МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра дискретной математики и алгоритмики

ИЛЬИН Андрей Викторович

РАЗРАБОТКА ПРОТОКОЛА ДОКИНГА БЕЛКОВЫХ МОЛЕКУЛ

Магистерская диссертация

специальность 1-31 81 09 «Алгоритмы и системы обработки больших объемов информации»

Научный руководитель:
Тузиков Александр Васильевич,
доктор физико-математических наук,
профессор

Научный консультант:
Хадарович Анна Юрьевна,
магистр физико-математических наук

Допущена к защите
“___” _____ 2020 г.

Зав. кафедрой дискретной математики и алгоритмики
Котов Владимир Михайлович,
доктор физико-математических наук, профессор

Минск, 2020

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Внедрение информационных технологий в различные сферы является в настоящее время довольно важным направлением развития экономики нашей страны. В частности, особый интерес представляют пограничные направления, в которых осуществляется синтез знаний из различных дисциплин. Данная работа занимает место на стыке биологии, химии и информатики – в относительно новой области под названием «биоинформатика».

Белок-белковые комплексы – объект исследования многих научных коллективов. В особенности, интерес представляет задача предсказания трёхмерной структуры белковых комплексов, основываясь на знании о структуре одиночных белков, в результате соединения которых данный комплекс получен, до начала процесса связывания. Алгоритм, позволяющий осуществлять такое предсказание, называют протоколом докинга, и именно разработке такого протокола посвящена данная магистерская диссертация.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Ключевые слова: белок-белковые комплексы, белок-белковые взаимодействия, протокол докинга.

Объектом данного исследования являются белковые комплексы, а предметом исследования – белок-белковые взаимодействия.

Цель исследования: разработка протокола докинга белковых молекул.

Задачи исследования:

- ознакомиться с различными уровнями строения белка, химическими и физическими свойствами белка, классификацией белков;
- изучить принципы белок-белкового докинга и основные компоненты, входящие в состав протоколов;
- разработать алгоритм предсказания структуры комплекса (другими словами, протокол докинга);
- создать программную систему, реализующую данный протокол и позволяющую проводить исследования и настройку составляющих этапов алгоритма.

Полученные результаты: разработан протокол докинга белковых молекул, использующий на этапе оптимизации алгоритм роевого интеллекта (метод роя частиц и его улучшение), выполнена его программная реализация. Тестирование протокола показало перспективные результаты, в связи с чем планируется проведение дальнейших экспериментов в данном направлении.

Работа состоит из трёх глав. Первая глава содержит общие сведения о белках и белковых комплексах, базах данных с информацией о белках и формате их представления в компьютере. Во второй главе рассказывается про протокол с теоретической стороны – приведены этапы протокола и используемые на каждом из них алгоритмы. Последняя, третья глава посвящена программной реализации протокола, который описан в предыдущей главе. В общей сложности, весь материал занимает 41 страницу и содержит 16 изображений, 1 таблицу и 9 формул. Список использованных библиографических источников содержит 31 позицию, включая собственную дипломную работу, которая была посвящена одному из этапов протокола.

АГУЛЬНАЯ ХАРАКТАРЫСТЫКА ПРАЦЫ

Ключавыя словы: бялок-бялковыя комплексы, бялок-бялковыя ўзаемадзеянне, пратакол докінга.

Аб'ектам дадзенага даследавання з'яўляюцца бялковыя комплексы, а прадметам даследавання – бялок-бялковыя ўзаемадзеянне.

Мэта даследавання: распрацоўка пратакола докінга бялковых малекул.

Задачы даследавання:

- азнаёміцца з рознымі ўзроўнямі будынку бялкоў, хімічнымі і фізічнымі ўласцівасцямі бялку, класіфікацыяй бялкоў;
- даследваць прынцыпы бялок-бялковага докінга і асноўныя кампаненты, якія ўваходзяць у склад пратаколаў;
- распрацаваць алгарытм прадказання структуры комплексу (іншымі словамі, пратакол докінга);
- стварыць праграмную сістэму, якая рэалізуе дадзены пратакол, дазваляе праводзіць даследаванні і наладу этапаў алгарытму.

Атрыманыя вынікі: распрацаваны пратакол докінга бялковых малекул, які выкарыстоўвае на этапе аптымізацыі алгарытм раявога інтэлекту (метад роя часціц і яго аптымізацыю), выкананая яго праграмная рэалізацыя. Тэсціраванне пратаколу паказала перспектыўныя вынікі, у сувязі з чым плануецца правядзенне далейшых эксперыментаў у гэтым напрамку.

Праца складаецца з трох раздзелаў. Першая частка змяшчае агульныя звесткі пра бялкі і бялковыя комплексы, базы дадзеных з інфармацыяй пра бялкі і фармаце іх прадстаўлення ў камп’ютары. У другой частцы распавядаецца пра пратакол з тэарэтычнага боку – прыведзены этапы пратаколу і алгарытмы, якія выкарыстоўваюцца на кожным з іх. Апошняя, трэцяя частка прысвечана праграмнай рэалізацыі пратаколу, які апісаны ў папярэдняй частцы. Увогуле, увесь матэрыял займае 41 старонку і змяшчае 16 малюнкаў, 1 табліцу і 9 формул. Спіс выкарыстаных бібліяграфічных крыніц складаецца з 31 пазіцый, уключаючы ўласную дыпломную працу, якая была прысвечана аднаму з этапаў пратаколу.

GENERAL DESCRIPTION OF WORK

Keywords: protein-protein complexes, protein-protein interactions, docking protocol.

The object of this study are protein complexes, and the subject of the study are protein-protein interactions.

Goal: to develop a protocol for docking protein molecules.

Objectives:

- familiarize with different levels of protein structure, chemical and physical properties of protein, protein classification;
- study the principles of protein-protein docking and the main components that make up the protocols;
- develop an algorithm for predicting the structure of the complex (in other words, a docking protocol);
- create a software system that implements this protocol, allows you to conduct research and configure component parts of the algorithm.

Key results: developed a protocol for docking protein molecules using the swarm intelligence algorithm (particle swarm method and its improvement) at the optimization stage, performed software implementation of it. Quality and performance testing showed promising results, in connection with which it is planned to conduct further experiments in this direction.

The work consists of three chapters. The first chapter contains general information about proteins and protein complexes, databases with information about proteins and the format of their presentation on a computer. The second chapter tells about the protocol from the theoretical side - the steps of the protocol and the algorithms used on each of them are given. The last, third chapter is devoted to the software implementation of the protocol, which is described in the previous chapter. In total, all the material takes 41 pages and contains 16 images, 1 table and 9 formulas. The list of used bibliographic sources contains 31 items, including my own Bachelors thesis, which was devoted to one of the stages of the protocol.

ГЛАВА 1 ОСНОВНЫЕ СВЕДЕНИЯ О БЕЛКАХ

1.1 Основные определения.

Белки – органические соединения, состоящие из аминокислотных остатков, последовательно соединенных пептидной связью (рисунок 1.1).

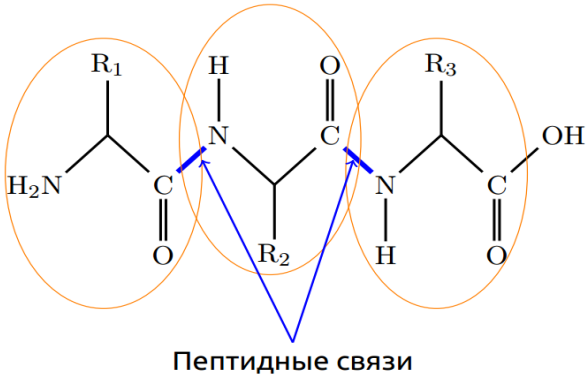

Рисунок 1.1 – Пример строения белка

Пептидная связь – вид химической связи, возникающей в результате взаимодействия карбоксильной группы (–COOH) одной аминокислоты с аминогруппой ($–NH_2$) другой аминокислоты.

Отличительными особенностями данного типа связи является затруднительное вращение вокруг данной связи (валентные углы у атомов C и N всегда равны примерно 120º), расположение атомов каждого пептидного звена в одной плоскости, повышенная прочность относительно других разновидностей амидной связи.

Для каждого из звеньев цепи в связи, таким образом, можно выделить лишь три связи, вокруг которых возможно вращение (рисунок 1.2). Однако, для данных углов также имеются ограничения, которые представляются в виде карты Рамачандрана (рисунок 1.3).

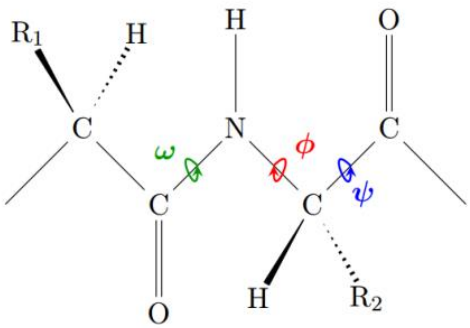

Рисунок 1.2 – Углы вращения вокруг связей в остове белка

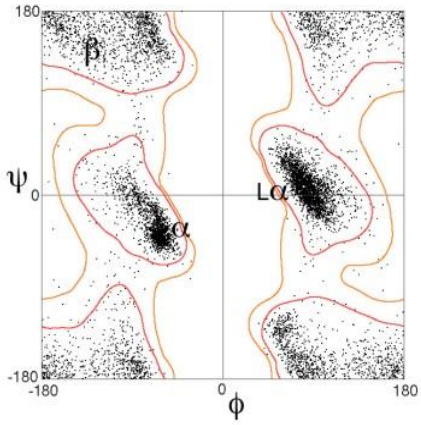

Рисунок 1.3 – Пример карты Рамачандрана для глицина

Всего в природе существует 20 стандартных остатков L-альфа-аминокислот, которые могут входить в состав белков. Аминокислотные остатки, не входящие в стандартный набор, либо не встречаются в составе белков вовсе, либо количество их вхождений в белковые цепи пренебрежимо мало, в связи с чем в большинстве исследований они не учитываются. Каждому из стандартных аминокислотных остатков ставятся в соответствие собственные одно- и трёхбуквенные обозначения (таблица 1.1). При этом, различные системы используют различные обозначения, из-за чего часто возникает необходимость перевода из одной системы в другую.

Название аминокислоты	Трёхбуквенная аббревиатура	Однобуквенная аббревиатура
Аланин	Ala	A
Аргинин	Arg	R
Аспарагин	Asn	N
Аспарагиновая кислота	Asp	D
Валин	Val	V
Гистидин	His	H
Глицин	Gly	G
Глутамин	Gln	Q
Глутаминовая кислота	Glu	E
Изолейцин	Ile	I
Лейцин	Leu	L
Лизин	Lys	K
Метионин	Met	M
Пролин	Pro	P
Серин	Ser	S
Тирозин	Tyr	Y
Треонин	Thr	T
Триптофан	Trp	W
Фенилаланин	Phe	F
Цистеин	Cys	C

Таблица 1.1 – Стандартные аминокислотные остатки и их аббревиатуры

В белках выделяют главную и боковые цепи. Главная цепь – наиболее длинная последовательность соединённых друг с другом атомов, содержащая карбоксильную и аминовую части всех входящих в состав белка аминокислот.

Боковые цепи – все другие цепи атомов, образованные радикалами аминокислот (рисунок 1.4).

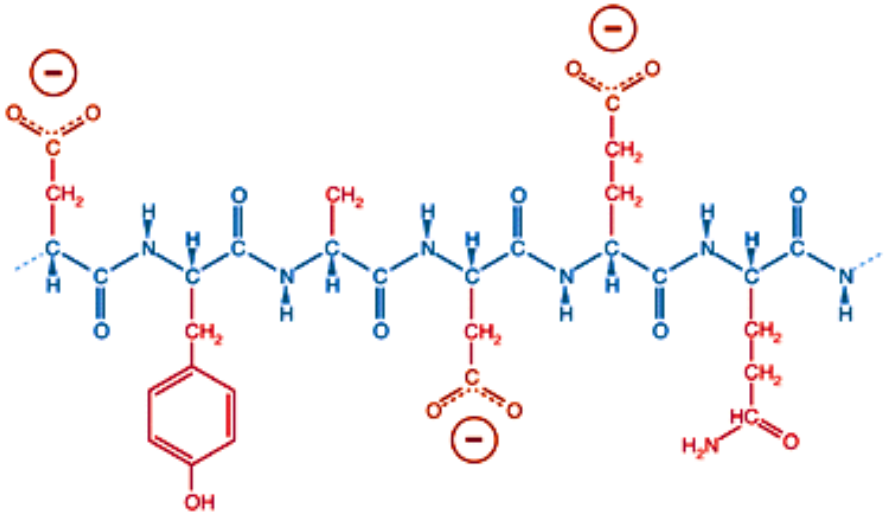

Рисунок 1.4 – Главная (выделенная синим) и боковые (выделены красным) белковые цепи.

Длина белковой цепи может быть различной – от 70 до нескольких тысяч мономеров. Основная белковая цепь, при этом, несимметрична, в связи с чем выделяют два её направления, которые различаются тем, какая группа является последней в цепи в данном направлении – аминная или карбоксильная.

1.2 Структура и классификация белков. Модель белка

Различают 4 уровня структурной организации белков:

- Первичная структура – последовательность аминокислотных остатков в белковой цепи.
- Вторичная структура – простое упорядочивание фрагментов цепи. Наиболее распространены такие типы вторичной структуры, как α-спирали (представляют собой плотные витки вокруг длинной оси) и β-листы (несколько зигзагообразных полипептидных цепей). Примеры вторичной структуры изображены на рисунке 1.5.
- Третичная (трёхмерная) структура – укладка вторичных структур одной полипептидной цепи в глобулу. Данный тип структурной организации стабилизируется различными дополнительными видами связей (дисульфидная, водородная, ионная), а также гидрофобным взаимодействием, которое играет наибольшую роль в построении структуры.
- Четвертичная структура представляет собой совокупность нескольких цепей третичной структуры, объединённых в составе белкового комплекса.

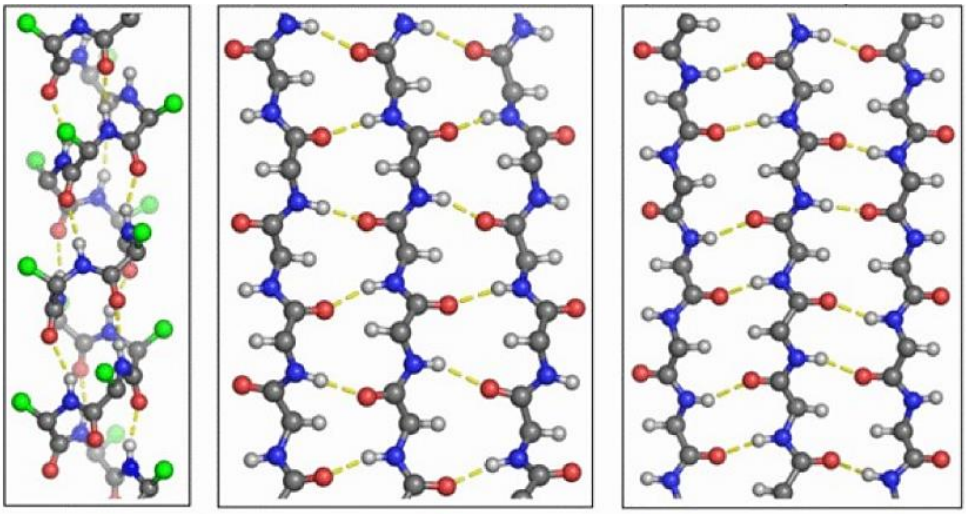

Рисунок 1.5 – Пример вторичной структуры в виде
α-спирали (слева) и β-листов (в центре и справа).

В связи с вышенаписанным, наиболее простой моделью белка является последовательность символов, каждый из которых определяет аминокислотный остаток на соответствующей позиции в белке (например, MAGTAVAHTLLPF). Более сложные модели включают также описание типа укладки, структуры петель, расположения боковых групп всех аминокислотных остатков. В общем случае трёхмерная структура белка представляется в виде списка всех входящих в его состав атомов с их координатами. Чаще всего трёхмерная структура изображается в виде «палочковой» модели, на которой изображены атомы и связи между ними, либо в виде поверхности, образованной сферами радиусов Ван-дер-Ваальса вокруг атомов (рисунок 1.6).

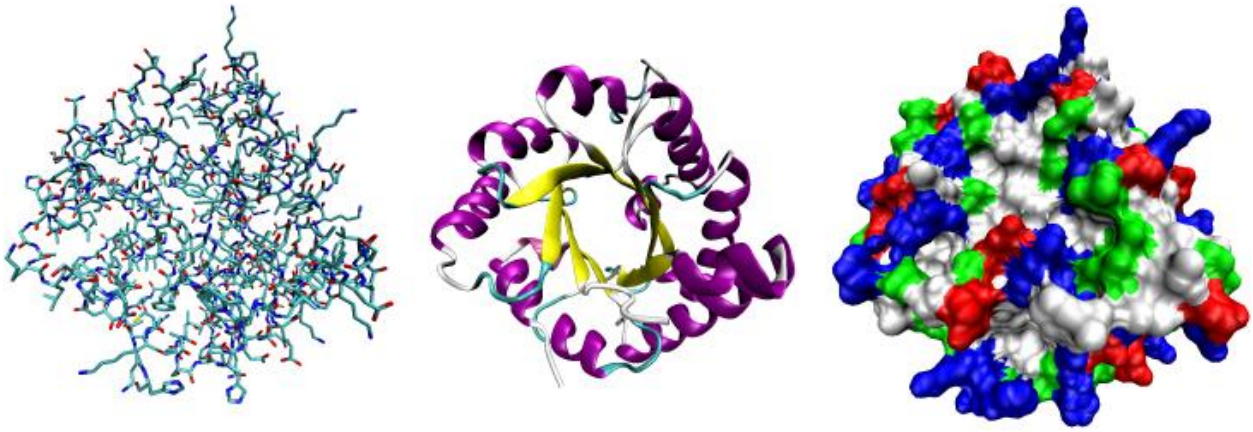

Рисунок 1.6 – Изображение трёхмерной структуры белка 1TIM в виде «палочковой» модели (слева), его вторичных структур (в центре) и контактной поверхности (справа)

Существует классификация белков по общему типу строения и расположению белков в клетке:

- Фибриллярные белки, образующие огромные агрегаты. По структуре такие белки высоко регулярны, их связи основаны чаще всего на взаимодействиях между разными цепями.
- Мембранные белки. Они находятся в мембране, где нет воды, но части их выступают из мембраны в водные растворы. Части таких белков, лежащие внутри клетки, как и фибриллярные белки, высоко регулярны и прошиты водородными связями, но размер этих регулярных частей ограничен толщиной мембраны.
- Водорастворимые, живущие в воде глобулярные белки наименее регулярны, их структура поддерживается взаимодействиями белковой цепи с самой собой, причем особенно важны взаимодействия далеких по цепи, но сблизившихся в пространстве углеводородных (гидрофобных) групп, а также взаимодействиями белковой цепи с кофакторами.
- Разупорядоченные белки – относительно недавно выделенный класс белков, не обладающих постоянной трехмерной структурой, либо приобретающих ее только на короткое время.

Белковые комплексы

Белковый комплекс – форма четвертичной структуры, представляющая собой объединение нескольких белковых молекул в единый комплекс (рис. 1.7). Жёлтым и зелёным на рисунке выделены участки белков, относящиеся к так называемому «интерфейсу». Именно эти аминокислотные остатки осуществляют связывание двух белков воедино.

Белковые комплексы являются основным объектом исследования для многих биологических процессов, ибо они выполняют широкий спектр функций в живых организмах. Благодаря пространственной близости, скорость взаимодействия и селективность связывания между комплексом и субстратами могут быть значительно выше, что, собственно, и является причиной более высокой клеточной эффективности.

Как и в случае одиночных белковых молекул, структура комплексов на атомном уровне может быть точно определена лишь с помощью экспериментальных методов, таких как рентгеновская кристаллография, ядерный магнитный резонанс и др.

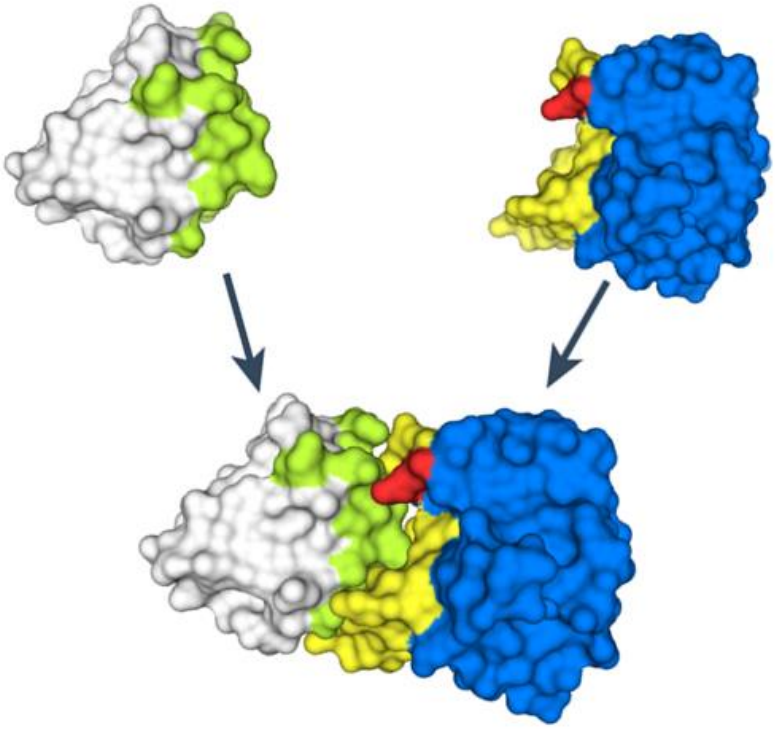

Рисунок 1.7 – Образование белкового комплекса из двух независимых пептидных цепочек.

1.3 Системы хранения и обработки информации о белках

Наиболее известной базой данных является Брукхейвенский банк пространственных структур (Protein Data Bank, PDB), в котором содержится информация о пространственной структуре белков. Данная база поддерживается совместно университетом Rutgers (США, штат Нью-Джерси); EBI (Англия) и BIRD (Institute for Bioinformatics Research and Development, Япония). Вся информация доступна бесплатно через интернет [26].

Идентификатор соединения в PDB – четырехзначный код, состоящий из цифр и букв латинского алфавита. Структура белка записывается в файл с расширением *.pdb в строго определенном формате. Наибольший интерес в данных файлах представляет информация в секции «ATOM» (рисунок 1.8), в которой содержатся номера и имена аминокислотных остатков, названия и трехмерные координаты атомов, название цепи белка и некоторая другая информация.

Другой базой данных, содержащей классифицированную и точно аннотированную информацию о последовательностях белков, является Uniprot [30]. Во многих случаях имеется соответствие между данными в PDB и Uniprot, однако в некоторых случаях информация немного отличается. Довольно много

информации содержится также в базе Национального центра Биотехнологической информации (NCBI) [19].

```
ATOM      1  N   HIS A  17     -12.690   8.753   5.446  1.00 29.32           N
ATOM      2  CA  HIS A  17     -11.570   8.953   6.350  1.00 21.61           C
ATOM      3  C   HIS A  17     -10.274   8.970   5.544  1.00 22.01           C
ATOM      4  O   HIS A  17     -10.193   8.315   4.491  1.00 29.95           O
ATOM      5  CB  HIS A  17     -11.462   7.820   7.380  1.00 23.64           C
ATOM      6  CG  HIS A  17     -12.551   7.811   8.421  1.00 21.18           C
ATOM      7  ND1 HIS A  17     -13.731   7.137   8.194  1.00 28.94           N
ATOM      8  CD2 HIS A  17     -12.634   8.384   9.644  1.00 21.69           C
ATOM      9  CE1 HIS A  17     -14.492   7.301   9.267  1.00 27.01           C
ATOM     10  NE2 HIS A  17     -13.869   8.058  10.168  1.00 22.66           N
ATOM     11  N   ILE A  18      -9.269   9.660   6.089  1.00 19.45           N
ATOM     12  CA  ILE A  18      -7.910   9.377   5.605  1.00 18.67           C
ATOM     13  C   ILE A  18      -7.122   8.759   6.749  1.00 16.24           C
ATOM     14  O   ILE A  18      -7.425   8.919   7.929  1.00 18.80           O
```

Рисунок 1.8 – Пример записей в разделе «ATOM» PDB-файла.

В ходе реализации разработанного алгоритма часто возникала необходимость проверки промежуточных данных. Для визуализации трёхмерных структур, при этом, использовались программы PyMol и Python Molecule Viewer (рисунок 1.9) [28].

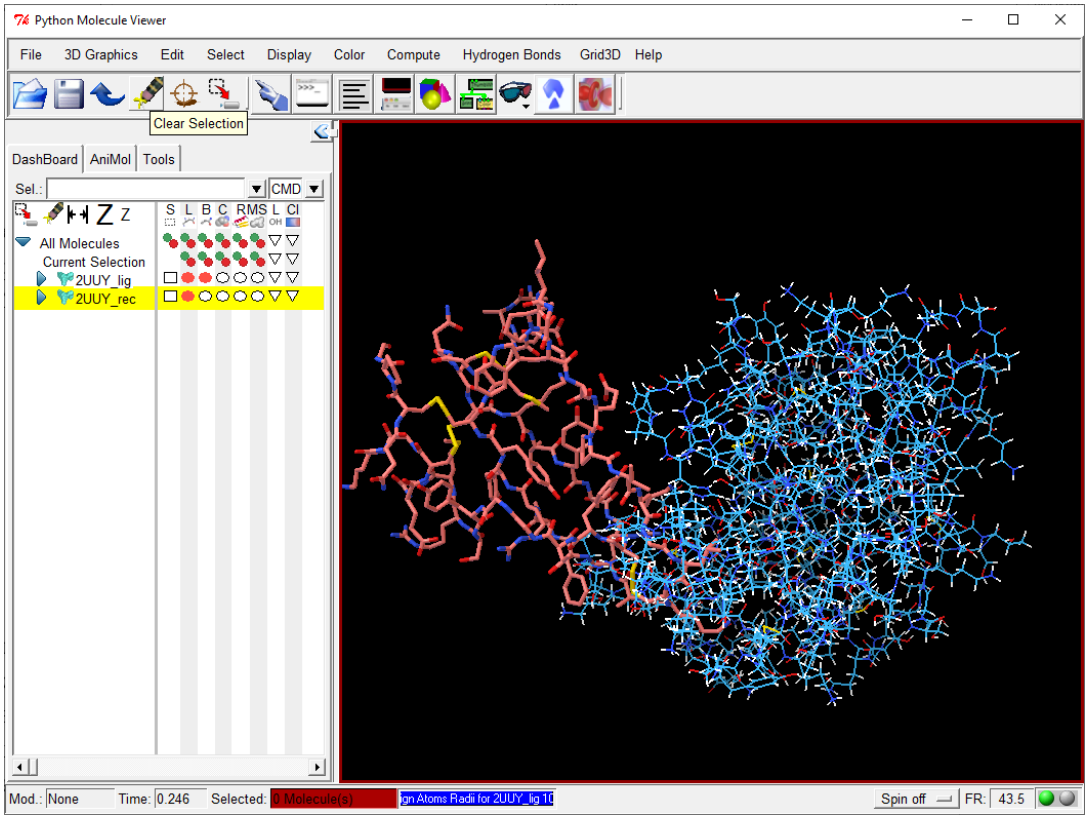

Рисунок 1.9 – Рабочее окно программы Python Molecule Viewer.

Для проверки точности работы протокола применяется набор данных Protein-Protein Docking Benchmark V5 [31]. Данный набор для каждого комплекса содержит по два PDB-файла с трёхмерной структурой обоих взаимодействующих белков, причём в двух версиях – состыкованные белки (для проверки результатов работы протокола) и случайным образом расположенные разделённые белки (для использования как входных данных).

ГЛАВА 2. ПРОТОКОЛ БЕЛОК-БЕЛКОВОГО ДОКИНГА

2.1 Постановка задачи

Входными данными является информация о двух независимых белковых цепях до начала их взаимодействия (в несвязанном состоянии): последовательность аминокислотных остатков, типы и координаты всех атомов. Данная информация предоставляется в виде двух файлов в формате PDB, соответствующих данным о белке-рецепторе и белке-лиганде.

Как правило, при образовании комплекса происходит объединение двух биохимических соединений, которые сильно отличаются друг от друга по размеру. Большее из них в таком случае называется рецептором, а меньшее – лигандом. С точки зрения постановки данной задачи принципиальной разницы нет, но знание о малых (по сравнению с рецептором) размерах лиганда пригодится нам в дальнейшим в при реализации протокола.

На основе предоставленных сведений требуется получить один файл в формате PDB, который будет описывать положение всех белковых цепей (и рецептора, и лиганда) после формирования белок-белкового комплекса. Таким образом требуется определить, в каком положении и с какой стороны будет выполнена стыковка белка-лиганда к белку-рецептору.

Результат работы алгоритма в дальнейшем используется для оценки моделей исследователями в области биохимии и служит для обнаружения потенциально подходящих под задачу комплексов и/или отсечения вариантов, проверка которых на практике маловероятно приведёт к положительному результату.

2.2 Общая схема работы протокола

Путь от двух несвязных структур к единому итоговому комплексу можно разбить на несколько этапов (рисунок 2.1). В рамках протокола решается ряд отдельных задач и замена алгоритма в каждой из них может быть осуществлена независимо от других. Данный факт позволяет нам построить систему, которая позволит в том числе проводить эксперименты с каждым из этапов, оценивая результаты в общем контексте.

Первый этап – предварительная обработка входных данных, который заключается в чтении входных из PDB-файлов, их разбор и удобное структурированное представления в памяти, а также аннотирование вспомогательной информацией, которая может быть необходима некоторым алгоритмам (площадь доступной растворителю поверхности и др.).

Рисунок 2.1 – Основные этапы протокола докинга.

Второй этап заключается в определении наиболее вероятных участков для взаимодействия белковых молекул (т.е. поиск белковых интерфейсов). Данная операция опциональна, т. е. может и не выполняться для получения хорошего результата. Тем не менее, данный этап позволяет найти те регионы вокруг поверхности рецептора, где правильное решение будет вероятнее всего находится. Отметим, что предсказание интерфейсов является само по себе отдельной темой для научных исследований и на данный момент не существует алгоритмов, позволяющих точно определять эти интерфейсы. В общем случае, практикуется подход, когда берётся не только самое лучшее из предсказаний на этом этапе, но некоторое количество различных лучших результатов, но тут следует уделять вниманию балансу между качеством и временем (чем больше возьмём, тем вероятнее там будет правильный вариант, но и время последующего поиска увеличится).

Третий этап заключается в нахождении оптимального положения белков, которое и будет служить результатом работы протокола. С научной точки зрения, на данном этапе требуется использовать решения сразу двух различных задач – оценка качества модели и поиск модели, дающий наилучшее значение этой оценки. Обе эти задачи не имеют точного решения, работающего за разумное время, так что результат третьего этапа у нас также будет содержать ошибочные данные. Тем не менее, мы можем воспользоваться идеей с предыдущего этапа и выдавать не один какой-то результат, а несколько различных наиболее вероятных – в любом случае осуществить анализ лишь этого фиксированного числа проще, чем всех возможных вариантов.

Последним этапом работы системы является постобработка результатов. К этому этапу можно в целом и отнести выделение наиболее вероятных ответов. При этом, поскольку схожие ответы нас интересуют не очень сильно, перед этим выполняется кластеризация полученных в ходе работы оптимизатора положений, а выбор лучших моделей осуществляется сначала независимо в рамках каждого из кластеров, а затем осуществляется аналогичная операция на глобальном уровне. После того, как результаты получены, следует осуществить их вывод в общепринятом текстовом формате хранения информации о структуре белков – PDB.

Полученные модели могут использоваться для анализа другими системами, которые определяют вероятность проявления тех или иных биофизических свойств, способностей к связям, действиям и т. п., либо могут быть загружены в специализированное программное обеспечение для графического отображения комплексов (например, чтобы визуально оценить, насколько та или иная конфигурация возможна, ведь, как уже говорилось, точной оценки правдоподобия для моделей не существует).

Рассмотрим далее некоторые из этапов алгоритма более подробно.

2.3 Определение вероятного интерфейса взаимодействия

Данный этап был ранее рассмотрен в рамках дипломной работы [1]. Приведём далее основные алгоритмы, которые возможно эффективно использовать для обнаружения интерфейсов. Отметим, что имеет смысл использовать комбинацию из различных алгоритмов для повышения точности – объединять регионы, на которые указал хотя бы один из них. Поскольку ложные срабатывания лишь увеличивают время обработки, но не уменьшают итоговое качества, а лишь наоборот – дают шанс его повысить, то данный подход кажется разумным.

Наиболее простым из алгоритмов на данной фазе является алгоритм, который основывается в своём предсказании исключительно на данных о последовательности аминокислотных остатков – двухэтапный алгоритм, где первым этапом является SVM над вектором признаков из типов соседних остатков, а вторым этапом – фильтр шумов на основе байесовской модели. Идея алгоритма зиждется на статистике – как показали исследования, принадлежащие к белковому интерфейсу аминокислотные остатки, в основном, образуют кластеры в белковой последовательности.

Ещё один способ решения задачи – алгоритм PredUs [22]. На первом этапе работы для каждого находящегося на поверхности белка аминокислотного остатка осуществляется выделение 31 признака, которые можно разделить на две подгруппы. Первая подгруппа – площадь доступной поверхности белка (рисунок 2.2). Данная характеристика вычисляется для рассматриваемого аминокислотного остатка и 14 его ближайших соседей, образуя, таким образом, 15 различных признаков. Для выбора ближайших соседей в данном алгоритме используется отношение пространственной близости, т. е. ближайшие соседи – другие аминокислотные остатки, а в качестве меры расстояния принимается минимальное из расстояний до входящих в их состав атомов.

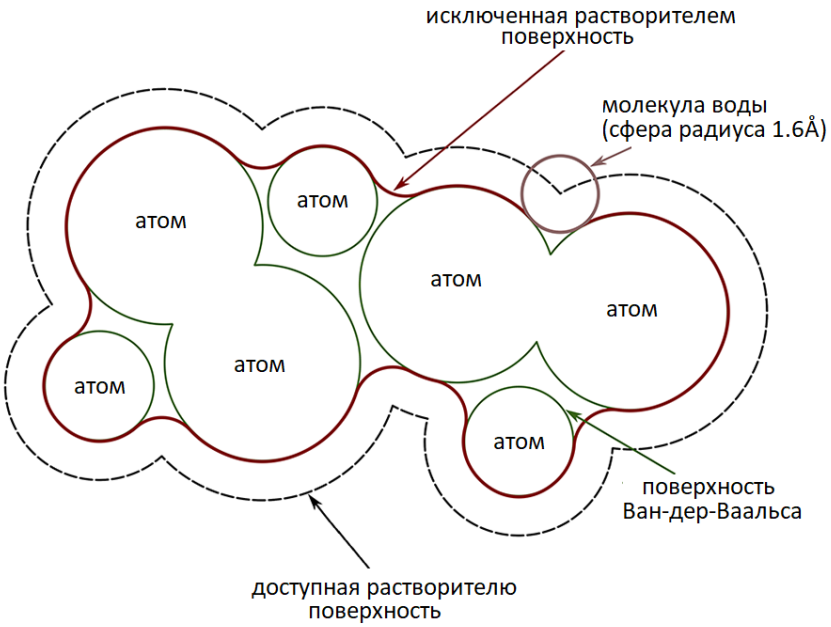

Рисунок 2.2 – Схема поверхности белковой молекулы в разрезе

Вторая группа признаков – частота встречаемости в интерфейсах аминокислотных остатков соответствующего типа. Данная характеристика также образует группу из 15 признаков: для рассматриваемого остатка и 14 наиболее пространственно близких к нему. Последним признаком является наибольшая из частот взаимодействия среди всех аминокислотных остатков обрабатываемого белка.

На втором этапе полученные наборы признаков обрабатываются методом опорных векторов: первым делом происходит переход к векторам высокой размерности с использованием радиальной базисной функции, а далее осуществляется попытка построения разделяющей гиперплоскости в полученном векторном пространстве.

В качестве меры вероятности принадлежности остатку используется расстояние до построенной гиперплоскости (расстояние берётся со знаком,

который определяет по какую из сторон от плоскости расположен каждый из векторов). По умолчанию, в PredUs все аминокислотные остатки, получившие положительную меру вероятности, считаются принадлежащими интерфейсу, однако порог принадлежности, при желании, может быть скорректирован пользователем.

Одним из наиболее новых алгоритмов поиска белок-белковых интерфейсов является алгоритм PAIRPred [17]. Данный алгоритм использует как пространственные характеристики, так и информацию о первичной белковой структуре для осуществления предсказания взаимодействия аминокислотных остатков.

Первая группа используемых признаков базируется на информации о площади доступной растворителю поверхности белковой молекулы. Второй характеристикой аминокислотных остатков является их глубина относительно поверхности белка – минимальное из расстояний от атомов рассматриваемого остатка до какого-нибудь из атомов, расположенных на поверхности белковой молекулы.

Следующая группа признаков основывается на работе Томаса Хамелрика, который показал, что геометрические и физико-химические свойства регионов белка в направлении боковой цепи и противоположном направлении могут в значительной степени отличаться. Каждый отдельный признак из данной группы равен количеству аминокислотных остатков конкретного типа, расположенных в соответствующей подгруппе – полусфере радиуса 8.0 Å.

Четвёртая группа признаков называется «индекс выступа». Для их определения вокруг каждого из атомов белковой цепи строится сфера радиуса 10.0 Å, подсчитывается объём свободного от каких-либо атомов пространства, а затем для каждого аминокислотного остатка в цепи происходит агрегация значений, полученных для входящих в его состав атомов, и нормализация этих значений.

Последняя группа признаков основана на информации о первичной структуре белка. Используя алгоритм PSI-BLAST, на основе информации из баз данных о белках было выполнено вычисление «позиционно-зависимой оценивающей матрицы» (Position Specific Scoring Matrix, PSSM) и «позиционно-зависимой частотной матрицы» (Position Specific Frequency Matrix, PSFM). Далее, методом скользящего окна шириной в 11 аминокислотных остатков происходит вычисление 220-мерного вектора признаков, который получил название «вектор профильных признаков».

2.4 Поиск оптимального положения лиганда

Алгоритм роя частиц

Для определения оптимальных положений лиганда мы можем использовать различные методы оптимизации. Примерами таких методов являются метод роя частиц (Particle Swarm Optimization, PSO) [14] и метод роя светящихся червей (Glowworm Swarm Optimization, GSO) [11]. Методы подобного рода изначально использовались для имитации социального поведения, но впоследствии была замечена возможность их применения и для других оптимизационных задач. Одним из важных моментов является то, что метод роя, не требует знание градиента оптимизируемой функции, а, значит, отлично применим к решаемой нами задаче. Метод основан на идее оптимизации функции путём поддержания набора возможных решений, называемых частицами, и перемещения этих частиц в пространстве решений согласно принципу наилучшего найденного, которое постоянно изменяется при нахождении частицами более выгодных положений. В нашем случае в качестве частиц роя выступают положения лиганда, наиболее оптимальное из которых мы и хотим найти.

Отметим, что хоть в данный момент в рассмотрении и находится лишь данный метод оптимизации, однако создаваемая система будет иметь возможность для задания собственного алгоритма оптимизации и проведения экспериментов в данном направлении.

Первым этапом работы алгоритма PSO является генерация начальных положений лиганда. В общем случае происходит случайная генерация начальных положений с равномерным распределением вокруг рецепторного белка (для покрытия всей области возможных решений), однако область поиска может быть сужена при наличии подсказок о возможных местах взаимодействия (белковых интерфейсах). Возможность задания данной области или алгоритма её нахождения также присутствует в разрабатываемой системе.

После завершения генерации начальных положений запускается непосредственно итеративная фаза исследования покрываемого пространства. Каждая итерация состоит из следующих шагов:

1. Пересчёт оценки для каждого положения из рабочего набора.
2. Определение того, какие изменения положений будут совершены.
3. Применение изменений.

Общей идеей перехода между итерациями является принцип, согласно которому все положения будут с большей вероятностью изменяться в сторону тех своих «соседей», которые имеют больший шанс попадания в экстремум (тех, для которых оценочная функция имеет большее значение). Тем не менее, по

аналогии с другими эвристическими подходами (напр., методом отжига) переход в сторону лучшего соседа не гарантирован и остаётся вероятность движения в сторону менее перспективных кандидатов.

Под «соседями» (N_i^{iter}) в предыдущем абзаце понимаются другие положения из рабочего набора, находящиеся на расстоянии, меньшем определённого значения ($r_{ij}^{iter} < r\_max_i^{iter}$). При этом количество рассматриваемых на одном шаге «соседей» также ограничено сверху некоторым количеством (N_max) для каждого состояния.

Существует несколько способов подсчёта расстояния между состояниями из набора, сравнение целесообразности применения которых также является одной из задач. Так, можно использовать разницу между центрами минимальных покрывающих фигур (напр., минимальных по объёму эллипсоидов, во внутреннюю часть которых входят абсолютно все атомы из лиганда) или метрики, базирующиеся на среднеквадратичном отклонении между атомами двух лигандов. Применение расстояния между центрами минимальных покрывающих эллипсоидов позволяет не учитывать при определении расстояния величину поворота модели, что положительным образом сказалось на качестве работы алгоритма.

Ограничение расстояния $r\_max_i^{iter}$ не является константой, а изменяется динамически по формуле

$$r\_max_i^{iter+1} = \max\{0, \min\{r\_max^{limit}, r\_max_i^{iter} + \beta\left(n^{iter} - \left|N_i^{iter}\right|\right)\}\},$$

где n^{iter} – общее количество различных положений в рабочем наборе на данной итерации. Данная величина также ограничена сверху лимитом в 20 Å (при использовании минимальных покрывающих эллипсоидов в качестве метрики расстояния).

После определения списка «соседей» для каждого положения лиганда происходит определение направления изменения, которое пропорционально разнице значений оценочных функций. Если говорит конкретнее, то переход от положения с номером i в сторону положения с номером j происходит с вероятностью

$$\mathbb{P}[i \to j]^{iter} = \frac{score_j^{iter} - score_i^{iter}}{\sum_{k \in N_i^{iter}}\left(score_k^{iter} - score_i^{iter}\right)}.$$

При этом, если вектор разницы между положениями под номерами i и j обозначить как Δ_{ij}^{iter} , то величина смещения в сторону положения j будет составлять $\alpha * \Delta_{ij}^{iter}$, где $\alpha \in (0, 1)$.

По аналогии с формулой изменения лимита расстояния для «соседства», значение оценки также используется в данном алгоритме не напрямую, а обладает несколько замедляющим изменения эффектом памяти:

$$score_i^{iter+1} = \omega * score_i^{iter} + \varphi * \mathrm{S}(state_i^{iter}),$$

где $\mathrm{S}(state_i^{iter})$ – применяемая функция оценки совместимости рецепторного белка и лигандного белка в положении под номером i, параметр ω определяет скорость замедления активности роя, а φ – влияние полученной оценки правдоподобия на притягательную способность конкретной частицы.

Оценка положения может производится с помощью любой оценочной функции, например, из рассмотренных ранее. Для гарантии сходимости к одному или нескольким наиболее вероятным экстремумам следует на всех итерациях использовать одну и ту же оценочную функцию (хотя чередование оценочных функций также вызывает интерес и планируется, что данный вопрос будет изучен в ходе дальнейшей работы).

Величины α, β, ω, φ, N_max, а также начальные значения для динамически изменяемых между итерациями параметров являются одним из предметов для дальнейшего исследования. Хотя оптимальные значения могут варьироваться между различными входными данным, в качестве значений по умолчанию (на которых в среднем результат неплох) были использованы значения $\beta = 0.2,\ \alpha = 0.5,\ \omega = \varphi = 0.6$, а в качестве лимита на количество соседей использовалось значение 7.

При использовании наиболее простой модели белок-белкового докинга, положение лиганда можно описать семью параметрами, три из которых будут определять смещение лиганда относительно своего начального положения, а другие четыре – его поворот в пространстве кватернионов. Кватернионы довольно широко применяются в вычислительной математике при работе с трёхмерными структурами, поэтому имеет смысл попробовать применить их и в данной работе. Такой подход позволит избежать некоторых возможных проблем, которые могут возникнуть при использовании эйлеровых углов поворота или полярной системы координат.

Изменение положения в данном случае будет также разделено на две части: для вектора смещения будет выполнено обычное сложение с $\alpha * \Delta_{ij}^{iter}$, а для углов поворота будет выполнена сферическая линейная интерполяция (spherical linear interpolation, SLERP) [10].

Теоретически, возможно применение и других способов описания положений. Например, помимо смещения и вращения лиганда можно также учитывать возможности деформации обоих белков при их стыковке, что позволит получить более точные результаты докинга для большего количества белок-белковых комплексов.

Генетический алгоритм

Альтернативным способом поиска решений, также основанном на случайностях, являются генетические алгоритмы. Как и ранее, параметрами, для которых осуществляется оптимизация, являются координаты центра лиганда и угол его поворота. Данный алгоритм является итеративным, при этом на каждом этапе выполняется несколько последовательных шагов. В основе заложена идея применения принципов естественного отбора из природы для оптимизации каких-либо функций.

Генетические алгоритмы оперируют с такой сущностью, как популяция – множество индивидуумов, имеющих свою ДНК – массив параметров, характеризующих конкретный вариант решения. Изначально для инициализации популяции применяются случайные числа из некоторого заданного диапазона.

Первым этапом при переходе к следующему поколению (следующей итерации) является кроссинговер – скрещивание индивидуумов. В ходе скрещивания, получается новый индивидуум, вектор параметров которого представляет собой объединение ДНК тех индивидуумов, которые его породили. При этом под объединением понимается то, что для каждой ячейки потомка независимо случайным образом выбирается, от какого из двух родителей этот параметр будет скопирован. Пары индивидуумов для скрещивания выбираются случайным образом из текущей популяции. Описанное действие повторяется некоторое количество раз, формируя новую популяцию некоторого заданного размера.

Следующим шагом является выполнение мутаций. Для каждого индивидуума случайным образом с некоторой фиксированной вероятностью выбирается, будет ли осуществлена его мутация. Если ответ на предыдущий вопрос положительный, то случайным образом определяется, в какой из ячеек ДНК будет выполнено данной изменение, и какое именно изменение следует осуществить.

Третьим основным этапом при переходе между итерациями является проведение отбора лучших решений. Для каждого индивидуума производится оценка его жизнеспособности – вычисление целевой функции, которую мы оптимизируем. Далее, одним из ряда способов, на основе этих оценок принимается решение, какие же из кандидатов перейдут в новую популяцию, а какие будут отброшены. Среди популярных стратегий можно отметить «выбрать X% лучших, а остальных – случайным образом из оставшихся». Другим популярным способом отбора является метод имитации отжига, вероятность движения в случайном направлении, в котором зависит от того, насколько долго

мы уже осуществляем наш итерационный процесс. Если точнее, то вероятность перехода от объекта *x* к *x** вычисляется по формуле

$$P(\overline{x^*} \to \overline{x_{i+1}} \mid \overline{x_i}) = \begin{cases} 1, & F(\overline{x^*}) - F(\overline{x_i}) < 0 \\ \exp\left(-\dfrac{F(\overline{x^*}) - F(\overline{x_i})}{Q_i}\right), & F(\overline{x^*}) - F(\overline{x_i}) \geqslant 0 \end{cases}$$

В связи с тем, что данный алгоритм является эвристическим, гарантировать нахождение оптимумов во всех случаях мы не можем. Тем не менее, согласно результатам исследований, применение такого подхода для белок-белкового докинга может давать хорошие результаты.

Другие оптимизационные алгоритмы

Помимо рассмотренных выше способов, имеются исследования применимости таких методов как геометрическое сопоставление, геометрическое распознавание на основе быстрого преобразования Фурье, скрытые Марковские модели и другие [21].

2.5 Оценка правдоподобия модели

В общем случае, функции оценки правдоподобия делятся на две категории – те, что основаны на измерении каких-то физико-химических характеристик, и те, что основаны на знаниях о других структурах.

Начнём рассмотрения вариантов решения задачи данного этапа с функции оценки, которая была применена в алгоритме PyDock. Для оценки правдоподобия комплекса первым делом для каждого атома рецептора и лиганда определяется ближайший атом в другой структуре (по евклидову расстоянию между их центрами) и вычисляется ряд значений:

- суммарная энергия электростатического взаимодействия комплекса – все пары атомов, находящиеся ближе, чем в 30 ангстремах, считаются взаимодействующими и дают вклад в размере $(E_{rec} + E_{lig}) / D^2$, где E_{rec} и E_{lig} – заряды соответствующих атомов рецептора и лиганда, а D – расстояние между ними;
- суммарная энергия взаимодействия через метрики Ван-дер-Ваальса атомов;
- площадь контактной поверхности (через площади доступной растворителю поверхности и расстояние между парами атомов).

На основании полученных значений вычисляется метрика как простая сумма упомянутых показателей с некоторым заранее фиксированными коэффицентами.

Оценочная функция HawkRank [7] представляет собой линейную взвешенную сумму Ван-дер-Ваальсовых потенциалов притяжения и отталкивания, электростатического притяжения и отталкивания, а также потенциалов десольватации. Значение метрики HawkRank может быть вычислено по формуле

$$S_{HawkRank} = \omega_{vdW_{attr}} \Delta E_{vdW_{attr}} + \omega_{vdW_{repu}} \Delta E_{vdW_{repu}} + \\ + \omega_{elec_{attr}} \Delta E_{elec_{attr}} + \omega_{elec_{repu}} \Delta E_{elec_{repu}} + \\ + \omega_{pdsol} \Delta E_{pdsol},$$

где ΔE - значение потенциалов, а ω - соответствующий вес потенциалов.

Как показано в уравнении ниже, потенциалы Ван-дер-Ваальса разбиты на привлекательную и отталкивающую части. Отталкивающая часть рассчитывалась по линейной формуле для уменьшения потенциальных локальных столкновений, вызванных дискретными ротамерами боковых цепей и фиксированной основной цепочкой, а привлекательная часть была определена как традиционная формула классического потенциала Леннарда-Джонса. Для повышения вычислительной эффективности рассчитываются только Ван-дер-Ваальсовы взаимодействия между атомами с расстоянием менее 12 Å.

$$E_{attr} = \varepsilon_{ij} \left\{ \left(\frac{\sigma_{ij}}{r_{ij}} \right)^{12} - 2 \left(\frac{\sigma_{ij}}{r_{ij}} \right)^{6} \right\} \quad \text{for } 0.89\sigma_{ij} < r_{ij} < 12$$

$$E_{repu} = 10.0 \left(1 - \frac{r_{ij}}{0.89\sigma_{ij}} \right) \quad \text{for } 0.89\sigma_{ij} > r_{ij}$$

где i и j – номера атомов, r_{ij} – межатомное расстояние между ними, σ_{ij} – сумма атомных радиусов, ε_{ij} - глубина скважины в силовом поле (Amber ff14SB force field) [16].

Для расчёта электростатических потенциалов используется формула

$$E_{elec} = 332 \frac{q_i q_j}{\varepsilon r_{ij}}$$

где i и j – атомы, r_{ij} – межатомное расстояние между ними, q_i и q_j – парциальные заряды атомов, создаваемые силовым полем. Диэлектрическая проницаемость ε была взята равной 1. Электростатические потенциалы также разбиты на притягивающую и отталкивающую части. Если E_{elec} меньше нуля, то он будет добавлен в потенциалы электростатического притяжения, в противном случае – в потенциалы электростатического отталкивания.

Для вычисления потенциалов сольватации несвязанного рецептора, несвязанного лиганда и комплекса использовалась модель на основе SASA (Solvent Accessible Surface Area). Для расчёта десольватации использовалась

разница полярной свободной сольватационной энергии комплекса и суммы энергий для отделённых рецептора и лиганда:

$$\Delta E_{pdsol} = psol_{SASA(com)} - \left(psol_{SASA(lig)} + psol_{SASA(rec)}\right)$$

В качестве функции подсчета правдоподобия в HADDOCK [6] используется линейная комбинацией различных энергий и скрытой площади поверхности. Формула будет немного различаться в зависимости от рассматриваемой фазы:

- Жёсткотельный докинг

$$S_{haddock-it0} = 0.01 * E_{vdW} + E_{elec} + E_{desol} + 0.01 * E_{air} - 0.01 * \text{BSA}$$

- Полугибкое уточнение

$$S_{haddock-it1} = E_{vdW} + E_{elec} + E_{desol} + 0.1 * E_{air} - 0.01 * \text{BSA}$$

- Явное уточнение растворителя

$$S_{haddock-water} = E_{vdW} + 0.2 * E_{elec} + E_{desol} + 0.1 * E_{air}$$

Формулы выше используют следующие величины:

1) E_{vdW} – Ван-дер-Ваальсова межмолекулярная энергия;
2) E_{elec} – электростатическая межмолекулярная энергия;
3) E_{desol} – энергия десольватации;
4) E_{air}– энергия ограничения расстояния;
5) BSA – скрытая площадь поверхности.

Кроме того, в работе упоминаются следующие дополнительные показатели, которые также можно использовать:

E_{rg} – радиус энергии ограничения вращения;
E_{sani} – прямая энергия сдерживания RDC;
E_{vean} – энергия удержания угла межвекторной проекции;
E_{pcs} – энергия сдерживания псевдоконтактного сдвига;
E_{dani} – диффузионная энергия анизотропии;
E_{cdih} – энергия удержания двугранного угла;
E_{sym} – энергия ограничения симметрии (NCS и C2 / C3 / C5);
ΔE_{int} – энергия связи ($E_{complex}$ – Sum[$E_{components}$])

Существуют также гибридные подходы, которые осуществляют смешивание результатов различных оценочных функций. В последнее время из данного направления стало популярным использование небольшой полносвязной нейронной сети для определения итоговой оценки по результатом каждой из функций.

Как будет сказано далее, наилучшие результаты среди рассмотренных функций показал способ оценки под названием DFIRE [15]. Первым делом,

вычисляется потенциал взаимодействия между каждой парой атомов белков (один из элементов пары – атом рецептора, другой – атом лиганда) по следующей формуле:

$$\bar{u}(i,j,r) = \begin{cases} -\eta RT * \ln \dfrac{N(i,j,r)}{\left(\frac{r}{r_{cut}}\right)^{\alpha}\left(\frac{\Delta r}{\Delta r_{cut}}\right) \quad N(i,j,r_{cut})}, & r < r_{cut} \\ 0, & r \geq r_{cut} \end{cases}$$

где i,j – типы выбранных атомов, r – расстояние между ними, R – универсальная газовая постоянная, приблизительно равная 8,31446261815324 Дж / (моль·К), T = 300K, η = 0.0157, α = 1.61, $r_{cut} = 14.5Å$, Δr – размер «корзины расстояний», а N(i, j, r) – количество пар из атомов (i, j), расположенных на расстоянии, меньшем r. Под «корзиной расстояний» понимается способ дискретизации значения расстояния, когда расстояние относится к одному из набора значений: если расстояние меньше 2 Å, то это первая «корзинка», далее расстояния между 2 Å и 8 Å разделяются с интервалом в 0.5 Å, а от 8 Å до 15 Å – с интервалом в один ангстрем. Значения параметров были подобраны авторами данной функции оценки экспериментально. Если же N(i, j, r) = 0, то в качестве потенциала используется значение 10η, либо 5η (по выбору пользователя).

После того, как все межатомные потенциалы вычислены, выполняется вычисление общего потенциала связи молекул. Эта величина равно среднему арифметическому описанных выше потенциалов по всем парам атомов в интерфейсе (тем парам атомов, для которых расстояние меньше порогового значения).

2.6 Кластеризация

Кластеризация занимает важное положение в описанной цепочке действий, независимо от функции оценки и первоначальных подсказок, ибо она позволяет избавится от излишних решений и тем самым внести большее разнообразие в выдаваемый на выходе протокола список наиболее вероятных структур для докинга.

В рамках данной работы применяется простая последовательная алгоритмическая схема (Basic Sequential Algorithmic Scheme) [29]. Согласно ей, мы, первым делом, выделяем представителя первого кластера (и сразу добавляем его в ответ) – лучшее с точки зрения показателя энергии положение. Далее мы последовательно обрабатываем все положения по мере увеличения энергии связи. Для каждого нового положения мы сначала пробуем отнести его к одному из существующих кластеров и в случае успеха просто добавляем наше положение к нему, а в противном случае – создаём новый кластер и добавляем

рассматриваемое положение лиганда в ответ. Считается, что два положения лиганда достаточно близки друг другу для отнесения к одному кластеру тогда и только тогда, когда среднеквадратичное расстояние между атомами в них не превышает пороговое значение в 4 Å.

Разумеется, в дальнейшем возможно применение и других алгоритмов кластеризации и оценки их качества, и соотношения «качество/скорость».

ГЛАВА 3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ СИСТЕМЫ

3.1 Общая структура системы

Для построения системы был выбран язык C++, в связи с высокой вычислительной нагрузкой и установкой на оптимизацию скорости работы системы. Каждая концептуально отдельная часть системы вынесена в исходном коде отдельно и будет рассмотрена в отдельном пункте. Имеется возможность с помощью единообразного интерфейса расширять систему и добавлять в неё новые алгоритмы.

3.2 Предобработка

Наименьшей единицей, с которой возможна работа в системе, является атом (*Atom*). Каждый атом имеет своё имя в соответствии с общепринятой международной номенклатурой. В общем случае имя состоит из трёх соединённых в одно слово частей – символьного кода химического элемента, указателя удаления атома от центра аминокислотной группы вдоль боковой цепи и цифровой идентификатор ветви в боковой цепи. Так, альфа-атом углерода для аминокислотной группы имеет имя CA (Carbon Alpha), соседствующие с ним в основной цепи атомы углерода, кислорода и азота имеют имена C, O и N. Примеры именования атомов в боковых цепях можно увидеть на изображении, приведённом ниже (рисунок 3.1).

Для каждого атома также доступны характеристики, определяемые химическим элементом, атом которого рассматривается. Наиболее важной характеристикой здесь является масса атома. Как ни странно, для некоторых атомов точного значения их массы найти нельзя, ибо для них существуют различные стабильные изотопы. В системе используются значения атомных масс, взятые из периодической системы химических элементов, размещённой на сайте Международного союза теоретической и прикладной химии (IUPAC).

Единая система кодирования имён атомов и аминокислотных остатков важна, так как некоторые этапы могут использовать сторонние системы, либо просто загружать предпросчитанные данные, содержащие данные обозначения.

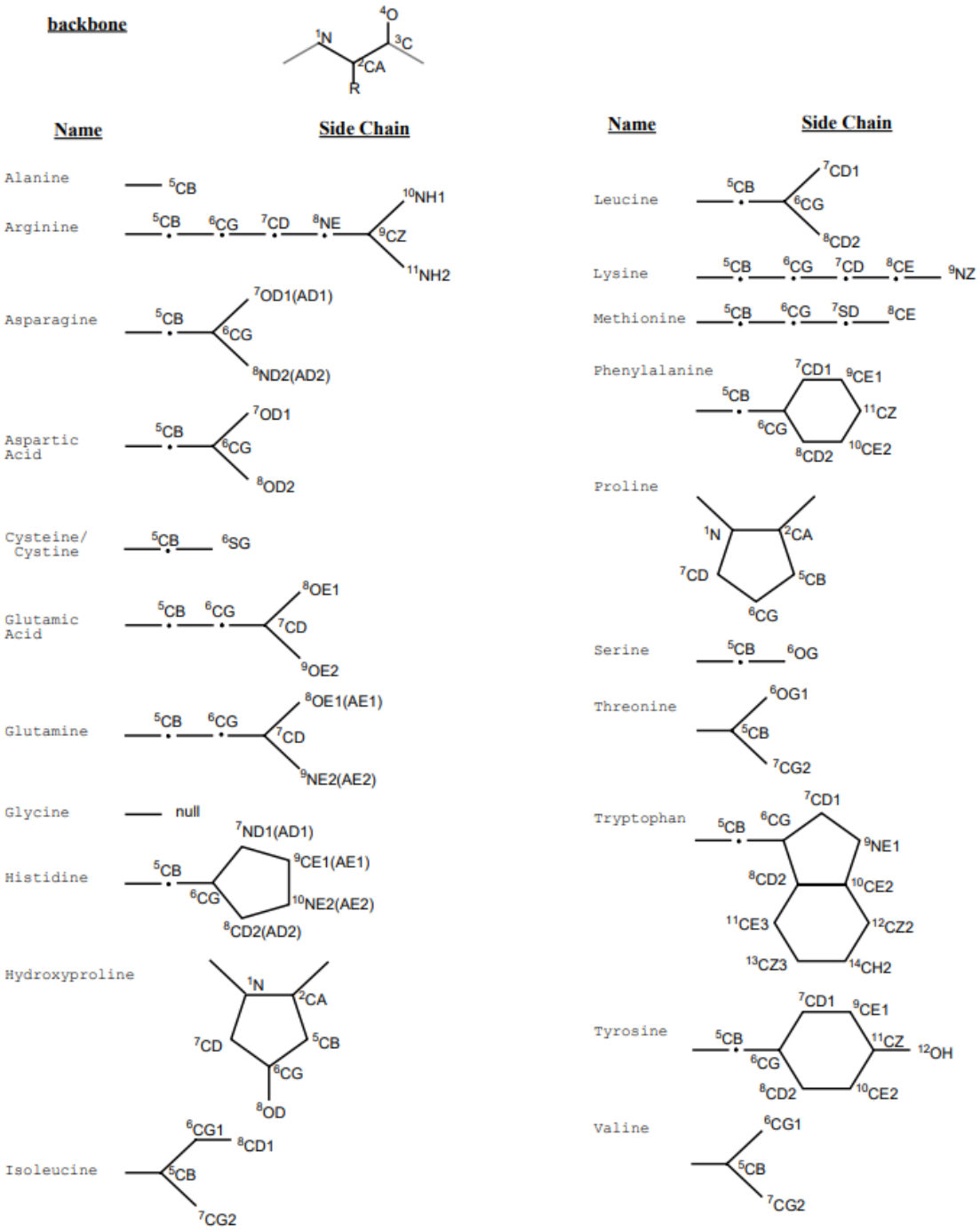

Рисунок 3.1 – Именование атомов в боковых цепях для различных аминокислотных остатков.

Другими важными характеристиками, являются трёхмерные координаты и фактор Дебая-Уоллера (также известный как B-фактор или температурный фактор). B-фактор является величиной, которая характеризует влияние тепловых колебаний кристаллической решётки на некоторые процессы, связанные с тепловым излучением.

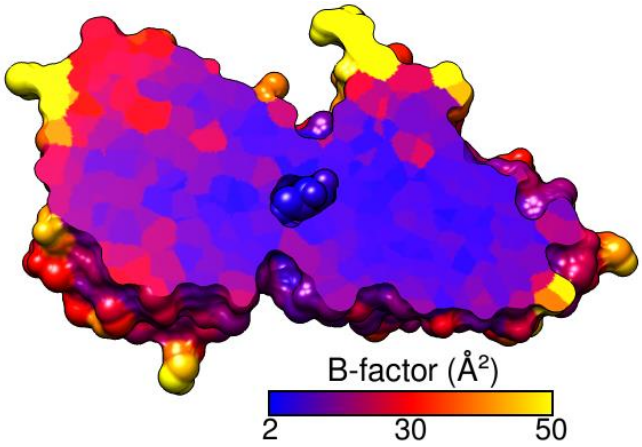

Рисунок 3.2 – Визуализация значения B-фактора
для атомов комплекса 2GBP

Атомы группируются в объектах класса *Residue*, каждый из которых соответствует аминокислотным остаткам. Данный класс представляет из себя просто контейнер, который также скрывает в себе реализацию часто используемых операции (напр., получение α-углерода, подсчёт центра масс атомов в аминокислотной группе, поиск ближайшего атома к заданному).

Аминокислотные группы, в свою очередь, формируют цепочки из последовательно идущих элементов (*Chain*), а белковые цепи формируют модель (*Model*). Помимо доступа к составляющим их объектам, данные классы предоставляют также удобный интерфейс для поворота и сдвига цепи/комплекса в трёхмерном координатном пространстве, подсчёт центра масс и т. п.

Для обеспечения взаимодействия с международными базами данных и другими программами анализа белков, в системе имеется также вспомогательные функции, позволяющие считывать и записывать структуры в формате PDB, осуществлять конвертацию между различными системами кодирования (например, существует два способа кодирования типа аминоксилотного остатка – однобуквенный и трёхбуквенный).

Помимо задания структур для внутреннего представления белков, ввода-вывода и аннотирования вспомогательной информацией, на данном этапе также происходит центрирование входных моделей. Для осуществления этого сначала для каждой из двух входных моделей происходит определение ограничивающего её прямоугольника (нахождение минимумов и максимумов среди всех атомов модели по каждой из трёх координат), а затем осуществляется сдвиг начала координат на среднее из этих двух величин. Разумеется, белковые молекулы могут быть сильно ассиметричны и иметь выступающие части,

однако, в любом случае, данная операция приводит к тому, что точка с координатами (0,0,0) будет находится примерно в центре молекулы.

3.3 Функции оценки правдоподобия

Как уже упоминалось ранее, в хоте работы алгоритма мы проводим оптимизацию некоторой функции правдоподобия. Каждая функция задаётся отдельным классом, объекты которого при необходимости могут хранить единожды проинициализированные общие значения. Все реализованные функции реализуют одинаковый интерфейс ScorerInterface (рисунок 3.3). При добавлении новых методов оценки требуется реализовать как минимум метод CalculateScore(…), который принимает в качестве параметров представления в памяти белка-рецептора и лиганда. Опционально можно добавлять конструктор и поля для загрузки при инициализации и дальнейшего хранения некоторых констант или статистических предпросчитанных величин.

```
#ifndef SCORING_SCORER_H_
#define SCORING_SCORER_H_

#include "core/structures/model.h"

class ScorerInterface {
 public:
  virtual ~ScorerInterface() = default;

  [[nodiscard]] virtual double CalculateScore(
      const Model& receptor, const Model& ligand) const = 0;
};

#endif  // SCORING_SCORER_H_
```

Рисунок 3.3 – Интерфейс класса, реализующего функцию оценки правдоподобия модели.

Использование единого интерфейса позволяет несложным образом расширять систему и переключаться между различными функциями оценки при проведении экспериментов.

3.4 Поиск оптимального положения

В общем случае предполагается, что добавление оптимизатора заключается в определении класса для одного решения, механизма их генерации в заданном регионе поиска, непосредственно оптимизационного алгоритма, а также механизма кластеризации.

Минимальный интерфейс, которая должна поддерживать реализация для использования извне, тем не менее, достаточно простой (рисунок 3.4) – достаточно лишь реализовать функцию Solve(…), которая принимает объект-оценщик правдоподобия и две модели – белка-рецептора и белка-лиганда.

В данной работе для демонстрации был пока что реализован лишь описанный во второй главе алгоритм оптимизации на основе роя частиц. В рамках одного из направлений для дальнейшего развития системы можно выделить как раз добавление и исследование других алгоритмов оптимизации: например, можно добавить генетические алгоритмы.

На начальном этапе работы алгоритма происходит разбиение области поиска на прямоугольные области фиксированного размера, в каждой из которых будет запущен отдельный рой частиц. Факт того, что это именно независящие друг от друга подзадачи пригодится нам при рассмотрении следующего пункта.

```cpp
#ifndef SOLVERS_SOLVER_INTERFACE_H_
#define SOLVERS_SOLVER_INTERFACE_H_

#include <vector>

#include "core/structures/model.h"

struct SolverResult {
  double score;
  std::unique_ptr<Model> model;
};

class SolverInterface {
 public:
  virtual ~SolverInterface() = default;

  virtual std::vector<SolverResult> Solve(
      const Model& receptor,
      const Model& ligand,
      std::shared_ptr<ScorerInterface> scorer) = 0;
};

#endif  // SOLVERS_SOLVER_INTERFACE_H_
```

Рисунок 3.4 – Минимальный интерфейс класса, реализующего поиск оптимальной модели.

Как отмечалось в самом начале 2 главы, как правило существует большой дисбаланс между размерами белка-рецептора и белка-лиганда. При реализации данный факт учитывался, в результате чего получился следующий подход: рецептор не меняет своего положения, оставаясь статичным на протяжении всего процесса, в то время как для лиганда как раз и подбираются смещение и угол поворота (оптимизируемые параметры), которые и определяют его положение.

Для получения оценки какого-либо решения осуществляется сдвиг и поворот модели лиганда, а затем запускается алгоритм оценки правдоподобия такого комплекса.

Помимо оценки правдоподобия моделей, довольно трудозатратным является определение того, в каком направлении будет происходить движение частиц. Можно применять некоторые оптимизации на данном шаге, но из-за сильной изменчивости (координаты атомов пересчитываются на каждой итерации), максимальный диапазон «видимости» у частиц также может изменяться.

В последней части для каждого роя происходит кластеризация полученных решений и нахождение экземпляров с наилучшей оценкой в каждом из них. Максимальное расхождение решений, которое всё ещё допустимо для отнесения решений к одному кластеру, передаётся в алгоритм параметром. Кластеризация осуществляется первоначально путём попарного сравнения всех решений, после которого осуществляется выделение кластеров и обработка каждого из них.

3.5 Распределение задач и масштабируемость

Поскольку, как было указано ранее, оптимизация – довольно длительный этап, то естественным вопросом является то, как же провести ускорение данного процесса. Тут нам пригодится знание о том, что, в общем и целом, оптимизационная задача естественным образом разбивается на несколько независимых частей – ведь, например, каждый из роев частиц в реализованном алгоритме функционирует независимо от других.

Для использование данного факта предлагается выполнять распределение вычислительной нагрузки на разные машины. Тогда, как можно заметить, все наши независимые задачи, при наличии достаточного количества машин в кластере, могут быть выполнены параллельно друг с другом. Даже если машин в кластере меньше, чем подзадач, такой подход позволит сократить время работы за счёт максимально эффективного использования имеющихся вычислительных ресурсов.

Взаимодействие вычислительных узлов осуществляется по клиент-серверной модели с выделенным узлом-контроллером. Когда на рабочем узле появляются свободные ресурсы, он подключается к контроллеру и запрашивает новую задачу для выполнения. Если задач в очереди на контроллере на этот момент нет, узел уходит в спячку на некоторый фиксированный период времени (по умолчанию, 1 минута), после чего повторяет описанное действие по запросу новой задачи. После завершения выполнения какой-либо задачи рабочий поток осуществляет передачу результата на контроллер по аналогичной схеме.

Помимо осуществления распределения задач для ускорения обработки, единый контроллер также позволяет использовать один вычислительный кластер для проведения различных экспериментов.

В зависимости от условий использования, возможны различные сценарии применения распределённой системы. Например, если в организации не практикуется отключение рабочих станций на ночное время, то имеет смысл в рабочие часы использовать некоторое количество выделенных для этого серверов для проведения срочных экспериментов, а некоторую часть вычислений осуществлять во время простоя рабочих машин. В целом, можно несложным образом дополнить текущую систему приоритетом задач, так чтобы какие-то срочные эксперименты проводились максимально быстро, а наименее важные выполнялись в фоновом режиме.

Помимо того, что можно разделить задания между машинами, стоит также отметить наличие нескольких потоков-исполнителей на каждой машине, ведь современные процессоры имеют несколько ядер, которые могут также осуществлять параллельную обработку задач. Для этих целей используется локальный менеджер задач, который осуществляет коммуникацию с сервером-контроллером и менеджмент потоков-исполнителей.

Осуществление взаимодействия между контроллером и рабочими узлами, а также между контроллером и пользовательским клиентом (работающим в режиме командной строки), осуществляется при помощи технологии удалённого вызова процедур (RPC) [8]. RPC позволяет программам вызывать функции или процедуры в другом адресном пространстве (которые при этом могут исполняться как на одной, так и на различных машинах). Данная технология позволяет простым с точки зрения программирования способом переложить всю нагрузку по организации сетевого взаимодействия на соответствующую библиотеку, которая будет реализовываться сериализацию данных и работу с сетью. В качестве библиотеки, организующей работу по технологии удалённого вызова процедур, используется библиотека gRPC от компании Google, использующая для осуществления сериализации и задания интерфейса сервера технологию Protocol Buffers [24].

Общая схема системы в приведена на рисунке 3.5.

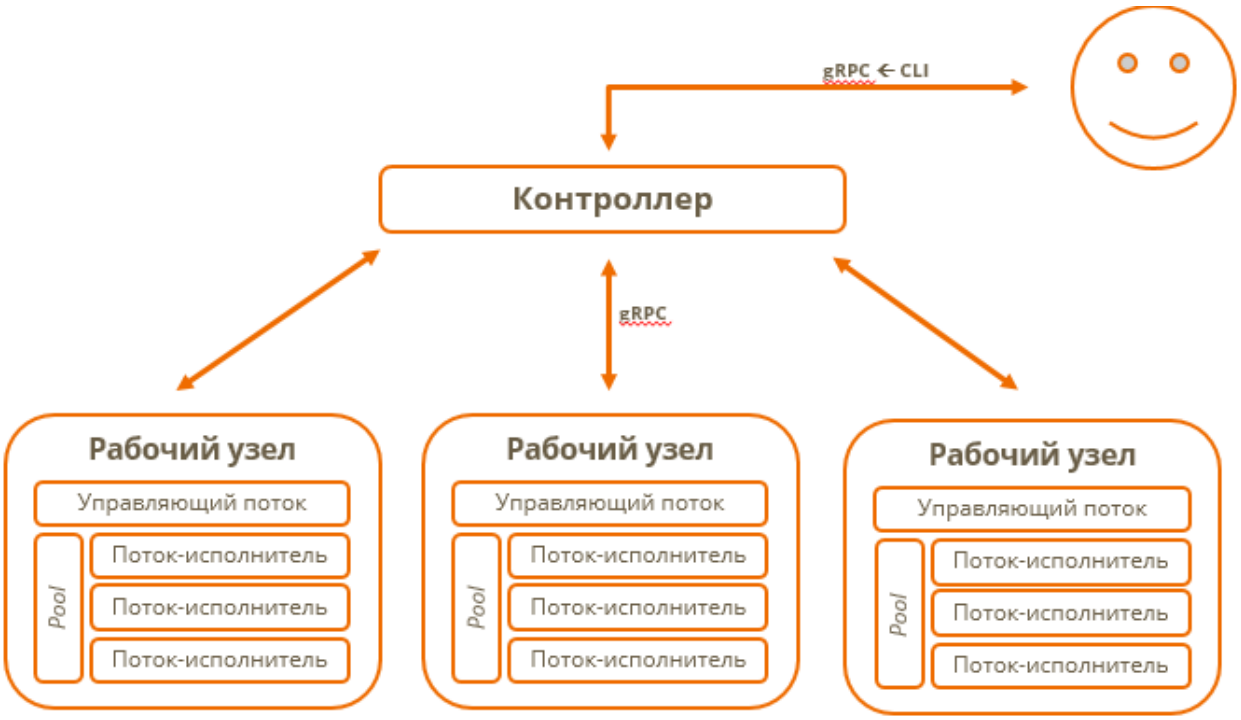

Рисунок 3.5 – Схема разбиения системы на компоненты
и взаимодействия между ними.

3.6 Тестирование системы

Для проверки правильности работы компонентов системы применялась технология юнит-тестирования. Тесты были реализованы с помощью библиотеки Google Test.

Качество протокола оценивалось при помощи специального набора данных «Protein–Protein Docking Benchmark 5.0» [28], который состоит из 230 белковых комплексов. В качестве метрики того, правильно ли выполнено предсказание, использовался принцип наличия близкого к верному комплексу среди лучших 5000 предсказаний. Под «близкими» в данном случае подразумевается положение, в котором среднеквадратичное отклонение расстояний между соответствующими атомами лиганда составляет менее 12 Å. Результаты при использовании DFIRE в качестве метрики правдоподобия и генерации порядка 200 000 частиц для каждого из комплексов (в среднем), составили около 73%. Аналогичная оценка для Топ-10 результатов, разумеется, даёт меньший процент – результат порядка 12%. Данные метрики хуже наилучших из существующих решений, но не сильно, так что использование роя частиц представляет интерес для дальнейших исследований и экспериментов, в особенности с различными функциями оценки.

Производительность распределённой системы пока не измерялось, однако время работы на одном компьютере ясно даёт понять о необходимости и актуальности данной возможности. Тестирование на всех 230 комплексах из

бенчмарка на компьютере с 8-ядерным Intel Xeon (3GHz) и 64 Гб DDR4 заняло порядка 3 суток непрерывной работы. В силу того, что используемый алгоритм легко масштабируется, разработанный протокол представляет собой интерес для организации быстрой оценки комплексов в случае доступности вычислительного кластера.

3.7 Направления дальнейшего развития

В дальнейшем планируется расширение возможностей системы путём добавления новых функций оценки правдоподобия моделей. Особый интерес представляют также эксперименты с различными смешанными функциями оценки. Этим летом планируется также провести тестирование производительности распределённого варианта системы и сравнения скорости работы реализации данного протокола с другими протоклами, большая часть которых реализована на более медленном, по сравнению с C++, языке Python (так что ожидаются положительные результаты от этого сравнения).

В текущей реализации реализованы лишь алгоритмы, результаты которых основаны на физических и химических характеристиках элементов белковой цепи. Внедрение методов машинного обучения на этапы поиска интерфейсов и оценки правдоподобия моделей в наши дни также представляется интересным направление для дальнейшего развития системы.

ЗАКЛЮЧЕНИЕ

В процессе подготовки данной диссертации была выполнена работа над решением проблемы белок-белкового докинга. Задача сведения воедино разрозненной информации о различных алгоритмах и этапах данного процесса выполнена в полном объёме. Разработан единый алгоритм действия для проведения всего процесса докинга целиком. Кроме того, реализована легко расширяемая и масштабируемая система, позволяющая простым образом проводить эксперименты с каждым из отдельных этапов протокола (что представляет практическую ценность для работающих в направлении биоинформатики людей), либо использовать систему для предсказания вероятных моделей комплексов для исследователей-биохимиков. Тестирование показало относительно неплохие результаты качества построенного протокола, что повышает интерес к дальнейшим экспериментам в данном направлении.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ильин, А. В. Поиск белковых интерфейсов / А. В. Ильин // Белорусский государственный университет – 2018, Минск.
2. Машинное обучение: курс лекций / К. В. Воронцов // Школа анализа данных [Электронный ресурс] – 2016-2017. – Режим доступа: https://wiki.school.yandex.ru/shad/MachineLearning – Дата доступа: 06.02.2018
3. Структурная Биоинформатика в ШАД: курс лекций / А. В. Головин // Школа анализа данных [Электронный ресурс] – 2012-2014. – Режим доступа: http://vsb.fbb.msu.ru/projects/edu/wiki/Shad – Дата доступа: 08.02.2018.
4. Финкельштейн, А. В. Введение в физику белка / А. В. Финкельштейн // Учебный центр Института белка РАН [Электронный ресурс] – 1999-2000. – Режим доступа: http://phys.protres.ru/lectures/protein_physics – Дата доступа: 11.02.2018
5. A series of PDB related databases for everyday needs. / W. G. Touw [et al.] // Nucleic Acids Research. – 2015. – Vol. 43, Database issue. – P. D364- D368.
6. Bonvin, A. HADDOCK2.2 scoring function / A. Bonvin, J. Rodrigues // Bonvin lab [Electronic resource] – 2019. – Mode of access: https://www.bonvinlab.org/software/haddock2.2/scoring/ – Date of access: 12.09.2019
7. Feng, T. HawkRank: a new scoring function for protein–protein docking based on weighted energy terms / T. Feng [et al.] // Journal of Cheminformatics – 2017. – Vol. 9
8. gRPC Framework // The Linux Foundation [Electronic resource] – 2014-2020. – Mode of access: https://grpc.io/ – Date of access: 19.04.2018
9. Heinig, M. STRIDE: a web server for secondary structure assignment from known atomic coordinates of proteins / M. Heinig, D. Frishman // Nucleic Acids Research. – 2004. – Vol. 32 – P. W500- W502.
10. Jafari, M. Spherical Linear Interpolation and Bezier Curves / M. Jafari, H. Molaei // General Scientific Researches. – 2014. – Vol. 2, No. 1. – P. 13-17
11. Krishnanand, K. N. Glowworm swarm-based optimization algorithm for multimodal functions with collective robotics applications / K. N. Krishnanand, D. Ghose // Multiagent and Grid Systems – An International Journal 2. – 2006. – P. 209–222.
12. Kyte, J. A simple method for displaying the hydropathic character of a protein. / J. Kyte, R. F. Doolittle // Journal of Molecular Biology. – 1982. – Vol. 157, No 1. – P. 105-132.
13. Li, X. Detection and refinement of encounter complexes for protein–protein docking: taking account of macromolecular crowding / X. Li [et al.] // Proteins. – 2010. – Vol. 78 – P. 3189–3196.

14. Lindfield, G. Particle Swarm Optimization Algorithms / G. Lindfield, J. Penny // Introduction to Nature-Inspired Optimization – 2017. – P. 49-68
15. Liu, S. Unbound Protein-Protein Docking Selections by the DFIRE-based Statistical Pair Potential / S. Liu, C. Zhang, Y. Zhou [Electronic resource] – 2004. – Mode of access: https://arxiv.org/pdf/q-bio/0406025.pdf – Date of access: 19.04.2020
16. Maier, J. A. ff14SB: Improving the Accuracy of Protein Side Chain and Backbone Parameters from ff99SB / J. A. Maier [et al.] // Journal of Chemical Theory and Computation – 2015. – Vol. 11(8) – P. 3696-3713
17. Minhas F. PAIRpred: Partner-specific prediction of interacting residues from sequence and structure / F. Minhas, B. J. Geiss, A. Ben-Hur // Proteins: Structure, Function, and Bioinformatics. – 2014. – Vol. 82, No 7. – P. 1142-1155.
18. Morrison, J. Quaternion interpolation with extra spins / J. Morrison // Graphics Gems III (IBM Version). – 1992. – P. 96–97.
19. National Center for Biotechnology Information online resources // National Center for Biotechnology Information [Electronic resource] – 2001-2018. – Mode of access: https://www.ncbi.nlm.nih.gov – Date of access: 19.04.2018
20. Neuvirth, H. ProMate: A Structure Based Prediction Program to Identify the Location of Protein–Protein Binding Sites / H. Neuvirth, R. Raz, G. Schreiber // Journal of Molecular Biology. – 2004. – Vol. 338. – P. 181-199.
21. Pagadala, N. S. Software for molecular docking: a review / N. S. Pagadala, K. Syed, J. Tuszynski // Biophysical Reviews. – 2017. – Vol. 9(2) – P. 91-102.
22. PredUs: a web server for predicting protein interfaces using structural neighbors / Q. C. Zhang [et al.] // Nucleic Acids Research. – 2011. – Vol. 39, Web Server issue. – P. W283-287.
23. Protein-Protein Interface Predictions by Data-Driven Methods: A Review / L. C. Xue1 [et al.] // FEBS Letters. – 2015. – Vol. 589, No 23. – P. 3516-3526.
24. Protocol Buffers // Google Inc. [Electronic resource] – 2008-2020. – Mode of access: https://developers.google.com/protocol-buffers/ – Date of access: 19.04.2018
25. PSAIA – Protein Structure and Interaction Analyzer / J. Michel [et al.] // BMC Structural Biology. – 2008. – Vol. 8, Article 21.
26. The Protein Data Bank / H. M. Berman [et al.] // Nucleic Acids Research – 2000. – Vol. 28, No 1. – P. 235–242.
27. The PyMOL Molecular Graphics System // Schrödinger, Inc. [Electronic resource] – 2011-2018. – Mode of access: https://pymol.org – Date of access: 15.03.2018.
28. The Python Molecule Viewer (PMV) // The Scripps Research Institute [Electronic resource] – 2005-2020. – Mode of access: http://mgltools.scripps.edu/packages/pmv – Date of access: 19.04.2020

29. Theodoridis, S. Sequential clustering algorithms / S. Theodoridis, K. Koutroumbas // Pattern Recognition. – Elsevier Academic Press. – 2008. – P. 433–437.
30. UniProt: the Universal Protein knowledgebase / The UniProt Consortium // Nucleic Acids Research – 2017. – Vol. 45, Database issue. – P. D158-D169.
31. Updates to the integrated protein-protein interaction benchmarks: Docking benchmark version 5 and affinity benchmark version 2. / T. Vreven [et al.] // Journal of Molecular Biology. – 2015. – Vol. 427, No 19. – P. 3031-3041.