УЧРЕЖДЕНИЕ РОССИЙСКОЙ АКАДЕМИИ НАУК
САНКТ-ПЕТЕРБУРГСКИЙ АКАДЕМИЧЕСКИЙ УНИВЕРСИТЕТ –
НАУЧНО-ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР НАНОТЕХНОЛОГИЙ РАН

На правах рукописи

Диссертация допущена к защите
Зав. кафедрой

_____

“ ”_____2012 г.

ДИССЕРТАЦИЯ

НА СОИСКАНИЕ УЧЕНОЙ СТЕПЕНИ

МАГИСТРА

Тема: Приближённые алгоритмы решения перестановочных задач

Направление: 010600.68 – Прикладные математика и физика

Магистерская программа: “Математические и информационные технологии”

Выполнил студент: А. Г. Головнёв

(подпись)

Руководитель:
к.ф.-м.н. А. С. Куликов

(подпись)

Рецензент:
к.ф.-м.н. С. И. Николенко

(подпись)

Санкт-Петербург
2012 г.

Реферат

Работа выполнена на 25 страницах в 2 главах, использован 41 источник.

Лучший известный алгоритм решения задачи о коммивояжёре имеет время работы $O^*(2^n)$, используя память $O^*(2^n)$ ($O^*(\cdot)$ скрывает полиномиальные множители от длины входа, т.е. $\mathrm{poly}(n, \log M)$, где n — количество вершин исходного графа, M — максимальный вес ребра). Если мы рассматриваем только алгоритмы с полиномиальной памятью, то лучшее решение требует время $O^*(4^n n^{\log n})$. Для задачи коммивояжёра с небольшими весами рёбер удобно использовать алгоритм со временем работы $O^*(1.657^n \cdot M)$ и памятью $O^*(M)$. Открытым вопросом является существование алгоритма со временем работы $O^*(2^n)$, использующем лишь полиномиальную память. В первой главе работы мы предлагаем алгоритм, который для любого $\varepsilon > 0$ находит $(1+\varepsilon)$-приближение общей ориентированной задачи о коммивояжёре за время $O^*(2^n\varepsilon^{-1})$, используя $O^*(\varepsilon^{-1}) = \varepsilon^{-1} \cdot \mathrm{poly}(n, \log M)$ память. То есть, для любого фиксированного ε, алгоритм за $O^*(2^n)$ шагов и полиномиальную память находит $(1+\varepsilon)$-приближение.

Лучшее приближение задачи о кратчайшей надстроке имеет фактор приближения 2.5. Во второй главе работы мы приедлагаем алгоритм, улучшающий фактор приближения для частного случая задачи о кратчайшей надстроке: для случая, когда длины всех входных строк не превосходят 7. В частности, алгоритм находит 4/3-приближение задачи о 3-надстроке.

Содержание

Введение

В работе рассматриваются приближённые алгоритмы решения перестановочных задач. Многие NP-трудные задачи (например, задача выполнимости булевой формулы, задача о вершинном покрытии, задача о покрытии множествами) имеют множество кандидатов на решение размера 2^n. Но даже для некоторых из перечисленных задач неизвестны алгоритмы со временем работы меньше 2^n. Перестановочные задачи — это задачи, кандидаты на решение которых имеют размер порядка $n!$. Интересно, что для многих перестановочных задач можно получить алгоритмы со временем работы 2^n.

В данной работе мы рассматриваем следующие перестановочные задачи: задача коммивояжёра и задача о кратчайшей надстроке. Эти задачи имеют как практический, так и теоретический интерес. Задача коммивояжёра широко используется в логистике, планировании, производстве интегральных схем. Задача о кратчайшей надстроке долгое время рассматривалась применительно к области биоинформатики. Также кратчайшие надстроки используются для эффективного хранения информации о графах.

Для точного решения задачи коммивояжёра известны алгоритмы, использующие

- время $O^*(2^n)$ и память $O^*(2^n)$;
- время $O^*(2^n \cdot M)$ и память $O^*(M)$;
- время $O^*(4^n n^{\log n})$ и полиномиальную память.

Открытым вопросом является существование точного алгоритма решения задачи коммивояжёра со временем $O^*(2^n)$, использующем лишь полиномиальную память. В первой главе этой работы получим промежуточный результат. Мы покажем, что для любого фиксированного ε, за время $O^*(2^n)$ и полиномиальную память может быть найдено $(1+\varepsilon)$-приближение общей задачи о коммивояжёре. Этот результат интересен ещё и тем, что в предположении P≠NP, общая задача о коммивояжёре не может быть приближена за полиномиальное время с любым полиномиально вычислимым фактором. Более того, если P≠NP, даже метрическая версия задачи о коммивояжёре не может быть приближена с любой точностью за поли-

номиальное время.

Обе исследуемые задачи являются NP-трудными. Следовательно, маловероятно существование полиномиальных алгоритмов точного решения задач. В связи с этим изучаются полиномиальные алгоритмы приближённого решения задачи коммивояжёра и задачи о кратчайшей надстроке.

Лучшее известное приближение задачи о кратчайшей надстроке имеет фактор приближения 2.5. Во второй главе мы предложим полиномиальный по времени приближённый алгоритм решения для задачи о кратчайшей k-надстроке. Задача о кратчайшей k-надстроке принимает на вход строки длины k. Новый алгоритм улучшает известные приближения для задачи о k-надстроке для $k < 8$. В частности, алгоритм находит 4/3-приближение задачи о 3-надстроке.

Глава 1. Экспоненциальная схема приближения для задачи о коммивояжёре

1.1. Определения

Гамильтонов цикл графа — это цикл, проходящий по каждой вершине графа ровно один раз. Задача коммивояжёра (ЗК) заключается в поиске гамильтонова цикла минимального веса в полном графе с целыми неотрицательными весами рёбер. Метрическая задача коммивояжёра — задача коммивояжёра на графах, удовлетворяющих неравенству треугольника. Ориентированная (асимметрическая) задача коммивояжёра — это задача коммивояжёра на ориентированных графах.

Обозначим через n количество вершин исходного графа, а через M — максимальный вес ребра. $O^*(\cdot)$ скрывает полиномиальные множители от длины входа, т.е. $\mathrm{poly}(n, \log M)$. Вес оптимального гамильтонова цикла в графе G обозначим через $\mathrm{OPT}(G)$. Мы будем опускать G в данной нотации, если из контекста будет понятно, с каким графом мы работаем. Наша цель — по заданному ε найти гамильтонов цикл длины не больше, чем $(1+\varepsilon)\mathrm{OPT}(G)$.

1.2. Известные результаты

Точные алгоритмы Bellman [1], Held и Karp [2] разработали алгоритм динамического программирования для ЗК. Этот алгоритм использует память и время $O^*(2^n)$ и до сих пор является самым быстрым алгоритмом решения общей задачи о коммивояжёре. Лучший известный алгоритм с полиномиальной памятью имеет время работы $O^*(4^n n^{\log n})$ (Gurevich и Shelah [3], Björklund и Husfeldt [4]).

Koivisto и Parviainen [5] показали, что ЗК может быть решена за время $O^*(2^{2n-t}n^{\log(n-t)})$ и память $O^*(2^t)$ для любого $t = n, n/2, n/4....$ Также авторы предложили алгоритм, который для любого $\sqrt{2} < S < 2$ решает ЗК за $O^*(T^n)$ шагов и $O^*(S^n)$ память, так что $TS < 4$.

Kohn, Gottlieb и Kohn [6], Karp [7], Bax и Franklin [8] предложили алгоритм для ЗК со временем работы $O^*(2^n \cdot M)$ и памятью $O^*(M)$, где M — максимальный вес ребра. Björklund [9] разработал алгоритм Монте-Карло со временем работы $O^*(1.657^n \cdot M)$ и экспоненциально маленькой вероятностью ошибки.

Даже для частного случая ЗК — метрической задачи коммивояжёра — неизвестны лучшие точные алгоритмы решения. Открытым вопросом является существование алгоритма со временем работы $O^*(2^n) = 2^n \cdot \mathrm{poly}(n, \log M)$ и полиномиальной памятью (см Open Problem 2.2.b, Woeginger [10]).

Eppstein [11] предложил алгоритм для ЗК в кубических графах со временем работы 1.260^n, для графов степени 4 — со временем работы 1.890^n. Iwama и Nakashima [12] улучшили оценку для кубических графов до 1.251^n. Gebauer [13] разработал алгоритм для графов степени 4 со временем работы 1.733^n и экспоненциальной памятью.

Björklund, Husfeldt, Kaski и Koivisto [14] разработали алгоритм со временем работы $O(2-\varepsilon)^n$ для ЗК в графах ограниченной степени (здесь ε зависит только от степени графа).

Для неориентированной версии задачи о гамильтоновом цикле (ГЦ) оценка $O^*(2^n)$ была улучшена. Björklund [9] предложил алгоритм, решающий ГЦ за время $O^*(1.657^n)$, используя лишь полиномиальную память. Более того, предложенный алгоритм решает задачу ГЦ на двудольных графах за время $O^*(2^{n/2})$.

Приближённые алгоритмы

Общая задача коммивояжёра Известно, что в предположении

P≠ NP, общая ЗК не может быть приближена с любым полиномиально вычислимым фактором (Sahni и Gonzalez [15]).

Задача называется сильно NP-трудной, если она остаётся NP-трудной, даже когда все её числовые параметры ограничены многочленом от длины входа (Garey и Johnson [16]). Если P≠NP, сильно NP-трудные задачи не имеют FPTAS (Vazirani [17]). ЗК и метрическая ЗК являются сильно NP-трудными задачами (см., например, Garey и Johnson [18]).

Неориентированная метрическая задача коммивояжёра Существует 2-приближающий алгоритм для метрической ЗК (Rosenkrantz, Stearns и Lewis [19]). Лучшее известное приближение 3/2 получено Christofides [20].

Самая сильная известная нижняя оценка принадлежит Papadimitriou и Vempala [21]: если P≠NP, неориентированная метрическая ЗК не может быть приближена за полиномиальное время с точностью лучше, чем $\frac{117}{116}$.

Ориентированная метрическая задача коммивояжёра Frieze, Galbiati, Maffioli [22] показали, что ориентированная версия метрической ЗК может быть приближена с фактором $\log n$, Bläser [23] улучшил оценку до $0.999 \log_2 n$. Kaplan, Lewenstein, Shafrir и Sviridenko [24] разработали $4/3 \log_3 n$-приближение. Feige и Singh [25] предложили $2/3 \log_2 n$-приближение. Недавно Asadpour, Goemans, Madry, Gharan и Saberi [26] улучшили фактор приближения до $O(\log n / \log \log n)$.

Papadimitriou и Vempala [21] доказали, что если P≠NP, то ориентированная метрическая ЗК не может быть приближена за полиномиальное время с точностью лучше, чем $\frac{220}{219}$

Другие частные случаи Полиномиальная приближённая схема (PTAS) — это алгоритм, который для любого фиксированного ε за полиномиальное время находит решение, не превосходящее $(1+\varepsilon)$OPT, где OPT — оптимальное решение задачи. Полностью полиномиальная приближённая

схема (FPTAS) — это полиномиальная приближённая схема, время работы которой полиномиально зависит от n и ε^{-1}. Для евклидовой версии ЗК найдена полиномиальная схема приближений (Arora [27] [28], Mitchell [29]).

Gharan, Saberi и Singh [30] разработали приближённый алгоритм с фактором $3/2 - \varepsilon$ для графической версии ЗК. Mömke, Svensson [31] привели 1.461-приближение для графической ЗК. Mucha [32] улучшил фактор приближения до 1.444.

(1,2)-ЗК — неориентированная задача коммивояжёра на графах, в которых веса всех рёбер равны 1 или 2. Даже этот случай задачи является MAX-SNP-трудным (Papadimitriou, Yannakakis [33]). Berman и Karpinski [34] разработали полиномиальный алгоритм, приближающий эту задачу с фактором 8/7.

Приближение ЗК за экспоненциальное время Boria, Bourgeois, Escoffier и Paschos [35] предложили следующие экспоненциальные $(1+\varepsilon)$-приближения метрической версии ЗК:

- за время $O^*(4^{(1-\varepsilon/2)n} n^{\log n})$ и полиномиальную память для любого $\varepsilon \leq 2/3$,
- за время $O^*(2^{(1-\varepsilon/2)n})$ и экспоненциальную память для любого $\varepsilon \leq 2/5$.

1.3. Идея алгоритма

Мы предлагаем алгоритм, который для любого фиксированного ε использует $O^*(2^n)$ шагов и полиномиальную память для вычисления $(1+\varepsilon)$-приближения общей ориентированной задачи коммивояжёра. Как было указано выше, лучшее известное полиномиальное приближение даже для неориентированной метрической версии ЗК — 1.5 (для ориентированной — $O(\frac{\log n}{\log \log n})$). Если P≠NP, то за полиномиальное время не может быть найдено $\frac{117}{116}$-приближение неориентированной метрической ЗК (см. раздел 1.2).

Если нам необходимо, например, $\frac{1001}{1000}$-приближение, то мы должны использовать точные алгоритмы, которые требуют либо экспоненциальную память, либо время $4^n n^{\log n}$. Экспоненциальные алгоритмы редко используются на практике, но в любом случае, мы способны их запустить и ждать ответа. Однако если алгоритм использует экспоненциальную память, то мы не имеем возможности даже запустить алгоритм на входах разумного размера. Предложенный алгоритм может найти 1001/1000-приближение общей ЗК за время $O^*(2^n)$, используя лишь полиномиальную память.

Разработанный алгоритм использует идею, предложенную для построения полностью полиномиальной приближённой схемы для задачи о рюкзаке в работе Ibarra, Kim [36]. Авторы используют псевдополиномиальный алгоритм для задачи о рюкзаке, работающий время $O(nW)$, где n — количество предметов, W — вместимость рюкзака. Полностью полиномиальная схема приближения задачи о рюкзаке сначала делит веса всех предметов на $\alpha(\varepsilon, n, W)$, затем вызывает псевдополиномиальный алгоритм. Полученный ответ может не оказаться оптимальным, т.к. некоторые веса не просто поделены на α, но и округлены. Однако простой анализ показывает, что округление не сильно сказывается на результате.

Мы используем похожую идею. Через OPT обозначим оптимальное рещение ЗК. Для того, чтобы получить полиномиальный по памяти приближённый алгоритм, мы сначала разделим веса всех рёбер на достаточно большое число α, затем запустим точный алгоритм, основанный на формуле включений-исключений. Время работы полученного алгоритма будет равно $2^n \cdot \mathrm{OPT}/\alpha$ и длина полученного цикла будет не больше $\mathrm{OPT} + \alpha n$. Теперь мы хотим выбрать α так, что $\alpha \geq \frac{\mathrm{OPT}}{\mathrm{poly(n)}}$ и $\alpha \leq \frac{\varepsilon \mathrm{OPT}}{n}$. Метрическая версия ЗК может быть приближена за полиномиальное время, то есть, мы можем найти β, что $\mathrm{OPT} \leq \beta \leq \mathrm{OPT} \cdot \log n$ и взять $\alpha = \frac{\beta \cdot \varepsilon}{n \log n}$. Тогда полученный алгоритм будет иметь время работы $O^*(2^n \varepsilon^{-1})$ и использовать $O^*(\varepsilon^{-1})$ памяти. Для неметрического случая задачи мы сначала находим

4-приближение за экспоненциальное время, а после запускаем описанный алгоритм.

Сравнивая предложенный алгоритм с алгоритмом из работы Boria et al. [35], отметим, что для любого фиксированного $\varepsilon > 0$, алгоритм Boria либо требует экспоненциальную память, либо имеет время выполнения $O^*(c^n)$, где $c > 2$.

1.4. Алгоритм включений-исключений

Алгоритм включений-исключений основан на следующей хорошо известной формуле (доказательство может быть найдено, например, в Bax [37]).

Теорема 1.1. Пусть X — конечное множество, f, g — функции, определённые на всех подмножествах X. Если для любого $A \subseteq X$ выполняется

$$g(A) = \sum_{B \subseteq A} f(B)\,.$$

то для любого $A \subseteq X$

$$f(A) = \sum_{B \subseteq A} (-1)^{|A|-|B|} g(B). \tag{1.1}$$

Например, для проверки существования в графе $G(V, E)$ гамильтонова пути из вершины s в вершину t за время $O^*(2^n)$ и полиномиальную память, можно определить для $A \subseteq V$, $f(A)$ как количество путей (не обязательно простых) длины $n-1$, проходящих по всем вершинам из A. Тогда $f(V)$ равно количеству гамильтоновых путей из s в t. Легко видеть, что $g(A)$ — количество путей (не обязательно простых) длины $n-1$, проходящих только по вершинам из A. Осталось заметить, что $g(A)$ может быть посчитано за полиномиальное время (динамическим программированием или возведением матрицы смежности, индуцированной A, в степень $n-1$).

Обычно работы Kohn, Gottlieb and Kohn [6], Karp [7], Bax and Franklin [8] цитируются как алгоритмы со временем работы $O^*(2^n \cdot M)$,

использующие $O(n \cdot M + n^2)$ память, где M — максимальнй вес ребра. Для начала нам необходимо показать, что время работы алгоритма включений-исключений может быть оценено как $O^*(2^n \cdot \mathrm{OPT})$, а память — как $O(n \cdot \mathrm{OPT} + n^2)$.

Задача распознавания для ЗК может быть сформулирована следующим образом: по параметру k необходимо определить, существует ли в данном графе G гамильтонов цикл длины $\leq k$. Мы покажем, что задача распознавания может быть решена за время $O^*(2^n k) = 2^n k \cdot \mathrm{poly}(n)$ и память $O^*(k) = k \cdot \mathrm{poly}(n)$.

За $c(U)$ обозначим количество циклов в $U \subseteq V$ длины $\leq k$, содержащих ровно n рёбер. Отметим, что $c(U)$ может быть вычислено динамическим программированием за время $O(kn^4)$ и память $O(kn^2)$ (можно заполнить таблицу D размера $n \times k$, где $D[v, j]$ — количество путей длины j, заканчивающихся в вершине v).

Алгоритм 1 InclExcl-Decision-TSP — решение задачи распознавания для ориентированной ЗК за время $O^*(2^n k)$ и память $O^*(k)$.

Input: $G = (V, E)$ — полный взвешенный ориентированный граф, k — параметр задачи распознавания.
Output: количество гамильтоновых путей длины $\leq k$.

```
res = 0
for U ⊆ V do
    res = res + (−1)^{|V|−|U|} c(U)
end for
return res
```

Корректность алгоритма InclExcl-Decision-TSP следует из (1.1). Время работы алгоритма можно оценить как $O^*(2^n k)$, а память — как $O^*(k)$.

Бинарный поиск по параметру k даёт следующий алгоритм (InclExcl-TSP) решения оптимизационной версии ориентированной ЗК.

Лемма 1.1. Алгоритм InclExcl-TSP решает ориентированную ЗК за время $O^*(2^n \cdot \mathrm{OPT})$ и память $O^*(\mathrm{OPT})$.

Доказательство. Мы используем двоичный поиск для поиска такого ми-

Алгоритм 2 InclExcl-TSP — решение ориентированной ЗК за время $O^*(2^n \cdot \mathrm{OPT})$ и память $O^*(\mathrm{OPT})$.

Input: $G = (V, E)$ — полный взвешенный ориентированный граф.
Output: длина кратчайшего гамильтонова цикла.

1: установить $k = 1$ и удваивать k, пока InclExcl-Decision-TSP(G, k)$= 0$
2: использовать двоичный поиск на полученном интервале для нахождения минимального такого k, что InclExcl-Decision-TSP(G, k)> 0

нимального k, что алгоритм InclExcl-Decision-TSP возвращает положительное количество циклов. Нам необходимо $O(\log \mathrm{OPT})$ вызовов InclExcl-Decision-TSP с параметром $k \leq 2\mathrm{OPT}$, а $O(\log OPT)$ полиномиально от длины входа. Следовательно, время работы составляет $O^*(2^n \cdot \mathrm{OPT})$, а память — $O(n \cdot \mathrm{OPT} + n^2) = O^*(\mathrm{OPT})$.

□

Замечание 1.1. Алгоритм InclExcl-TSP был предложен в работе Karp [7]. Мы лишь подчеркнули, что время работы и использованная память могут быть оценены как $O^*(2^n \cdot \mathrm{OPT})$ и $O^*(\mathrm{OPT})$, соответственно.

Замечание 1.2. Алгоритм InclExcl-TSP может быть модифицирован для нахождения самого цикла, а не просто установления факта его существования.

1.5. Алгоритм решения метрической задачи коммивояжёра

Обозначим через Approx-Asym-TSP $(\log n)$-приближающий алгоритм ориентированной метрической задачи коммивояжёра (алгоритмы перечислены в разделе 1.2). Мы предлагаем следующую приближённую схему для ориентированной метрической ЗК.

Лемма 1.2. Алгоритм Approx-Metric-TSP находит цикл длины $\leq (1 + \varepsilon)\mathrm{OPT}$ в графе G.

Доказательство. Пусть OPT и OPT′ — длины оптимальных гамильтоновых циклов в графе G до деления весов рёбер на α и после, соответственно.

Алгоритм 3 Approx-Metric-TSP — $(1+\varepsilon)$-приближение метрической ЗК за время $O^*(2^n\varepsilon^{-1})$ и память $O^*(\varepsilon^{-1})$.

Parameter: $\varepsilon > 0$ — фактор приближения.
Input: $G = (V, E)$ — полный взвешенный ориентированный граф.
Output: гамильтонов цикл веса $\leq (1+\varepsilon)\mathrm{OPT}(G)$.

1: $\beta = $ Approx-Asym-TSP(G)
2: разделим веса всех рёбер на α: $w(e) = \lceil w(e)/\alpha \rceil$, где

$$\alpha = \frac{\beta \cdot \varepsilon}{n \log n}$$

3: **return** InclExcl-TSP(G)

Длина полученного алгоритмом пути $\mathrm{RES} \leq \alpha \cdot \mathrm{OPT}'$. Обозначим через $I = (e_1, \ldots, e_n)$ рёбра оптимального цикла в исходном графе G. Тогда OPT' не превосходит $\sum_{i \in I} \lceil w(e_i)/\alpha \rceil$. Следовательно,

$$\mathrm{RES} \leq \alpha \cdot \mathrm{OPT}' \leq \alpha \cdot \sum_{i \in I} \lceil w(e_i)/\alpha \rceil \leq \sum_{i \in I} (w(e_i) + \alpha) = \mathrm{OPT} + \alpha n.$$

Из $\beta \leq \mathrm{OPT} \cdot \log n$ получаем $\alpha n \leq \varepsilon \mathrm{OPT}$ и $\mathrm{RES} \leq (1+\varepsilon) \cdot \mathrm{OPT}$. □

Лемма 1.3. Время работы алгоритма Approx-Metric-TSP равно $O^*(2^n\varepsilon^{-1})$, память — $O^*(\varepsilon^{-1})$.

Доказательство. Время работы шага 1 полиномиально (см. раздел 1.2). По Лемме 1.1, время работы шага 7 составляет $O^*(2^n \cdot \mathrm{OPT}')$, а память — $O^*(\mathrm{OPT}')$. Оценка $\beta \geq \mathrm{OPT}$ даёт

$$\mathrm{OPT}' \leq \frac{\mathrm{OPT}}{\alpha} + n \leq \frac{n \log n}{\varepsilon} + n = O^*(\varepsilon^{-1}).$$

Следовательно, время работы алгоритма Approx-Metric-TSP $O^*(2^n\varepsilon^{-1})$, память — $O^*(\varepsilon^{-1})$. □

1.6. Алгоритм решения общей задачи коммивояжёра

В предыдущем разделе мы рассматривали только метрическую версию задачи коммивояжёра, т.к. для генерации оценки β мы использовали

полиномиальный приближённый алгоритм. Сейчас мы избавимся от вызова полиномиального алгоритма и расширим результат на общую задачу коммивояжёра. Единственное отличие алгоритмов состоит в поиске 4-приближения общей ЗК за экспоненциальное время (шаги 2-5).

Алгоритм 4 Approx-TSP — $(1+\varepsilon)$-приближение общей ЗК за время $O^*(2^n\varepsilon^{-1})$ и память $O^*(\varepsilon^{-1})$.

Parameter: $\varepsilon > 0$ — фактор приближения.
Input: $G = (V, E)$ — полный взвешенный ориентированный граф.
Output: гамильтонов цикл длины $\leq (1+\varepsilon)\mathrm{OPT}(G)$.

1: $\beta = 1$
2: **repeat**
3: Пусть граф G' получен из G делением всех весов рёбер на α': $w'(e) = \lceil w(e)/\alpha' \rceil$, где

$$\alpha' = \frac{\beta}{n}$$

4: $\beta = 2\beta$
5: **until** InclExcl-Decision-TSP$(G', 2n) > 0$
6: Разделим все веса рёбер на α: $w(e) = \lceil w(e)/\alpha \rceil$, где

$$\alpha = \frac{\beta \cdot \varepsilon}{4n}$$

7: **return** InclExcl-TSP(G)

Лемма 1.4. Алгоритм Approx-TSP находит гамильтонов цикл длины $\leq (1+\varepsilon)\mathrm{OPT}$ в графе G.

Доказательство. Нам необходимо показать только то, что шаги 2-5 находят 4-приближение OPT. После мы используем такую же технику, как и в алгоритме Approx-Metric-TSP.

Если G' содержит цикл длины $\leq 2n$, то G содержит цикл длины $\leq 2n\alpha' = 2\beta$. Если в G' отсутствуют циклы длины $\leq 2n$, то все циклы в G имеют длину $\geq \beta$. Действительно, если бы G содержал цикл длины $\leq \beta$, то G' содержал бы цикл длины $\leq \beta/\alpha' + n = 2n$.

Следовательно, 2β является 4-приближением OPT. □

Лемма 1.5. Время работы алгоритма Approx-TSP $O^*(2^n\varepsilon^{-1})$, память — $O^*(\varepsilon^{-1})$.

Доказательство. Время работы шага 5 составляет $O^*(2^n \cdot 2n) = O^*(2^n)$, память — $O^*(2n) = O^*(1)$. Количество вызовов InclExcl-Decision-TSP — $O(\log(\mathrm{OPT} n)) = \mathrm{poly}(n, \log M)$. По Лемме 1.3, время работы алгоритма Approx-TSP оценивается как $O^*(2^n \varepsilon^{-1})$, память — $O^*(\varepsilon^{-1})$. □

Глава 2. Полиномиальный приближённый алгоритм для задачи о надстроке

2.1. Определения

Пусть нам задан набор из n строк $S = \{s_1, \ldots, s_n\} \subseteq \Sigma^+$. Задача о кратчайшей надстроке заключается в поиске строки s наименьшей длины, содержащей каждую s_i как подстроку. Задача о кратчайшей r-надстроке принимает на вход n строк длины r. Компрессией называется разность сумм длин всех строк $|s_1| + \ldots + |s_n|$ и длины кратчайшей надстроки $(s_1, \ldots, s_n)$.

Будем называть перекрытием $\mathrm{overlap}(s, t)$ строк s и t наибольший суффикс s, который является префиксом t. Например, $\mathrm{overlap}(\texttt{ABACBA}, \texttt{BABCA}) = \texttt{BA}$. Через префикс $\mathrm{prefix}(s, t)$ обозначим строку, полученную из s вычёркиванием $\mathrm{overlap}(s, t)$.

Каждой входной строке поставим в соответствие вершину графа. Граф перекрытий строится следующим образом: для любых двух вершин s и t, вес дуги (s, t) равен $\mathrm{overlap}(s, t)$. Граф префиксов имеет веса, равные $\mathrm{prefix}(s, t)$

Задача о кратчайшей надстроке является типичной перестановочной задачей: если мы знаем порядок следования исходных строк в оптимальной надстроке, то мы можем восстановить оптимальную надстроку.

2.2. Известные результаты

Gallant, Maier, Astorer [38] доказали, что задача о кратчайшей 2-надстроке может быть решена за полиномиальное время, а задача о кратчайшей 3-надстроке является NP-трудной.

Лучшее известное полиномиальное приближение задачи о надстроке имеет фактор приближения 2.5 (Sweedyk [39]; Kaplan, Lewenstein, Shafrir, Sviridenko [24]). Задача о компрессии может быть приближена с фактором 2/3 (Kaplan, Lewenstein, Shafrir, Sviridenko [24]). Karpinski и Schmied [40] доказали, что в предположении P≠NP задача о кратчайшей надстроке не может быть приближена с фактором, меньшим 1.0029, задача о компрессии — с фактором 1.0048. Из α-приближения задачи о надстроке над бинарным алфавитом следует α-приближение задачи о надстроке над любым алфавитом (Vassilevska, [41]).

Можно рассматривать задачу о кратчайшей надстроке, как задачу о кратчайшем гамильтоновом пути в графе префиксов. Также задача о надстроке может быть рассмотрена как задача о максимальном гамильтоновом пути в графе перекрытий. Лучший известный алгоритм решения задачи о надстроке использует алгоритм включений-исключений для задачи о коммивояжёре (Kohn, Gottlieb и Kohn [6]; Karp [7]; Bax и Franklin [8]). Следовательно, задача о кратчайшей надстроке может быть решена за время 2^n с использованием лишь полиномиальной памяти.

Открытыми вопросами остаются вопрос о существовании точного алгоритма со временем работы $< 2^n$ и о существовании полиномиальногго алгоритма с фактором приближения < 2.5.

В этой главе мы предложим полиномиальный приближённый алгоритм для задачи о кратчайшей k-надстроке. Алгоритм имеет фактор приближения < 2.5 для $k < 8$.

2.3. Идея алгоритма

Пусть $\mathcal{S} \subseteq \Sigma^3$ — множество строк длины 3.

Отметим, что $n + 2 \leq |\mathrm{OPT}(\mathcal{S})| \leq 3n$ (равенства достигаются в случае, когда перекрытия всех строк будут равны 2 и когда оптимальная строка не содержит строк с перекрытиями). Отметим, что в этих двух слу-

чаях ($\mathrm{OPT}(\mathcal{S}) = n + 2$ или $\mathrm{OPT}(\mathcal{S}) = 3n$) решение может быть найдено за полиномиальное время. Действительно, если $\mathrm{OPT}(\mathcal{S}) = n + 2$, то $\mathcal{S}$ — множетво всех подстрок длины 3 какой-то строки длины $n + 2$. Такая надстрока может быть найдена проходом по эйлерову циклу графа де Брюйна на $\mathcal{S}$, $\mathrm{DG}(\mathcal{S})$. С другой стороны, если $\mathrm{OPT}(\mathcal{S}) = 3n$, то исходные строки не пересекаются, следовательно, любая их конкатенация даёт оптимальную надстроку.

Заметим, что случай $\mathrm{OPT}(\mathcal{S}) = n + 2$ соответствует максимально возможной компрессии, в то время как случай $\mathrm{OPT}(\mathcal{S}) = 3n$ соответствует минимальной (т.е., нулевой) компрессии.

Алгоритм работает следующим образом. Сначала построим граф перекрытий $\mathrm{OG}(\mathcal{S})$ и найдём 2/3-приближение максимального гамильтонова пути в нём. Такой путь даёт хорошее приближение $\mathrm{OPT}(\mathcal{S})$ в случае, когда $\mathrm{OPT}(\mathcal{S})$ достаточно велико.

Теперь построим граф де Брюйна $\mathrm{DG}(\mathcal{S})$. Заметим, что этот граф может рассматриваться как граф де Брюйна набора строк длины 2 над алфавитом Σ^2. А именно, строка `ABC` представлялась, как ребро из `AB` в `BC`. Но это ребро ещё может быть рассмотрено как строка `(AB)(BC)` длины 2 над новым алфавитом. Теперь мы находим оптимальное решение для построенного набора 2-строк (напомним, что задача о 2-надстроке может быть решена за полиномиальное время) и перейдём от решения к решению исходной задачи. Это даёт хорошее приближение в случае, когда $\mathrm{OPT}(\mathcal{S})$ мало. Мы будем использовать этот приём, когда исходные строки сильно перекрываются. Тогда соответствующие 2-строки также пересекаются сильно и могут быть найдены полиномиальным алгоритмом для задачи о 2-надстроке.

2.4. Алгоритм

Алгоритм 5 Приближённый алгоритм для задачи о r-надстроке

$\mathcal{S} = \{s_1, \ldots, s_n\} \subseteq \Sigma^r$

{сначала находим длинный гамильтонов путь в графе перекрытий}

1: пусть $\pi \in S_n$ — это 2/3-приближение максимального гамильтонова пути в графе $\mathrm{OG}(\mathcal{S})$

{теперь находим короткий путь задачи о деревенском почтовом в графе де Брюйна}

2: пусть $\mathcal{S}' = \{s'_1, \ldots, s'_n\} \subseteq \Sigma_1^2$ — множество 2-строк над алфавитом $\Sigma_1 = \Sigma^{r-1}$; s'_i — 2-строка, состоящая из префикса s_i длины $r-1$ и суффикса s_i длины $r-1$

3: найдём $\pi_1 \in S_n$ — оптимальное решение задачи о 2-надстроке для $\mathcal{S}'$

4: **return** лучшее из решений π и π_1

Теорема 2.1. Алгоритм 5 находит $\alpha(r)$-приближение задачи о r-надстроке, где

$$\alpha(r) = \max_{0 \le x \le r-1} \left\{ \min \left\{ \frac{r - 2x/3}{r-x}, \frac{(r^2 - 2r + 2) - (r-1)x}{r-x} \right\} \right\} .$$

Доказательство. Пусть H — оптимальный гамильтонов путь в графе $\mathrm{OG}(\mathcal{S})$. Тогда

$$\mathrm{OPT}(\mathcal{S}) = rn - w(H) .$$

2/3-приближение максимального гамильтонова пути имеет вес $\ge 2w(H)/3$. Следовательно, перестановка π даёт надстроку длины $\le rn - 2w(H)/3$. Соответствующий фактор приближения:

$$\frac{rn - 2w(H)/3}{rn - w(H)} . \tag{2.1}$$

Пусть u обозначает количество рёбер веса $\le (r-2)$ в H. Тогда количество рёбер в H веса ровно $(r-1)$ равно $(n-u)$. Получаем $w(H) \le (r-1)(n-u) + (r-2)u$ и, следовательно,

$$u \le (r-1)n - w(H) . \tag{2.2}$$

Отметим, что

$$\mathrm{overlap}(s'_i, s'_j) = \begin{cases} 1 & \text{, если } \mathrm{overlap}(s_i, s_j) = r-1, \\ 0 & \text{, иначе.} \end{cases}$$

Тогда длина оптимальной надстроки для $\mathcal{S}'$ имеет длину $\le n + u$. Следовательно, π_1 даёт надстроку длины не больше, чем $rn - (r-1)(n-u)$ для

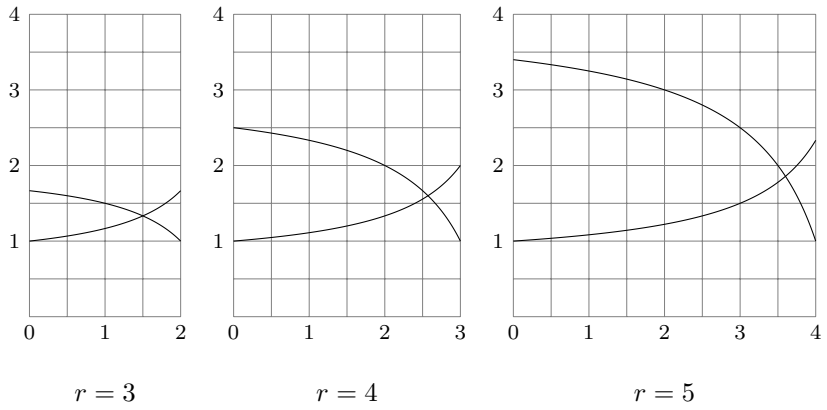

Рис. 2.1: графики $\frac{r-2x/3}{r-x}$ и $\frac{r^2-2r-(r-1)x}{r-x}$ для $r = 3, 4, 5$ и $0 \le x \le r-1$.

$\mathcal{S}$. Из (2.2), это не больше

$$rn - (r-1)(n-(r-1)n + w(H)) = (r^2 - 2r + 2)n - (r-1)w(H)\,.$$

Соответствующий фактор приближения:

$$\frac{(r^2 - 2r + 2)n - (r-1)w(H)}{rn - w(H)}\,. \tag{2.3}$$

Теперь теорема следует из (2.1), (2.3) и $0 \le w(H)/n \le (r-1)$. □

Следствие 2.1. Алгоритм 5 находит 4/3-приближение 3-надстроки, 8/5-приближение для 4-надстроки, 13/7-приближение для 5-надстроки.

Доказательство. Это следует из несложных вычислений. Рис. 2.1 демонстрирует графики рассматриваемых функций. □

Заключение

В данной работе предложены следующие алгоритмы:

- $(1+\varepsilon)$-приближение общей ЗК за время $O^*(2^n\varepsilon^{-1})$ и память $O^*(\varepsilon^{-1})$;
- полиномиальное по времени 4/3-приближение для задачи о 3-надстроке;
- алгоритм, имеющий фактор приближения < 2.5 для задачи о k-надстроке для $k < 8$.

Дальнейшие направления исследования включают в себя следующие задачи:

- разработать точный алгоритм для ЗК со временем работы $O^*(2^n)$, использующий лишь полиномиальную память;
- разработать полиномиальный приближённый алгоритм для задачи о надстроке с фактором приближения < 2.5;
- разработать алгоритм решения задачи о надстроке со временем работы $< 2^n$.

Список литературы

[1] Richard Bellman. Dynamic Programming Treatment of the Travelling Salesman Problem. *J. ACM*, 9:61–63, January 1962.

[2] Michael Held and Richard M. Karp. The Traveling-Salesman Problem and Minimum Spanning Trees. *Mathematical Programming*, 1:6–25, 1971.

[3] Yuri Gurevich and Saharon Shelah. Expected computation time for Hamiltonian path problem. *SIAM J. Comput.*, 16:486–502, June 1987.

[4] Andreas Björklund and Thore Husfeldt. Exact Algorithms for Exact Satisfiability and Number of Perfect Matchings. *Algorithmica*, 52:226–249, 2008.

[5] Mikko Koivisto and Pekka Parviainen. A space-time tradeoff for permutation problems. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 484–492, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

[6] Kohn, Samuel and Gottlieb, Allan and Kohn, Meryle. A generating function approach to the traveling salesman problem. In *Proceedings of the 1977 annual conference*, ACM '77, pages 294–300, New York, NY, USA, 1977.

[7] Richard M. Karp. Dynamic Programming Meets the Principle of Inclusion and Exclusion. *Operations Research Letters*, 1(2):49 – 51, 1982.

[8] Eric Bax and Joel Franklin. A Finite-Difference Sieve to Count Paths and Cycles by Length. *Inf. Process. Lett.*, 60:171–176, November 1996.

[9] Andreas Björklund. Determinant Sums for Undirected Hamiltonicity. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS'10, pages 173–182, Washington, DC, USA, 2010. IEEE Computer Society.

[10] Gerhard J. Woeginger. Open Problems Around Exact Algorithms. *Discrete Appl. Math.*, 156:397–405, February 2008.

[11] David Eppstein. The traveling salesman problem for cubic graphs. In Frank Dehne, Jorg-Rudiger Sack, and Michiel Smid, editors, *Algorithms and Data Structures*, volume 2748 of *Lecture Notes in Computer Science*, pages 307–318. Springer Berlin / Heidelberg, 2003.

[12] Kazuo Iwama and Takuya Nakashima. An improved exact algorithm for cubic graph tsp. In Guohui Lin, editor, *Computing and Combinatorics*, volume 4598 of *Lecture Notes in Computer Science*, pages 108–117. Springer Berlin / Heidelberg, 2007.

[13] Heidi Gebauer. Finding and enumerating hamilton cycles in 4-regular graphs. *Theoretical Computer Science*, 412(35):4579–4591, August 2011.

[14] Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. The Travelling Salesman Problem in Bounded Degree Graphs. In Luca Aceto, Ivan Damgard, Leslie Goldberg, Magnus Halldorsson, Anna Ingolfsdottir, and Igor Walukiewicz, editors, *Automata, Languages and Programming*, volume 5125 of *Lecture Notes in Computer Science*, pages 198–209. Springer Berlin / Heidelberg, 2008.

[15] Sartaj Sahni and Teofilo Gonzalez. P-Complete Approximation Problems. *J. ACM*, 23:555–565, July 1976.

[16] M. R. Garey and D. S. Johnson. "Strong" NP-Completeness Results: Motivation, Examples, and Implications. *J. ACM*, 25:499–508, July 1978.

[17] Vijay V. Vazirani. *Approximation Algorithms*. Springer, 2003.

[18] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[19] Daniel J. Rosenkrantz, Richard Edwin Stearns, and Philip M. Lewis. An Analysis of Several Heuristics for the Traveling Salesman Problem. *SIAM J. Comput.*, 6(3):563–581, 1977.

[20] N Christofides. Worst-Case Analysis of a New Heuristic for the Traveling Salesman Problem. Technical Report 338, Graduate School of Industrial Administration, CMU, 1976.

[21] Christos Papadimitriou and Santosh Vempala. On The Approximability Of The Traveling Salesman Problem. *Combinatorica*, 26:101–120, 2006.

[22] A. M. Frieze, G. Galbiati, and F. Maffioli. On the Worst-Case Performance of Some Algorithms for the Asymmetric Traveling Salesman Problem. *Networks*, 12(1):23–39, 1982.

[23] Markus Bläser. A New Approximation Algorithm for the Asymmetric TSP with Triangle Inequality. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '03, pages 638–645, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[24] Haim Kaplan, Moshe Lewenstein, Nira Shafrir, and Maxim Sviridenko. Approximation Algorithms for Asymmetric TSP by Decomposing Directed Regular Multigraphs. *J. ACM*, 52:602–626, July 2005.

[25] Uriel Feige and Mohit Singh. Improved Approximation Ratios for Traveling Salesperson Tours and Paths in Directed Graphs. In Moses Charikar, Klaus Jansen, Omer Reingold, and Jose Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 4627 of *Lecture Notes in Computer Science*, pages 104–118. Springer Berlin / Heidelberg, 2007.

[26] Arash Asadpour, Michel X. Goemans, Aleksander Madry, Shayan Oveis Gharan, and Amin Saberi. An $O(\log n/\log\log n)$-approximation Algorithm for the Asymmetric Traveling Salesman Problem. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 379–389, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

[27] Sanjeev Arora. Polynomial Time Approximation Schemes for Euclidean TSP and Other Geometric Problems. In *In Proceedings of the 37th IEEE Symposium on Foundations of Computer Science (FOCS'96)*, pages 2–11, 1996.

[28] Sanjeev Arora. Nearly Linear Time Approximation Schemes for Euclidean TSP and Other Geometric Problems. In *In Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS'97)*, pages 554–563, 1997.

[29] Joseph S. B. Mitchell. Guillotine Subdivisions Approximate Polygonal Subdivisions: A Simple Polynomial-Time Approximation Scheme for Geometric TSP, k-MST, and Related Problems. *SIAM J. Comput.*, 28:1298–1309, March 1999.

[30] Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. A Randomized Rounding Approach to the Traveling Salesman Problem. In *FOCS*, pages 550–559, 2011.

[31] Tobias Mömke and Ola Svensson. Approximating Graphic TSP by Matchings. In *FOCS*, pages 560–569, 2011.

[32] Marcin Mucha. Improved Analysis for Graphic TSP Approximation via Matchings. *CoRR*, abs/1108.1130, 2011.

[33] Christos H. Papadimitriou and Mihalis Yannakakis. The Traveling Salesman Problem with Distances One and Two. *Math. Oper. Res.*, 18:1–11, February 1993.

[34] Piotr Berman and Marek Karpinski. 8/7-approximation Algorithm for (1,2)-TSP. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, SODA '06, pages 641–648, New York, NY, USA, 2006. ACM.

[35] N. Boria, N. Bourgeois, B. Escoffier, and V. Th. Paschos. Exponential approximation schemata for some network design problems. Cahier du LAMSADE 303, LAMSADE, Universite Paris-Dauphine, 2011.

[36] Oscar H. Ibarra and Chul E. Kim. Fast Approximation Algorithms for the Knapsack and Sum of Subset Problems. *J. ACM*, 22:463–468, October 1975.

[37] Eric T. Bax. Recurrence-Based Reductions for Inclusion and Exclusion Algorithms Applied to #P Problems. 1996.

[38] John Gallant, David Maier, and James A. Storer. On finding minimal length superstrings. *Journal of Computer and System Sciences*, 20(1):50 – 58, 1980.

[39] Z. Sweedyk. A $2\frac{1}{2}$-approximation algorithm for shortest superstring. *SIAM J. Comput.*, 29(3):954–986, December 1999.

[40] Marek Karpinski and Richard Schmied. Improved lower bounds for the shortest superstring and related problems. *CoRR*, abs/1111.5442, 2011.

[41] Virginia Vassilevska. Explicit inapproximability bounds for the shortest superstring problem. In Joanna Jedrzejowicz and Andrzej Szepietowski, editors, *Mathematical Foundations of Computer Science 2005*, volume 3618 of *Lecture Notes in Computer Science*, pages 793–800. Springer Berlin / Heidelberg, 2005.