**Слайд 0 – Титульная страница**

**Слайд 1 – Поиск гомологий в биологических последовательностях**

В генетике под гомологиями понимаются участки белков или ДНК, имеющие сходную последовательность аминокислот или нуклеотидов. Обычно существа, у которых есть гомологичные участки белков или ДНК, имеют общего предка, от которого они и получили такой участок. Поскольку в процессе эволюции ДНК подвергается мутациям, эти участки не обязательно идентичны. В них могут быть случайно заменены, добавлены или удалены нуклеотиды или аминокислоты.

**Слайд 2 – Задача выравнивания**

Выравнивание аминокислотных или нуклеотидных последовательностей - это процесс сопоставления сравниваемых последовательностей для такого их взаиморасположения, при котором наблюдается максимальное количество совпадений аминокислотных остатков или нуклеотидов.

На слайде представлены две последовательности ДНК: ДНК предка и его потомка. Красным цветом обозначены замены, оранжевым – инсерции и фиолетовым – делеции нуклеотидов.

**Слайд 3 – Задача выравнивания**

Различают два вида выравнивания: парное (выравнивание двух последовательностей ДНК, РНК или белков) и множественное (выравнивание трех и более последовательностей).

Качество выравнивания оценивают, назначая штрафы за несовпадение букв и за наличие пробелов (когда приходится раздвигать одну последовательность для того, чтобы получить наибольшее число совпадающих позиций), например, через расстояние Левенштейна --- минимальное число элементарных операций (вставка, удаление или замена символа в строке), чтобы превратить одну строку в другую. При сравнении ищется такой вариант выравнивания, чтобы итоговый счет был максимален. В такой постановке задача называется поиском <<**глобального выравнивания**>>. Необходимо отметить, что для полных геномов глобальное выравнивание не работает, так как при мутации, помимо вставок, удалений и замен, бывают нелинейные перестройки, которые могут менять порядок и ориентацию целых геномных блоков. Для решения, аналогично задаче поиска глобального выравнивания, формулируют задачу поиска <<**локального выравнивания**>>: для двух произвольных строк A и B найти две самые похожие подстроки и их выравнивание.

Алгоритмы множественного выравнивания, аналогично алгоритмам парного выравнивания, представляют собой инструмент для установления функциональных, структурных или эволюционных взаимосвязей между биологическими последовательностями. Несмотря на то, что задача множественного выравнивания была сформулирована более 20 лет назад, она до сих пор не теряет своей актуальности. Если говорить о множественном глобальном выравнивании, то, по сравнению с парным выравниванием, практически ничего не меняется: необходимо расставить разрывы в выравниваемых строках таким образом, чтобы <<счет по столбцам>> был максимален. Счет по столбцу можно считать перебирая все пары символов. Множественное локальное выравнивание обобщить на многомерный случай не так просто. Во-первых, какие-то подстроки могут быть не во всех последовательностях. Во-вторых, последовательности могут содержать дуплицированные участки. Поэтому для решения такой задачи **необходимо более точно сформулировать условия выравнивания**.

**Слайд 4 – Классические методы поиска гомологий**

Таким образом, две главные составляющие автоматических методов выравнивания --- это непосредственно алгоритм и функция оценки качества полученного результата. На сегодняшний день можно выделить два основных алгоритма выравнивания биологических последовательностей: алгоритм Нидлмана-Вунша и алгоритм Смита-Ватермана. Они представляют собой классический пример задачи динамического программирования.

**Слайд 5 – Алгоритм Нидлмана-Вунша**

Для своей работы алгоритм использует матрицу сходства, которая указывает, насколько схожими можно считать разные нуклеотиды. Использование матрицы позволяет придавать разный вес разным заменам нуклеотидов. Обычно используется симметричная матрица, однако, применение несимметричной матрицы позволяет различать замены в одну и в другую стороны.

Еще один параметр алгоритма --- штраф за разрыв последовательности. Он может выражаться произвольной функцией от длины и/или направления разрыва. Для определенности будем рассматривать линейный штраф за разрыв, определяющийся параметром d (за разрыв длинны n будет начислен штраф d\*n).

**Слайд 6 – Алгоритм Смита-Ватермана**

Алгоритм Смита-Ватермана аналогичен алгоритму Нидлмана-Вунша, но решает задачу локального выравнивания: находит подстроки первой и второй строк, обладающие максимальным сходством.

**Слайд 7 –Множественное выравнивание. Выравнивание в кубе**

Можно заметить, что каждая грань куба --- это парное выравнивание двух последовательностей с учетом некоторой части третьей, что и дает в итоге полный перебор всех возможных вариантов. Нулевые грани куба F[0,j,k], F[i,0,k] и F[i,j,0] заполняются аналогично алгоритму Нидлмана-Вунша.

**Слайд 8 – Выравнивание выравниваний**

Другой подход заключается в получении парного выравнивания между первыми двумя последовательностями, после чего полученный результат выравнивается с третьей и так далее. То есть, если f --- функция вычисления парного выравнивания, а A1, … ,An --- выравниваемые последовательности, то алгоритм можно условно записать формулой (рисунок на слайде)

Очевидно, что результат алгоритма будет зависеть от порядка исходных последовательностей. Существуют различные соображения по поводу наиболее правильного выбора этого порядка. Можно не ограничиваться выравниваниями типа <<последовательность против выравнивания>>, но также производить выравнивание <<выравнивание против выравнивания>>. Например, если есть четыре последовательности, из которых первая очень похожа на четвертую, вторая --- на третью, а гомология между остальными парами (1-2, 1-3, 2-4, 3-4) более слабая, то разумно сначала сделать два парных выравнивания: первой последовательности с четвертой и второй с третей, а затем уже выровнять эти два выравнивания друг с другом.

Похожим образом работает Clustal --- один из самых популярных алгоритмов множественного выравнивания. По сути, это жадный алгоритм с <<умным>> способом выбора пар. Сначала происходит построение всех парных выравниваний, после чего по полученным результатам строится <<дерево-подсказка>>. На слайде пример возможного дерева. Для четырех последовательностей A1, A2, A3 и A4 строится таблица (на рисунке слева), числа в которой обозначают их схожесть друг с другом. Видно, что самые близкие последовательности --- A1 и A3, и их выравнивание будет первым, затем оно выравнивается с A4, а последнее --- с A2.

**Слайд 9 – Открытые раки считывания**

Изменению числа нуклеотидных пар в цепи ДНК способствуют воздействия на генетический материал некоторых химических веществ. Деформируя структуру двойной спирали ДНК, они приводят к вставке дополнительных оснований или их выпадению при репликации. Однако, куда более вероятно возникновение ошибки на этапе секвенирования.

Рассмотренные ранее алгоритмы множественного и парного выравниваний применимы для любых, не обязательно биологических, последовательностей, например, текстов статей или исходных кодов программ на предмет поиска плагиата. Изложенные выше методы подходят к задаче выравнивания исключительно на математическом уровне, в том плане, что они производят поиск выравнивания с максимальным счетом, совершенно не опираясь на логический смысл входных данных. Возвращаясь непосредственно к задачам биоинформатики, для поиска <<правильного>> выравнивания последовательностей необходимо использовать более сложные алгоритмы, учитывающие открытые рамки считывания.

Один из самых простых способов построения <<правильного>> выравнивания --- произвести трансляцию нуклеотидной последовательности в аминокислотную по всем возможным рамкам считывания, после чего произвести выравнивание <<классическими>> алгоритмами, и, в завершение, транслировать полученный белок обратно в последовательность нуклеотидов. Основным недостатком этого трехступенчатого подхода является его неспособность справляться с неожиданной заменой рамки считывания. Все последующие этапы алгоритма после неправильной первой трансляции уже никак не смогут это исправить. В лучшем случае, этот ошибочный перевод быстро приведет к появлению стоп-кодона, который будет сигналом для предупреждения пользователя о неправильной трансляции. В худшем случае, программа построит выравнивание, которое будет очень сильно расходиться с действительностью.

В 1994 году был предложен еще один подход для решения этой задачи. Автором была предложена модель, по которой штраф за выравнивание являлся сочетанием двух штрафов: на аминокислотном и нуклеотидном уровнях. Он рассмотрел частный случай идеализированной эволюции исходных последовательностей, при котором инсерции допустимы только на аминокислотном уровне (запрет на сдвиг рамки считывания), а штраф за выравнивание вычислялся просто как сумма штрафов на обоих уровнях. Предложенный алгоритм выравнивания двух последовательностей длины n и m имел сложность O(n^2m^2). Позже этот алгоритм был оптимизирован и перенесен на модель с аффинными штрафами и итоговой сложностью O(nm). Эти улучшения казались многообещающими, так как асимптотическая сложность алгоритма получилась точно такая же, как и у классических методов выравнивания. Однако, необходимо отметить, что постоянный множитель, спрятанный в оценке сложности O, ограничивает применение этого алгоритма на практике. Для получения парного выравнивания алгоритму необходимо вычислить примерно 400nm значений, что, к сожалению, делает его неприменимым для задачи множественного выравнивания.

MACSE (Multiple Alignment of Coding SEquences Accounting for Frameshifts and Stop Codons) --- это программа для множественного выравнивания кодирующих последовательностей с учетом существующих рамок считывания и стоп-кодонв. Кроме задачи выравнивания, она может быть применена для обнаружения недокументированных рамок считывания в публичных базах данных.

Алгоритм MACSE основывается на идее двухуровневого выравнивания, но требует меньше времени на вычисление парного выравнивания, благодаря чему возможно его расширение на многомерный случай. Для получения многомерного выравнивания n строк S\_1 … S\_n MACSE производит выравнивание выравниваний, выбирая порядок через дерево-подсказку, как и алгоритм Clustal.