Применение многомерной квантильной функции в задаче пептид-белок докинга

С.В. Полуян

ГБОУ ВО МО «Университет «Дубна»,
институт системного анализа и управления

Настоящая работа посвящена исследованию применения стохастических эволюционных алгоритмов оптимизации к задаче пептид-белок докинга. Продемонстрированы основные особенности рассматриваемой задачи и возникающие трудности применения эволюционных алгоритмов оптимизации. Предложен способ применения эволюционных алгоритмов, основанный на использовании эмпирической квантильной функции. Описаны схемы применения параллельных вычислений для построения и использования многомерной эмпирической квантильной функции. Проведена реализация предложенного способа с применением вычислений на графических ускорителях. Представлены результаты разнообразных численных экспериментов.

Ключевые слова: глобальная оптимизация, эволюционные алгоритмы, эмпирическая квантильная функция, докинг.

1. Введение

В задаче пептид-белок докинга необходимо найти оптимальное место связывания белка и пептида при взаимодействии друг с другом, а также соответствующую этой связи конформацию комплекса. Традиционными экспериментальными методами определения места связывания и соответствующих конформаций белка и пептида являются кристаллография, ядерный магнитный резонанс, а также другие техники [1]. Несмотря на их точность и эффективность, они требуют значительных лабораторных ресурсов и материальных затрат. Более того, пептид-белок комплексы сложнее кристаллизовать, чем отдельный белок. В то время как докинг менее затратный, это лишь метод предсказания структуры комплекса. В связи с этим вычислительные методы приобретают все большую популярность. Большинство из них на различных этапах работы включает в себя разнообразные методы стохастической оптимизации [2,3]. Одним из основных преимуществ использования стохастических методов оптимизации является возможность напрямую использовать различную статистическую информацию. Кроме того, использование методов оптимизации более привлекательно в вычислительном отношении, чем, например, применение методов молекулярной динамики.

В основе большинства подходов к докингу лежит термодинамическая гипотеза Анфинсена, основное утверждение которой следующее: оптимальное состояние комплекса уникально и находится в глобальном минимуме свободной энергии. Поэтому задача пептид-белок докинга может быть рассмотрена как задача глобальной оптимизации, в которой необходимо найти конформацию комплекса с минимальной энергией.

2. Постановка задачи

В общем случае задачи пептид-белок докинга решаются комбинированными методами, включающими в себя несколько различных по структуре этапов и учитывающих разнообразную статистическую информацию. Такого рода комбинированные предсказания выходят за рамки текущего исследования. В большинстве случаев заключительным этапом является

поиск в полноатомном разрешении оптимальной структуры комплекса в окрестности места связывания, так называемый прямой докинг. Именно на этом этапе используются стохастические методы оптимизации в сочетании с методами локальной оптимизации. Важно отметить, что применяемые на данном этапе методы оптимизации (как глобальной, так и локальной) обладают высокой степенью универсальности относительно решаемой задачи, т.е. структура и параметры алгоритмов, как правило, независимы от сложности целевой функции и соответствующего энергетического ландшафта. Примером, подчеркивающим указанную универсальность, может служить протокол докинга Rosetta FlexPepDock [2], структура и применяемые алгоритмы которого не зависят от фундаментально меняющегося состава стандартной скоринг-функции силового поля.

Необходимо отметить, что на заключительном этапе поиск оптимальной структуры комплекса ведется, как правило, с учетом структурных особенностей [1] предполагаемого места связывания. Здесь необходимо подчеркнуть специфику рассматриваемой задачи. В силу структурых особенностей пептиды обладают высокой гибкостью. Торсионные углы главной цепи каждого аминокислотного остатка пептида являются ротамерами. В связи с этим докинг даже простейших пептидов длиной 2-5 аминокислотных остатка в полноатомном разрешении представляет собой сложную (иногда невыполнимую) задачу даже для специально разработанных пакетов [1].

Поиск в окрестности места связывания довольно просто организовать с помощью методов сэмплирования. Однако, предлагаемые в настоящее время эвристические подходы к глобальной оптимизации, в частности, эволюционные алгоритмы, требуют непрерывного пространства поиска без ограничений, кроме границ поиска для каждого параметра. Возникает вопрос, каким образом не допустить значительного смещения пептида в окрестности области поиска и сохранить непрерывность пространства поиска? При этом избежать грубого подхода с использованием штрафных функций и сохранить условия для прямого применения эволюционных алгоритмов оптимизации. Ответом может послужить применение многомерного квантильного преобразования.

Настоящая работа посвящена исследованию применения эволюционных алгоритмов оптимизации к задаче пептид-белок докинга с использованием квантильного преобразования. При этом ставится задача удержания пептида в некоторой локальной окрестности области поиска.

Задача оптимизации формулируется как задача минимизации энергии связывания (1), которая вычисляется как разница между энергией комплекса в связанном состоянии и энергией в свободном состоянии, т.е. когда белок и пептид друг с другом не взаимодействуют.

$$E_{\text{binding energy}} = E_{\text{complex}} - (E_{\text{protein}} + E_{\text{peptide}}). \quad (1)$$

Взаимодействие между пептидом и белком может быть описано целевой функцией. В численных экспериментах использовалось силовое поле Rosetta 3.8 [4]. Выбор силового поля обусловлен широкой распространенностью и ориентированностью к проблеме пептид-белок докинга. Детальное описание постановки задачи пептид-белок докинга, описание степеней свободы пептида и белка представлено в [5, 6]. В настоящей работе эксперименты проводились с комплексом 1JWG (код PDB) с линейным интерфейсом связывания [1, 7]. Комплекс представлен на рис. 1.

3. Многомерная эмпирическая квантильная функция

Определение многомерной эмпирической квантильной функции (или эмпирического квантильного преобразования) естественно выводится из определения эмпирической функции распределения. Впервые понятие многомерной квантильной функции введено в [8], однако, наиболее распространеное определение приведено в [9]. Здесь будет приведено краткое «рекурсивное» описание структуры квантильной функции.

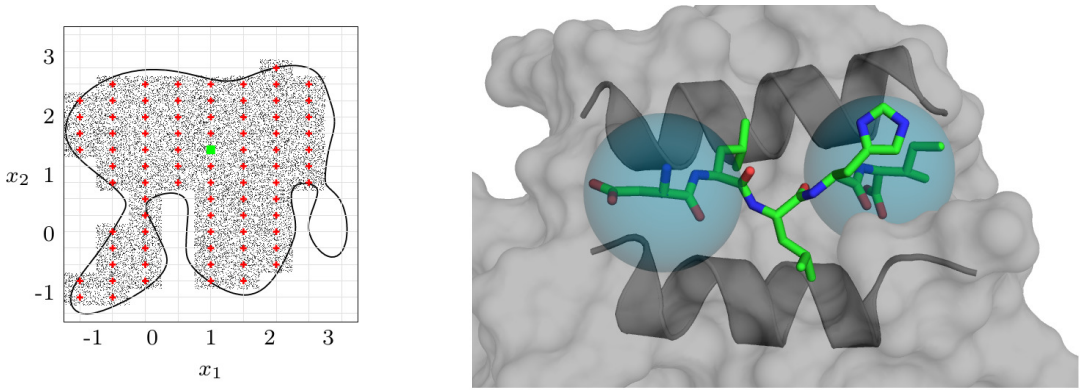

Рис. 1. Непрерывная область поиска, сетка, исходный узел, выборка и $2 \cdot 10^4$ распределенных точек. Стартовая позиция и границы смещения пептида комплекса 1JWG (код PDB).

Пусть дано вероятностное пространство и на нем определена случайная величина X. Функцией распределения случайной величины X назовем функцию $F_X : [0,1] \to \mathbf{R}$, задаваемой формулой $F_X(x) = \mathrm{P}(X \leq x)$. Квантильное преобразование для заданной функции определяется следующей формулой:

$$F_X^{-1}(p) = \inf\{x \in \mathbf{R} : P(X \leq x) \geq p\}. \tag{2}$$

Определим эмпирическую функцию распределения следующим образом:

$$\hat{\mathrm{F}}_X(u) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(\mathbf{x}_{[i]} \leq u), \tag{3}$$

где $\mathbf{1}$ – индикаторная функция.

Важно отметить, что здесь и далее рассматриваются случайные величины, распределённые на сетке. Это связано с двумя факторами. Во первых, в процессе дальнейшего применения построенного преобразования важна только доступность той или иной области поиска, поэтому достаточно равномерного распределения. Во вторых, если значения будут распределены просто равномерно, а не по сетке, то в формуле (3) количество найденных элементов может стать равно нулю.

Процедура использования одномерного квантильного преобразования для непрерывного числа из отрезка $[0,1]$ выглядит следующим образом (см. рис. 2). Для заданной выборки *sample* на сетке *grid* и всех значений в узлах сетки выполняется процедура двоичного поиска необходимого узлового значения сетки. Вначале выбирается значение середины сетки, производится подсчет количества элементов в выборке меньше данного значения середины, которое затем делится на общее количество элементов в выборке. Аналогичные действия производятся для соседнего узла сетки. Затем производится шаг, аналогичный двоичному поиску: если непрерывное значение меньше полученного числа, то меняется верхняя граница поиска по сетке. В противном случае аналогично меняется нижняя граница. Однако, если непрерывное значение больше узлового значения и меньше соседнего значения, то процедура поиска нужного значения сетки заканчивается. Затем для поддержания непрерывности используется линейная интерполяция.

Используя определение многомерной эмпирической функции –

$$\hat{\mathrm{F}}_{X_1,X_2,\ldots,X_d}(u_1,u_2,\ldots,u_d) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(\mathbf{x}_{[i,1]} \leq u_1, \mathbf{x}_{[i,2]} \leq u_2, \ldots, \mathbf{x}_{[i,d]} \leq u_d), \tag{4}$$

где d – размерность, n – размер выборки, можно определить многомерную квантильную функцию $[0,1]^d \to \mathbf{R}^d$. Пусть F – d-мерная функция распределения и $X_1, \ldots, X_n$ – выборка.

```
float getval(vector<float> &sample, vector<float> &grid, float val01){
    size_t l = 0, r = grid.size() - 1, m = 0, c1 = 0, c2 = 0;
    float f1, f2, n = sample.size();
    while(l <= r){
        m = l + (r - l) / 2;
        c1 = 0; c2 = 0;
        for(size_t i = 0, n = sample.size(); i != n; ++i){
            if(sample[i] < grid[m]) ++c1;
            if(sample[i] < grid[m + 1]) ++c2;
        }
        f1 = c1/n; f2 = c2/n;
        if((val01 < f2) && (val01 > f1)) break;
        if(val01 > f1) l = m + 1; else r = m - 1;
    }
    float x0 = grid[m], y0 = f1, x1 = grid[m + 1], y1 = f2;
    return x0 + (val01 - y0) * (x1 - x0) / (y1 - y0);
}
```

Рис. 2. Одномерное квантильное преобразование по сетке с линейной интерполяцией.

Используя одномерное квантильное преобразование (2) и выбрав вектор $z \in [0,1]^n$ можно определить рекурсивно квантильную функцию $Y = \tau_F^{-1}(z)$:

$$Y_1 = F_1^{-1}(z_1), \tag{5}$$

$$Y_k = F_{k|1,\ldots,k-1}^{-1}(z_k|Y_1,\ldots,Y_{k-1}), \quad 2 \le k \le d. \tag{6}$$

3.1. Построение выборки

Построение выборки для квантильной функции выглядит следующим образом. Пептид помещается в произвольную область поиска в окрестности места связывания. Поскольку структура интерфейса связывания известна, пептид располагается в приблизительно линейной структуре. Как указывалось выше, рассматривается только заключительный этап оптимизации, и, в общем случае, структура может быть произвольной. В координатах атома α-углерода первого и последнего аминокислотного остатка пептида создаются области поиска, которые будут определять границы степени свободы для положения выбранных атомов, которые зависят от оптимизируемых параметров. Поскольку положение первого α-углерода определяет смещение пептида относительно белка для отображения в сферу используется трёхмерное квантильное преобразование, построенное по плотности распределения, которое уже использовалось в [5]. Положение пептида в определенной сферами области поиска зависит от параметров смещения пептида, угла и вектора поворота, а также торсионных углов главной цепи пептида. Для каждого параметра создается собственная сетка. Важно отметить, что границы каждого параметра переведены в диапазон $[0,1]$. Параметры, определяющие смещение пептида и часть углов главной цепи пептида, уже находятся в диапазоне $[0,1]$. Остальных параметры переводятся в этот диапазон и преобразуются в искомые с помощью линейной интерполяции. В итоге пространство поиска сведено в единичный гиперкуб.

Для каждого параметра, принадлежащего гиперкубу, производится разбиение на независимое заданное число равных частей, которое будет определять узлы сетки. Для каждого параметра определяется ближайшее значение в полученной сетке. Если область поиска в ближайших узлах сетки не найдена, то происходит поиск в n-мерной окрестности фон Неймана или Мура.

Теперь можно приступить к процедуре построения выборки. Для этого используется n-мерный аналог алгоритма заливки Flood-fill [10] с использованием n-мерной окрестности фон Неймана. Алгоритм Flood-fill в процессе своей работы может посетить одно и тоже значение узла в сетке несколько раз. В связи с тем, что смещение пептида и присвоение

ему параметров требует вычислительных ресурсов, посещённые и вычисленные значения сетки добавляются в префиксное дерево. Поиск и проверка требуют значительно меньших ресурсов.

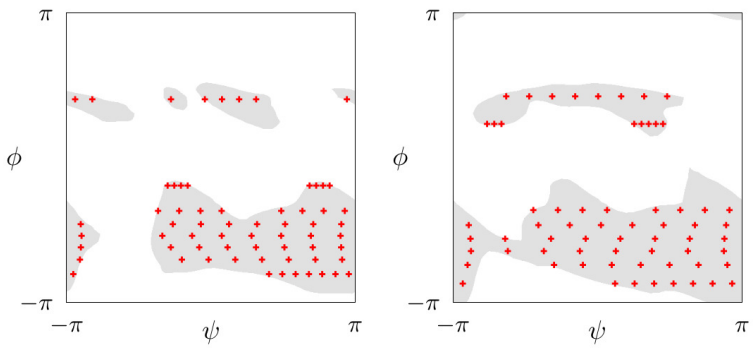

Рис. 3. Покрытие сеткой распределений углов главной цепи пептида в комплексе 1JWG (код PDB).

Важно отметить, что сам процесс построения выборки является обходом графа со структурой «решетка», но без определения самого графа. Выбор алгоритма обусловлен простотой реализации и исследованиями в [10], а также собственными экспериментами. Также необходимо отметить, что при обходе сетки невозможно использовать окрестность Мура для всех параметров в силу высокого количества соседних узлов. Например, в рассматриваемой задаче количество параметров в гиперкубе $n = 15$, для каждой точки окрестность фон Неймана $2n$ узлов, окрестность Мура 3^n узлов. Однако, алгоритм Flood-fill позволяет использовать окрестность Мура для части параметров.

На рис. 1 представлен результат построения выборки по двумерной сетке (9,18) с помощью реализованного алгоритма Flood-fill для произвольной области. Также на рис. 1, выражаясь в терминах компьютерной графики, отмечен «затравочный» узел и результат квантильного преобразования равномерно распределенных векторов $2 \cdot 10^4$. На рис. 3 показан результат покрытия параметров равномерной сеткой. Сами параметры кодируются в отрезке $[0, 1]$, на рис. 3 показаны получаемые в результате преобразования [5] узлов сетки значения.

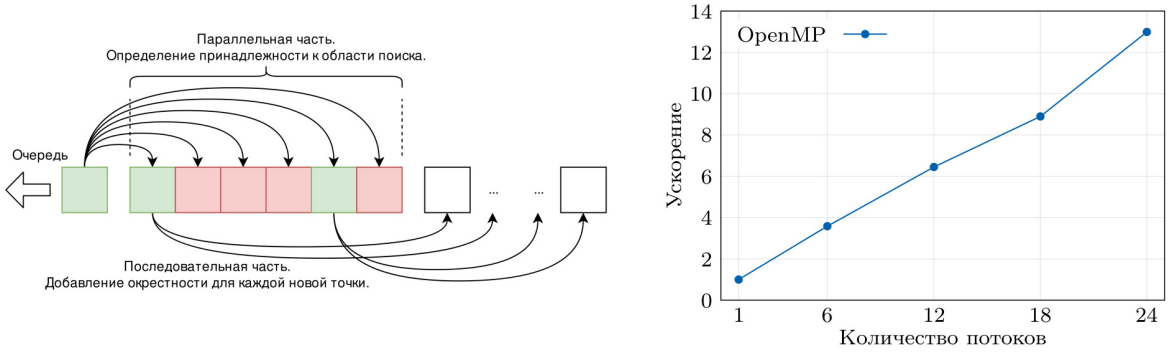

Рис. 4. Используемая схема распараллеливания при построении выборки и её производительность.

Необходимо отметить, что в приведенных примерах явно указаны основные недостатки построения выборки по сетке и квантильного преобразования. Во первых, алгоритм подходит только для определения связанной области. В случае несвязанности области из других областей для алгоритма также нужен исходный узел. Во вторых, значительная часть области поиска может оказаться недоступной из-за большого шага сетки и использования окрестности фон Неймана. В третьих, часть области поиска, рядом с границей непрерывной области, недоступна. В четвертых, граница поиска может выйти за границы непрерывной

области. Однако, последний недостаток не столь важен, так как незначительный выход за границы поиска несущественен.

В настоящей работе произведена параллельная реализация алгоритма построения выборки с применением технологии OpenMP. Произведено сравнение с последовательной версией алгоритма. На рис. 4 представлена схема распараллеливания и получаемое ускорение. Каждый узел сетки, который необходимо проверить на принадлежность к области поиска, добавляется в очередь. Если текущий узел принадлежит области, в очередь добавляется его окрестность, за исключением уже посещённых узлов. В отличие от последовательной версии, где проверка и добавление в дерево с вычисленными значениями происходит на каждом шаге, в параллельной реализации некоторые узлы вычисляются несколько раз.

3.2. Параллельная реализация

При использовании многомерной квантильной функции (5-6) для преобразования одного вектора $z \in [0,1]^d$ необходимо d раз пройти по всей выборке размера n и каждый раз отмечать удовлетворяющие всем предыдущим условиям значения, которые затем будут использоваться в одномерном квантильном преобразовании. Каждый элемент выборки – d-мерный вектор. Таким образом производится покоординатное сравнение вектора из выборки с двумя известными векторами (верхней и нижней границей). В случае невыполнения хотя бы одного условия при сравнении, т.е. непопадания координаты вектора в диапазон значений между границами, вектор из выборки не рассматривается.

На данном этапе довольно просто применить параллельные вычисления. Поскольку заранее неизвестно количество удовлетворяющих всем условиям векторов в текущей реализации одномерное квантильное преобразование выполняется на хосте.

В результате выполненной работы реализованы три различных способа выбора необходимых значений с использованием технологии OpenCL. Во всех трех случаях вся выборка располагается на графическом ускорителе. При этом используется тип данных $float$.

В первом случае используется простейшая схема распараллеливания. Ядро OpenCL запускается с максимальным возможным размером рабочей группы. Используя единственный глобальный номер потока в ядре происходит покоординатное сравнение с использованием цикла и условного оператора if. В случае невыполнения хотя бы одного условия происходит выход из ядра. Если все условия выполнены, последнее значение, для одномерного преобразования, записывается в возвращаемый с ускорителя на хост вектор длины n. При этом в параметры ядра передается только два необходимых для сравнения вектора. Данная реализация обозначена на рис. 6 как GPU 1.

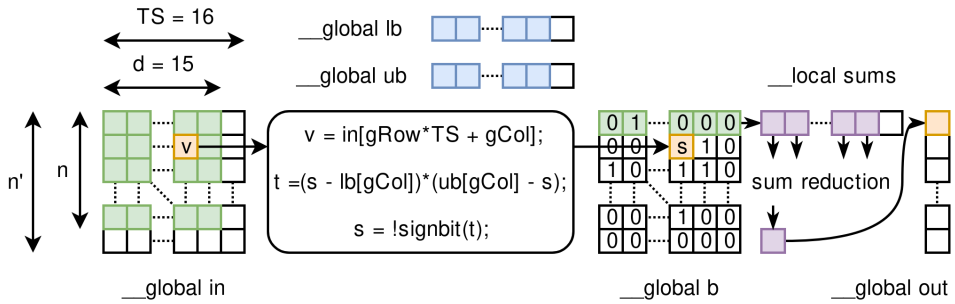

Рис. 5. Используемая схема распараллеливания GPU 2 с дополнительным массивом.

Во втором случае используется дополнительный массив типа *char* аналогичный выборке. В данном массиве сохраняются знаки покоординатного сравнения, которое производится по формуле $(x_i - min_i) \cdot (max_i - x_i)$. При запуске ядра в параметрах используются рабочие группы 16x16. Поскольку рассматриваемая задача имеет размерность 15, массив расширен до размера рабочей группы. Следующее ядро выполняет процедуру покоординатного сло-

жения каждой строки дополнительного массива, так называемый SumReduction. При этом используется локальный массив и обеспечивается когерентность запросов. Полученные значения суммы записываются в вектор длины n и копируются с ускорителя на хост. Искомый вектор найден, если сумма битов равна текущей размерности задачи. Данная реализация обозначена на рис. 6 как GPU 2. Также на рис. 5 представлена схема распараллеливания.

В третьем случае исключается работа с дополнительным массивом. Ядро выполняет процедуру покоординатного сравнения аналогично GPU 2 и сразу же выполняет процедуру сложения полученных знаковых битов. При этом также используется локальный массив и обеспечивается когерентность запросов. Полученные значения суммы записываются в вектор длины n и копируются с ускорителя на хост. Данная реализация обозначена на рис. 6 как GPU 3.

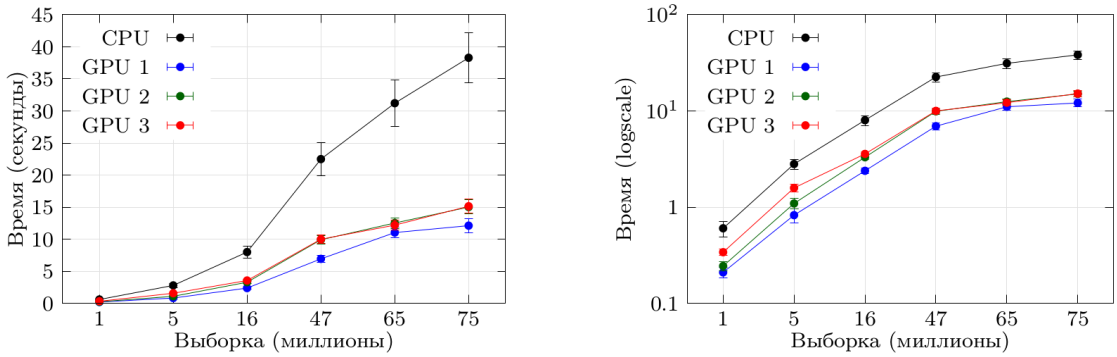

Рис. 6. Производительность применяемых схем параллельных вычислений.

Результаты предложенных схем приведены в 6. Для каждой выборки выполнено 10 запусков. Представлено среднее арифметическое время работы одного квантильного преобразования и среднеквадратическое отклонение. Несмотря на детерминированность получаемых выборок, предсказать их размер довольно трудно. Этим обусловлены представленные округленные размеры. В представленных результатах учитывается время полного квантильного преобразования, т.е. при каждом запуске произошло 15 проходов по выборке и столько же раз с ускорителя на хост скопирован результирующий вектор. Также учитывается произведенное 15 раз одномерное квантильное преобразование.

Несмотря на полученное во всех случаях ускорение, оно остается приблизительно постоянным для каждой выборки. Максимально полученное ускорение дает первая, простейшая, реализация, со средним ускорением в 3.14 раза. Худший результат со средним ускорением в 2.18 раза дает версия с дополнительным массивом.

Причины плохой производительности следующие. Во первых, при каждом преобразовании 15 раз с ускорителя на хост копируется массив, равный размеру выборки. Во вторых, высокая скорость выполнения операции сравнения. В третьих, большое количество обращений к глобальной памяти ускорителя. В первой, простейшей реализации, меньше всего такого рода обращений, поскольку при невыполнении хотя бы одного условия происходит выход из функции. Этим обусловлено максимально полученное ускорение этой реализации.

Все вычисления выполнены на кластере ОИЯИ HybriLIT [11] с использованием одного ускорителя NVIDIA Tesla K40s. Необходимо отметить, что используемый вычислительный узел имеет три графических ускорителя. Используя дополнительные ресурсы можно несколько увеличить ускорение, разбив выборку на равные части.

4. Результаты численных экспериментов

В настоящих экспериментах выполнялся докинг пептида DLLHI в белок 1JWG:B (см. рис. 1). Размерность задачи составила 54 параметра. Квантильное преобразование приме-

нялось к 15-ти параметрам, которые отвечают за положение пептида в определенной выше окрестности. Радиус сфер равен четырем ангстремам.

Сравнение адаптивной дифференциальной эволюции JADE производилось с протоколом Rosetta FlexPepDock [2]. Условием применения данного протокола является присутствие пептида в радиусе пяти ангстрем от места связывания. Работа протокола включает в себя несколько различных этапов, заключительная стадия которого – алгоритм Монте-Карло с локальной оптимизацией. Выбор эволюционного алгоритма JADE обусловлен проведенными в [5, 6] исследованиями.

На рис. 7 представлены результаты использования алгоритма JADE, его модификации с использованием квантильной функции qJADE и протокола Rosetta FlexPepDock (FPD). Указано среднеквадратичное отклонение атомов главной цепи пептида относительно нативной структуры, которая прошла процедуру локальной оптимизации стандартными средствами пакета Rosetta. Нативное состояние комплекса имеет значение скоринг-функции приблизительно -280.

Начальная позиция пептида для эволюционных алгоритмов и FPD представляла собой перевернутый на 180 градусов относительно нативного состояния вдоль места связывания пептид. Приемлемым результатом докинга является субангстремное значение отклонения.

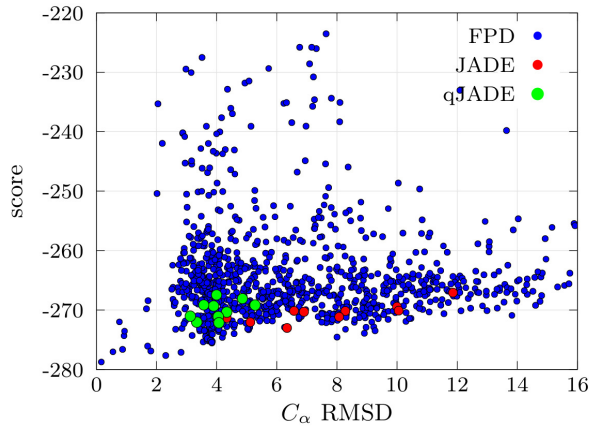

Рис. 7. Результаты десяти независимых запусков JADE, qJADE и 10^3 запусков FlexPepDock.

Результаты проведенных численных экспериментов показывают, что с поставленной задачей докинга справился только протокол FPD. Применением квантильного преобразования удалось сократить пространство поиска и добиться лучших результатов для эволюционного алгоритма JADE. Под сокращением пространства поиска имеется в виду не сокращение размерности, а удержание пептида в окрестности места связывания. На это указывает меньшее среднеквадратичное отклонение атомов пептида. Для худшего найденного значения результат изменился приблизительно в два раза. Однако в среднем для лучших найденных значений энергия связывания практически аналогична. Это указывает и неспособность используемого эволюционного алгоритма оптимизации преодолеть сложный энергетический ландшафт.

В данном случае использовалась выборка размером приблизительно 75 миллионов. Построение такой выборки для приведенного пептида на одном узле заняло приблизительно 5 часов. Задаваемое количество вызовов целевой функции выбиралось из расчета сопоставимости времени вычислений.

5. Заключение

В результате выполненной работы проведена реализация многомерной эмпирической квантильной функции. Предложен сеточный подход к построению детерминированный вы-

борки, произведена параллельная реализация и получено приемлемое ускорение. Произведена параллельная реализация квантильной функции.

На основании проведенных исследований можно заключить, что с помощью квантильной функции возможно свести задачу пептид-белок докинга в непрерывный единичный гиперкуб. При этом учитываются остальные параметры, которые также проходят процедуру преобразования [5]. Такая постановка задачи позволяет создать платформу для объективного сравнения различных алгоритмов глобальной оптимизации, таких как эволюционные, роевые, алгоритмы оценки распределений, алгоритмы со множественной оценкой и без оценки константы Липшица.

Недостатком использования квантильной функции является экспоненциальный рост выборки. Проведенные исследования показывают, что используемая размерность 15 параметров является верхней допустимой границей. Необходимо отметить, что во многих актуальных задачах пептид-белок докинга необходимо рассматривать пептиды длиной 10-15 аминокислотных остатков. Таким образом, размерность степеней свободы белка возрастает в 2-3 раза, что приводит к невозможности использования квантильной функции в текущей постановке.

Целью дальнейшей работы является расширение возможностей применения квантильной функции в задаче пептид-белок докинга. Смещение пептида определяет позицию первого α-углерода. В настоящей работе использовалось преобразование позиции пептида только в одну из двух ограничивающих сфер. Однако, довольно просто построить отображение в две. Таким образом, можно рассматривать одновременно два потенциальных положения пептида в линейном интерфейсе связывания.

Важно отметить, что предложенный подход с применением квантильной функции может быть применён для широкого спектра задач со схожей формулировкой.

Литература

1. Rentzsch R., Renard B.Y. Docking small peptides remains a great challenge: an assessment using AutoDock Vina // Briefings in Bioinformatics. 2015. Vol. 16, No. 6. P. 1045–1056. DOI: 10.1093/bib/bbv008.

2. Raveh B., London N., et al. Rosetta FlexPepDock ab-initio: Simultaneous Folding, Docking and Refinement of Peptides onto Their Receptors // PLoS ONE. 2011. Vol. 6. No. 4. DOI: 10.1371/journal.pone.0018934

3. Lopez-Camacho E., Garcia Godoy M.J., et al. Solving molecular flexible docking problems with metaheuristics: A comparative study // Applied Soft Computing. 2015. DOI: 10.1016/j.asoc.2014.10.049

4. Alford R.F., Leaver-Fay A., Jeliazkov R., et al. The Rosetta all-atom energy function for macromolecular modeling and design. 2017. DOI: 10.1101/106054.

5. Полуян С.В., Ершов Н.М. Применение параллельных эволюционных алгоритмов оптимизации в задачах структурной биоинформатики // Вестник УГАТУ. 2017. Т. 21, № 4. С. 143–152.

6. Poluyan S., Ershov N. Parallel evolutionary optimization algorithms for peptide-protein docking // EPJ Web of Conferences. 2018. Vol. 173. P. 06010–06010. DOI: 10.1051/epjconf/201817306010

7. Sellers M.S., Hurley M.M. XPairIt Docking Protocol for peptide docking and analysis. // Molecular Simulation. 2015. Vol. 42. P. 149–161. DOI: 10.1080/08927022.2015.1025267.

8. O'Brien G.L. The Comparison Method for Stochastic Processes. // The Annals of Probability. 1975. Vol. 3, No. 1. P. 80–88. DOI: 10.1214/aop/1176996450

9. Einmahl J.H.J., Mason D.M. Generalized Quantile Processes. // The Annals of Statistics. 1992. Vol. 20, No. 2. P. 1062–1078. DOI: 10.1214/aos/1176348670

10. Vučković V., Arizanović B., Le Blond S. Generalized N-way iterative scanline fill algorithm for real-time applications // Journal of Real-Time Image Processing. 2017. DOI: 10.1007/s11554-017-0732-1.

11. Heterogeneous Computing Cluster HybriLIT. URL: http://hybrilit.jinr.ru/en/ (дата обращения: 15.04.2018).