpos (from NEP),
;  T = completed transcription count (for genes),
;  P = completed transcription count by Kors (from PEP),
;  N = completed transcription count by Rpos (from NEP),
;  O = number of binding attempts (all agents but genes),
;  B = number of successful bindings (all agents but genes)
;  D = number of RNAP terminated by Repr (main strand),
;  K = number of RNAP terminated by Repr (complement strand)
Record= # LSP.B HSP1.B Phe.T HSP2.B 12S.T Val.T mTERF.B mTERF.D mTERF.K 16S.T
Leu.T Ile.T Gln.T Met.T Trp.T Ala.T Asn.T Cys.T Tyr.T Ser.T Asp.T Lys.T Gly.T
Arg.T His.T Ser2.T Leu2.T Glu.T Thr.T Pro.T
; Output sample records (all trajectories) if true
Sample=true
; Output average record if true
Average=true
; Output std deviation record if true
StDev=false
; Number of decimal places in average/stdev values (0 default)
ResultPrecision=2
; Number of decimal places in parameter values (2 default)
ParamPrecision=3
```

```
; Dynamic parameter changes
; Set absolute value - Dynamic Name.X time value [time value ...]
Dynamic LSP.R       0 500
Dynamic HSP1.R      0 500
Dynamic HSP2.R      0 500
Dynamic mTERF.M     0 0.33
Dynamic mTERF.Q     0 0.01
; Set relative value: DynLinear Name.X time value1 value2 [time value ...]
; New value = preset * value1 + value2
; Rates @ T:      39C *3.82   35C *0.93
DynLinear LSP.R   180 3.82 0  270 0.93 0
DynLinear HSP1.R  180 3.82 0  270 0.93 0
DynLinear HSP2.R  180 3.82 0  270 0.93 0

; Begin of the job section
[Job to run]
; Job lines follow. The parameter values must be specified in the Names line
order.
;      LSP.L  HSP1.L HSP2.L mTERF.L
       0.02   0.15   0.05   0.84
       0.02   0.15   0.25   0.66
...
; End of file
```

В преамбуле файла заданий допускаются следующие параметры:

Task Имя файла задачи (п. 4.2), для которой необходимо выполнить пакет заданий. Имя может включать абсолютный или относительный путь к файлу, составленный по правилам операционной системы. По умолчанию файл ищется в текущем каталоге. Данный параметр не обязателен (задача может перед запуском задания уже быть загружена в GUI или имя задачи может быть указано в командной строке запуска), но если используется, то должен предшествовать параметрам **Names** и **Record**.

Result Имя для выходных файлов программы. В общем случае программа формирует файл протокола (с расширением .log) и один и более файлов результатов в формате CSV (с расширением .csv). В основе имени всех выходных файлов лежит значение данного параметра, которое может включать абсолютный или относительный путь к файлу, составленный по правилам операционной системы. По умолчанию файл создается в каталоге программы. Данный параметр не обязателен; значение может быть установлено по умолчанию (совпадает с именем задачи), либо задано в GUI или командной строке.

Append Режим записи в выходные файлы. Возможные значения – true (режим дозаписи в конец существующего файла) или false (старое содержимое файла стирается). В обоих случаях, если файл не существует, он создается заново.

Granularity Указывает периодичность обновления информации на экране во время моделирования траектории. Данный параметр не влияет на результаты моделирования, но может сказываться на быстродействии программы. Разные варианты программы интерпретируют указанное значение по-разному:

– программа Rivals (в пакетном режиме) обновляет значения *всех* счетчиков какого-либо объекта, как только значение *любого* его счетчика становится кратно указанному значению (не обновляя показания счетчиков для прочих объектов).

– программа cpRivals в однопроцессорном режиме выдает на консоль короткую запись с указанием физического времени траектории, длины

очереди и номера итерации всякий раз, когда номер итерации становится кратен указанному значению (последующая запись затирает предыдущую).

– программа cpRivals в мультипроцессорном режиме игнорирует этот параметр и не выдает никакой информации оператору по ходу траектории, поскольку на кластере параллельно может моделироваться большое число траекторий.

Repeat Задает число траекторий в пучке, которые должны быть получены при одинаковых значениях параметров для последующего усреднения результатов. Может указываться и в других местах; если параметр нигде не задан, то программа работает в режиме одиночной траектории.

Names В этом (обязательном) предложении перечисляются параметры модели, которые ниже варьируются в этом файле заданий. Параметры перечисляются через пробел или запятую (также в качестве разделителя здесь допускаются символ табуляции и точка с запятой). Каждый параметр должен иметь вид <*name*>.<*parameter*> , где <*name*> – имя объекта задачи, а <*parameter*> – код одного из допустимых параметров этого объекта, выделенный в Табл. 1 полужирным шрифтом. Допускается указывать только <*name*> (что эквивалентно <*name*>.L). Вместо точки в качестве разделителя допускаются также символы ? ! %.

Scratch Существующий каталог для временных файлов (с символом / или \ на конце, в зависимости от ОС). Параметр используется только программой cpRivals.

Pminutes Моделируемое физическое время в минутах (длительность траектории). Значение не должно превышать 35 791 394 мин. Параметр может также задаваться в других местах (см. 3.3).

Startup Длительность в минутах (моделируемого физического времени) фазы разгона модели, во время которой подсчёт событий не ведётся. Параметр может также задаваться в командной строке и в файле конфигурации (см. 3.3).

Interval Периодичность выдачи промежуточных данных по ходу траектории, в минутах (моделируемое физическое время). Значение 0 интерпретируется как отсутствие выдачи. Параметр может также указываться в командной строке. Если указано ненулевое значение этого параметра, в конце каждой строки файла результатов будет автоматически добавляться поле со значением текущего физического времени в минутах.

Seed Базовое значение («зародыш») генератора псевдослучайной последовательности. Может задаваться также в других местах (см. 3.3). Значение 0 интерпретируется как установка начального значения на основании таймера (рандомизация); этот режим действует по умолчанию.

Decision Число траекторий пучка, после которого начинает проверяться критерий вырожденности задания. Значение 0 выключает фильтрацию заданий. Если указано ненулевое значение *s*, то первые *s* траекторий в пучке моделируются в безусловном порядке для каждого задания в пакете. По завершении *s*-й и каждой последующей траектории (вплоть до конца пучка) выполняется следующая операция: подсчитывается суммарное число N_i транскрипций каждого гена во всех выполненных траекториях и суммарное число транскрипций всех генов во всех выполненных траекториях, $N = \sum_i N_i$.

Отношение наименьшей из первых величин ко второй величине,

$\rho = \min_i N_i / N$ используется в критерии фильтрации (см. параметр **Minratio**).

Данный параметр также может задаваться в командной строке.

Minratio Пороговое значение критерия фильтрации заданий (см. описание параметра **Decision**). Если при моделировании пучка траекторий на каком-то этапе вычисленная величина ρ оказывается меньше значения, указанного этим параметром, то дальнейшая обработка пучка не проводится (задание признается вырожденным, т.к. экспрессия какого-то гена слишком мала по сравнению с остальными). Если указано Decision=0, то значение данного параметра игнорируется. Параметр также может задаваться в командной строке.

Joint Устанавливает режим учета транскрипций генов. Возможные значения – true или false. Если указано true, в файл результатов выдается суммарное число транскрипций каждого гена, без разделения по типам РНК-полимераз. Если указано false, то подсчитывается и выдается число начатых/завершенных транскрипций раздельно по полимеразам эубактериального типа (с PEP-промоторов) и полимеразам фагового типа (с NEP-промоторов). Этот параметр учитывается только в том случае, когда запись файла результатов формируется по умолчанию, т.е. в файле заданий нет предложения **Record**.

BothCSV Управляет набором счетчиков, значения которых по умолчанию выводятся в файл результатов. Возможные значения – true или false. Если указано true, в файл результатов выдаются значения *двух* счетчиков для каждого объекта, например, для генов – число инициированных и завершенных транскрипций, для промоторов – число попыток посадки полимеразы и число успешных посадок, и т.д. Если указано false, то выдается *только* значение счетчика успешных событий (завершенных транскрипций, посадок полимеразы и т.п.). Этот параметр учитывается только в том случае, когда запись файла результатов формируется по умолчанию, т.е. в файле заданий нет предложения **Record**.

Record В этом предложении пользователь может вместо стандартной по умолчанию задать желаемую структуру записи (т.е. строки) файла результатов в формате CSV. (Это не касается используемого разделителя полей, который можно изменить только в файле конфигурации – см. п. 4.1). В безусловном порядке, в начало каждой записи включаются значения переменных параметров в той очередности, как они записаны в предложении **Name**. Содержанием остальной части записи пользователь может управлять. Параметры в правой части предложения перечисляются через пробел или запятую (также в качестве разделителя здесь допускаются символ табуляции и точка с запятой). Каждый параметр должен иметь вид <*name*>.<*parameter*> , где <*name*> – имя объекта задачи, а <*parameter*> – код одного из возможных параметров этого объекта, представленный в Табл. 1. Вместо точки для отделения имени объекта от его параметра также могут использоваться символы ? ! %.

Исключением является особый вид параметра, # , который указывает, что в данном поле требуется выдавать не параметр какого-либо объекта, а порядковый номер траектории в пучке (без этого результаты от разных траекторий оказываются трудно различимыми). (Программа cpRivals также выводит в это поле признаки итоговых строк для всего пучка, если они были запрошены). В случае выдачи промежуточных результатов поле физического времени выдаётся автоматически и здесь не фигурирует.

Sample Если в данном файле заданий запрошено моделирование пучков траекторий (т.е. указано значение параметра **Repeat**, большее 1), это предложение определяет, будут ли выводиться в выходные файлы результаты по отдельным траекториям (если указано true) или только усреднённые (false). По умолчанию используется значение false. Следует помнить, что усреднённые данные может выводить в файлы .csv только программа cpRivals, а программа Rivals выводит их только в файл протокола.

Average При условии что в данном файле заданий запрошено моделирование пучков траекторий, если указать true, для каждого пучка в файлы результатов будет выведена запись со средними значениями всех счётчиков по этому пучку траекторий (этот режим используется по умолчанию). Если указано false, запись со средними значениями не выдаётся. Программа Rivals игнорирует данный параметр.

StDev При условии что в данном файле заданий запрошено моделирование пучков траекторий, если указать true, для каждого пучка в файл результатов будет выведена запись со стандартными отклонениями значений всех счётчиков по этому пучку траекторий (этот режим используется по умолчанию). Если указано false, запись со стандартными отклонениями не выдаётся. Программа Rivals игнорирует данный параметр.

ResultPrecision Параметр указывает требуемое число десятичных знаков после запятой в выдаваемых значениях среднего и стандартного отклонения (по умолчанию значения округляются до целого). Программа Rivals игнорирует данный параметр.

ParamPrecision Параметр указывает требуемое число десятичных знаков после запятой в значениях варьируемых параметров в файле результатов (по умолчанию выдаются два знака после запятой). Программа Rivals игнорирует данный параметр.

Dynamic Это предложение позволяет проводить ступенчатое изменение некоторого параметра модели в ходе каждой траектории (подробнее см. раздел 1.4.2). Разрешается динамически изменять любые параметры, которые можно указывать в предложениях **Names** и **Record**. Первый аргумент в этом предложении задаёт параметр в таком же формате <*name*>.<*parameter*> , где <*name*> – имя объекта задачи, а <*parameter*> – код одного из параметров этого объекта (Табл. 1). Далее следуют до 16 пар числовых значений, первое из которых указывает момент моделируемого физического времени (в минутах), а второе – то значение, которое должно быть придано этому параметру в указанный момент времени. В частности, если указано время 0, то значение будет присвоено параметру перед началом моделирования траектории. Любой параметр может присутствовать только в одном предложении **Dynamic**. Максимальное число таких предложений – 64.

DynLinear Это предложение служит для той же цели, что и **Dynamic,** обеспечивая однотипное ступенчатое изменение параметров модели на протяжении каждой траектории. Отличие состоит в том, что в первом предложении новое значение параметра указывалось в виде числа, одинакового для всех наборов параметров, которые идут далее в этом файле заданий. В данном же предложении новое значение параметра устанавливается по фиксированному закону с учётом начального значения, присвоенного в начале траектории, например, среди параметров конкретного задания из этого файла. В данной версии программы реализована только линейная зависимость от начального значения. Конкретно, первый аргумент в этом предложении задаёт параметр

модели в том же формате <*name*>.<*parameter*>, что и в других предложениях, а далее следуют до 16 *троек* числовых значений. В каждой тройке первое число указывает момент моделируемого физического времени (в минутах), второе задаёт коэффициент, а третье – свободный член линейной зависимости. Иными словами, если параметр имеет значение *x* в начале траектории, в момент времени, указанный первым числом, ему будет присвоено значение *xV2+V3*, где *V2* и *V3* соответственно второе и третье число в данной тройке значений. Всего в файле заданий может быть указано также не более 64 таких предложений.

С помощью предложений **Dynamic** и **DynLinear** удаётся моделировать сложные экспериментальные сценарии, например, эксперименты по тепловому шоку, в которых температура и зависящие от неё параметры процесса транскрипции изменяются на протяжении моделирования. Кроме того, они позволяют удобнее задавать общие параметры для всех заданий.

[Job to run] Заголовок пакета заданий

Далее до конца файла следуют строки пакета заданий. Каждая строка (не являющаяся строкой комментариев) должна содержать значения всех переменных параметров задачи в том количестве и в той последовательности, как они указаны в предложении **Names**. В качестве разделителей допускаются пробел, символ табуляции, запятая и точка с запятой (несколько разделителей подряд трактуется как один). Число строк в пакете не ограничено. *Примечание:* Рекомендуется начинать каждую строку с набором параметров как минимум с одного пробела, потому что программа отмечает успешно выполненные задания символом комментария (точка с запятой) в первой позиции строки.

4.4. Файл протокола

Пример выдачи на консоль протокола обработки тестового задания из дистрибутивного комплекта программы cpRivals в однопроцессорном режиме показан на Рис. 26. Запуск производился командой **make test** (или **cprivals -nompi -rjob3.rjb --j**).

```
cpRivals: Model of regulation by competing promoters (CLI&MPI) V1.5.3.3
 Copyright (C) 2009-2012 Lev Rubanov <rubanov@iitp.ru>
 Copyright (C) 2009-2012 Institute for Information Transmission Problems RAS
 Made in Laboratory of Mathematical Methods and Models in Bioinformatics,
 Head of the Lab: Vassily Lyubetsky <lyubetsk@iitp.ru>  http://lab6.iitp.ru
 This is free software, it is distributed WITHOUT ANY WARRANTY. See GNU GPL.

[0000] J   1/6 R   1/3 1K5 1K5   0 1K4 1K4 1K3 414 1K5 1K4 1K5 Stop 0'
[0000] J   2/6 R   1/3 1K4 1K4   2 1K4 1K3 1K2 770 1K6 1K6 1K6
[0000] J   2/6 R   2/3 1K5 1K5   5 1K4 1K4 1K3 654 1K6 1K5 1K6
[0000] J   2/6 R   3/3 1K5 1K5   3 1K4 1K4 1K3 593 1K5 1K5 1K6 Done 0'
[0000] J   3/6 R   1/3 1K5 1K5   3 1K4 1K4 1K2 906 1K7 1K6 1K7
[0000] J   3/6 R   2/3 1K4 1K4   5 1K3 1K3 1K2 886 1K7 1K6 1K7
[0000] J   3/6 R   3/3 1K5 1K5   3 1K4 1K3 1K2 846 1K7 1K6 1K7 Done 1'
[0000] J   4/6 R   1/3 1K5 1K4   1 1K4 1K4 1K2 1K2 1K8 1K8 1K9 Stop 0'
[0000] J   5/6 R   1/3 1K5 1K5   5 1K4 1K4 815 1K5 1K9 1K8 1K9
[0000] J   5/6 R   2/3 1K5 1K5   2 1K4 1K4 888 1K6 1K9 1K9 2K0
[0000] J   5/6 R   3/3 1K5 1K5   1 1K4 1K4 787 1K5 1K9 1K9 1K9 Done 1'
[0000] J   6/6 R   1/3 1K4 1K4   2 1K3 1K3 587 1K7 2K0 1K9 2K0
[0000] J   6/6 R   2/3 1K5 1K5   2 1K4 1K2 359 1K8 2K0 1K9 2K0
[0000] J   6/6 R   3/3 1K5 1K5   3 1K4 1K3 509 1K8 2K0 2K0 2K1 Done 0'
[0000] Job file processing completed after 1 min.
```

Рис. 26. Консольный протокол обработки задания.

После заголовочной части протокола программа выдает по одной строке после выполнения каждой траектории моделирования. Структура выдачи следующая: [0000] – номер параллельной ветви программы (всегда 0 в однопроцессорном режиме); J *x*/*y* – порядковый номер задания в пакете (*x*) и общее число заданий в пакете (*y*); R *m*/*n* – порядковый номер траектории в пучке (*m*) и размер всего пучка (*n*). Далее приводится число транскрипций каждого гена задачи в порядке слева направо с учетом обеих цепей; для экономии ширины строки представлено приближенное значение, так чтобы оно занимало ровно 3 символа (см. Табл. 4). Точные значения записываются в файл результатов.

Если по результатам данной траектории задание было отсеяно, то далее в строке выдается слово Stop и общее время выполнения этого задания в минутах (до отсева). Если задание не было отсеяно, то вслед за данными последней траектории в пучке пучка выдается слово Done и общее время выполнения этого задания в минутах. В данном примере были отсеяны задания 1 и 4. По завершении всего пакета заданий выдается общее время его обработки.

Табл. 4. Представление значений счетчиков транскрипций на консоли.

№	Минимум	Максимум	Запись минимума	Запись максимума
1	0	999	0	999
2	1000	9949	1K0	9K9
3	9950	99499	10K	99K
4	99500	994999	M10	M99
5	995000	9949999	1M0	9M9
6	9950000	99499999	10M	99M

В результате обработки задания будет записан файл протокола AT1.log следующего содержания (эталонный файл с именем AT1_chk.log имеется в составе дистрибутивного комплекта программы):

```
Job=job3.rjb (6 remain) Task=AT1.tsk Runs=3 Time=60 Startup=0 Seed=12345 v.1.5.3.3
    0m: Job    1 (line  102) degenerated, time =    0.1 min
    0m: Job    2 (line  103) completed,   time =    0.3 min
    1m: Job    3 (line  104) completed,   time =    0.3 min
    1m: Job    4 (line  105) degenerated, time =    0.1 min
    1m: Job    5 (line  106) completed,   time =    0.3 min
    1m: Job    6 (line  107) completed,   time =    0.3 min
Job file processing completed after 1 min.
```

В первой, заголовочной строке указано имя файла заданий (job3.rjb), число подлежащих выполнению заданий в пакете (6), имя файла задачи (AT1.tsk), число траекторий в пучке (3), длительность траектории в минутах (60), длительность фазы разгона в минутах (0), начальное значение для инициализации генератора псевдослучайной последовательности (12345) и номер версии программы. Затем выдается по одной строке для каждого обработанного задания в пакете, со следующим содержимым: порядковый номер задания, номер строки в файле заданий, результат обработки (“degenerated” для отсеянных заданий, “completed” для заданий, выполненных целиком) и продолжительность обработки. Если файл заданий был обработан до конца (т.е. программа не была остановлена оператором или вследствие сбоя), то в последней строке указывается общее время обработки пакета заданий.

Приведенный пример демонстрирует минимальный объем информации, который выдается в файл протокола по умолчанию. Программа Rivals способна выдавать значительно более подробный протокол по каждой траектории, для чего используются возможности меню **Edit > Options > Miscellaneous** (п. 2.3.1.6). Эти средства могут быть полезны при отладке и детальном анализе поведения модели, в данном руководстве они не рассматриваются.

4.5. Файл результатов

Файл результатов формируется в формате CSV (значения с разделителями) и предназначен для представления результатов работы программы и для возможной последующей автоматизированной обработки с привлечением сторонних программ и скриптов. В частности, если такой файл содержит не более 65535 строк, он может быть загружен непосредственно в электронную таблицу Excel. Обработка результатов находится в ведении пользователя и здесь не рассматривается.

При обработке рассмотренного в разделе 4.4 примера файла заданий job3.rjb будет сформировано два файла результатов с именами AT1.csv и AT1_rejected.csv соответственно (для контроля их эталонные копии с именами AT1_chk.csv и AT1_rejected_chk.csv включены в дистрибутивный комплект программы cpRivals). Первый файл содержит результаты успешно выполненных заданий, второй – частичные результаты для тех заданий, которые были отфильтрованы после некоторой траектории пучка. Формат обоих файлов одинаковый и описывается на примере представленного ниже содержимого файла AT1.csv.

```
"T1.L";"T2.L";"P.L";"P*.L";"P^.L";"N.L";"#";"psbB.T";"psbT.T";"psbN.T";"psbH.T";"petB.T";"petD.T";"rpoA.
T";"rps11.T";"rpl36.T";"trnI.T";"Minutes";
0.20;0.25;0.85;0.05;0.05;0.55;"Mean";0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0;
0.20;0.25;0.85;0.05;0.05;0.55;"StDev";0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0;
0.20;0.25;0.85;0.05;0.05;0.55;"Mean";731.3;729.7;3.0;680.3;665.0;593.7;279.3;736.7;741.3;817.7;30;
0.20;0.25;0.85;0.05;0.05;0.55;"StDev";17.8;18.2;1.7;22.6;21.1;2.1;57.0;36.1;5.9;8.0;30;
0.20;0.25;0.85;0.05;0.05;0.55;"Mean";1469.0;1465.0;3.3;1368.7;1355.0;1238.3;672.3;1582.3;1550.7;1619.7;60;
0.20;0.25;0.85;0.05;0.05;0.55;"StDev";26.0;25.2;1.5;16.5;17.3;37.6;89.9;43.6;8.3;10.0;60;
0.20;0.25;0.85;0.05;0.05;0.60;"Mean";0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0;
0.20;0.25;0.85;0.05;0.05;0.60;"StDev";0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0;
0.20;0.25;0.85;0.05;0.05;0.60;"Mean";711.0;708.0;3.7;662.3;648.7;569.0;403.7;789.7;773.0;846.7;30;
0.20;0.25;0.85;0.05;0.05;0.60;"StDev";9.6;10.8;1.2;9.0;8.5;33.0;36.8;18.2;5.3;6.4;30;
0.20;0.25;0.85;0.05;0.05;0.60;"Mean";1460.7;1458.0;3.7;1358.7;1342.3;1204.7;879.3;1678.3;1626.3;1713.0;60;
0.20;0.25;0.85;0.05;0.05;0.60;"StDev";15.2;15.7;1.2;39.1;36.7;23.4;30.6;5.5;13.9;16.1;60;
0.20;0.25;0.85;0.05;0.05;0.70;"Mean";0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0;
0.20;0.25;0.85;0.05;0.05;0.70;"StDev";0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0;
0.20;0.25;0.85;0.05;0.05;0.70;"Mean";724.7;721.7;2.7;677.0;660.7;416.7;673.0;882.7;874.7;964.3;30;
0.20;0.25;0.85;0.05;0.05;0.70;"StDev";18.6;18.6;2.1;19.5;20.5;139.8;159.5;53.3;40.8;31.4;30;
0.20;0.25;0.85;0.05;0.05;0.70;"Mean";1465.3;1462.0;2.7;1385.0;1370.3;830.0;1547.3;1898.3;1863.3;1955.0;60;
0.20;0.25;0.85;0.05;0.05;0.70;"StDev";9.7;10.8;2.1;16.1;16.2;52.1;72.0;41.3;28.2;32.2;60;
0.20;0.25;0.85;0.05;0.05;0.75;"Mean";0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0;
0.20;0.25;0.85;0.05;0.05;0.75;"StDev";0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0;0;
0.20;0.25;0.85;0.05;0.05;0.75;"Mean";729.0;725.3;2.0;677.7;635.7;307.3;774.3;932.0;927.0;1026.3;30;
0.20;0.25;0.85;0.05;0.05;0.75;"StDev";5.3;5.0;1.0;5.8;18.0;63.5;36.1;24.3;36.4;29.7;30;
0.20;0.25;0.85;0.05;0.05;0.75;"Mean";1464.0;1460.7;2.3;1365.7;1274.7;485.0;1764.3;1977.3;1948.7;2043.0;60;
0.20;0.25;0.85;0.05;0.05;0.75;"StDev";23.6;22.4;0.6;24.0;62.6;115.9;14.2;8.5;27.0;25.1;60;
```

В файлах .csv данные записываются в текстовом формате; каждая запись представляется отдельной строкой, состоящей из полей с разделителями. Число полей во всех записях одно и то же. В качестве символа-разделителя полей в формате CSV используется тот символ, который был указан в файле конфигурации программы (п. 4.1), в данном примере – точка с запятой. Числовое поле дается в обычной десятичной записи, целая часть отделяется от дробной части точкой (это необходимо учитывать при использовании локализованных версий сторонних программ). Содержимое символьного поля заключается в кавычки.

Первая строка файла содержит имена полей, которые указываются в такой же форме, как и в предложениях **Names** и **Record** в файле заданий (см. п. 4.3). Поле с именем # содержит порядковый номер траектории внутри пучка (в данном примере их выдача не запрошена), а для строк с усреднёнными данными по пучку траекторий – слово “Mean” (если в строке приведены средние значения) или “StDev” (если приведены стандартные отклонения). В

конце каждой записи может стоять ещё поле “Minutes” (момент моделируемого физического времени от начала траектории, которому соответствуют приведённые данные), если в файле заданий или командной строке был запрошен вывод промежуточных данных по ходу траектории. В данном примере была запрошена выдача промежуточных результатов каждые 30 мин., поэтому результаты выдаются трижды – в начале, середине и конце траектории. Если выдаются данные отдельно по каждой траектории пучка, то данные разных пучков отделяются пустой строкой.

5. Кое-что на закуску

В этой главе собрана различная дополнительная информация, которая отражает наш опыт использования программы и возможные направления ее дальнейшего развития.

5.1. О быстродействии программы

Жизненный цикл любого программного обеспечения – от сложнейших операционных систем до простеньких утилит – демонстрирует общую печальную закономерность. Если программа оказывается неспособной решать даже простые задачи, для которых она изначально предназначалась, или делает это неприемлемо долго, то она не находит практического применения и через короткий срок тихо «умирает», уступая место более удачным реализациям. Если же программа работает достаточно быстро (иногда даже быстрее, чем предполагал разработчик), то это провоцирует пользователей поручать ей значительно более сложные или трудоемкие задачи, чем изначально предполагалось – и в итоге быстродействие разработанного программного обеспечения опять становится недостаточным, что вынуждает искать пути оптимизации программы, совершенствовать алгоритм и т.д.

Описанная в настоящем руководстве разработка, которая ведётся с 2008 г., не стала исключением. Первоначально программа Rivals с GUI разрабатывалась с ориентацией скорее на удобство работы пользователей при подготовке исходных данных и несложные эксперименты с ними, а не на массовые расчеты. Для типичной задачи, с локусом из 10 генов, содержащим 4 промотора со средними значениями интенсивности связывания, программа затрачивает на моделирование одной траектории с физической продолжительностью 1 час порядка минуты компьютерного времени, выполняя при этом свыше 20 млн. итераций модели, что не так уж медленно. Однако для решения обратной задачи методом перебора такие расчёты необходимо проводить многократно, меняя каждый раз значения параметров, что утомительно делать в интерактивном режиме. Результатом стало добавление в программу возможности работы в пакетном режиме, с заранее подготовленным набором значений параметров. При таком применении все интерактивные возможности программы уже становятся ненужными и только тормозят работу.

Далее выявилось, что во многих случаях для получения статистически достоверных результатов требуются траектории значительно большей физической продолжительности, от 10 час. до нескольких суток, и что к тому же результаты необходимо усреднять по большому числу таких траекторий, порядка сотен и тысяч. Повысить быстродействие программы на 3–4 порядка не представлялось возможным. Результатом стала разработка новой программы, cpRivals, которая являет собой утилиту командной строки, без интерактивности и без привязки к конкретной платформе. Такая программа уже сама по себе работает быстрее и, кроме того, позволяет распараллеливать вычисления. Так была достигнута скорость работы, позволившая нам исследовать ряд локусов с помощью высокопроизводительных кластеров.

Однако по колоссальным наборам данных, которые формирует программа cpRivals, часто бывает трудно выявить закономерности влияния отдельных параметров на ход процесса. Это

затрудняет планирование вычислений с подбором многочисленных параметров модели. Поэтому в версии 1.5.x.x алгоритм работы интерактивной программы был существенно ускорен, и оба варианта были сделаны более совместимыми, что позволяет строить задачу и проводить начальный подбор параметров с помощью GUI, и лишь потом переходить к программе cpRivals для получения более достоверных окончательных результатов.

Таким образом, в данной разработке использование одного варианта программы стимулирует развитие и совершенствование другого варианта. Поэтому на каждом этапе они неизбежно оказываются не идентичными, однако эти отличия некритичны.

Учитывая конечность доступных вычислительных ресурсов, пользователю важно учитывать те резервы, которые могут содержаться в исходных данных, предлагаемых программе. Для этого ниже в порядке убывания важности перечисляются те факторы, от которых зависит скорость работы программы. Мы оставляем за скобками два важнейших фактора – физическое время одной траектории и число траекторий в пучке, поскольку время вычислений зависит от каждого из этих параметров в первом приближении линейно. Задача пользователя – по результатам предварительных расчетов решить, обмениваются ли эти параметры (и если да, то в каком соотношении), и какие их значения минимально необходимы для достоверного результата решения конкретной задачи (см. также раздел 5.2).

1) **Длина очереди событий**
   Это важнейший фактор, определяющий быстродействие программы при выбранном алгоритме реализации модели. Очередь должна быть всегда упорядочена хронологически (по возрастанию значения свойства **TouchTime** всех динамических объектов), и на каждой итерации моделирования указанное свойство модифицируется по меньшей мере у одного объекта. В отличие от прежних версий программы, в которых зависимость трудоёмкости обслуживания очереди от её длины была в лучшем случае линейной (а в среднем – медленнее), в текущей версии очередь построена на основе двоичной кучи с вычислительной сложностью добавления и удаления событий $O(\log q)$, где q – длина очереди. Это несколько снизило критичность данного фактора, но, конечно, не отменило его. Пользователь не может прямо повлиять на очередь, но может наблюдать за ее длиной по информации, которую программа cpRivals выдает на консоль в однопроцессорном режиме, а программа Rivals – в составе расширенной выдачи в файл протокола. Однако все перечисляемые ниже факторы, находящиеся уже под контролем пользователя, в большей или меньшей степени влияют именно на длину очереди событий.

2) **Число динамических объектов**
   Здесь имеются в виду прежде всего неподвижные динамические объекты, т.е. промоторы, репрессоры, терминаторы и внешние источники РНК-полимераз, а также динамически изменяемые параметры объектов (фактически, каждый из таких параметров также образует самостоятельный динамический объект для каждой смены его значения). Эти объекты, в свою очередь, уже могут порождать подвижные динамические объекты – холоферменты, кор-ферменты и полимеразы фагового типа. В общей сложности, чем больше динамических объектов, тем длиннее будет очередь событий. Поэтому в первом приближении следует стремиться к уменьшению числа неподвижных динамических объектов, особенно промоторов и внешних источников. Один из используемых для этого приемов изложен в п. 5.4. Наряду с прочим, при уменьшении числа объектов снижается и размерность пространства неизвестных параметров, в котором ищется решение задачи.

3) **Интенсивность динамических объектов**
   Чем выше интенсивность любого динамического объекта, тем чаще он оказывается на первом месте в очереди событий. Даже если при этом не порождается новый подвижный

объект (например, когда сайт посадки промотора перекрыт), это все равно замедляет работу программы. Поэтому не следует увеличивать интенсивности сверх предполагаемых границ. При увеличении интенсивности рекомендуется наблюдать за долей успешных попыток (т.е. за обоими счетчиками, например, попыток посадки полимеразы на промотор и успешных посадок). Если увеличение интенсивности не приводит к заметному повышению числа успешных попыток (а увеличивается только число неудачных), то дальше увеличивать интенсивность не имеет смысла. Следует также учитывать характер действия конкретного объекта. Например, для промотора или внешнего источника рост интенсивности связывания способствует увеличению числа полимераз, т.е. ведет к удлинению очереди. Напротив, для репрессора и терминатора с ростом интенсивности зависимость обратная, т.к. больше полимераз сбрасывается с ДНК.

4) **Длина последовательности**
РНК-полимераза любого типа после ее порождения (т.е. связывания с ДНК) существует в модели до тех пор, пока она либо не сорвется в результате коллизии с другим объектом, либо не выйдет за границу последовательности. Поэтому чем длиннее рассматриваемый участок последовательности, тем дольше в среднем существуют полимеразы, и, значит, тем длиннее будет очередь событий. Чтобы избежать ненужных затрат при моделировании, программа автоматически уничтожает полимеразу, когда она выходит за пределы задачи, т.е. правее самого правого объекта или левее самого левого. Тем не менее пользователь должен всегда стремиться к минимальной длине задачи, например, не включать в нее гены на краях, если их экспрессия не особенно важна для изучаемого эффекта, размещать внешние источники полимераз не очень далеко от крайних объектов задачи и т.п. Всё вышесказанное в этом пункте не относится к случаю обработки в кольцевом режиме, когда длина последовательности не подлежит изменению. Однако следует оценить, насколько существенно использовать в задаче именно этот режим (грубо говоря, появляются ли в интересующей области РНК-полимеразы, прошедшие полный круг). Во многих случаях сравнительно короткие участки кольцевых последовательностей большой длины могут с успехом моделироваться в линейном режиме, возможно – с внешними источниками. См. также раздел 5.5.

5) **Число генов**
С увеличением числа генов программа тратит больше времени на проверку того, что РНК-полимераза входит в область гена или покидает её, что необходимо для ведения соответствующих счётчиков. По этой причине для больших локусов может оказаться полезным включать в задачу не все промежуточные гены в составе длинных цепочек, особенно если конкретные данные по каждому гену не особенно важны.

5.2. Об интерпретации результатов

Как обычно при стохастическом моделировании, полученные результаты важно еще правильно проинтерпретировать. Выше уже неоднократно упоминалось усреднение на пучке траекторий. **Пользователь программы в ходе предварительных экспериментов должен сам оценить статистические характеристики получаемых данных и желаемую погрешность, после чего выбрать подходящие физическую длительность траекторий и размер их пучка.** Следует иметь в виду, что общеупотребительные методы обработки результатов измерений в применении к данной модели могут приводить к неправильным выводам, поскольку они часто неявно базируются на предположении, что результаты измерения суть реализации нормально распределенной случайной величины, для которой (при таком предположении) с заданной погрешностью находятся, например, математическое ожидание и дисперсия или стандартное отклонение.

Как показывают эксперименты, распределение числа транскрипций генов в нашей модели часто заметно отличается от нормального. В качестве примера на Рис. 27 приведены гистограммы числа транскрипции трех генов одного из рассмотренных нами локусов в хлоропласте природного типа *Arabidopsis thaliana*, полученные на пучке из 1000 траекторий, физическая продолжительность каждой траектории составляла 10 часов. По горизонтальной оси в каждом случае отложено число транскрипций (точнее – правая граница соответствующей ячейки гистограммы), а по вертикальной оси – количество траекторий, в которых число транскрипций гена попадало в эту ячейку. Из рисунка хорошо видно, что у первых двух генов распределение имеет значительную асимметрию, а для третьего гена распределение вообще больше похоже на пуассоновское, чем на нормальное. Не пытаясь дать конкретные рецепты для подобных ситуаций, мы лишь хотим привлечь внимание пользователя к важности грамотной обработки и правильной интерпретации результатов, полученных с помощью описываемой программы.

5.3. Об ограничениях текущей версии программы

В программе всюду используется плавающая арифметика двойной точности и 32-битная целочисленная арифметика. Единственные 64-битные переменные в программе – это номер итерации (т.к. при представляющих интерес физических продолжительностях траекторий число итераций моделирования иногда превышает 2^{31}) и аккумуляторы, применяемые при вычислении средних значений и стандартных отклонений (поскольку пучок может состоять из большого числа траекторий). Все прочие переменные не должны выходить за 32-битный предел, иначе могут возникать трудно обнаруживаемые ошибки, так как фиксированное переполнение обнаруживается не на всех платформах и сама программа не отслеживает такие события.

Сказанное в первую очередь относится к счетчику физического времени моделирования (в секундах) и к счетчикам событий у всех объектов (например, счетчикам числа транскрипций у генов). Занимаясь моделированием траекторий с постепенным увеличением физического времени, скажем, до года, следует внимательно следить за характером изменения всех выдаваемых значений, чтобы своевременно обнаруживать опасность переполнения. Другой возможный прием состоит в выдаче промежуточных результатов в середине траектории, что поможет обнаруживать нарушение монотонности их роста.

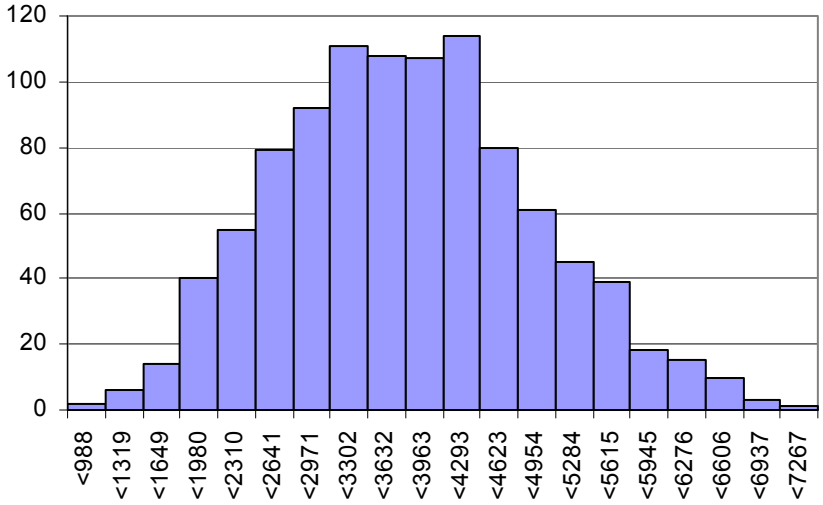
С использованием программы Rivals в интерактивном режиме связаны ограничения, которые обусловлены тем, что в её GUI не предусмотрены возможности управления размером шрифта или самостоятельного планирования рабочей области главного окна. Из-за этого при отображении могут обрезаться длинные имена объектов или очень большие значения счетчиков. На работе модели и содержимом выходных файлов это не сказывается.

В программе также ограничены общее число динамически изменяемых параметров (<= 64) и точек изменения на протяжении траектории (<=16), см. раздел 1.4.2.

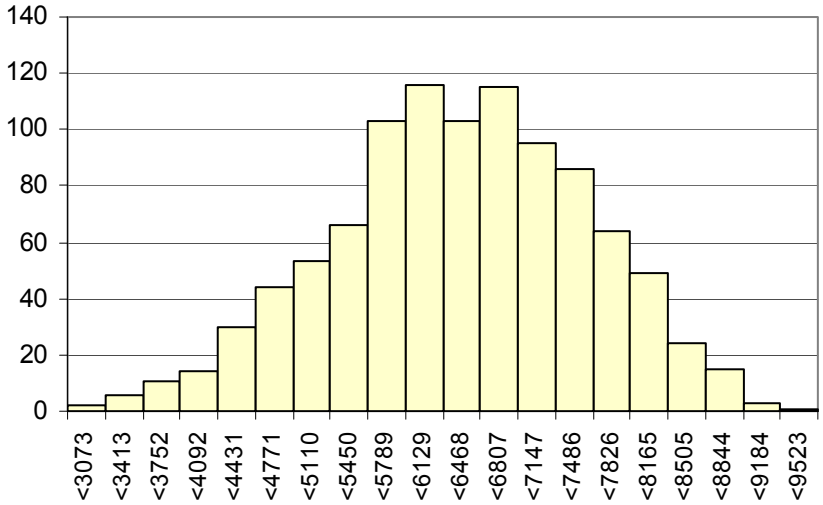
5.4. О представлении неспецифичных PEP-промоторов

Некоторые PEP-промоторы являются неспецифичными, т.е. зависят сразу от нескольких сигма-субъединиц, которые могут связываться с таким промотором. В нашей модели PEP-промотор всегда представляется как специфичный, т.к. для него можно указать только один тип сигма-субъединицы. Прямолинейный способ преодолеть такое ограничение уже указывался в разделе 1.3.3: можно ввести в задачу нужное число промоторов, занимающих одни и те же позиции локуса; каждый из них будет зависеть от своей сигма-субъединицы и иметь свою интенсивностью связывания.

а)



б)



в)

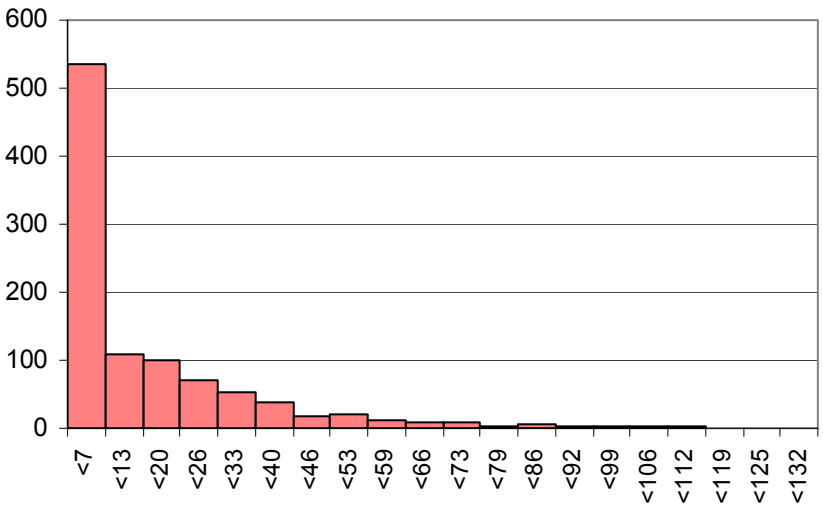


Рис. 27. Гистограммы распределения числа транскрипций генов:
а) *ycf1* (число транскрипций 658 – 7266); б) *ndhF* (2734 – 9522); в) *rpl32* (0 – 131).

С точки зрения скорости моделирования такое решение будет неэффективным (что следует из п. 5.1). *Если сайты посадки таких специфичных промоторов в точности совпадают*, т.е. длина сигма-субъединиц одна и та же, *и позиции старта транскрипции одинаковы*, то возможно более эффективное решение. Оно вытекает из того теоретического факта, что суперпозиция двух и более независимых пуассоновских потоков однородных событий с интенсивностями, соответственно, $\lambda_1, \lambda_2, \ldots, \lambda_k$ есть также пуассоновский поток с интенсивностью $\lambda = \sum_{i=1}^{k} \lambda_i$. Это значит, что при соблюдении отмеченных условий вместо того, чтобы вносить в задачу несколько PEP-промоторов взамен одного неспецифичного, достаточно внести только один PEP-промотор с суммарным значением интенсивности связывания.

Сказанное выше имеет еще одно важное для практики следствие: если раздельные значения интенсивностей λ_i неизвестны, и будет проводиться перебор с целью нахождения точных значений, то этот перебор может быть одномерным – достаточно перебирать только значения λ, поскольку всевозможные значения суммы уже заключают в себе все сочетания значений слагаемых.

Поясним это на реальном примере. Предположим, что в природном типе некоторый PEP-промотор зависит от двух сигма-субъединиц и одновременно с ним рассматривается мутант, в котором выполнен нокаут по одной сигма-субъединице. Желая перебрать всевозможные сочетания значений интенсивности связывания с каждой сигма-субъединицей, например, в интервале [0, 1) с шагом 0.01, мы должны были бы провести $100 \times 100 = 10\,000$ экспериментов над задачей с двумя промоторами. Однако всю необходимую информацию можно получить, варьируя в задаче с одним промотором уже суммарную интенсивность в интервале [0, 2) с тем же шагом 0.01 (менее 200 экспериментов), т.е. получаем выигрыш более чем в 50 раз только для одного промотора!

Рассмотрим, как это скажется на обработке результатов, например, при решении следующей типичной обратной задачи: найти интенсивности λ_1, λ_2, если известно отношение частот транскрипции некоторого гена (или генов) в природном типе и при нокауте по одной сигма-субъединице (для определённости, пусть при нокауте $\lambda_2 = 0$). В первом варианте моделирования надо значение экспрессии в каждой точке перебора, имеющей вид $(\lambda_1, 0)$, сопоставить со значениями экспрессии во всех точках (λ_1, λ_2), т.е. вычислить порядка 10000 отношений, из которых выбрать наилучшее. Во втором варианте требуется вычислить аналогичные отношения по всем парам точек $\langle \lambda_x, \lambda_y \rangle$, таким что $\lambda_x \le \lambda_y$ (первое значение соответствует суммарной интенсивности при нокауте, второе – в природном типе). Поскольку суммарных значений λ вдвое больше, то таких пар будет порядка 20000 (из-за упорядоченности значений), поэтому трудоемкость обработки результатов возрастает вдвое, что вполне приемлемо, учитывая большой выигрыш при моделировании. После выбора лучшей пары $\langle \lambda_x, \lambda_y \rangle$ легко найти искомые раздельные значения интенсивности: $\lambda_1 = \lambda_x,\ \lambda_2 = \lambda_y - \lambda_x$.

5.5. О работе с кольцевой ДНК

Некоторые геномы органелл имеют кольцевую ДНК, которая представлена в файле ГенБанка в разрезанном виде. Описываемая версия программы позволяет обрабатывать целую такую последовательность в *кольцевом режиме*, который можно включить через GUI или задать предложением **Circular** в файле задачи (см. раздел 4.2). Однако во многих случаях

рассматривать всю длинную кольцевую последовательность избыточно, тем более что это сильно замедляет моделирование. Достаточно просто выделить интересующий локус кольца и рассматривать его как линейный (если заметное число РНК-полимераз делают более одного круга, для выделенного локуса это можно пытаться смоделировать с помощью внешних источников полимераз, п.п. 1.3.6, 1.3.9). Работа с такой последовательностью в программе не имеет особенностей, пока исследуемый локус не содержит точку, в которой была разрезана кольцевая ДНК.

Если встречается такая ситуация, то часть генов задачи (включая ее левый край) окажется в конце разрезанной последовательности, а другая часть (в том числе правый край задачи) – в начале последовательности. Для текущей версии программы требуется, чтобы в линейном режиме номер позиции левого края задачи был меньше номера позиции правого края, что оказывается нарушенным. Тем не менее, такую задачу можно моделировать с помощью текущей версии, используя следующий обходной путь. Точная последовательность рекомендуемых действий такова:

1. С помощью программы Rivals создать новую задачу и ввести в нее данные всех генов интересующего локуса обычным образом, как описано в разделе 2.3.3.1, не обращая внимания, что вид последовательности генов в рабочей области главного окна программы отличается от желаемого.
2. Сохранить задачу с нужным именем и выйти из программы.
3. Записать точную длину последовательности l и выбрать удобный «круглый» номер позиции d между двумя фрагментами локуса. Например, при длине последовательности $l \approx 200000$ удобно выбрать $d = 100000$.
4. Открыть сохраненный файл задачи в обычном текстовом редакторе (например, **Notepad/Блокнот)** и по очереди отредактировать все разделы [Gene] в конце файла (см. раздел 4.2). Изменению подвергаются только значения X параметров **Begin/Left** и **End/Right**.
5. Если указанное в файле значение X больше, чем d, заменить его значением $X - d$.
6. Если указанное в файле значение X меньше, чем d, заменить его значением $X + l - d$.
7. Сохранить измененный файл и закрыть текстовый редактор.
8. Снова запустить программу Rivals и загрузить в ней отредактированный файл задачи.
9. Убедиться, что гены локуса представлены в рабочей области в правильном порядке.
10. Добавить в задачу остальные желаемые объекты, как описано в разделах 2.3.3.1, 2.3.2. Во избежание ошибок, при указании местоположения объекта следует использовать только относительную локализацию (задавая смещение от конца или начала некоторого гена).
11. Сохранить задачу и далее использовать ее обычным образом.

5.6. О регистрации типов файлов в Windows

При работе в системе Windows квалифицированный пользователь может для большего удобства зарегистрировать в ней типы используемых программой файлов задач и заданий, так чтобы двойной щелчок левой кнопкой мыши по имени файла инициировал подходящие действия, например:

- для файла задачи – открывал эту задачу в программе Rivals;
- для файла заданий – запускал его обработку программой cpRivals в однопроцессорном режиме.

Например, в системе Windows XP для этого можно взять за основу файл **tskrjb.reg** из комплекта поставки программы. Предварительно следует редактором **Notepad/Блокнот** исправить имеющиеся этом файле пути, так чтобы они указывали на папку с программами (файл составлен в предположении, что программы помещены в папку **C:\WinApp**). В этой же папке должен лежать файл конфигурации **rivals.ini**.

Примечание**:** Данная процедура предполагает, что расширение имени файла **.tsk** ранее не было зарегистрировано в системе. Следует убедиться в этом с помощью программы **Explorer/Проводник** (**Tools > Folder Options > File Types**) или **regedit**, и внести необходимые изменения, если такой тип файла уже зарегистрирован (например, удалить или переименовать ранее зарегистрированный тип файла **TSK**).

Затем можно двойным щелчком мыши запустить файл **tskrjb.reg**, подтвердив намерение добавить информацию из файла в реестр Windows.

После этого, как только пользователь в первый раз дважды щелкнет мышью по файлу задачи или файлу заданий, будет предложено выбрать программу – соответствующая программа (rivals.exe или cprivals.exe) будет стоять первой в списке известных программ для данного типа файла. Необходимо выделить ее и отметить пункт **Always use the selected program to open this kind of file**, после чего нажать кнопку **OK**. В дальнейшем такой запрос появляться не будет, а система сразу выполнит необходимое действие.

Примечания:

1. Для такого использования файла заданий в нём предложением **Task** обязательно должен быть указан файл задачи.
2. Пользователь лишается возможности указывать параметры командной строки для модификации параметров, указанных в файле заданий.
3. Текущей папкой при запуске программы будет не папка с программой, а папка с файлом, который запускался двойным щелчком. В частности, по умолчанию в этой папке будет искаться файл задачи и в нее будут записаны выходные файлы.

Для отмены сделанных изменений в реестре Windows используется файл **~tskrjb.reg**.

5.7. Об использовании программы Rivals в Linux

Если работа происходит под управлением Linux на платформе с архитектурой x86, то в этой среде с помощью пакета **wine** также можно использовать исполняемый модуль rivals.exe (с дружественным графическим интерфейсом), хотя он изначально откомпилирован для Windows. Точная последовательность действий зависит от дистрибутива и версии Linux. В качестве примера укажем один из успешно нами опробованных сценариев.

ОС Mandriva Linux Free 2009.1 с графической оболочкой KDE4 и пакетом wine 1.1.32.

Установить в системе пакет **wine** и затем настроить его с помощью входящего в состав пакета скрипта **winetricks**, заказав установку следующих дополнительных пакетов, которые эмулируют необходимые программе компоненты Windows:

- **dotnet20** (MS .NET 2.0 framework)
- **vcrun2008** (библиотеки MS Visual C++ 2008 sp1)
- **corefonts** (встроенные шрифты Windows)

После этого практически все функции программы действуют, как описано в главе 2, а скорость работы уменьшается примерно на 10% по сравнению с работой программы в среде Windows XP на том же компьютере.

6. Литература

1. Lyubetsky V.A., Zverkov O.A., Rubanov L.I., Seliverstov A.V. Modeling RNA polymerase competition: the effect of sigma-subunit knockout and heat shock on gene transcription level. *Biology Direct* 2011, 6:3. http://www.biology-direct.com/content/6/1/3
2. Lyubetsky V.A., Zverkov O.A., Pirogov S.A., Rubanov L.I., Seliverstov A.V. Modeling RNA polymerase interaction in mitochondria of chordates // *Biology Direct*, 2012, 7:26. http://www.biology-direct.com/content/7/1/26