Имплементация алгоритма Пападимитриу–Яннакакиса приближения задачи коммивояжёра для графа с весами рёбер 1 и 2 с точностью приближения $\frac{7}{6}$

Аннотация

В статье рассматривается реализация приближения частного случая задачи коммивояжёра для графов с весами рёбер 1 или 2. Имплементируется алгоритм Пападимитриу–Яннакакиса, позволяющий получить приближение с точностью $\frac{7}{6}$ в худшем случае. Проводится анализ точности приближения.

1. Введение

Задача коммивояжёра является одной из самых известных задач комбинаторной оптимизации. Цель задачи в отыскании кратчайшего пути, проходящего через заданные города и возвращающегося в исходный город. Сформулируем задачу на математическом языке:

Рассмотрим полный граф на n вершинах, где n — число городов. Поставим вершины графа в соответствие городам. Каждому ребру сопоставляется его вес (который в аналогии с городами и дорогами может восприниматься как время в пути, стоимость проезда, длина пути и т.д.). Таким образом, если между двумя городами нет пути, то соответствующему ребру можно присвоить максимальный вес. Задача коммивояжёра называется симметричной, если она моделируется неориентированным графом, и метрической, если она является симметричной и для любых трёх вершин i, j, k графа вес ребра (i, k) меньше или равен, чем сумма весов рёбер (i, j) и (j, k). В нашем случае рассматривается полный неориентированный граф с возможными значениями весов рёбер 1 или 2, то есть задача является симметричной и метрической. Решение задачи коммивояжёра в нашем случае — это нахождение гамильтонова цикла минимального веса. Гамильтоновым циклом называется цикл в графе, включающий каждую вершину графа ровно один раз. В действительности, мы хотим найти гамильтонов цикл с наименьшим числом рёбер веса 2. Задача, которую мы хотим решить, является **NP**-полной [PY, Ka]. Однако существуют полиномиальные алгоритмы, приближающие решение (например, с точностью $\frac{3}{2}$, [Ch]). Мы же представим алгоритм, описанный в [PY], приближающий решение с точностью $\frac{7}{6}$, с доказательством точности в худшем случае и анализом точности в среднем и в краевых случаях.

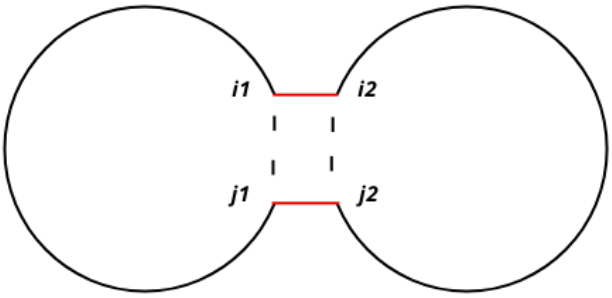
2. Описание алгоритма

2.1. Входные данные. Входные данные: симметричная матрица d размера $n \times n$ из 1 и 2, задающая граф $G = (V, E)$, где $V = \{1, 2, \ldots, n\}$ и для любых $i, j \in V$ ребро (i, j) имеет вес $d_{i,j}$.

Цель: найти циклическую перестановку $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$, минимизирующую $\sum_{i=1}^{n} d_{i,\pi(i)}$

2.2. Сведение задачи к объединению циклов. Найдём в графе минимальный по весу подграф, содержащий все вершины исходного графа, в котором степени всех вершин равны 2. Такой подграф можно найти, воспользовавшись алгоритмом, указанном в [H]. Более того, такой алгоритм позволяет найти оптимальный подграф, не содержащий треугольников. Обозначим этот подграф через G'. Заметим, что G' состоит из нескольких несвязанных циклов. Если цикл один, то это оптимальный гамильтонов цикл, а значит, мы решили задачу. Иначе объединим циклы в один. Перед тем, как описать процедуру объединения, посмотрим, как меняется вес подграфа при объединении двух циклов.

2.3. Объединение двух циклов. Рассмотрим два произвольных цикла. Выберем по ребру из каждого цикла: (i_1, j_1) и (i_2, j_2), удалим их, добавим ребра $(i_1, i_2), (j_1, j_2)$. Мы получили один цикл. При этом вес подграфа изменился на разницу весов двух добавленных и двух удалённых рёбер. Если в каком-то цикле изначально есть ребро веса 2, то выбираем его в качестве удаляемого: тогда вес увеличится не больше, чем на 1. Если же в обоих циклах есть ребра веса 2, то при их удалении вес точно не увеличится.

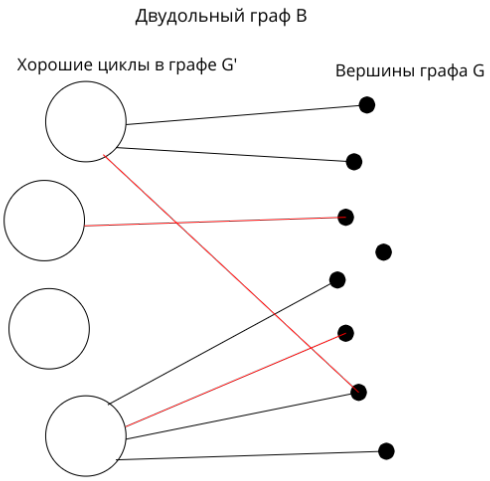


2.4. Хорошие циклы. Будем называть цикл хорошим, если он состоит только из рёбер с весами 1. Тогда, если в графе G был гамильтонов путь, проходящий только по ребрам веса 1, то граф G' состоит только из хороших циклов (Так как граф G' — оптимальный граф, в котором степени всех вершин равны 2, то его вес не превосходит вес любого гамильтонова цикла). Иначе,

можно считать, что в графе G' только один цикл нехороший, а также, что из вершин, инцидентных ребрам веса 2, не может идти ребро веса 1 в хороший цикл. Действительно, если есть такие ребра или более одного нехорошего цикла, то мы, не увеличивая вес подграфа, объединяем циклы: при объединении двух нехороших циклов мы удаляем два ребра веса 2, добавляем тоже два ребра веса 2 (иначе противоречие с оптимальностью G'); при объединении нехорошего и хорошего цикла, между которыми в графе есть ребро веса 1, мы удаляем ребро веса 2 и ребро веса 1 и добавляем ребро веса 1 и ребро веса 2 (второе добавляемое ребро будет иметь вес 2, иначе противоречие с оптимальностью G').

2.5. Построение двудольного графа. Обозначим множество циклов в G' через C. Построим двудольный граф B: вершины первой доли — хорошие циклы из C, вершины второй — вершины исходного графа G. Цикл $c \in C$ соединён с вершиной v второй доли в графе B, если v не принадлежит c и v соединена с некоторой вершиной цикла c ребром веса 1.

Теперь каждый цикл первой доли соединён с несколькими (0 или больше) вершинами графа G. Найдём в двудольном графе B максимальное паросочетание.



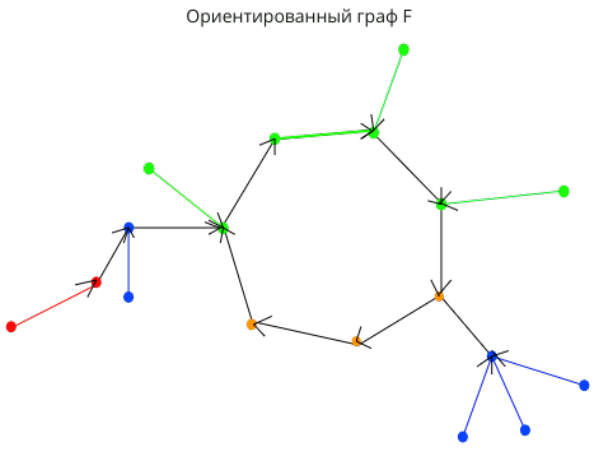
Если для цикла c ребро (c, v) принадлежит паросочетанию, то рассмотрим цикл c', которому принадлежит вершина v(этот цикл необязательно хороший).

Построим ориентированный граф F с вершинами C, в котором из каждого цикла $c \in C$, являющегося вершиной ребра в паросочетании, идёт ребро в соответствующий цикл c', описанный выше. Если цикл не является вершиной ребра в паросочетании, то из него не выходит ребро (Из нехорошего цикла по

построению графа B не может выходить ребро). Заметим, что каждое ориентированное ребро (c, c') графа F появилось, так как в графе G было ребро (v, v') веса 1, соединяющее циклы.

Лемма 1. *Лемма о разбиении. Построенный ориентированный граф F можно разбить на поддеревья глубины 1 и один путь длины 2.*

Доказательство. Рассмотрим все компоненты слабой связности по отдельности. Каждая такая компонента представляет из себя либо просто дерево, либо цикл, к некоторым вершинам которого подвешены деревья. Начнём с самых глубоких листов подвешенных деревьев (дерева, если цикла нет). Рассматриваем предка листа, для него берём всех потомков и отрезаем поддерево глубины 1. Продолжаем, пока не дойдём до вершины, предком которой является вершина цикла (если дерево одно, то мы дошли до корня и разбили граф на поддеревья глубины 1). Рассмотрим узлы цикла, к которым всё ещё что-то подвешено. Мы хотим отрезать поддерево глубины 1, при этом заметим, что для вершин цикла мы можем включать или не включать предыдущую по циклу вершину в список листов, отрезая поддерево. Тогда будем отрезать так, чтобы между двумя вершинами цикла, к которым ещё что-то подвешено, оставалось чётное (возможно нулевое) число вершин. Тогда от цикла останется несколько (возможно 0) путей чётной длины (по числу вершин) и не больше одного цикла нечётной длины. Циклы чётной длины разбиваем на пути длины 1 (то есть деревья глубины 1, состоящие из корня и одного листа), цикл нечётной - на пути длины 1 и один путь длины 2. ■



Теперь, когда мы разбили граф F на путь длины 2 и деревья глубины 1, можно объединять.

2.6. Объединение дерева из циклов глубины один. Как мы помним, каждое ориентированное ребро (c, c') графа F появилось, так как в графе G было ребро (v, v') веса 1, соединяющее циклы, причём, даже если в графе F есть два ребра (c_1, c') и (c_2, c'), то в цикле c' соответсвующие им вершины v'_1, v'_2 разные (так как мы строили максимальное паросочетание, то концы рёбер не могут совпадать). То есть все листья дерева(циклы) $c_1, \ldots, c_t$ соединены ребрами с разными вершинами корневого цикла. Обойдём корневой цикл против часовой стрелки, начиная с вершины, не соединённой ребром ни с каким циклом, если такая есть, иначе — с произвольной(направление обхода на самом деле неважно, так как у нас неориентированный граф G).

1) Если две подряд идущие вершины v_i, v_j соединены ребрами e_i, e_j с циклами c_i, c_j, то объединяем корневой цикл и циклы c_i, c_j в один, удаляя ребро (v_i, v_j) и добавляя ребро, соединяющее циклы (в цикле c_i выбираем следующую вершину после вершины, инцендентной e_i, по обходу по часовой стрелке, в c_j - после вершины, инцендентной e_j, против часовой). (Пример на рис.1).

2) Если в корневом цикле после вершины v_i идёт вершина v_j, не соединённая ребром с циклом, то объединяем корневой и c_i циклы, удаляя ребро (v_i, v_j), добавляя ребро e_i и ребро, соединяющее вершину v_j с вершиной цикла c_i, следующей по обходу по часовой стрелке за вершиной, инцендентной e_i, удаляя соответствующее ребро в цикле c_i. (Пример на рис.2)

2.7. Объединение трёх связанных циклов (пути длины 2). Обозначим циклы через c_1, c_2, c_3. Уже есть два ребра (v_1, v_2), (v'_2, v_3), соединяющие циклы c_1 и c_2, c_2 и c_3 (Возможно, вершины v_2 и v'_2 совпадают). Добавим ещё два ребра как на рис.3, удалим соответствующие ребра в циклах.

2.8. Завершение построения цикла. Объединяем все циклы, полученные для отдельных связных компонент, в один. Если остались изолированные циклы, то добавляем их к уже построенному как сказано в пункте 2.3. В итоге мы получили один цикл.(Если построенного цикла не оказалось, то у нас только хорошие циклы, между которыми в графе G только ребра веса 2, а значит можно просто соединить два цикла как в пункте 2.3.и считать, что это построенный цикл. В этом цикле точно будет ребро веса 2).

2.9. Доказательство точности. Пусть G' содержит k рёбер веса 2, где $0 \leq k \leq n$. Тогда вес оптимального гамильтонова цикла не меньше, чем $n+k$ (так как G' оптимальный граф, в котором степени всех вершин равны 2, то гамильтонов цикл не может весить меньше). Рассмотрим оптимальный гамильтонов цикл $(v_1, v_2, \ldots, v_n, v_1)$. С помощью него можно построить следующее паросочетание в графе B: цикл c соединим с вершиной v_{i+1}, если $v_i \in c$, $v_{i+1} \notin c$ и $d[i, i+1] = 1$.

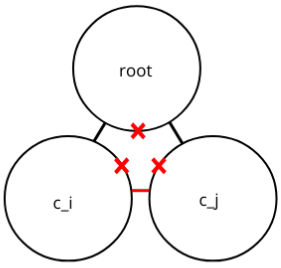


Рис. 1

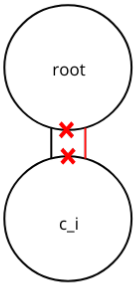


Рис. 2

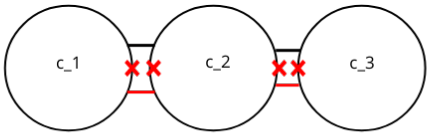


Рис. 3

Тогда два цикла точно не будут соединены с одной вершиной. Оптимальное паросочетание не может быть хуже данного. При таком построении число изолированных циклов в F не больше, чем количество вершин v_i в оптимальном гамильтоновом цикле, таких, что $d[i, i+1] = 2$. Обозначим число таких вершин через p. Тогда, очевидно, $n + p \leq n + k$. Обозначим через r количество циклов, из которых не выходит ребро в F, через m - суммарное число вершин в этих циклах.

Посмотрим, как изменяется вес при объединении неизолированных циклов (циклов в слабо связной компоненте F). При объединении трёх циклов, один из которых является корневым в дереве глубины 1, добавляется одно ребро и удаляется одно, вес увеличивается не больше, чем на 1 (на самом деле удаляются ещё два ребра из листовых циклов и добавляются два ребра, соответствующие ребрам ориентированного графа, но все эти ребра имеют вес 1, поэтому мы их не учитываем; в остальных случаях рассуждение аналогичное). При этом в листовых циклах больше или равно восьми вершин (так как изначально в C нет треугольников), то есть на каждую вершину приходится $\leq \frac{1}{8}$. При объединении двух циклов, один из которых - корневой, добавляется одно ребро, вес увеличивается не больше, чем на 1. Если рассмотреть листовой цикл и две вершины в корневом, к которым он присоединяется, то получается больше или равно шести вершин, то есть на каждую вершину приходится $\leq \frac{1}{6}$. При объединении пути длины 2 в F, то есть трёх последовательных циклов, добавляется два новых

ребра, вес увеличивается не больше, чем на 2, при этом вершин больше или равно двенадцати, то есть на каждую вершину приходится $\leq \frac{1}{6}$.

Во время объединения неизолированных циклов количество вершин, для которых увеличивается вес, приходящийся на них, не больше $n - m - k$, так как даже если нехороший цикл неизолированный, то в нём не увеличивается вес, приходящийся на вершины, инцидентные ребрам веса 2, в силу того, что такие вершины не связаны с хорошими циклами ребрами веса 1 по построению. При объединении циклов, полученных для слабо связных компонент, суммарный вес не увеличивается по пункту 2.3, так как каждый цикл имеет ребро веса 2 (если это не так, то можно заменить ребро веса 1 на ребро веса 2, так как мы закладываем такую возможность при подсчёте оценки для объединения неизолированных циклов). При добавлении изолированных циклов: добавление хорошего цикла увеличивает вес не больше, чем на 1, так как в уже построенном цикле уже есть ребро веса 2; добавление нехорошего цикла не увеличивает вес, так как в нём тоже есть ребро веса 2. Тогда для общего веса полученного цикла выполнено:

$$cost \leq n + k + \frac{1}{6}(n - m - k) + r$$

где $n + k$ — вес графа G', $\frac{1}{6}(n - m - k)$ — увеличение веса при объединении неизолированных циклов в F (так как вес, приходящийся на каждую вершину, увеличивается не больше, чем 1 раз), r — увеличение веса от хороших изолированных циклов. Тогда, так как $r \leq \frac{m}{4}$ и $r \leq p$, то

$$cost \leq \frac{7}{6}n + \frac{5}{6}k - \frac{m}{6} + r \leq \frac{7}{6}max(n + p, n + k)$$

то есть данный алгоритм действительно приближает задачу коммивояжёра с точностью не хуже, чем $\frac{7}{6}$.

3. Код алгоритма

Код алгоритма можно посмотреть по ссылке:

https://github.com/artyomnurgaliev/TSPApproximation.

Представление графа: Graph.h

Представление двудольного графа и поиск максимального паросочетания: BipartiteGraph.h

Представление цикла: Cycle.h

Представление ориентированного графа, поиск компонент слабой связности и циклов: DirectedGraph.h

Основной класс решения: TSPApproximation.h

В основном классе: на вход подаётся представление графа в виде матрицы смежности и набор циклов, образующих минимальный по весу подграф, содержащий все вершины исходного графа, в котором степени всех вершин равны 2. Происходит первоначальное объединение циклов, описанное в пункте 2.4. Строится двудольный граф. Находится максимальное паросочетание. Строится ориентированный граф, вершинами которого являются циклы. Ориентированный граф разбивается по лемме о разбиении. Происходит объединение циклов.

4. Тестирование

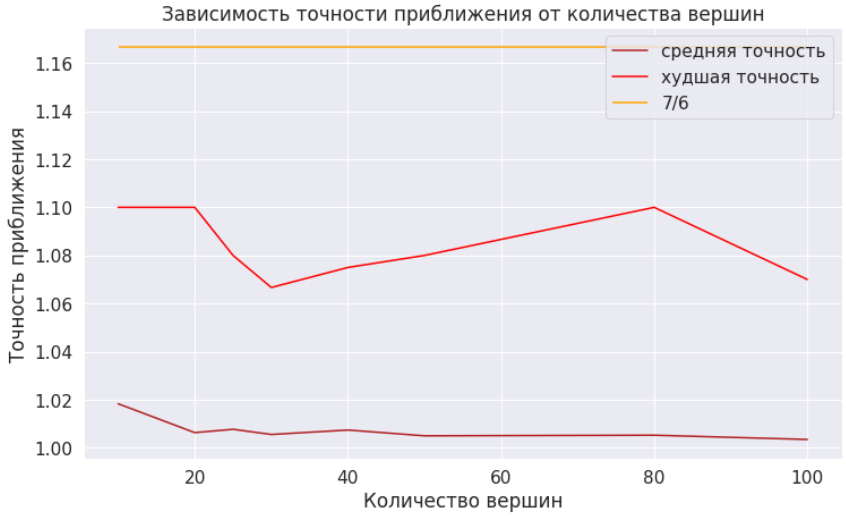
Код тестов можно посмотреть в tests.ipynb и main.cpp

4.1. Идея. Тестировать алгоритм будем следующим образом: так как находить истинное решение задачи коммивояжёра для больших графов довольно долго, будем действовать другим способом. Берём граф, в котором веса всех рёбер равны 2. Рассматриваем произвольную циклическую перестановку вершин графа (цикл). Очевидно, что для полного графа, веса всех рёбер которого одинаковы, этот цикл - решение задачи. Положим веса некоторых рёбер, принадлежащих решению, равными единице. Цикл при этом останется решением задачи. Разобъём цикл на несколько циклов, проведя несколько раз операцию, обратную объединению. При этом изменим веса добавляемых рёбер таким образом, чтобы разность весов двух добавляемых и двух удаляемых рёбер при каждом разбиении была равна нулю. Получили минимальный по весу подграф, содержащий все вершины исходного графа, в котором степени всех вершин равны 2(набор циклов). Запускаем алгоритм и, зная вес оптимального цикла, понимаем точность приближения для конкретного графа. (Примечание: заметим, что если уменьшать веса рёбер так, что разность весов двух добавляемых и двух удаляемых рёбер будет меньше нуля, то изначальный цикл может перестать быть решением задачи). Очевидно, что если изначальный цикл будет содержать много рёбер веса 2, то даже обычное последовательное попарное объединение циклов даст хорошую оценку точности. Поэтому особый интерес представляют случаи, когда все или практически все рёбра изначального цикла имеют единичный вес.

Исходя из всего написанного выше, проведём следующие тесты (посмотрим на среднюю и наихудшую точности приближения):

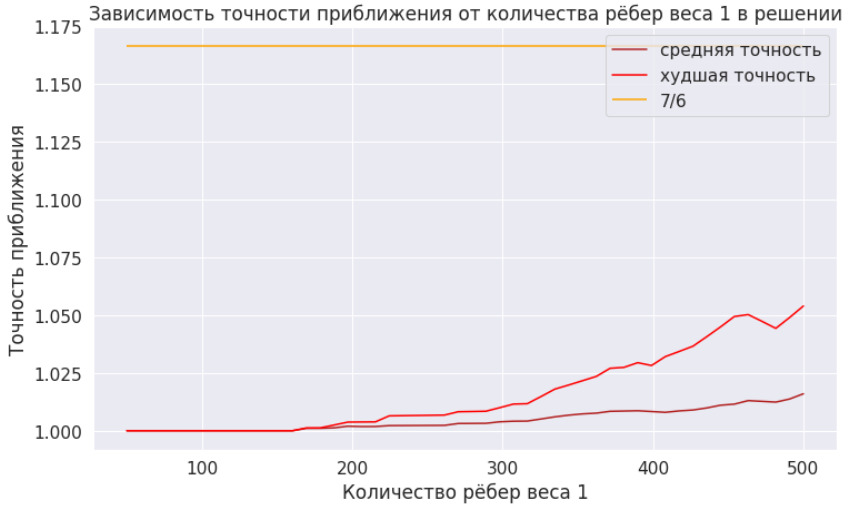
- Зависимость точности приближения от количества вершин.
- Зависимость точности приближения от количества рёбер веса 1 в решении при фиксированном числе вершин.
- Зависимость точности приближения от количества циклов при фиксированном числе вершин и рёбер веса 1 в решении.

4.2. Зависимость точности приближения от количества вершин. Для каждого количества вершин переберём некоторые значения числа рёбер веса 1 и количества циклов. Посчитаем среднюю и худшую точность.



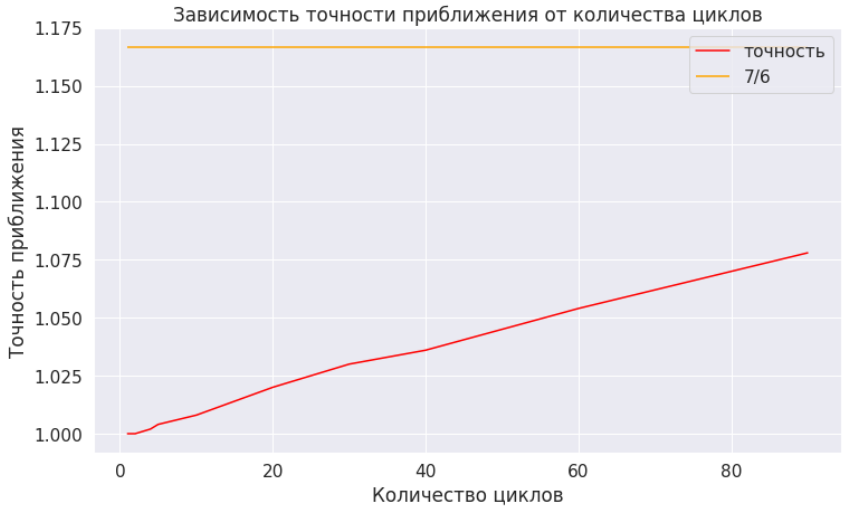
Естественно, перебирая варианты для числа рёбер веса 1 и количества циклов, мы не можем перебрать все возможные варианты. Поэтому рассматриваем некоторые средние значения и важные краевые случаи, такие как: цикл целиком состоит из рёбер веса 1, в цикле нет рёбер веса 1, количество циклов равно одному, количество циклов равно максимально возможному. Смотря на график, делаем вывод, что в среднем алгоритм хорошо приближает решение, особенно вспоминая о том, что краевые случаи, которые в данном случае вносят существенный вклад, в реальности встречаются намного реже. Худшая точность не превышает $\frac{7}{6}$, чего мы и ожидаем.

4.3. Зависимость точности приближения от количества рёбер веса 1 в решении при фиксированном числе вершин. В предыдущем случае мы перебирали значительное количество вариантов, поэтому для адекватного времени тестирования брали графы с числом вершин, не превыщающим 100. В данном случае можем взять большее число вершин (берём 500). Гамильтонов цикл, содержащий все вершины данного графа, имеет 500 рёбер, поэтому максимальное число рёбер в этом цикле, имеющих вес 1, равно 500. Ожидаем, что чем больше рёбер веса 1, тем хуже точность приближения, так как по алгоритму вес решения увеличивается только в случае, когда добавляются рёбра веса 2 вместо рёбер веса 1 (средняя точность берётся по различным разбиениям гамильтонова цикла на циклы).



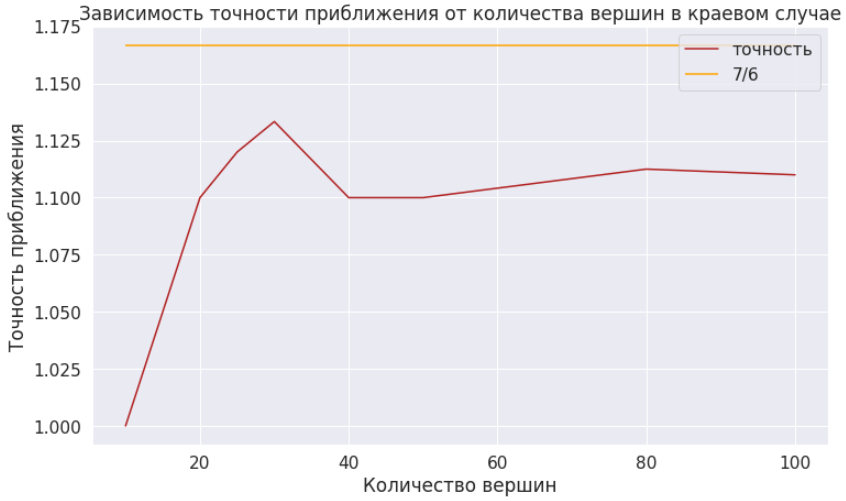
Глядя на график, понимаем, что наше предположение подтвердилось: и средняя, и худшая точность становятся хуже при увеличении числа рёбер веса 1.

4.4. Зависимость точности приближения от количества циклов при фиксированном числе вершин и рёбер веса 1 в решении. В предыдущем пункте выяснили, что наихудшая точность в случае, когда число рёбер веса 1 в цикле максимально. Рассмотрим этот случай. Так как вес решения может увеличиться только при объединении циклов, то ожидаем, что чем больше циклов, тем хуже точность приближения.



Действительно, из графика следует, что чем больше циклов в наборе циклов, подаваемых на вход алгоритму, тем хуже точность приближения.

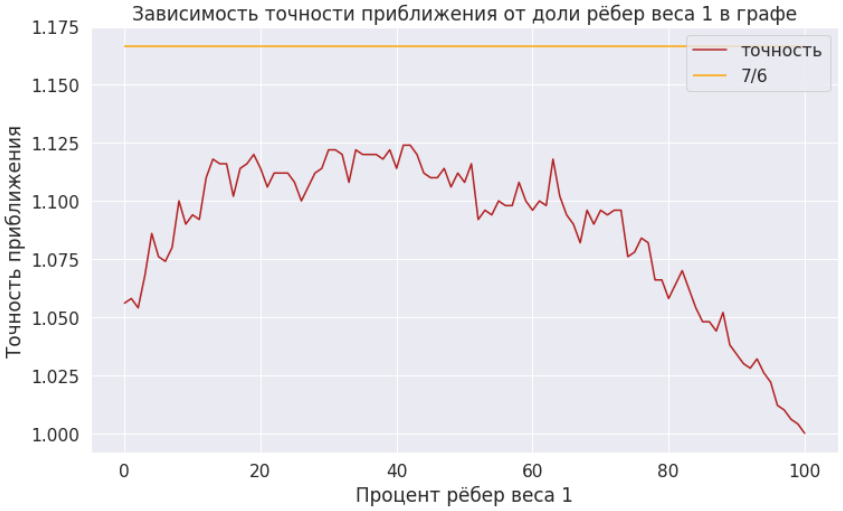
4.5. Зависимость точности приближения от количества вершин в краевом случае. Теперь, когда мы выяснили, что худшая точность достигается при наибольшем числе рёбер веса 1 в истинном решении и наибольшем числе циклов, рассмотрим график зависимости точности приближения от количества вершин только в этом случае.



Действительно, видим, что начиная с некоторого момента точность в этом случае и худшая точность из первого теста практически совпадают.

4.6. Тест с хорошим гамильтоновым циклом. Заметим, что если гамильтонов цикл, являющийся решением задачи, является хорошим, то мы можем изначально заменить вес любой доли рёбер на 1, не поменяв при этом вес решения. Посмотрим на зависимость точности приближения от доли хороших рёбер в графе в данном случае. Ожидаем, что чем когда в графе много рёбер веса 1 и ещё больше рёбер веса 2, то большая вероятность поменять ребро веса 1 на ребро веса 2 и тем самым увеличить вес решения.

На самом деле доля рёбер веса 1 в графе не может быть меньше, чем доля рёбер хорошего гамильтонова цикла от всех рёбер. В нашем случае вершин в графе 500, а значит рёбра гамильтонова цикла составляют $\frac{500}{(500*499)/2} = 0.004$. Заметим, что наше предположение оказалось верно: действительно, худшая точность при доле рёбер веса 1 от 0.2 до 0.6.



5. Заключение

Алгоритм в среднем даёт хорошее приближение задачи коммивояжёра, а в худшем случае приближает с точностью не хуже, чем заявленная $(\frac{7}{6})$

Ссылки

PY. Christos H. Papadimitriou, Mihalis Yannakakis. *The Traveling Salesman Problem with Distances One and Two* Mathematics of Operations Research , Feb., 1993, Vol. 18, No. 1 (Feb., 1993), pp. 1-11

Ka. Karp, R. M.. *Reducibility among Combinatorial Problems. in Complexity of Computer Computations*. R. E. Miller, and J. W. Thatcher (Eds.), Plenum, New York (1972)

Ch. Christofides, N. *Worst-Case Analysis of a New Heuristic for the Traveling Salesman Problem*. Technical Report, GSIA, Carnegie-Mellon University (1976)

H. Hartvigsen, D. B. *Extensions of Matching Theory*. Ph.D. Thesis, Carnegie-Mellon University (1984)