ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ И НАУКИ
САНКТ-ПЕТЕРБУРГСКИЙ АКАДЕМИЧЕСКИЙ УНИВЕРСИТЕТ —
НАУЧНО-ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР НАНОТЕХНОЛОГИЙ
РОССИЙСКОЙ АКАДЕМИИ НАУК

На правах рукописи

Диссертация допущена к защите
Зав. кафедрой
Омельченко А.В.

_____
”__” _____ 2014 г.

Д И С С Е Р Т А Ц И Я
НА СОИСКАНИЕ УЧЕНОЙ СТЕПЕНИ
М А Г И С Т Р А

Тема: Аннотирование и гуманизация последовательностей вариабельных доменов иммуноглобулинов

Направление: 010900.68 – Прикладные математика и физика

Выполнил студент П.А. Яковлев

(подпись)

Руководитель А.В. Карабельский
к.б.н., доцент

(подпись)

Рецензент Ю.Б. Порозов
к.м.н., доцент

(подпись)

Санкт-Петербург
2014 г.

Реферат

С. 39, рис. 13, табл. 3.

В данной работе представлены алгоритмы для аннотации последовательностей вариабельных доменов тяжелых и легких цепей иммуноглобулинов. Рассмотренные методы позволяют быстро и точно определять структурные семейства рассматриваемых последовательностей, а также находить ближайшие исходные образующие гены и ближайшие гомологи в любых используемых базах данных. Помимо этого рассматриваются методы получения более детальной аннотации для каждой позиции последовательности. Примером таких аннотаций в данной работе являются принадлежность каждой позиции к одному из семи регионов вариабельного домена, а также поиск вариантов замен для проведения гуманизации таких последовательностей.

Рассмотренный метод использует особенности последовательностей вариабельных доменов иммуноглобулинов и, в отличие от других методов поиска по гомологии, не требует множества попарных выравниваний со всеми элементами используемой референсной базы. Это обеспечивает более высокую скорость работы и уменьшенное потребление памяти при сохранении точности, сопоставимой и часто превосходящей имеющиеся аналоги.

Ключевые слова: иммуноглобулины, структура белка, префиксное дерево, выравнивание последовательностей

ОГЛАВЛЕНИЕ

Введение

Основой для разработки большинства белковых препаратов для терапии аутоиммунных и онкологических заболеваний являются вариабельные домены иммуноглобулинов [29]. Изучение и дизайн подобных элементов включает в себя поиск исходных кодирующих генов, определение структурных семейств и регионов, поиск сайтов иммуногенности и посттрансляционных модификаций, а также получение трехмерных структур этих доменов и вариантов их оптимизации с целью увеличения аффинности и специфичности к рассматриваемой мишени. Все указанные операции базируются на аннотации последовательностей вариабельных доменов иммуноглобулинов, то есть описании их свойств на основании использования референсных данных [16]. Аннотация в данном случае происходит как всей последовательности в целом, так и ее составных частей, вплоть до каждой нуклеотидной или аминокислотной позиции.

Эта задача осложняется необходимостью сравнения с огромными референсными базами, что приводит к значительному времени работы и потреблению памяти. Существующие решения стараются избегать этой проблемы путем снижения точности предсказания, а также уменьшением рассматриваемой области на вариабельном домене [16; 23; 27]. Также эти решения созданы для работы на заведомо высококачественных данных, а потому неприменимы для обработки неисправленных результатов секвенирования и данных с ошибками вообще.

В данной работе представлен метод, решающий задачу аннотации вариабельных доменов иммуноглобулинов без применения эвристик, снижающих качество получаемых результатов. Он также применим для работы на данных любого качества, что позволяет использовать его для очень широкого круга задач: от проверки качества результатов секвенирвоания до высокоточного поиска регионов и вариантов замен для гуманизации кандидатов.

Метод основан на использовании префиксного дерева [11], что позволяет значительно сжимать референсную базу. Также предложены алгоритмы поиска и выравнивания с использованием подобной структуры, являющие-

ся максимально эффективными по времени работы и затрачиваемой памяти. Данный метод с сопутствующими алгоритмами реализованы на языке Scala в качестве библиотеки под лицензией BSD, что позволяет использовать ее для любых проектов.

Диссертация состоит из трех глав. В первой главе приводится историческая справка по изучению и применению терапевтических антител, а также дается постановка задачи и обзор существующих методов ее решения. Во второй главе приводится описание метода и детально рассматриваются алгоритмы, лежащие в его основе. Третья глава содержит экспериментальные результаты: выбор оптимального количества референсов для точного поиска регионов на антителах, а также сравнение с существующими методами решения данной задачи.

1. Постановка задачи и обзор существующих методов

1.1 Краткая историческая справка

Еще в начале XX века Нобелевкий лауреат по физиологии и медицине Пауль Эрлих представил концепцию “magic bullet” [29]. Эрлих считал, что если будет найден компонент, способный селективно связывать мишень в организме, то с этим компонентом к патогенам можно было бы доставлять токсин. Такая терапия была бы гораздо эффективнее и безопаснее, чем разработанная им же химиотерапия.

Белки, обладающие такой селектирующей функцией, были известны еще с 1890 года благодаря работам Беринга и Кинзато по изучению сыворотки крови животных. Основываясь на предположении, что эти белки являются антагонистами патогенам, они были названы *антителами*, однако их природа, механизмы появления и образования специфичности были неизвестны. Только в 1937 году Тиселиус и Кабат начали изучение молекулярной природы антител, получивших второе название – *иммуноглобулины*, в соответствии с классификацией их белковой природы.

К 1970-ым была установлена связь между B-лимфоцитами и выработкой антител. Тогда же были получены гибридомы – бессмертные B-клетки, вырабатывающие только один тип антител, получивших название моноклональных. Жорж Келлер и Сезар Мильштейн разработали метод, позволивший вырабатывать любое антитело как моноклональное. Это исследование принесло им Нобелевскую премию, а также начало эпоху терапевтических антител.

Методика получения антител, специфичных к конкретной мишени, состояла в иммунизации животных (то есть введения им патогена) и получения в их организме иммунного ответа. После этого у животного бралась кровь, из нее выделялась фракция B-клеток, которые уже по методу Келлера и Мальштейна преобразовывались в формат для выработки моноклональных антител. Первый коммерческий препарат был получен из крови мышей уже в середине 1980-ых годов, но тогда же были обнаружены сложности с данным видом терапии.

Во-первых, многие животные быстро погибали, не успевая выработать требуемые антитела. Во-вторых, помимо требуемых антител в крови живот-

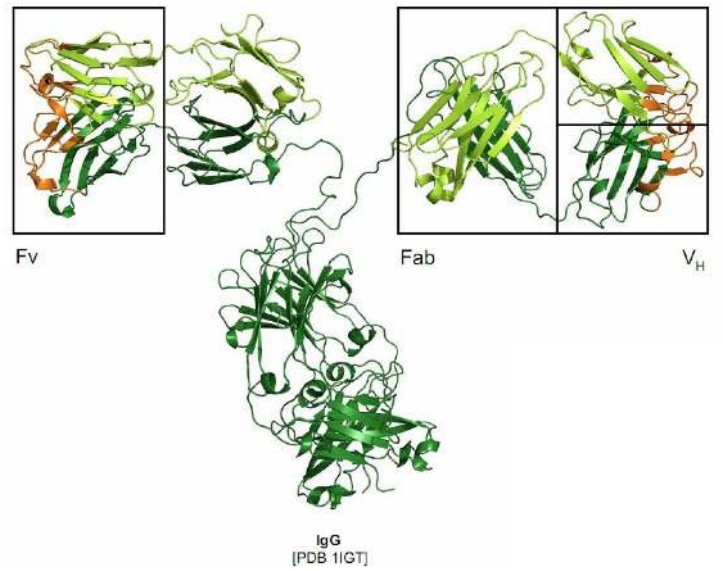


Рисунок 1.1 — Структура антитела типа IgG. Fc - кристаллический фрагмент; Fab - антиген-связывающий фрагмент; Fv - вариабельная часть антиген-связывающего фрагмента

ных всегда содержались антитела и на другие мишени, а значит, после забора крови должен был наступать этап селекции, по сей день являющийся одной из самых длительных и затратных операций в разработке антительных препаратов. Главная же проблема заключалась в том, что человеческий организм часто реагировал на антитела, полученные из мышиных гибридом, вырабатывая на них собственный иммунный ответ вплоть до анафилактического шока, что снижало терапевтическую эффективность препаратов.

Первое решение этой проблемы было предложено в [24] и было названо химеризацией антител. Ричманном и Винтером было определено, что наибольшую иммуногенность представляет кристаллический фрагмент (см. рис. 1.1), при этом за специфичность антитела отвечают только два вариабельных домена тяжелой и легкой цепи, лежащие на краю антиген-связывающего фрагмента. Был создали метод, позволяющий путем генной инженерии пришивать к Fc фрагменту человеческого антитела Fab фрагменты антитела иммунизированного животного, что сильно снижало иммуногенность такой рекомбинантной молекулы.

Такие антитела так же обладали некоторой иммуногенностью, для снижения которой без потери позитивных свойств антитела потребовалось уже более глубокое изучение вариабельного фрагмента (см. рис. 1.2).

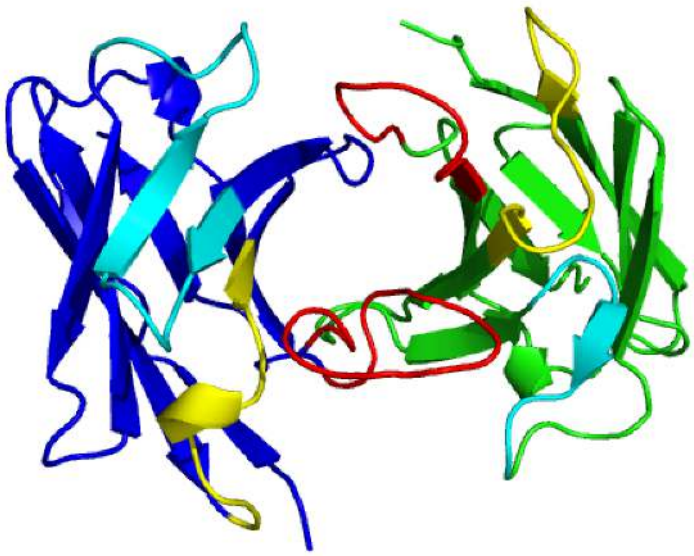


Рисунок 1.2 — Структура Fv: домены тяжелой (синяя) и легкой (зеленая) цепи

Вариабельный фрагмент наиболее изученных антител – класса IgG – обычно состоит из двух доменов: тяжелой цепи (VH) и легкой цепи (VL), каждый из которых состоит из 100 – 150 аминокислот. Открытие комбинаторного образования генов, кодирующих эти домены [20], позволило подойти к задаче их исследования с более системным, информатическим подходом.

Одной из главных особенностей антител является процесс их образования с помощью V(D)J рекомбинации, схема которой изображена на рис. 1.3. Сборку антитела можно представить как выбор по одному гену *гермлайну* из трех “кассет” в случае тяжелой цепи или двух в случае легкой. После этого на стыках этих генов может произойти добавление и удаление произвольного количества нуклеотидов. Финальной стадией является соматический мутагенез, когда в любой точке образовавшейся конструкции может произойти мутация: как замена нуклеотида (чаще), так и вставка и удаление нуклеотидов (реже).

Такой механизм позволяет иммунной системе создавать огромное количество разнообразных структур способных обеспечить специфичность к любому антигену. Так, например, у человека может образовываться $51VH \times 75DH \times 6JH = 22950$ тяжелых цепей без учета соматических мутаций.

В работах лаборатории Цайруса Хотии были описаны [19] структурные семейства вариабельных доменов антител, а в [32] были представлены аминокислотные последовательности гермлайнов человека, а также варианты ге-

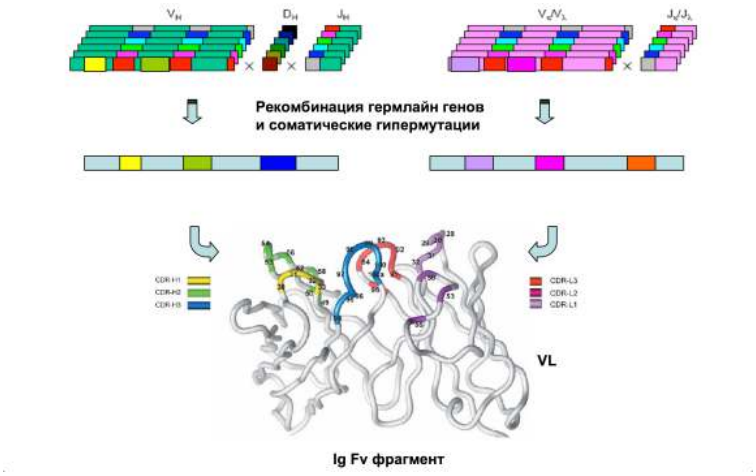


Рисунок 1.3 — Образование тяжелой и легкой цепи в ходе V(D)J рекомбинации

нов, их кодирующих. Все антитела можно было разделить на группы по семействам и исходным гермлайнам, а зная их, стало возможно делать предварительные выводы об их структурных и функциональных свойствах. Появление этих работ определило необходимость создания новых инструментов для анализа последовательностей и структур вариабельных доменов антиген-связывающих фрагментов антител.

1.2 Задача поиска CDR и FR регионов

Вторичная структура вариабельного домена антитела представляет протяженную β-складку [22], состоящую из 4 листов, перемежающихся тремя петлями, ориентированными наружу. Такую структуру принято делить на, соответственно, *framework* регионы (FR), поддерживающие пространственную структуру вариабельного домена, и *complementarity determining* регионы (CDR), участвующие непосредственно в распознавании и связывании с антигеном (см. рис. 1.4).

Сложность состоит в том, что определить область контакта абсолютно точно можно лишь при рассмотрении кристаллической структуры комплекса антиген-антитело. Получение таких структур методами кристаллографии весьма сложно, а их моделирование либо сильно проигрывает по точности, либо требует для проведения знания границ регионов. В связи с этим многие лаборатории определили различные варианты разделения первичной структуры вариабельного домена на структурные регионы [17; 25]. Обычно такие

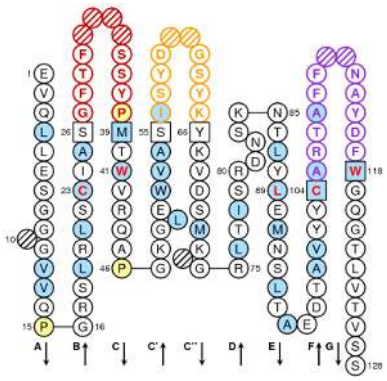


Рисунок 1.4 — Вторичная структура VH домена в номенклатуре IMGT [17]

схемы или *номенклатуры* вводятся на гермлайнах, а далее предсказываются для зрелых антител. Это определило необходимость в создании методов быстрого определения семейства и поиска гермлайнов для любой последовательности вариабельного домена, а так же разметки ее на регионы в любой возможной номенклатуре.

1.3 Гуманизация вариабельных доменов

После определения свойств структурных регионов вариабельных доменов начался дальнейший процесс по снижению имуногенности рекомбинантных моноклональных антител. Предположение о том, что FR регионы совсем никак не влияют на свойства антитела, определило создание методики *CDR grafting* [1] – пересадки CDR иммунных животных в наиболее гомологичные человеческие каркасы. Однако такие структуры чаще всего получались нестабильными и сильно теряли свои свойства ввиду изменения относительного расположения петель и их взаимодействия с каркасными регионами. В связи с этим выработался новый подход к гуманизации антител – осуществление минимального числа замен в каркасных регионах для сильного снижения иммуногенности антител.

С середины 1990-ых вышло множество статей (e.g. [13; 14]) и патентов (e.g. [3; 9; 15]) описывающих методы точечной гуманизации антител. В основном в них описывались экспериментальные подходы к осуществлению замен, но все они требовали механизма получения статистической информации и возможных замен каждой аминокислотной позиции как в как в антителах организма-источника, так и в антителах человека.

Для получения такой информации потребовалось создание инструментов, способных работать с огромными гермлайными, наивными и иммунными библиотеками антител, выделяя из ближайших гомологов необходимую информацию.

1.4 Существующие методы

Целью данной работы является разработка алгоритмов и создание инструмента для исследования последовательностей вариабельных доменов антител для их детальной аннотации и гуманизации. Для этого требуется решать следующие задачи для нуклеотидных и аминокислотных последовательностей Fv:

1. определение ближайших генов гермлайнов и семейства антитела;
2. поиск регионов на последовательности, содержащей вариабельный домен или его часть;
3. точечная гуманизация антител путем внесения замен в FR регионы.

Так как моноклональные антитела с момента открытия являются предметом повышенного интереса фармацевтических компаний, решения подобных задач в большей мере производилось именно их сотрудниками или финансируемыми академическими учреждениями. Наибольший интерес для таких организаций представляют методы гуманизации последовательностей. В связи с этим невозможно найти какие-либо инструменты, реализующие хоть один алгоритм гуманизации, в открытом доступе. Алгоритмы поиска ближайших гермлайнов и регионов являются составной частью крупных комплексных решений и не существуют как самостоятельные утилиты или библиотеки, которые было бы возможно внедрить в свои пайплайны или инструменты.

Пытаясь достигнуть максимальной скорости, многие инструменты [23; 26; 27] используют различные методы нечеткого поиска паттернов границ регионов. При этом такие паттерны могут быть извлечены как на основании эмпирических наблюдений, так и с помощью методов машинного обучения, запущенных на достаточно большой выборке. Так, сервис ROSIE использует поиск одиночных аминокислот в конкретных позициях: *Cys* в L22, L92 и

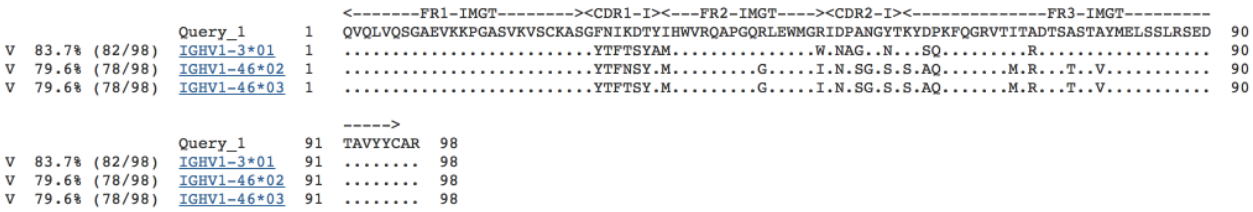


Рисунок 1.5 — Пример вывода инструмента IgBLAST [16]

H22, H92; *Trp* в L35, H36 и H103 [26]. Утилита proABC использует метод случайного леса [7] для поиска границ регионов, основываясь на обширной референсной базе, что позволяет достичь им точности и полноты в 80% [23].

Этот метод позволяет находить только внутренние границы, то есть только границы CDR, и не работает на данных с ошибками. Помимо ошибок, нарушение паттернов может происходить и в естественной природе в связи с гипервариабельной природой Fv. Таким образом, эти инструменты не работают в принципе на достаточно большом классе антител, но дают достаточно точные результаты в случае, если последовательность не содержит ошибок и имеет не слишком большое различие с последовательностями, на основании которых строились правила.

Более точным методом является аннотирование, основанное на гомологии. Этот же метод, помимо разметки на регионы, позволяет получить варианты замен в каждой позиции, а также определить семейство и вид, к которому относится анализируемый иммуноглобулин. Для аннотирования последовательностей подобным методом используется размеченная референсная база последовательностей, в которой про каждую позицию или их группу записана необходимая информация для аннотирования. Аннотирование происходит в два этапа: выбор ближайших референсов и выравнивание на эти референсы. Для выравнивания обычно используется метод [28], позволяющий получить наиболее похожие области между последовательностью-запросом и референсом. Этот метод не так сильно подвержен ошибкам из-за большого различия в последовательностях, а также дает более точные результаты, хоть и сильно проигрывает первому методу по скорости работы.

В связи с большей точностью и широкой применимостью, метод гомологичного аннотирования используется как в академических инструментах анализа последовательностей иммуноглобулинов [8; 16; 33], так и в современ-

```
...LAISGLQSEDEADYHCMGSGIAVF...
...LTIKNIQEEDESDYYC-GSGIVVF...
...---FR3---------> <--CDR3...
```

Рисунок 1.6 — Пример потери аннотации аминокислоты в граничной позиции при использовании одного референса

ных инструментах предсказания структур, где требуется высокая точность соответствия размеченных границ принятым в системе моделирования [6; 18]. Пример вывода таких инструментов приведен на рис. 1.5. Несмотря на широкую распространенность данного метода, существующие решения по-прежнему обладают большим количеством проблем и недостатков.

В первую очередь, с ростом референсной базы стремительно растет вычислительное время, требующееся для сравнения со всеми референсами. Решения, использующие в своей основе BLAST-алгоритм [5], избегают перебирания всех последовательностей, используя эвристику, построенную на поиске общих k-меров. Однако гипервариабельная природа Fv часто делает невозможным ее применение, в связи с чем даже IgBLAST – специальная версия BLAST для антител [16] – исключает эвристический поиск для основной гермлайной базы данных, оставляя его только для дополнительных подключаемых баз.

Из-за невозможности быстро обрабатывать большие референсные базы, многие инструменты, построенные на таком принципе, рассматривают только V-ген, что влечет за собой невозможность нахождения CDR3 региона, или ищут все гены раздельно, что может привести к проблеме неправильного картирования гена D на последовательности. Это связано с тем, что размер D-гена может оказаться меньше CDR1 или CDR2, а в силу вариабельности последних возможна ситуация, когда какой-нибудь из генов выровняется с высоким score. Здесь же проявляется проблема множественного предсказания генов. Если каждый из трех (V, D, J) генов тяжелой цепи имеет по 3 предсказания, то общее число вариантов, которое требуется рассмотреть человеку, работающему с системой – 27, тогда как при оценке выравниваний на комбинаторные референсы, точность получаемых результатов возрастает, а их количество уменьшается.

Следующей проблемой является использование имеющимися инструментами только одного ближайшего референса для аннотирования. Это может вызвать проблему потери аннотации для граничной аминокислоты (см. рис. 1.6). Для решения этой проблемы требуется выравнивание сразу нескольких комбинаторных гермлайновых референсов и получение аннотации, используя их все, а не только ближайший.

Последнее важное препятствие в использовании имеющихся инструментов состоит в том, что они являются web-сервисами [8; 16; 26], предоставляющими удобный интерфейс для аннотирования одиночных последовательностей, но абсолютно неприменимыми для обработки больших данных, появляющихся, например, после высокопроизводительного секвенирования. Еще одна проблема инструментов как сервиса – это потеря конфиденциальности передаваемых сторонней организации данных, являющаяся критичным минусом для фармацевтической отрасли.

Невозможность использования ни одного из существующих решений в высокопроизводительных пайплайнах и оригинальных протоколах гуманизации потребовала разработки собственного метода для решения обозначенных задач. В результате выполнения данной работы были разработаны и реализованы алгоритмы, позволяющие выполнять быструю и высокоточную обработку последовательностей вариабельных доменов иммуноглобулинов.

2. Разработанные методы

2.1 Общий обзор методов и алгоритмов

Для решения поставленных задач был выбран метод аннотирования по гомологии. Поиск гомологов и аннотаций потребовал создание эффективной структуры данных, способной хранить большое количество референсов с их аннотациями. Чтобы обеспечить максимальную компрессию хранимых данных, были рассмотренны особенности последовательностей вариабельных доменов иммуноглобулинов.

При проведении исследований чаще всего рассматриваются антитела, специфичные к одной мишени. Такие антитела имеют, как правило, один общий V-ген-предок. Последовательности вариабельных доменов этих антител часто обладают достаточно длинным общим префиксом, поскольку большинство соматических мутаций происходит на границе генов (V, D, J или V, J), в самом же V-гене их гораздо меньше [20]. В случае, когда про рассматриваемые антитела ничего не известно, обычно обращаются к гермлайновым базам ближайшего родственного организма, как к универсальному референсу. Нормальной практикой также является использование гермлайнов человека, так как большинство рассматриваемых животных имеют с ними гомологию свыше 70%. V-гены гермлайнов разделены на структурные семейства, внутри которых так же наблюдается наличие длинных общих фрагментов. Например, 5 из 51 человеческих VH гермлайнов имеют общий префикс *QVQLVQSGAEVKKPGASVKVSCKASGYTFT* [32].

Такие наблюдения подсказали выбрать в качестве структуры для хранения последовательностей префиксное дерево или *бор* [11]. Эта структура часто используется в поисковых автоматах, когда требуется работать одновременно с большим количеством строк. В частности, широко применимы модификации алгоритма [4], использующиеся для составления черных или белых списков слов, а так же поиска заранее определенных паттернов в тексте. Считается, что бор эффективен для обширного списка хранимых строк даже в случае отсутствия между ними какой-либо определенной корреляции. В следствии приведенных выше соображений, при работе с иммуноглобулинами эффективность его использования очень высока.

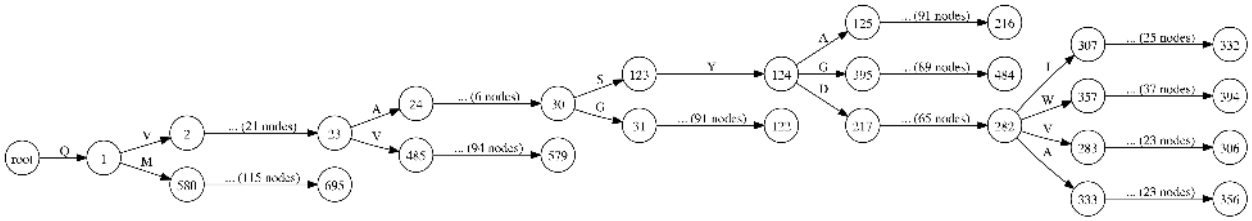


Рисунок 2.1 — Префиксное дерево, построенное по 10 иммуноглобулинам (сжатый вид)

В дополнение к префиксному дереву используется хэш-таблица, хранящая для каждого k-мера каждого референса множество вершин, в которых он заканчивается. Подобная вспомогательная структура позволяет осуществлять быстрый поиск, а также выбирать некоторое поддерево, представляющее уменьшенную референсную выборку для последующих аннотации или гуманизации.

Алгоритмы поиска ближайших гермлайнов, аннотирования и поиска вариантов замен для гуманизации осуществляются с помощью выравниваний [12; 21; 28], модифицированных для работы на структуре префиксного дерева. Сами аннотации не требуются для работы алгоритмов, а потому могут подгружаться только по мере необходимости индивидуально для каждого референса. Это позволяет сократить потребление памяти и работать с огромными референсными базами, используя из них только те последовательности, которые реально требуются для аннотирования последовательностей-запросов.

2.2 Детальное описание алгоритмов

2.2.1 Построение префиксного дерева и индекса k-меров

Рис. 2.1 показывает префиксное дерево, построенное по 10 случайно выбранным последовательностям иммуноглобулинов. Узлы дерева помечены числовыми индексами, а ребра промаркированы буквами последовательности, в данном случае, аминокислотами. Любой путь от корня к листу дерева задает один референс, таким образом, используя любой обход этого дерева, можно перебрать все референсы. Предпочтительным в этом случае является DFS – поиск в глубину, поскольку он позволяет сделать это наиболее удоб-

ным для обработки способом с пространственной сложностью $O(max_{r \in R}|r|)$, где R – множество всех референсов, хранящийся в дереве.

Построение такого дерева – online алгоритм, таким образом возможно добавлять референсы по мере их поступления, причем на работу всех описываемых далее алгоритмов над данной структурой не повлияет добавление новых последовательностей. Алгоритм добавления новых последовательностей в дерево может быть описан следующим псевдокодом:

function $add\_sequence(sequence)$
  $counter \leftarrow$ reference to a global counter
  $node \leftarrow root\_node$
  for $symbol$ in $sequence$ **do**
    $node \leftarrow add\_symbol(node, symbol, counter)$
  end for
end function

При этом добавление каждого символа последовательности происходит в соответствии со следующей процедурой:

function $add\_symbol(node, symbol, counter)$
  if $node$ has no outgoing edge labeled with $symbol$ **then**
    $new\_node \leftarrow create(label = counter, inedge = symbol)$
    Add child new_node to $node$
    Increment $counter$
  end if
  return Get child of $node$ by $symbol$ edge
end function

Все узлы непрерывно нумеруются с помощью счетчика. Учитывая, что удаление из дерева не допускается, ссылки на узлы можно располагать в линейной памяти. Такой индексирующий массив наращивается по ходу добавления узлов в структуру и позволяет обращаться к любому узлу за константное время, зная только его номер. Также в самом узле хранятся номер его предка и список всех исходящих ребер, перейдя по которым можно спуститься на уровень ниже по дереву.

Помимо самих узлов в аналогичном *массиве данных* можно хранить ссыл-

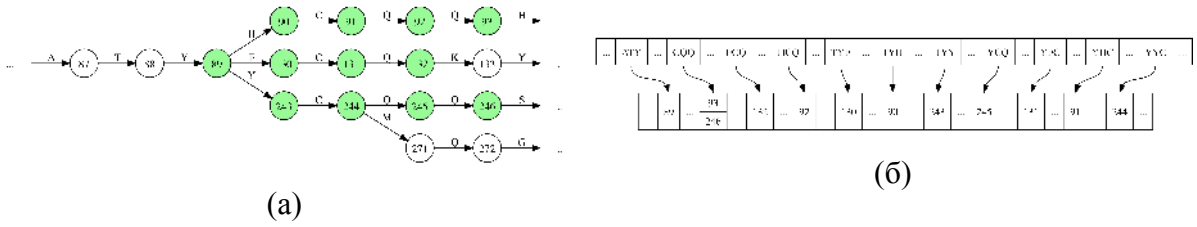
ки на произвольную стороннюю информацию. Например, в обычном состоянии в нем могут хратиться ссылки на аннотации узлов. Если возникает необходимость временно сохранять какую-то другую информацию об узлах дерева, то такой массив можно скопировать, не копируя само дерево. Это обеспечивает возможность добавление новых референсов даже во время работы алгоритмов поиска и выравнивания, что делает рассматриваемую структуру потокобезопасной.

В дополнение к дереву по мере добавления последовательностей заполняется хэш-таблица k-меров. Для этого в момент инициализации контейнера должен быть задан алфавит, который используется в референсных последвоательностях. Помимо классических нуклеотидов и аминоксилот, алфавит может содержать так же метасимволы (wildcard), заменяющие собой целую группу символов. Это удобно для работы с вырожденными нуклеотидами, такими как, например, $S = \{C, G\}$ или $N = \{A, C, G, T\}$.

Алфавит используется для эффективного построения таблицы k-меров с помощью рекурсивного кольцевого хэша [10]. При добавлении нового узла в дерево, его индекс также добавляется в таблицу по ключу, соответствующему k-меру, который в этом узле заканчивается. Длина k-мера может быть задана в момент инициализации. По умолчанию испольуются значения 7 для нуклеотидов и 3 для аминокислот.

2.2.2 Поиск паттернов

Построенная хэш-таблица, сопоставляющая каждому k-меру набор узлов, в которых он заканчивается, позволяет осуществлять быстрый поиск паттерна во всех хранящихся референсах. Идея алгоритма заключается в *покраске* всех узлов, соответствующих k-мерам искомого паттерна, и последующим подсчетом последовательно расположенных *окрашенных* узлов. Здесь покраской узла называется установление метки *true* во вспомогательном boolean массиве в ячейке, соответсвующей этому узлу. Такой массив содается при инициализациии работы алгоритма поиска и подменяет массив данных, описанный в предыдущем пункте, для возможности осуществления поиска параллельно с добавлением новых референсов в дерево.



(а) (б)

Рисунок 2.2 — (а) часть бора, покрашенная при поиске паттерна *ATY**X**CQQ*; (b) часть хеш-таблицы, содержащей k-меры, соответствующие искомому паттерну

Алгоритм делится на два этапа. На первом этапе последовательность паттерна P бьется на k-меры и осуществляется установка метки для всех соответствующих им узлов. Если хоть для одного k-мера не найдено ни одного узла, то поиск останавливается, так как данного паттерна нет референсной выборке. В противном случае для всех узлов k-мера устанавливается метка в массиве данных.

Отдельным случаем является ситуация, когда рассматриваемый k-мер содержит метасимвол. В этом случае требуется проверить все возможные варианты замены этого символа. Чтобы не перебирать экспаненциально возрастающее количество вариантов, алгоритм разрешает использовать только по одному метасимволу на k-мер. В связи с этим, для обеспечения возможности использования разумного количества метасимволов, выбранный стандартный размер k-мера меньше, чем используемый в BLAST-алгоритме [5]. Однако дальнейшее его уменьшение не рекомендуется, чтобы не допустить ситуации, когда каждому k-меру будет соответствовать слишком большое количество узлов.

Второй этап алгоритма начинается, когда покраски были завершены. От каждого узла, соответствующего последнему k-меру паттерна осуществляется $|P| - k + 1$ шагов по направлению к корню. Если на протяжении всех шагов метка была установлена во всех узлах, то искомый паттерн найден и заканчивается в выбранном узле.

Таким образом все вхождения паттерна в референсы ищется в среднем за $O(p{\cdot}\#occurrences)$. Стоит заметить, что как установка, так и проверка меток в первом и втором этапах соответственно могут быть произведены независимо, а потому возможно использование параллелизма по данным, что еще сильнее сокращает затрачиваемое время.

Работа данного алгоритма проиллюстрированна на рис. 2.2, а упрощенный алгоритм без обработки метасимволов может быть описан следующим псевдокодом:

function $search\_pattern(P, D, K, k)$
    Switch trie $Data$ array (D) to empty $Colors$ array
    for $i \in 1 \dots length(D)$ **do**
        $Colors(i) \leftarrow False$
    end for
    for $kmer$ in $get\_kmers(P)$ **do**
        for all $node\_id \in K(kmer)$ **do**
            $Colors(node\_id) \leftarrow True$
        end for
    end for
    $candidats \leftarrow K(last\_kmer(P))$
    $result \leftarrow \varnothing$
    for all $node\_id \in candidats$ **do**
        $current\_node \leftarrow node\_id$
        for $i \in 1 \dots |P| - k + 1$ **do**
            $current\_node \leftarrow parent(current\_node)$
            if $Colors(current\_node) == False$ **then**
                $break$
            end if
        end for
        if previous cycle ended without $break$ **then**
            Add $node\_id$ to $result$
        end if
    end for
    return $result$
end function

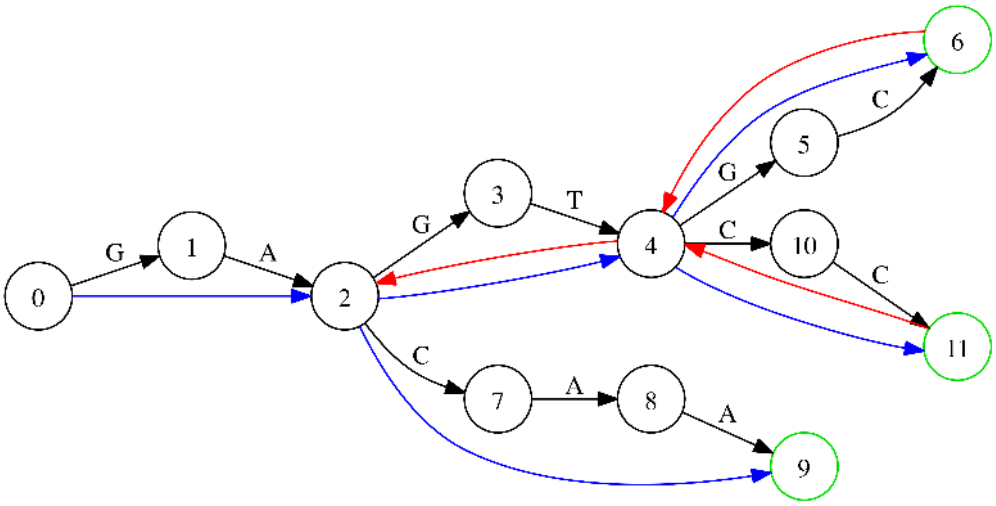


Рисунок 2.3 — Выполнение выравнивания на структуре дерева

2.2.3 Выравнивание one-vs-many

Сравнение последовательности запроса и референсов осуществляется с помощью парных выравниваний. Выравнивание относится к задачам с оптимальной подструктурой, а потому к нему применим метод динамического программирования. Классический оптимальный алгоритм выравнивания двух строк был описан в [34]. Параллельно были опубликованы аналогичные алгоримы [21; 28], ставшие первым приложением динамического программирования к обработке биологических последовательностей.

Во всех упомянутых алгоритмах задача поиска выравнивания двух строк S и T решается путем построения матрицы оптимальных выравниваний всех префиксов S_i и T_j, и имеет сложность $O(|S| \cdot |T|)$. Матрица может строиться итеративно, строчка за строчкой. Так, на итерации $j \cdot |S| + i$ будут известны выравнивания всех префиксов вплоть до S_i и T_j, а для ее подсчета требуется не вся строка T, а только ее префикс T_j. Это означает, что выравнивание можно выполнять в online режиме по мере поступления символов последовательно T. На этом наблюдении может быть построен алгоритм, эффективно позволяющий выравнивать последовательность на все референсы, содержащиеся в префиксном дереве.

Основная идея алгоритма состоит в переиспользовании матриц выравнивания на общие части ветвей, представляющих в дереве различные референсы. Такой принцип легко реализуется путем использования стэка матриц вы-

равнивания, в котором операция *push* осуществляется на развилках и листьях дерева. Рисунок 2.3 иллюстрирует работу алгоритма.

При обходе дерева в глубину пройденные ребра накапливаются в строку до достижения развилки или листа. Такой путь показан красными ребрами. Далее для накопленной строки строится матрица выравнивания и добавляется в стэк. Для корректного построения матрицы требуется осуществлять проверку стэка на пустоту.

Если стэк пуст, добавляется классическая матрица выравнивания последовательности запроса на переданную подстроку, являющуюся префиксом некоторого набора референсов. В случае, когда стэк не пуст, передаваемая подстрока берется из середины набора референсов, а потому новая матрица выравнивания не может строиться классическим способом. Корректная матрица получается при использовании последей строки матрицы с верхушки стэка в качестве предыдущей строки для новой матрицы.

При достижении листа (зеленые вершины на рис. 2.3) производится обратный проход (backtrace) по всем матрицам стэка в соответствии с используемым типом выравнивания. Полученный результат передается в callback-функцию, которая уже осуществляет его обработку.

Далее алгоритм обхода в глубину делает отбратный шаг (синие ребра на рис. 2.3), на котором из стэка последовательно убираются все матрицы выравнивания соответствующие пройденным развилкам. Эти матрицы более не требуются, так как все референсы с подобным префиксом к этому моменту уже были обработаны.

Ниже представлен псевдокод описанного алгоритма.

function $align(query, trie, callback)$
    $alicont \leftarrow empty\_alignment\_container(query, max\_depth(trie))$
    $previous\_node \leftarrow None$
    $fork\_stack \leftarrow empty\_stack()$
    $target \leftarrow empty\_string()$
    for $node$ in $dfs\_order(nodes(trie))$ **do**
        if $is\_leaf(previous\_node)$ **then**
            while $top(fork\_stack) \neq parent(node)$ **do**
                $pop(fork\_stack)$

```
            pop(alicont)
        end while
    end if
    Push symbol(node) to the end of target
    if is_fork(node) then
        push(fork_stack, node)
        align_and_push(alicont, target)
        target ← empty_string()
    end if
    if is_leaf(node) then
        align_and_push(alicont, target)
        result ← get_alignment(alicont)
        pop(alicont)
        callback(result)
    end if
    previous_node ← node
  end for
end function
```

Как видно из приведенного кода, данный алгоритм является классическим конечным автоматом, в котором в качестве состояний используются тип текущего узла, а выходных значений – результаты проведенных выравниваний. Подобное выравнивание последовательности на префиксное дерево позволяет заменить сложность $O(|S| \cdot max_{r \in R}|r| \cdot |R|)$ на $O(|S| \cdot |Tree|)$, где R – множество всех референсов, а $|Tree|$ – количество узлов в дереве. При достижении большой степени сжатия, реальный разрыв между этими величинами оказывается довольно значительным. В данном алгоритме также было сделано несколько оптимизирующих изменений, не меняющих его ассимптотическую сложность, но повышающих скорость работы.

Во-первых, порядок обхода узлов дерева кэшируется, вырезая все промежуточные узлы обратного хода. Такое кэширование позволяет избежать переполнения программного стэка и в два раза сократить время обхода дерева. При реальном использовании дерево заполняется референсами один раз перед началом работы, а далее используется для аннотирования многих по-

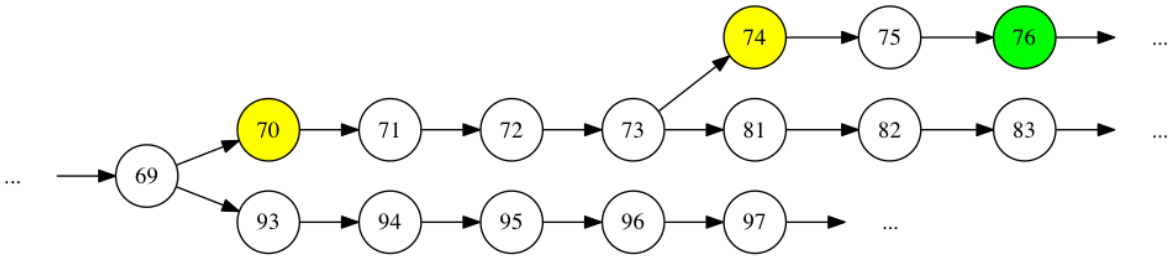


Рисунок 2.4 — Использование хеш-таблицы k-меров для выбора ветвей

следовательнсотей. В связи с этим кэш практически никогда не требуется перестраивать, а значит оптимизация работает почти все время.

Во-вторых, с помощью построенной хэш-таблицы k-меров можно сократить количество рассматриваемых ветвей и не обходить все дерево. Для этого выравниваемая последовательность бьется на k-меры аналогично тому, как это делается при поиске паттернов. Далее помечаются ветви, которые ведут к узлам, соответствующим этим k-мерам. Способ пометки ветвей показан на рис. 2.4. Если в узле 76 кончается некоторый k-мер из последовательности, то помечаются все узлы (e.g. 70, 74), следующие за каждой развилкой, ведущей от корня к данному узлу. Это позволяет быстро выбирать ветвь с k-мерами из запроса при нахождении в узле-развилке. Если исходящие из развилки ребра ведут хоть в один помеченный узел, то обходятся только ветви, начинающиеся с помеченных узлов. Если среди рассмотренных узлов помеченных нет, то обходятся все ветви.

2.2.4 Получение аннотаций и вариантов замен

Получение аннотаций или вариантов замен осуществляется в два этапа. На первом этапе требуется получить некоторое количество парных выравниваний аннотируемой последовательности на референсную базу, а на втором собрать аннотации для каждой позиции в этой последовательности.

Для выбора из всех референсов ближайших можно воспользоваться одним из двух вариантов функции callback, введенной в предыдущем пункте: выбор N лучших последовательностей или выбор всех последовательностей со степенью схожести больше указанной. Второй вариант является тривиальным, а для реализации первого можно воспользоваться очередью с приоритетами на основе двоичной кучи. Если в качестве приоритетов использо-

Рисунок 2.5 — Использование glocal и local выравниваний для аннотирования и поиска вариантов

вать значение, обратное *score* выравнивания, то сравнивать с минимальным элементом новые выравнивания можно за константу, а обновлять очередь за логарифмическую сложность.

Сборка аннотаций зависит от использованного алгоритма выравнивания (см. рис. 2.5). При использовании полуглобального (glocal) выравниваний [12] выбираются последовательности, наиболее близкие по всей протяженности вариабельного домена, а аннотации выбираются путем консенсуса. Для каждой позиции последовательности выбираются аннотации, подтвержденные наибольшим количеством ближайших референсов. Применении локального (local) выравнивания [28] реализует иную логику: для каждой позиции выбирается аннотация с референса, наиболее близкого к запросу в этой позиции.

Учитывая возможность запуска на “грязных” данных с ошибками, а также отсутствующими или лишними фрагментами, использование глобального (global) выравнивания [21] является наименее предпочтительным, поскольку ведет к неправильному аннотированию.

Получение списка вариантов идет аналогичным способом и тоже имеет две различные логики реализации. В первом случае используется полуглобальное выравнивание полностью аналогично алгоритму определения аннотаций. Во втором случае происходит предварительное разделение последовательности на регионы по уменьшенной выборке референсов и последующий поиск вариантов для каждого региона индивидуально. При этом на малых фрагментах могут использоваться все три вида выравнивания, поскольку гуманизируются достоверные последовательности, которые обычно уже не содержат ошибок и неполных регионов.

Второй метод может привести к неправильному поиску вариантов, поскольку разные ближайшие референсы для небольших последовательностей

регионов могут найтись в разных структурных семействах. В связи с этим такой поиск осуществляется для каждого семейства независимо.

Полученные варианты из последовательностей иммуноглобулинов человека и исходного организма используются в алгоритмах предсказания необходимости выполнения замен. Эти алгоритмы в данной работе не приводятся.

2.3 Программная реализация

Все описанные в предыдущем параграфе алгоритмы были реализованы в качестве библиотеки на языке Scala. В реализацию были включены следующие модули:

- алгоритмы глобального, локального и полуглобального выравниваний в вариациях с простыми и аффинными гэпами;
- стэк матриц для выполнения итеративных выравниваний;
- контейнер для хранения последовательностей (абстракция над префиксным деревом, хэш-таблицей k-меров и загрузкой аннотаций);
- алгоритмы поиска, выравнивания, аннотирования и получения вариантов замен, работающие над структурой контейнера.

Для проверки работы алгоритмов и сравнения метода с существующими данная библиотека была использована для реализации системы, выполняющей следующие функции:

- определение ближайших гомологов среди гермлайнов и любых дополнительных баз данных;
- определение структурного семейства, к которому принадлежит вариабельный домен;
- определение структурных регионов;
- получение вариантов замен на основании гермлайнов, сторонних человеческих и животных баз референсных последовательностей.

В реализованном инструменте принадлежность к региону определяется для каждой позиции в последовательности запроса, после чего результаты сглаживаются для получения диапазона, принадлежащего региону. В случае возникновения неоднозначности на границе двух регионов, позиция приписывается ближайшему CDR, поскольку невозможность определения позиции говорит, скорее, о ее гипервариабельности.

3. Результаты и сравнения с существующими решениями

3.1 Дизайн экспериментов и выбор тестовых данных

3.1.1 Выбор методов и метрик сравнения

При сравнении методов поиска ближайших гермлайнов и гомологичных вариабельных доменов иммуноглобулинов в сторонних базах, разработанный алгоритм в 100% случаев давал те же результаты, что и IgBLAST [16]. Осуществить сравнение предлагаемых вариантов замен оказалось невозможно ввиду отсутствия в открытом доступе инструментов, решающих данную задачу.

В связи с этим тестирование алгоритмов проводилось по критерию поиска FR и CDR регионов. Для тестов использовались последовательности тяжелых цепей, так как в них вариабельность CDR3 выше за счет использования дополнительного D-гена, следовательно сложность определения границ регионов возрастает. Сравнение проводилось с предсказаниями, сделанными инструментами IgBLAST и ROSIE [26]. Последний инструмент предназначен для поиска третичной структуры белка, но в процессе работы ищет регионы.

Для сравнения результатов аннотирования было введено несколько метрик. Каждая метрика применялась раздельно к каждой аннотации, то есть к началу и концу каждого рассмотренного региона. Чтобы понять общую картину, была введена метрика, определяющая разницу между предсказаниями – частоту различных предсказаний:

$$\text{DDR} = \frac{D}{T},$$

где D – количество различно предсказанных границ данного региона, а T – общее количество сделанных предсказаний данного региона.

Помимо установления самого факта различия между предсказаниями, требуется оценка характера различия и его величины. Другими словами, требуется определить, как сильно предсказания различаются, когда они различаются. Для этого введены метрики, показывающие средний сдвиг границ ре-

гионов в случае установления разных предсказаний:

$$AS_s = \frac{\sum(B_1 - B_2)}{\mathrm{D}},\ AS_u = \frac{\sum|B_1 - B_2|}{\mathrm{D}},$$

где B_i – предсказанные границы регионов разными инструментами, а D – количество различных предсказаний границ данного региона.

Наличие знаковой и беззнаковой версий этой метрики обсуловленно желанием определять тип рассматриваемых сдвигов. В случае близости абсолютных значений знаковой и беззнаковой версии можно говорить о преобладающем сдвиге в одну сторону, а значит о систематическом различии. Если же эти значения сильно различаются, то различие, скорее, носит случайный характер.

3.1.2 Выбор данных для проведения сравнений

В качестве референсной базы были использованы все комбинации из 51 VH, 75 DH и 6 JH последовательностей протранслированных человеческих генов, взятых из [32]. Из того же источника была взята разметка всех этих последовательностей в номенклатуре Kabat. Это обеспечило проаннотированную референсную базу в $51 \times 75 \times 6 = 22950$ последовательностей для поиска регионов в Kabat-номенклатуре.

Помимо гермлайнов были выбраны два набора последовательностей вариабельных доменов зрелых иммуноглобулинов. Первая база была собрана из успешно проаннотированных запросов в публичной очереди сервиса ROSIE. В момент сбора данных в очереди находилось 336 уникальных пар последовательностей (тяжелая и легкая цепи), проаннотированных в номенклатуре Chothia. Этот набор использовался для сравнения результатов определения регионов методом поиска паттернов (ROSIE) и разработанного метода. Для перевода номенклатуры в Kabat использовалась методика, описанная в [2].

Второй набор из 302 пар последовательностей был взят в базе IgBase [30; 31]. Эти данные не были проаннотированы. Так как аннотация такого большого числа последвоательностей через сервис ROSIE невозможна, аннотации в Kabat были получены с помощью IgBLAST. В силу ограничений этого

инструмента, размечены были регионы только с FR1 по FR3. Сравнение результатов аннотирования IgBLAST и разработанного метода в связи с этим также проводилось только по обозначенным регионам.

3.2 Выбор оптимального количества референсов

В процессе тестирования была выдвинута гипотеза, что для точного определения регионов не требуется использовать полную гермлайную базу из 22950 последовательностей, а можно обойтись меньшим числом. Для проверки этой гипотезы было проведено тестирование с целью определения количества референсов, необходимых для получения точных границ регионов в рамках используемой номенклатуры.

Проверка осуществлялась на гермлайновых данных, поскольку для них есть эталонная разметка, с которой можно было проводить сравнение. По полученным результатам из 10 тестов были построены графики зависимости ошибки предсказания от использованного количества референсов.

Рисунок 3.1 показывает зависимость количества неправильно проаннотированных последовательностей от количества используемых референсов. В данном случае ошибкой считалось наличие хотя бы одного несовпадения в границах регионов. Интересным наблюдением является тот факт, что практически все ошибки аннотирования пришлись на границу FR3 и CDR3 регионов.

Перед каждым из десяти тестов готовились данные тренировочного и проверочного наборов. В качестве проверочных данных выбирались 1000 случайных последовательностей из гермлайновых данных. Чтобы определить зависимость, требовалось создать несколько тренировочных наборов разного размера. Такие данные были получены путем последовательного добавления по 50 элементов из оставшихся после выбора проверочного набора гермлайнов. Были сгенерированы наборы размеров от 50 до 1000 с шагом 50. Эти наборы были использованы в качестве референсов для запуска алгоритма аннотирования.

Алгоритм был запущен со следующими параметрами:

- полуглобальное выравнивание с аффинными гэпами;

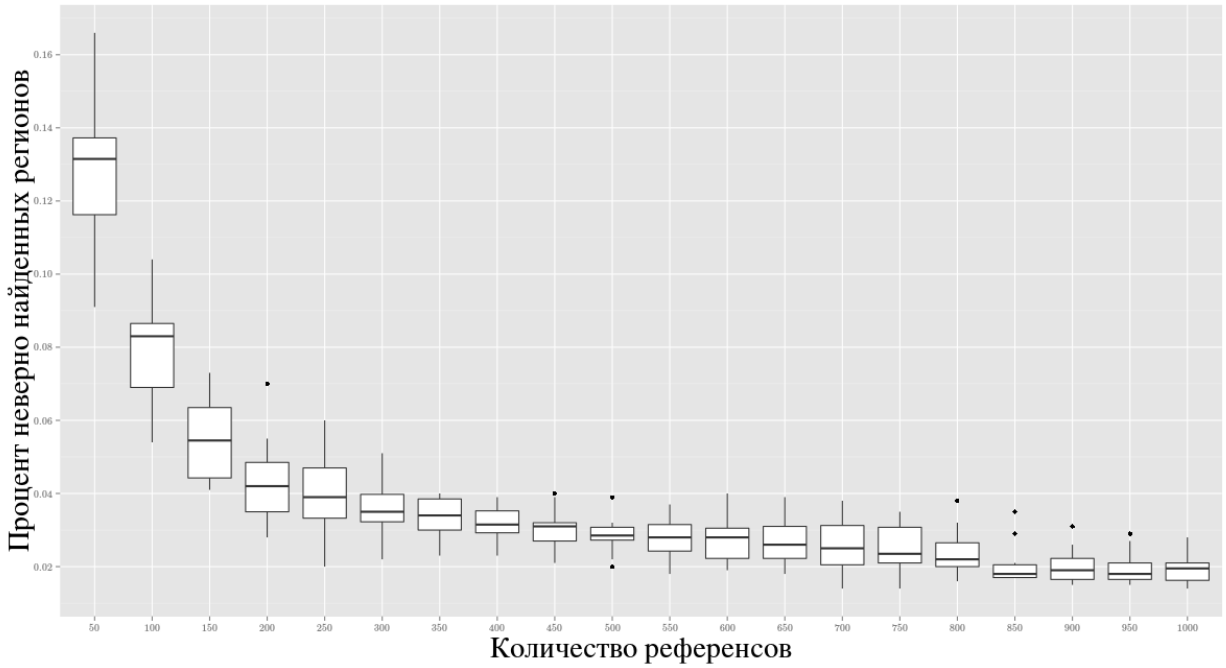


Рисунок 3.1 — Зависимость количества ошибок от размера используемой референсной выборки

- матрица выравнивания BLOSUM62;
- стоимость гэпов: открывающий -10, продолжающий -1;
- количество гомологов, используемых для аннотации 3.

Полученные результаты показывают, что ошибка аннотирования быстро убывает с ростом числа референсов, и становится менее 2 процентов уже после 850 референсов. Это позволяет использовать уменьшенные референсные выборки без значительного снижения точности для быстрого предсказания регионов, например, при обработке большого количества некачественных последовательностей, полученных после высокопроизводительного секвенирования с целью получить репертуар какого-нибудь конкретного региона.

3.3 Сравнение с существующими решениями

Таблица 3.1 показывает, как различались предсказания, сделанные описываемым методом, с предсказаниями, сделанными инструментом IgBLAST. Использованные метрики соответствуют введенным в первом параграфе этой главы.

Таблица 3.1 — Различия в предсказаниях с IgBLAST

регион	DDR (%)	AS_s	AS_u
FR1 начало	8.88	1.19	1.19
FR1 конец	1.32	-2.50	2.50
FR2 начало	0.00	0.00	0.00
FR2 конец	0.33	2.00	2.00
FR3 начало	0.33	-4.00	4.00
FR3 конец	20.39	-1.34	1.34

Так как значение границ одного региона строго зависит от значений предсказания другого, то рассматривались только границы FR регионов. Разница предсказаний для начала CDR соответствует разнице для конца предшествующего FR, а для конца CDR - началу следующего FR. В силу ограничений IgBLAST, результаты для FR4 региона не были получены и отсутствуют в сравнении.

В полученных результатах основного внимания заслуживают различия, полученные в начале FR1 и конце FR3 регионов. Как видно из таблицы 3.1, IgBLAST часто обрезает по одной аминокислоте с каждой стороны от рассматриваемого фрагмента. Это связано, скорее всего, с использованием в данном инструменте локального выравнивания, которому свойственно терять концы в случае их малого совпадения. Учитывая, что в IgBASE хранятся точные последовательности вариабельных доменов иммуноглобулинов без лишних аминокислот, в данном случае разработанный метод дает лучшие результаты в предсказании регионов. Прочие отличия встречаются не чаще, чем в полутора процентах случаев, а потому являются незначительными.

Также было проведено сравнение с IgBLAST с использованием только V-генов в качестве референсов, а также простого локального выравнивания в качестве алгоритма. В этом случае никаикх различий в аннотациях найдено не было, что подверждает выдвинутую выше гипотезу о природе полученных результатов.

Аналогичные результаты для сравнения с инструментом ROSIE приведены в таблице 3.2. Различия в аннотациях начала и конца вариабельного домена обсулавливается методом поиска регионов в Rosetta Antibody, лежащего

Таблица 3.2 — Различия в предсказаниях с ROSIE

регион	DDR (%)	AS_s	AS_u
FR1 начало	6.86	-1.30	1.30
FR1 конец	1.27	0.55	1.00
FR2 начало	<0.01	1.24	1.75
FR2 конец	<0.01	0.22	1.00
FR3 начало	0.00	0.00	0.00
FR3 конец	12.85	-0.47	1.49
FR4 начало	0.11	-0.64	1.13
FR4 конец	11.61	1.00	1.00

в основе инструмента ROSIE. Поиск осуществляется по паттернам границ CDR регионов, и потому в случае использования входной последовательности иммуноглобулина с дополнительными аминокислотами (например, конец лидерной последовательности перед началом домена и начало константного домена после конца домена), данные аминокислоты просто приписываются к началу и концу FR1 и FR4 региона. Остальные различия имеют случайный характер. Большие различия в определении FR3 региона лежит, скорее всего, в высокой вариабельности последних двух аминокислотных позиций в данном регионе. Ручной анализ случайно выбранных кандидатов говорит о более правильной аннотации методом выравнивания, однако однозначных выводов в данном случае сделать нельзя ввиду недостаточного количества данных для анализа.

Таблица 3.3 — Различия в предсказаниях между IgBLAST и ROSIE

регион	DDR (%)	AS_s	AS_u
FR1 начало	6.80	-1.26	1.26
FR1 конец	0.03	-1.00	1.00
FR2 начало	0.03	-1.00	1.00
FR2 конец	23.7	1.97	1.97
FR3 начало	0.80	1.00	1.00
FR3 конец	99.7	2.01	2.01

В таблице 3.3 приводится сравнение двух рассмотренных инструментов между собой на наборе даннных ROSIE. Как видно из таблицы, частота различных предсказаний между этими двумя инструментами не ниже, чем между представленным в данной работе и каждым из них в отдельности.

Заключение

Особенности последовательностей тяжелых и легких цепей вариабельных доменов иммуноглобулинов позволило использовать использовать механизм сжатия таких последовательностей с высокой эффективностью. Для этого была использована структура данных префиксного дерева, широко известная и применяемая в задачах обработки текстов на натуральном языке. Применение данной структуры позволило использовать огромные референсные базы полных последовательностей вариабельных доменов для аннотирования антител без применения эвристик, способных потерять часть близких последовательностей. Высокая скорость и точность полученного метода позволяет применять его для работы с результатами высокопроизводительного скрининга и других операций, требующих обработки больших объемов данных.

Описанные методы были воплощены в библиотеке на языке Scala, свободно доступной для использования в любых программных комплексах, как открытых, так и коммерческих. Построенный на базе данных библиотек инструмент поиска регионов внедрен во все протоколы обработки последовательностей антител в компании BIOCAD, на базе которой проводилась данная работа. Алгоритмы поиска вариантов замен в базах исходного организма и человеческих гермлайнов и зрелых последовательностей применены в той же компании для построения алгоритмов гуманизации, которые уже были использованы для обработки нескольких кандидатов для лечения аутоиммунных и онкологических заболеваний.

Дальнейшее развитие данной работы может быть направлено на применение описанной в задачах поиска и оценки сайтов иммуногенности. Другим направлением может быть применение данного метода для других классов белков, обладающих схожими особенностями, таких как Т-клеточные рецепторы и PDZ-доменные белки.

По результатам проделанной работы была подана статья на конференцию ECCB 2014.

Список литературы

1. A comparison of two murine monoclonal antibodies humanized by CDR-grafting and variable domain resurfacing / M. A. Roguska [et al.] // Protein Engineering. — 1996. — Vol. 9, no. 10. — Pp. 895–904.
2. Abhinandan K., Martin A. C. Analysis and improvements to Kabat and structurally correct numbering of antibody variable domains // Molecular Immunology. — 2008. — Vol. 45, no. 14. — Pp. 3832–3839.
3. Adair J., Athwal D., Emtage J. Humanised antibodies. — Jan. 1999 ; — US Patent 5,859,205.
4. Aho A. V., Corasick M. J. Efficient String Matching: An Aid to Bibliographic Search // Commun. ACM. — New York, NY, USA, 1975. — June. — Vol. 18, no. 6. — Pp. 333–340.
5. Basic local alignment search tool / S. F. Altschul [et al.] // J. Mol. Biol. — 1990. — Oct. — Vol. 215, no. 3. — Pp. 403–410.
6. Beard H. [et al.] Applying Physics-Based Scoring to Calculate Free Energies of Binding for Single Amino Acid Mutations in Protein-Protein Complexes // PLoS ONE. — 2013. — Dec. — Vol. 8, no. 12. — e82849.
7. Breiman L. Random Forests // Machine Learning. — 2001. — Vol. 45, no. 1. — Pp. 5–32.
8. Brochet X., Lefranc M.-P., Giudicelli V. IMGT/V-QUEST: the highly customized and integrated system for IG and TR standardized V-J and V-D-J sequence analysis // Nucleic Acids Research. — 2008. — Vol. 36, suppl 2. — W503–W508.
9. Carter P., Presta L. Preparing humanized immunoglobulin from complimentarity and human framework region; obtaining human and mammalian variable domain sequences, detecting, replacing, and adjusting sequences, detect binding, prepare immunoglobulin. — Apr. 2000 ; — US Patent 6,054,297.
10. Cohen J. D. Recursive Hashing Functions for n-Grams // ACM Trans. Inf. Syst. — 1997. — Vol. 15, no. 3. — Pp. 291–320.
11. Fredkin E. Trie Memory // Commun. ACM. — New York, NY, USA, 1960. — Sept. — Vol. 3, no. 9. — Pp. 490–499.

12. Glocal alignment: finding rearrangements during alignment / M. Brudno [et al.] // Bioinformatics. — 2003. — Vol. 19, suppl 1. — Pp. i54–i62.

13. Human-engineered monoclonal antibodies retain full specific binding activity by preserving non-CDR complementarity-modulating residues / G. M. Studnicka [et al.] // Protein Engineering. — 1994. — Vol. 7, no. 6. — Pp. 805–814.

14. Humanization of the anti-CD18 antibody 6.7: an unexpected effect of a framework residue in binding to antigen / C. Caldas [et al.] // Molecular Immunology. — 2003. — Vol. 39, no. 15. — Pp. 941–952.

15. Humanized immunoglobulins / C. Queen [et al.]. — Dec. 1997 ; — US Patent 5,693,762.

16. IgBLAST: an immunoglobulin variable domain sequence analysis tool / J. Ye [et al.] // Nucleic Acids Research. — 2013. — Vol. 41, W1. — W34–W40.

17. IMGT unique numbering for immunoglobulin and T cell receptor variable domains and Ig superfamily V-like domains / M.-P. Lefranc [et al.] // Developmental & Comparative Immunology. — 2003. — Vol. 27, no. 1. — Pp. 55–77.

18. Inc. C. C. G. Molecular Operating Environment (MOE), 2011.10.

19. Al-Lazikani B., Lesk A. M., Chothia C. Standard conformations for the canonical structures of immunoglobulins // Journal of Molecular Biology. — 1997. — Vol. 273, no. 4. — Pp. 927–948.

20. Market E., Papavasiliou F. N. V(D)J Recombination and the Evolution of the Adaptive Immune System // PLoS Biol. — 2003. — Oct. — Vol. 1, no. 1. — e16.

21. Needleman S. B., Wunsch C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins // Journal of Molecular Biology. — 1970. — Vol. 48, no. 3. — Pp. 443–453.

22. Pauling L., Corey R. B. Configurations of Polypeptide Chains With Favored Orientations Around Single Bonds: Two New Pleated Sheets // Proc. Natl. Acad. Sci. U.S.A. — 1951. — Nov. — Vol. 37, no. 11. — Pp. 729–740.

23. Prediction of site-specific interactions in antibody-antigen complexes: the proABC method and server / P. P. Olimpieri [et al.] // Bioinformatics. — 2013. — Vol. 29, no. 18. — Pp. 2285–2291.

24. Reshaping human antibodies for therapy / L. Riechmann [et al.] // Nature. — 1988. — Mar. 24. — Vol. 332, no. 6162. — Pp. 323–327.

25. Sequences of Proteins of Immunological Interest / E. Kabat [et al.]. — Diane Publishing Company, 1992.

26. Serverification of Molecular Modeling Applications: The Rosetta Online Server That Includes Everyone (ROSIE) / S. Lyskov [et al.] // PLoS ONE. — 2013. — May. — Vol. 8, no. 5. — e63906.

27. Sircar A., Kim E. T., Gray J. J. RosettaAntibody: antibody variable region homology modeling server // Nucleic Acids Research. — 2009. — Vol. 37, suppl 2. — W474–W479.

28. Smith T., Waterman M. Identification of common molecular subsequences // Journal of Molecular Biology. — 1981. — Vol. 147, no. 1. — Pp. 195–197.

29. Strebhardt K., Ullrich A. Paul Ehrlich's magic bullet concept: 100 years of progress // Nature reviews. Cancer. — 2008. — June. — Vol. 8, no. 6. — Pp. 473–480.

30. Structural classification of CDR-H3 revisited: a lesson in antibody modeling / D. Kuroda [et al.] // Proteins. — 2008. — Nov. — Vol. 73, no. 3. — Pp. 608–620.

31. Systematic classification of CDR-L3 in antibodies: implications of the light chain subtypes and the VL-VH interface / D. Kuroda [et al.] // Proteins. — 2009. — Apr. — Vol. 75, no. 1. — Pp. 139–146.

32. T.M. Tomlinson S.C. Williams S. C., Winger G. VBASE Sequence Directory // MRC Centre for Protein Engineering. — 1996. — Vol. 1, no. 1.

33. VBASE2, an integrative V gene database / I. Retter [et al.] // Nucleic Acids Research. — 2005. — Vol. 33, suppl 1. — Pp. D671–D674.

34. Wagner R. A., Fischer M. J. The String-to-String Correction Problem // J. ACM. — New York, NY, USA, 1974. — Jan. — Vol. 21, no. 1. — Pp. 168–173.