Министерство образования и науки Российской Федерации
Московский физико-технический институт (государственный университет)
Факультет управления и прикладной математики
Кафедра «Интеллектуальные системы»
при Вычислительном центре им. А. А. Дородницына РАН

Подкопаев Александр Сергеевич

Параметризованные полуопределенные релаксации и их приложения

010900 — Прикладные математика и физика

Бакалаврская диссертация

Научный руководитель:
и.о. н.с. ИППИ РАН, к.ф.-м.н.
Максимов Ю. В.

Москва

2016 г.

Содержание

Аннотация

В работе рассматриваются эффективные методы приближенного решения дискретных оптимизационных задач, основанные на их релаксации к задачам выпуклой оптимизации. Доминирующим подходом в этой области являются методы полуопределенного программирования. Вместе с тем скорость сходимости указанных методов существенно падает с ростом размерности, что делает их малоприменимыми в задачах большой размерности. В работе исследуется рад способов снижения вычислительной сложности методов полуопределенного программирования, основанных на хордальной декомпозиции исходной задачи и идеях случайного прореживания данных. Разработанные методы апробировались на различных модельных и реальных данных (задача о максимальном разрезе в случайных графах и задача прогнозирования третичной структуры белков). Показано, что в ряде задач предложенные алгоритмы позволяют повысить качество и/или скорость решения в сравнении с алгоритмами, использовавшимися ранее.

Ключевые слова: дискретная оптимизация, полуопределенное программирование, хордальные графы, третичная структура белка.

1 Введение

Задачи дискретной оптимизации возникают во многих прикладных задачах, в числе которых задачи из области статистической физики и вычислительной биологии. Доминирующим подходом является сведение к задачам полуопределенного программирования. Однако такие методы плохо масштабируемы под сложность решаемой проблемы, что делает малоэффективным применение этих методов в случае решения реальных задач, поскольку велик размер области поиска решений. Основной мотивацией исследования является задача упаковки в многомерные комплексы.

В основе каждого белка лежит полипептидная цепь, которая организована в трехмерную структуру. Различают четыре уровня пространственной организации белка: первичная, вторичная, третичная и четвертичная структуры белковых молекул. Под четвертичной структурой понимают макромолекулярное образование из нескольких полипептидных цепей в составе единого мультимерного белкового комплекса.

В работе решается задача предсказания упаковки боковых цепей белковой структуры при помощи выпуклой оптимизации в предположении известного положения основной цепи.

В работах по прогнозированию структур белков [1, 2] основное внимание уделяется взаимодействиям между парами белков. В качестве результата используемые методы выделяют наиболее правдоподобных кандитатов, которые впоследствии при помощи целевой функции ранжируются. На данный момент среди наиболее популярных подходов к решению данной задачи следует отметить метод прогнозирования структуры в приближении жестких тел с последующим применением алгоритма полярной корреляции Фурье, описанный в статье [1], а также предложенный в работе [2] подход, основанный на методе Монте-Карло. Особое внимание в литературе уделяется решению задачи предсказания упаковки боковых цепей. В статьях [3, 4] описаны одни из лучших по качеству и скорости методов решения задачи упаковки боковых цепей. Важно отметить, что задача прогнозирования структур является $NP-$трудной задачей квадратичной оптимизации. Проблема упаковки белковых молекул в комплекс связана с исследованием большого числа всевозможных вариантов структур белка (большой размер базового пространства поиска).

Перейдем к формальной постановке задачи: предположим, что в цепи имеется n белков, причем каждый $i-$ый белок может располагаться в одной из p позиций: $\vec{x^i} = [x_1^i, x_2^i, ..., x_p^i]$, где $x_k^i \in \{0, 1\}$. Для типичных макроструктур предполагается $n \sim 10$, $p \sim 100$. Тогда одному комплексу будет соответствовать вектор $\vec{x} = [\vec{x^1}, \ldots, \vec{x^m}]^T$. Любой паре позиций x_p^i и x_m^j можно приписать функцию энергии q, которая обусловлена силовым взаимодействием. Каждой позиции белка также соответствует собственное значение энергии b_0 (эти значения получены из экспериментальных данных). Обозначим через N – число всевозможных позиций белков, составляющих один комплекс.

Тогда задача оптимальной упаковки может быть записана в виде задачи минимизации:

$$\vec{x^T}\mathbf{Q_0}\vec{x} + \vec{b_0^T}\vec{x} \to \min_{\vec{x}\in\{0,1\}^N} \tag{1.1}$$

при условии, что $\mathbf{A}\vec{x} = \vec{1}_n$, где

$$\mathbf{A} = \begin{bmatrix} 1 & \cdots & 1 & 0 & \cdots & 0 & \cdots\cdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1 & \cdots & 1 & \cdots\cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \cdots\cdots & \vdots & \ddots & \vdots \\ \underbrace{0 \cdots 0}_{p_1} & & & \underbrace{0 \cdots 0}_{p_2} & & & \cdots\cdots & \underbrace{1 \cdots 1}_{p_n} & & \end{bmatrix}$$

и

$$\mathbf{Q}_0 = \begin{bmatrix} [0] & [q_{12}(p_1, p_2)] & \cdots & [q_{1n}(p_1, p_n)] \\ [q_{21}(p_2, p_1)] & [0] & \cdots & [q_{2n}(p_2, p_n)] \\ \vdots & \vdots & \ddots & \vdots \\ [q_{n1}(p_n, p_1)] & [q_{n2}(p_n, p_2)] & \cdots & [0] \end{bmatrix}$$

В общем случае, матрица $\mathbf{Q}_0$ не является положительно полуопределенной. Одна из формулировок задачи записывается как невыпуклая оптимизация. Изначально задача задана на булевом кубе: $\mathbf{B}_N = \{\vec{x} : x_i = \{0, 1\}\ i \in \{1, ..., N\}\}$ Замечая, что $\vec{x^T}\mathbf{Q}_0\vec{x} = \langle \mathbf{Q}_0, \vec{x}\vec{x^T}\rangle_{N,N}$, переходим к задаче выпуклой оптимизации. С использованием различных методов релаксации можно попытаться получить решение исходной задачи за полиномиальное время.

Решение задачи определяется размерностями и свойствами матрицы $\mathbf{Q}_0$: в случае малых размерностей достаточно воспользоваться переборными методами (жадные алгоритмы), а в случае больших размерностей предлагается воспользоваться методами выпуклой оптимизации: например, полуопределенная релаксация, релаксация Лагранжа. Целью данной статьи является построение алгоритма меньшей вычислительной сложности с использованием предложенных методов [5, 6, 7, 8] .

2 Постановка задачи

В силу введенных в формальной постановке задачи предположений получаем, что вектор $\vec{x}$ имеет блочную структуру, то есть он может быть поделен на малое количество блоков $n \sim 10$, а на каждый блок размера $p \sim 100$ накладывается ограничение вида $||x||_\infty = 1$. Вектор $\vec{x}$ представляет собой набор вероятностей того, что составные белки занимают определенное положение в пространстве, а матрицы $\mathbf{Q}_i$ и вектора $\vec{b}_i$ предоставляют данные о ранее известных белковых структурах. Пусть задана выборка из N матриц энергий: $\{\mathbf{Q}_1, \ldots, \mathbf{Q}_N\}$. Квадратные матрицы энергий $\mathbf{Q} = \{q_{ij}\}_{i,j=1,\ldots,m}$ имеют размеры порядка $m = np \sim 1000$ и являются сильно разреженными (доля нулевых элементов составляет порядка 99%).

Целью первой части работы является исследование задачи квадратичной оптимизации, возникающей при решении проблемы оптимальной упаковки белков, и повышение эффективности методов выпуклой оптимизации за счет ее структуры.

Во второй части работы предлагается рассмотреть оптимизационные задачи, заданные на лапласианах случайных графов. Пусть имеется граф $G(n,p) = (V,E)$, где $p = \frac{1+\varepsilon}{n}$, которому соответствует лапласиан $\mathbf{L}$. Тогда предлагается рассмотреть задачу `Max-Cut`:

$$\underset{\mathbf{X}\succeq 0, \mathbf{X}_{ii}=1 \ \forall i\in V}{\text{maximize}} \quad \frac{1}{2}\sum_{(i,j)\in E} \omega_{ij}(1-\mathbf{X}_{ij}) \tag{2.1}$$

Если рассматривать случай, когда $\omega_{ij} = \omega_{ji}$ и $\omega_{ij} = 0$, если вершины i, j не соединены ребром, то задача принимает следующий вид:

$$\underset{\mathbf{X}=\mathbf{X}^T\succeq 0, \mathbf{X}_{ii}=1 \ \forall i\in V}{\text{maximize}} \quad \frac{1}{4}\sum_{i,j\in V} \omega_{ij}(1-\mathbf{X}_{ij}) \tag{2.2}$$

Введем в рассмотрение лапласиан графа $\mathbf{L}$ (подробнее о том, как он вводится, рассказано в следующей главе) и учтем, что тогда:

$$\sum_{i,j} \omega_{ij}(1 - \mathbf{X}_{ij}) = \sum_{i,j} \omega_{ij} - \sum_{i \neq j} \omega_{ij}\mathbf{X}_{ij} = \sum_{i} \left(\sum_{j} \omega_{ij} \right) + \sum_{i \neq j} \mathbf{L}_{ij}\mathbf{X}_{ij} =$$

$$= \sum_{i} \mathbf{L}_{ii}\mathbf{X}_{ii} + \sum_{i \neq j} \mathbf{L}_{ij}\mathbf{X}_{ij} = \sum_{i,j} \mathbf{L}_{ij}\mathbf{X}_{ij} = tr(\mathbf{LX})$$

Опуская множитель $\frac{1}{4}$, получаем задачу:

$$SDP_p = \underset{\mathbf{X} \succeq 0, d(\mathbf{X}) = \mathbf{1}_n}{\text{maximize}} \quad tr(\mathbf{LX}) \tag{2.3}$$

Если ввести $\mathbf{D}(\xi)$ как диагональную матрицу, для которой $\mathbf{D}_{ii} = \xi_i$, то двойственная задача записывается как:

$$\underset{\mathbf{D}(\xi) \succeq L}{\text{minimize}} \quad tr(\mathbf{D}(\xi), \mathbf{I}) \tag{2.4}$$

или

$$SDP_d = \underset{\mathbf{D}(\xi) \succeq L}{\text{minimize}} \quad \vec{1}^T \xi \tag{2.5}$$

Тогда оценка для решения двойственной задачи записывается в виде:

$$SDP^+ = \underset{\eta : T(\eta) \succeq L}{\text{minimize}} \quad \max_{x_i^2 = 1} \left(\vec{x}^T T(\eta) \vec{x} \right) \tag{2.6}$$

Для полученной задачи предлагается применить идею случайного «прореживания» к графу G. Если изначально вероятность появления ребра в случайном графе больше значения $\frac{1}{n}$, то вероятность того, что в графе присутствует «гигантская» компонента связности, стремится к единице [9]. Фиксируется структура исходного графа. Далее для имеющегося графа предлагается начать удалять ребра из него с вероятностью q, где q подбирается из условия: $p(1-q) < \frac{1}{n}$. В результате «прореживания» G с «гигантской» компонентой связности бьется на блоки (малые компоненты связности), а задача сводится к задаче минимизации по отдельным блокам много меньших размерностей. Для подобного сведения используются свойства случайных графов, о которых будет рассказано в следующей главе.

3 Используемые алгоритмы и их эффективность

Для снижения вычислительной сложности исходной невыпуклой задачи квадратичной оптимизации, являющейся $NP-$полной, осуществляется переход к графовой структуре данных (построение описано ниже), а затем задача декомпозируется на задачу «в деревьях» (*tree decomposition*), в процессе используются различные эвристические методы - например, работа с хордальным расширением полученного графа. В результате это приводит к решению задачи за меньшее время в сравнении с обычной полуопределенной релаксацией [10]. Перейдем непосредственно к описанию используемых методов.

3.1 Полуопределенное программирование

При использовании методов выпуклой оптимизации для решения задачи 1.1 невыпуклые ограничения заменяются на более слабые, но выпуклые ограничения. Тогда мы переходим к выпуклой задаче, методы решения которой хорошо известны. В результате удается получить хорощую оценку в задачах поиска минимума значения целевой функции. Основным методом выпуклой релаксации, используемым в этой работе, является полуопределенная релаксация.

Полуопределенную релаксацию можно представить в виде:

$$\underset{\mathbf{X}\in\mathcal{S}_+^N,\ \vec{x}\geqslant\vec{0}_N}{\text{minimize}} \quad \sum_{i,j}\mathbf{Q}_{ij}\mathbf{X}_{ij} \tag{3.1}$$

где $\mathcal{S}_+^N = \{\mathbf{X}_{N,N} : \mathbf{X} = \mathbf{X}^T \succeq 0\}$ и, кроме того, $\mathbf{X} \succeq \vec{x}\vec{x}^T, X_{ii} = x_i,\ i = 1, ..., N$ и $\mathbf{A}\vec{x} = \vec{1}_n$.

Используя лемму Шура, перепишем задачу в виде эквивалентной:

$$\underset{\mathbf{X}\in\mathbb{R}^{N\times N},\ \vec{x}\geqslant\vec{0}_N}{\text{minimize}} \quad \sum_{i,j}\mathbf{Q}_{ij}\mathbf{X}_{ij} \tag{3.2}$$

при дополнительных условиях $[\mathbf{X}\ \vec{x}; \vec{x}^T\ 1] \in \mathcal{S}_+^{N+1}, X_{ii} = x_i,\ i = 1, ..., N, \mathbf{A}\vec{x} = \vec{1}_n$.

Метод реализуется на MatLab при использовании пакета *cvx*.

Алгоритм 3.1. Elimination Game

Вход: граф $G = (V, E)$ и число вершин $n = |V|$

1: $G_1 = G; E' = \emptyset$;

2: **для** $i = 1$ до n

3: Добавляем ребра к G_i так, чтобы все соседи вершины i стали попарно смежными, добавляем эти ребра к E', убираем вершину i и получаем граф G_{i+1}

4: **return** $G' = (V; E \cup E')$

3.2 Хордальная структура данных

Для снижения вычислительной сложности алгоритма воспользуемся результатами из теории графов. Заметим, что каждой квадратной матрице $\mathbf{Q}_0$ порядка n в соответствие можно поставить некоторый граф $\mathbf{G}$ с n вершинами, ребра которого строятся по следующим правилам: наличию ненулевого элемента $\{i, j\}$ в матрице $\mathbf{Q}_0$ соответствует ребро, соединяющее вершины i и j, а значит, единица в таблице смежности графа, нулевому элементу будет соответствовать ноль в соответствующих ячейках таблицы смежности.

Определение 1. Граф называется хордальным, если каждый из его циклов, имеющий четыре и более дуг, имеет хорду, которая является ребром, соединяющим две вершины, не смежные в цикле.

Поставим задачу поиска максимальных клик в графе $\mathbf{G}$. Поскольку про структуру таблицы смежности исходного графа ничего не известно, то можно считать, что граф $\mathbf{G}$ является произвольным. В общем случае задача поиска максимальных клик является $NP-$полной. Однако для исходных графов можно построить хордальное расширение и, воспользовавшись свойствами этих графов, получить решение этой задачи за полиномиальное время. В нашей работе для построения расширения использовался алгоритм 1.

Для полученного хордального расширения воспользуемся теоремой [11]:

Теорема 1.

Пусть матрице $\mathbf{X}$ соответствует граф $\mathbf{G}$, $\mathbf{K_1}, ..., \mathbf{K_d}$-максимальные клики графа $\mathbf{G}$. Тогда следующие условия эквивалентны:

- Можно дополнить $\mathbf{X}$ до положительно полуопределенной матрицы.

- $\mathbf{X}_{\mathbf{K_i},\mathbf{K_i}} \succeq 0, i = 1, ..., d$ - подматрица $\mathbf{X}$ с номерами строк и стобцов, принадлежащими K_i.

Тогда при решении задачи 1.1 для тех матриц энергий, для которых размер максимальной клики у соответствующего ей графа существенно меньше размера этого графа, заменим исходную полуопределенную релаксацию на ее модификацию: поиск минимума не по всем $\mathbf{X}$, удовлетворяющим необходимым условиям, а по отдельным подматрицам существенно меньших размеров, соответствующих вершинам, входящим в состав максимальных клик.

Приведем теоретические факты и алгоритмы, используемые для вычисления максимальных клик в хордальных графах за полиномиальное время [12, 13].

Из теории хордальных графов использовались следующие теоремы:

Теорема 2. Неориентированный граф является хордальным тогда и только тогда, когда алгоритм `MCS` (Maximum Cardinality Search) вычисляет (G, σ), где σ является совершенным порядком исключения для графа G.

Теорема 3. Каждая максимальная клика хордального графа $G = (V, E)$ имеет вид $\{v\} \cup X(v)$, где

$X(v) = \{w \in adj(v) | \sigma(v) < \sigma(w)\}$

Теорема 4. Хордальный граф имеет не более, чем n максимальных клик.

Приведено описание работы алгоритма `MCS` (Maximum Cardinality Search), который сопоставляет вершинам графа некоторый порядок – perfect elimination order), а также алгоритма 3, который использует результаты работы алгоритма 2, и далее используется для нахождения максимальных клик в хордальном графе.

В результате от $NP-$полной задачи нахождения максимальных клик переходим к задаче для хордальных графов, решаемой за полиномиальное время.

3.3 Лапласиан графа

Пусть задан граф G на n вершинах. Тогда его лапласиан определяется как:

$$\mathbf{L} = \mathbf{D} - \mathbf{A},$$

где $\mathbf{D}$ - матрица, на главной диагонали которой степени вершин графа, а остальные элементы - нули, а $\mathbf{A}$ - матрица смежности графа G.

Алгоритм 3.2. Maximum Cardinality Search

Вход: граф $G = (V, E)$ и число вершин $n = |V|$;

для всех $v \in V$
  положить $label[v] = 0$;
для i **от** n **до** 1
  Выбрать непронумерованную вершину v с наибольшим значением $label$.
  Положить $\sigma(v) = i$ — тем самым нумеруем эту вершину
  для всех непронумерованных w, **смежных** v,
    Увеличить $label[w]$ на единицу.
return (G, σ)

Алгоритм 3.3. Maximal Cliques (G, σ)

Вход: хордальный граф $G = (V, E), n = |V|$

для всех $v \in V$
  положить $size[v] = 0$
для i **от** 1 **до** n
  $v = \sigma^{-1}(i)$.
  если $deg(v) = 0$ **то**
    вывести максимальную клику $\{v\}$
  $X = \{w \in adj(v) | \sigma(v) < \sigma(w)\}$
  если $X \neq \emptyset$ **то**
    если $size[v] < |X|$ **то**
      вывести максимальную клику $\{v\} \cup X$
    $m[v] = \sigma^{-1}(min\{\sigma(w) | w \in X\})$ $size[m[v]] = max\{size[m[v]], |X| - 1\}$
return набор максимальных клик

Приведем некоторые свойства Лапласиана графа:

- **L** - симметричная матрица
- **L** -положительно-полуопределенная матрица
- **L** обладает свойством диагонального доминирования, то есть $\forall i \in \{1, \ldots, n\}$:

$$\mathbf{L}_{ii} \geqslant \sum_{j=1, j\neq i}^{n} |\mathbf{L}_{ij}|$$

В случае рассмотрения взвешенного графа также определяют Лапласиан как:

$$\mathbf{L}_{ij} = \mathbf{L}(\omega)_{ij} = \begin{cases} \sum_k \omega_{ik}, i = j \\ -\omega_{ij}, i \neq j \end{cases}$$

3.4 Декомпозиции случайных графов

Пусть задано n изолированных вершин $V = \{1, \ldots, n\}$. На этом множестве предлагается построить граф по следующим правилам. Предлагается рассмотреть схему Бернулли, при которой вероятность возникновения ребра между любой фиксированной парой вершин в графе есть $p : p \in [0, 1]$. Рассматриваются неориентированные графы без кратных ребер. Пусть все ребра в графе появляются в графе независимо. Пусть E - случайное множество ребер, возникающее в результате реализации этой схемы. Тогда вероятность реализации случайного графа $G(n, p)$ с $|E|$ ребрами есть:

$$P(G) = C_{C_n^2}^{|E|} p^{|E|} (1-p)^{C_n^2 - |E|}$$

Вероятность p не фиксируется, а рассматривается как функция от числа вершин: $p = p(n)$. Характер этой зависимости определяет свойства «фазовых переходов», которые оказываются крайне полезными в приложениях. Некоторые из полезных свойств случайных графов будут рассмотрены в следующем параграфе.

Особое внимание в теории случайных графов уделяется фазовым переходам, когда при переходе через критическую точку p^*, резко меняются свойства наблюдаемого графа. Предлагается рассмотреть две теоремы, которые определяют характер компонент связности в случайном графе для рассматриваемой модели Эрдеша - Реньи.

Теорема 3.1. *Рассмотрим модель $G(n, p)$. Пусть $p = \frac{c \ln n}{n}$. Если $c > 1$, то с вероятностью, стремящейся к единице при $n \to \infty$, случайный граф связен. Если $c < 1$, то почти всегда случайный граф не является связным.*

Этот результат имеет важные применения в вопросах надежности сетей сообщения. В нашей работе предлагается воспользоваться следующим свойством. Предлагается рассмотреть крайние случаи: при $p = 0$ мы наблюдаем граф из n изолированных вершин, а при $p = 1$ мы перейдем к случаю полного графа. Если постепенно увеличивать p от нуля до единицы, то известно, что существует такая критическая точка p^*, что при превышении этого значения в графе наблюдается возникновение «гигантской компоненты связности». Имеет место следующая теорема:

Теорема 3.2. *Рассмотрим модель $G(n, p)$. Пусть $p = \frac{c}{n}$. Если $c < 1$, то найдется такая константа $\beta = \beta(c)$, что с вероятностью, стремящейся к единице при $n \to \infty$, размер каждой связной компоненты случайного графа не превосходит $\beta \ln n$. Если $c > 1$, то найдется такая константа $\gamma = \gamma(c)$, что с вероятностью, стремящейся к единице при $n \to \infty$, в случайном графе есть ровно одна компонента связности размера $\geqslant \gamma n$.*

Существуют уточнения этой теоремы. Например, при $c > 1$, кроме единственной «гигантской» компоненты, в случайном графе все остальные компоненты имеют логарифмический по отношению к общему числу вершин размер. Тем самым, имеет место «эволюция» графа в зависимости от значения $p(n)$. При $p < \frac{1}{n}$ наблюдаются некоторые малые компоненты связаности логарифмического размера. При переходе через точку $\frac{1}{n}$ возникается «гигантская» компонента связности, а при переходе через значение $\frac{\ln n}{n}$ весь граф становится связным.

Для иллюстрации этого материала предалагается построить случайные графы из 50 и 300 вершин. Пусть $p^*(n) = \frac{1}{n}, \overline{p} = \frac{\ln n}{n}$. Для каждого из графов были рассмотрены описанные этапы «эволюции», проведены расчеты и построены соответствующие графики при помощи пакета $GraphViz$, который позволяет визуализировать результаты. Результаты представлены на рис. 1 – 6.

Обратимся к задаче о максимальном разрезе. Предалагалось изначально генерировать случайные графы с $p(n) = \frac{1+\varepsilon}{n}$. В соответствии с вышесказанным в сгенерированных графах с большой вероятностью присутствовала «гигантская» компонента связности. В соответствии с идеей «случайного» прореживания ребра из графа удаляются с вероятностью q, где q выбирается из условия $p_1(1 - q) = p_2$, а значение p_2 принимает значение уже меньшее, чем $\frac{1}{n}$. В результате такого прореживания вместо

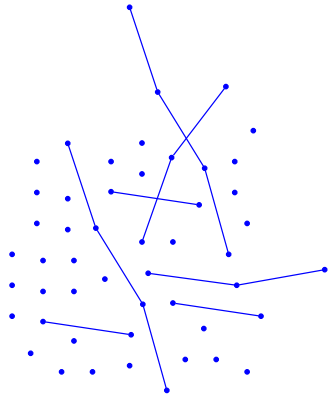

Рис. 1: $n = 50; p = 0.01$

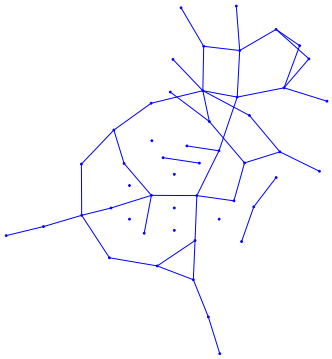

Рис. 2: $n = 50; p = 0.03$

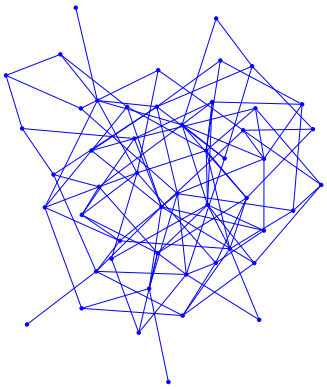

Рис. 3: $n = 50; p = 0.08$

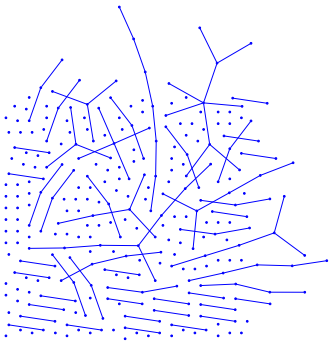

Рис. 4: $n = 300; p = 0.002$

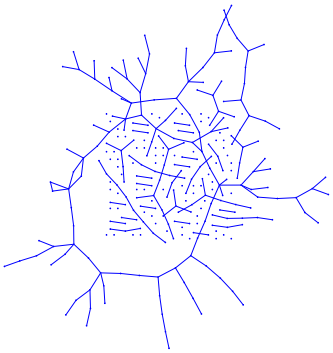

Рис. 5: $n = 300; p = 0.004$

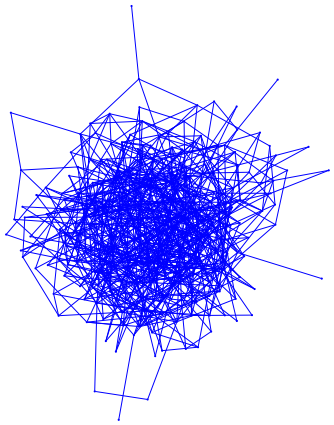

Рис. 6: $n = 300; p = 0.02$

задачи:

$$SDP^{+} = \underset{\eta: T(\eta) \succeq L}{\text{minimize}} \quad \max_{x_i^2 = 1} \left(\vec{x}^T T(\eta) \vec{x} \right) \tag{3.3}$$

решается задача:

$$SDP_t = \underset{\eta: T(\eta) \succeq L}{\text{minimize}} \quad \sum_j \max_{x_i^2 = 1} \left(\vec{x}^T T_j(\eta) \vec{x} \right), \tag{3.4}$$

где ведется суммирование по блокам. В рамках вычислительного эксперимента исследуется качество решения полученной задачи в сравнении с решением исходной.

4 Вычислительный эксперимент

4.1 Эффективность хордальной декомпозиции

4.1.1 Задача прогнозирования структур белка

Матрицы энергий (379 штук) из PDB (Protein Database Bank) были разбиты на группы по количеству вершин в соответствующих графах: до 500 (малых размеров), 500-1000 (средних размеров), больше 1000 (больших размеров). Для графов из каждой категории (73 - в первой категории, 145 - во второй и 161 - в третьей соответственно) были построены хордальные расширения и вычислены размеры максимальных клик. Приведем результаты в табл. 1.

Таблица 1: Соотношение размеров графов и их максимальных клик в исследуемой выборке

Размер клики	$<$**12**	**12-16**	**16-20**	$>$**20**
Менее **500** вершин	**4**	**29**	**32**	**8**
500-1000 вершин	**1**	**41**	**87**	**16**
более **1000** вершин	**0**	**16**	**120**	**25**

Для исходной матриц энергий из выборки приведем графики зависимости, на которых по оси абсцисс откладывается фиксированные размеры, а по оси ординат отложим число графов, размер максимальных клик которых не превосходит это фиксированное число.

4.1.2 Сравнение алгоритмов

Учитывая приведенные выше обоснования, делаем вывод, что на исходной выборке выборке следует попробовать применить модификацию исходной полуопределенной релаксации. На исходной выборке из PDB можно сравнить скорость и точность работы полуопределенной релаксации с/без использования хордальной структуры данных, а сопоставить их методу Монте-Карло.

Вычислив максимальные клики для графов, соответствующих матрицам энергий, можно перейти к работе с методами релаксации. Была изучена зависимость

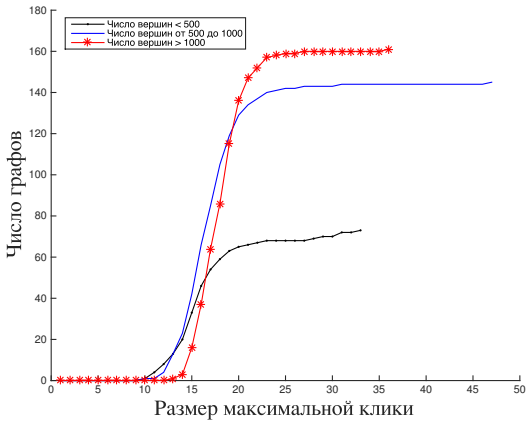

Рис. 7: Размеры максимальных клик

времени работы SDP, SDP с учетом хордальной структуры данных и Монте-Карло от размеров исходных матриц.

Сравнивались средние времена работы исходного SDP и модифицированной задачи на всей выборке из 379 матриц. Было получено, что в среднем на любой матрице из выборки модифицированная задача решается в 3.24 раза быстрее.

Также можно посмотреть, в скольких случаях модифицированный SDP дает значения целевой функции быстрее метода Монте-Карло. На нашей выборке первый метод давал результаты быстрее на 11% исходной выборки. В среднем, метод работал быстрее на квадратных матрицах размера 200-300. Если задать допустимую точность вычисления значения целевой функции $\delta = 5\%$, то расчет показывает, что в сравнении со стохастическим методом Монте-Карло модифицированный SDP дает более точное значение целевой функции на 2.84% матриц, на которых он работает быстрее.

Приведена таблица с указанием времени работы работы трех методов на части выборки, для которой соответствующие квадратные матрицы энергий имеют размеры до 500 строк (столбцов), и полученным значением минимизируемого функционала. Под `id` понимается метка данной белковой структуры, а под размером - размер соответствующей матрицы энергий $\mathbf{Q}_0$. В табл. 2 приводятся результаты, в которых хордальная модификация показала наиболее высокие результаты по сравнению с другими методами.

Вместе с тем следует отметить, что в ряде случаев качество методов полуопределенной релаксации существенно выше, чем качество методов Монте-Карло. Пять

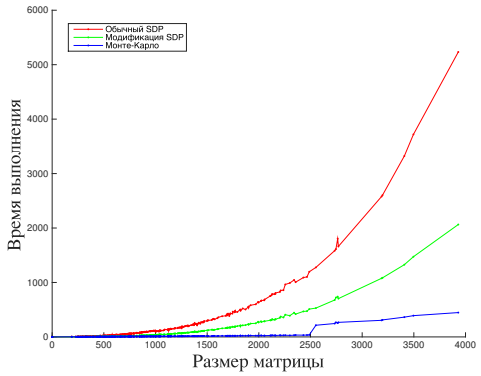

Рис. 8: Вычислительная сложность для матриц всех размерностей

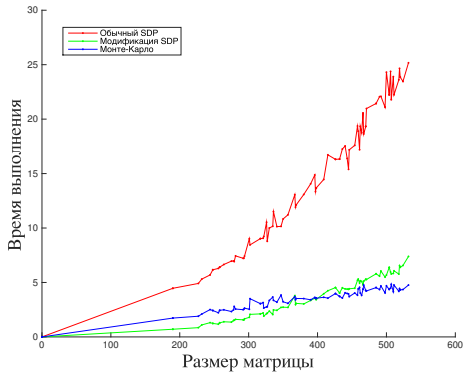

Рис. 9: Вычислительная сложность для матриц меньших размерностей

Таблица 2: Задачи с наибольшим выигрышем в скорости модифицированного SDP алгоритма

id	SDP		Модифицированный SDP		Монте-Карло		Размер
	Значение	Время	Значение	Время	Значение	Время	
1ado	13452	4.91	13624	0.85	12439	1.9	227
1b9w	15995	11.51	16201	2.48	15416	3.36	336
1cei	15830	10.16	15088	2.06	14717	3.66	335
1f94	13937	5.69	14006	1.31	13131	2.51	244
1gvp	16686	9.22	15184	1.91	15696	2.65	322
1oai	12595	6.62	11859	1.28	11599	2.56	258
1uln	14269	7.44	14079	1.61	13699	2.56	281
1ulr	16295	7.42	14294	1.66	15004	2.64	294
1wm3	14278	7.22	13305	1.57	12751	2.5	292
1yu5	12412	6.19	11704	1.31	12255	2.32	258
1yzm	8757.2	4.54	8197.3	0.73	7995.7	1.7	190
2g30	11522	6.96	11578	1.36	11508	2.33	275
2gqv	11427	4.39	11609	0.68	10371	1.76	190

задач с наибольшим выигрышем в качестве решения методами полуопределенного программирования приведены в табл. 3.

Таблица 3: Задачи с наибольшим выигрышем в качестве полуопределенной релаксации

id	SDP		Модифицированный SDP		Монте-Карло		Размер
	Значение	Время	Значение	Время	Значение	Время	
2hpl	18169	13.52	17641	3.62	19199	3.95	398
2o0q	20229	14.06	17866	3.35	18621	3.41	390
2i49	16099	10.83	14929	2.73	15683	3.23	350
2fl4	18492	11.91	17055	2.97	18310	3.41	368
1ulr	16295	7.42	14294	1.66	15004	2.64	294

4.2 Эффективность «случайного» прореживания

4.2.1 Задача о максимальном разрезе

В данной работе в ходе вычислительного эксперимента генерировались случайные графы $G_1 = G(n, p_1)$, где вероятность появления ребра в случайном графе – p_1 подбиралась в следующем диапазоне: $p_1 \in \{\frac{2}{n}, \frac{3}{n}, \ldots, \frac{6}{n}\}$. Рассматривались случайные графы с числом вершин $n : n \in \{150, 160, \ldots, 200\}$.

В соответствии со сказанным в разделе 3.4.1 в сгенерированных графах с большой вероятностью присутствовала «гигантская» компонента связности. В соответствии с идеей «случайного» прореживания предлагалось удалять ребра из графа G_1 с вероятностью q, где q выбиралось из условия $p_1(1-q) = p_2$. Для проведения вычислительного эксперимента предлагалось рассмотреть значения $p_2 \in \{\frac{1}{2n}, \frac{1}{4n}, \frac{1}{6n}, \frac{1}{8n}\}$. В рамках вычислительного эксперимента проводилось случайное прореживание, а далее сравнилось время и точность решения исходной и прореженной задачи.

4.2.2 Результаты эксперимента

В соответствии с теоремой 3.2 при заданном диапазоне p_2 с высокой вероятностью после прореживания в случайном графе будут наблюдаться компоненты связности логарифмического относительно общего числа вершин размера. Поэтому делается вывод о целесообразности исследования решения модифицированной задачи по отношению к решению исходной. Результаты сравнения скорости и качества решения исходной и прореженной задач приведены в табл. 4–5. В ячейках этих таблиц

указаны средние (по n) значения точности решения прореженной задачи относительно исходной, а также отношения времени решения прореженной задачи к времени решения исходной для различных пар p_1 и p_2.

Таблица 4: Результаты для графов $G(n, p_1)$ при $p_1 = \frac{2}{n}, \frac{3}{n}, \frac{4}{n}$

p_1	$\frac{2}{n}$				$\frac{3}{n}$				$\frac{4}{n}$			
p_2	$\frac{1}{2n}$	$\frac{1}{4n}$	$\frac{1}{6n}$	$\frac{1}{8n}$	$\frac{1}{2n}$	$\frac{1}{4n}$	$\frac{1}{6n}$	$\frac{1}{8n}$	$\frac{1}{2n}$	$\frac{1}{4n}$	$\frac{1}{6n}$	$\frac{1}{8n}$
$\left\langle \frac{SDP_t}{SDP_d} \right\rangle$, %	91	83,7	82,1	80,4	90,5	82,6	81,5	78,9	83,5	80,4	77,5	76,8
$\frac{t_{\text{прореженное}}}{t_{\text{без прореживания}}}$	0,2	0,18	0,18	0,17	0,22	0,19	0,18	0,18	0,19	0,18	0,18	0,17

Таблица 5: Результаты для графов $G(n, p_1)$ при $p_1 = \frac{5}{n}, \frac{6}{n}$

p_1	$\frac{5}{n}$				$\frac{6}{n}$			
p_2	$\frac{1}{2n}$	$\frac{1}{4n}$	$\frac{1}{6n}$	$\frac{1}{8n}$	$\frac{1}{2n}$	$\frac{1}{4n}$	$\frac{1}{6n}$	$\frac{1}{8n}$
$\left\langle \frac{SDP_t}{SDP_d} \right\rangle$, %	82,3	80,2	78,5	76,7	80,6	78,9	77,3	75,9
$\frac{t_{\text{прореженное}}}{t_{\text{без прореживания}}}$	0,19	0,18	0,17	0,17	0,18	0,18	0,18	0,17

Вместе с тем следует отметить, что с уменьшением значения p_2, то есть увеличением вероятности удаления ребер случайного графа, имели место рост числа блоков и уменьшение их размеров, наблюдался несущественный рост скорости решения задач, который, однако, сопровождался падением качества. Также задача, в целом, решалась хуже с увеличением вероятности появления ребра p_1 в исходном графе. В результате проведения вычислительного эксперимента был сделан вывод об эффективности прореживания до значения p_2 чуть меньшего $\frac{1}{n}$, однако вопрос об эффективности проведения дальнейшего прореживания остается открытым, поскольку на исследованной выборке наблюдалось несущественное улучшение скорости в сравнении с существенными потерями в качестве решения задачи при уменьшении значения p_2.

5 Заключение

В данной работе рассмотрены методы сведения оптимизационных задач больших размерностей к последовательности задач меньших размерностей. В первой части работы изучались алгоритмы прогнозирования структур белков, построенных на основе предположения о минимизации суммарной энергии. Для невыпуклой задачи оптимизации были рассмотрены методы выпуклой релаксации как без учета хордальной структуры данных, так и с учетом этой структуры. Был проведен вычислительный эксперимент на основе данных из PDB. Было проведено сравнение результатов работы этих двух алгоритмов и метода Монте-Карло. Показано, что в задачах малой размерности использованние выпуклой релаксации с хордальной структурой позволяет повысить скорость решения. В целом точность решения методами полуопределенной релаксации и методом Монте-Карло примерно равна, что по всей видимости связано с особенностями порождения данных задачи. Во второй части работы рассматривалась идея декомпозиции оптимизационных задач на меньшие подзадачи методом «случайного» прореживания. Был проведен вычислительный эксперимент на сгенерированных случайных графах, а также было изучено качество решения прореженных задач в сравнении с исходными. Было показано, что в ряде случаев оно является эффективным.

Использованная в первой части работы идея из [14] заключалась в разбиении исходной задачи полуопределенной релаксации на подзадачи определенным способом и последующим решением подзадач также методами полуопределенной релаксации. Дальшейшее повышение точности результатов возможно за счет явного представления оптимума в небольших кликах в виде последовательности максимумов, решаемой явно и встраивании данной конструкции в двойственную задачу.

В работе было показано, что в ряде задач структурной биологии данные обладают необходимыми свойствами для проведения эффективной декомпозиции исходной задачи на задачи меньшей вычислительной сложности (например, использованная в статье хордальная структура данных). В ряде случаев удается получить существенный выигрыш в скорости и качестве решения поставленной задачи.

Список литературы

[1] *Moughon G., Wang, Kuhlman B.* Protein-protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations // *Journal of Molecular Biology.* — 2003.

[2] *Topf M., Lasker K., Wolfson H.* Protein structure fitting and refinement guided by cryo-em density // *Structure.* — 2008. — Vol. 16, no. 2. — Pp. 295 – 307.

[3] *Krivov G., Shapovalov M., Dunbrack R.* Improved prediction of protein side-chain conformations with scwrl4 // *Proteins.* — 2009. — Pp. 778–795.

[4] *Zhichao M., Cao Y., Jiang T.* Rasp: rapid modeling of protein side chain conformations // *Bioinformatics.* — 2011. — Pp. 3117–3122.

[5] *Freund R.* Introduction to Semidefinite Programming. — 2004.

[6] *Boyd S., Vandenberghe L.* Convex Optimization. — Cambridge University Press, 2004.

[7] *Nesterov Y.* Introductory lectures on convex optimization. — Springer Science and Business Media, 2004. — Vol. 87.

[8] *Nesterov Y.* Quality of semidefinite relaxation for nonconvex quadratic optimization: CORE Discussion Papers 1997019: Universite catholique de Louvain, Center for Operations Research and Econometrics (CORE), 1997.

[9] *Райгородский А. М.* Модели случайных графов // *М.: МЦНМО.* — 2011.

[10] *Подкопаев А., Карасиков М., Максимов Ю.* Прогнозирование структур белков методами полуопределенного программирования // *Труды МФТИ.* — 2015. — Vol. 7, no. 4. — Pp. 66–73.

[11] *Grone R., Johnson C. R., Wolkowicz H.* Positive definite completions of partial hermitian matrices // *Linear Algebra and its Applications.* — 1984. — Vol. 58. — Pp. 109 – 124.

[12] *Rose D., Lueker G., Tarjan R.* Algorithmic aspects of vertex elimination on graphs // *SIAM Journal on Computing.* — 1976. — Vol. 5. — Pp. 266–283.

[13] *Tarjan R. E., Yannakakis M.* Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs // *SIAM J. Comput.* — 1984. — Vol. 13, no. 3. — Pp. 566–579.
`http://http://dx.doi.org/10.1137/0213035`

[14] *Klerk E. D.* Exploiting special structure in semidefinite programming: A survey of theory and applications // *European Journal of Operational Research.* — 2010. — Vol. 201. — Pp. 1–10.