# Исследование эволюционных процессов белков, генов и межгенных областей с помощью графов

Выполнили студенты группы 6306
Колышкин Е.С. и Атрошенко Я.П.

# Цель

Составить программу, которая на основе информации о генетическом коде белков формирует граф, преобразовывает его в филогенетическое дерево и строит эволюционную цепочку. А также визуализирует и анализирует построенные графы.

# Справка о биоинформатике

Биоинформа́тика — совокупность методов и подходов, включающих в себя:

- математические методы компьютерного анализа в сравнительной геномике (геномная биоинформатика).
- разработку алгоритмов и программ для предсказания пространственной структуры биополимеров (структурная биоинформатика).
- исследование стратегий, соответствующих вычислительных методологий, а также общее управление информационной сложности биологических систем

Главная задача биоинформатики — способствовать пониманию биологических процессов. Поэтому одной из основных целей биоинформатики является построение филогенетических деревьев.

# Филогенетические деревья

Филогенетическое дерево (эволюционное дерево, дерево жизни) — дерево, отражающее эволюционные взаимосвязи между различными видами

Такие деревья делятся на 2 типа:

1)Укоренённое дерево — дерево, содержащее выделенную вершину — корень.

2)Неукоренённое дерево не содержит корня и отражает связь листьев без предполагаемого положения общего предка.

# Структура генетического кода белков

Белки практически всех живых организмов построены из аминокислот 20 видов.

Каждый белок представляет собой цепочку аминокислот, соединенных в строго определенной последовательности. Эта последовательность определяет строение белка и все его биологические свойства.

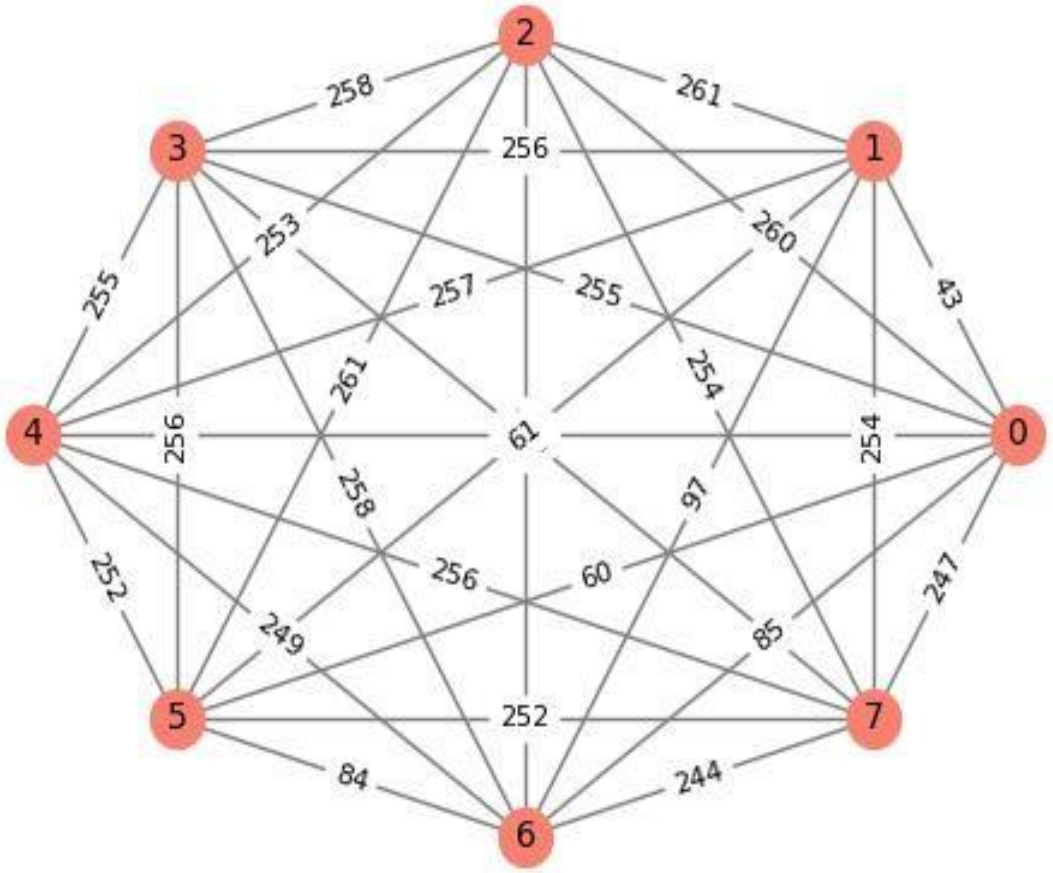
Мы используем первичную структуру белка, которая представляется в виде строки, содержащей только заглавные латинские буквы. Каждый символ такой строки является аминокислотой.

которая кодируется от 1 до 6 кодонами(триплетами), которые в

свою очередь являются последовательностью из 3 нуклеотидов.

Пример: CUG – каждая буква в этой последовательности является нуклеотидом, всю такую последовательность принято называть кодоном или триплетом. Дальше одним или несколькими кодонами кодируется одна аминокислота. Любая аминокислота представляется одной латинской буквой(первая буква её названия). Таким образом получается цепочка, составляющая генетический код белка.
Пример такой цепочки:
MAAIAFIGLGQMGSPMASNLLQQGHQLRVFDVNAEAVRHLVDKGATPAANP
AQAAKDAEF…
С данной строкой довольно просто работать. Сравнивая такие цепи поэлементно можно определить
генетическую разницу между ними.

# Граф




Вершины этого графа символизируют белки,
а ребра генетическую разницу между ними.
Веса ребер были получены посимвольным сравнением строк, содержащих генетический код этих белков.

# Принцип построения филогенетического дерева

Алгоритм, широко применяющийся в биоинформатике - Метод невзвешенного попарного среднего -UPGMA(Unweighted Pair Group Method with Arithmetic Mean).

Перед началом работы алгоритма рассчитывается матрица расстояний между объектами. Примем каждую вершину за кластер, тогда расстояние между такими кластерами будет определяться соответствующим значением в матрице весов. На каждом шаге в матрице расстояний ищется минимальное значение, соответствующее расстоянию между двумя наиболее близкими кластерами.

Найденные кластеры u и v объединяются, образуя новый кластер k.

Строки и столбцы, соответствующие кластерам u и v, выбрасываются из матрицы расстояний, и добавляется новая строка и новый столбец, соответствующие кластеру k. В результате матрица сокращается на одну строку и один столбец. Эта процедура повторяется до тех пор, пока размерность матрицы расстояний больше 1x1.

Пусть кластеры u, v и k содержат $T_u$ , $T_v$ и $T_k$ объектов, соответственно. Кластер k образован путем объединения кластеров u и v, тогда $T_k = T_u + T_v$ . Необходимо рассчитать удаленность кластера k от некоторого кластера w. Расстояние между этими кластерами определяется согласно формуле:

$$D\Big(\big(u,v\big),w\Big) = \frac{T_u D_{u,w} + T_v D_{v,w}}{T_u + T_v}$$

Где k = (u,v)

и записывается в матрицу, в качестве новой строки и нового столбца, соответствующего кластеру k.

Пример шага алгоритма: Пусть нам дана матрица весов:

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 2.06 | 4.03 | 6.32 | 2.08 |
| 2 | 2.06 | 0 | 3.5 | 4.12 | 5.43 |
| 3 | 4.03 | 3.5 | 0 | 2.25 | 3.65 |
| 4 | 6.32 | 4.12 | 2.25 | 0 | 4.81 |
| 5 | 2.08 | 5.43 | 3.65 | 4.81 | 0 |

Объединение происходит между кластерами, расстояние между которыми наименьшее. На этом шаге объединяются кластеры 1 и 2. Расстояние объединения – 2.06. Необходимо произвести перерасчет матрицы расстояний с учетом нового кластера:

| | 1,2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 1,2 | 0 | 3.765 | 5.22 | 3.755 |
| 3 | 3.765 | 0 | 2.25 | 3.65 |
| 4 | 5.22 | 2.25 | 0 | 4.81 |
| 5 | 3.755 | 3.65 | 4.81 | 0 |

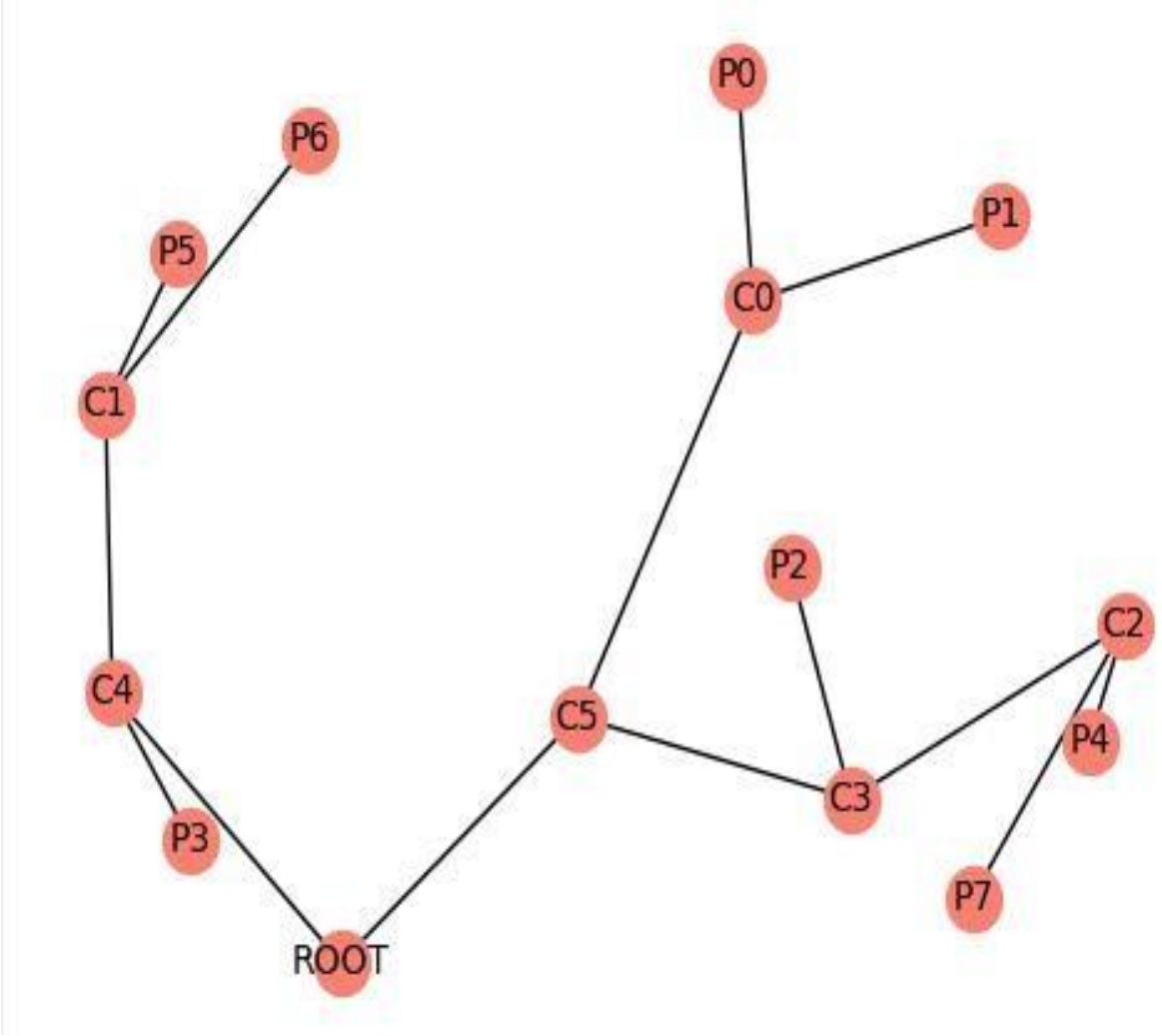
Приведем пример расчета расстояния между кластерами k = (1,2) и w = 3.

Кластер k образован путем объединения кластеров u = 1 и v = 2. Расстояния $D(u,w)$ и $D(v,w)$ берем из начальной матрицы расстояний. Подставив полученные значения в формулу, получим:

$$D = \frac{1 \times 4.03 + 1 \times 3.50}{1 + 1} = 3.765$$

# Дерево

В результате работы алгоритма UPGMA строится дерево:

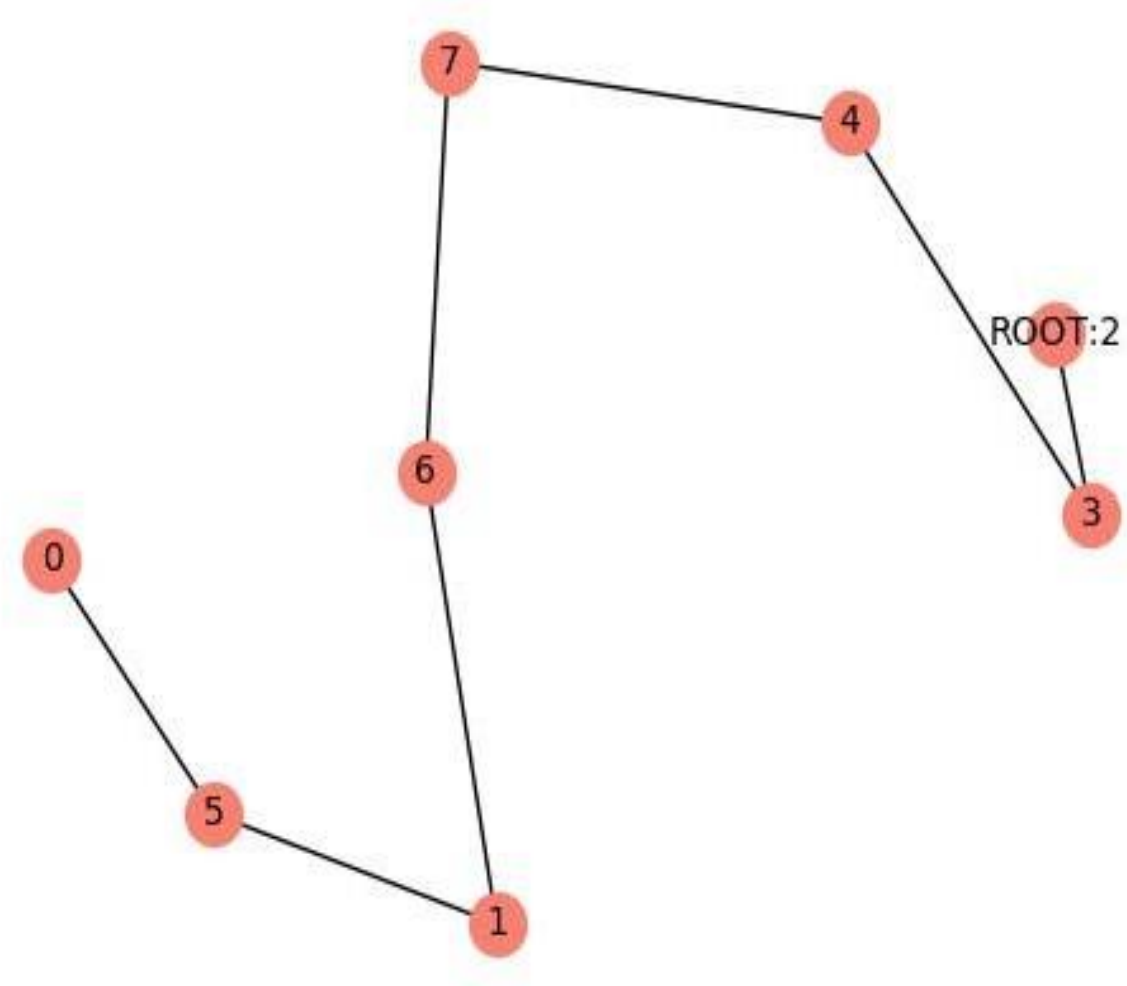



Вершины с индексами $P_n$ являются исходными белками, а вершины с индексами $C_n$ являются предками этих белков. Их мы получаем на каждом шаге алгоритма, в результате объединения исходных вершин в кластер. Вершина "ROOT" является общим предком для всех белков. Она получается на последнем шаге алгоритма, когда мы объединили 2 последних кластера.

# Преимущества UPGMA

- В отличие от других алгоритмов построения филогенетических деревьев UPGMA строит укорененное дерево
- Достаточно высокая скорость алгоритма
- Многочисленные тесты показывали правдоподобность деревьев, построенных с помощью этого алгоритма

# Построение эволюционной цепочки




Вершинами графа являются исходные белки
Цепочка построена по разработанному нами алгоритму.
На каждом шаге определяется вершина с наибольшей суммарной генетической разницей относительно остальных, такая вершина добавляется в цепочку и исключается из выборки.
Первая, найденная таким образом вершина получает приписку "ROOT".
Алгоритм продолжает работать, пока все вершины не будут добавлены в цепочку.
Эта цепочка позволяет определить порядок образования белков.

# Центральность вершин

Центральность относится к группе метрик, целью которых является определение «значительности» или «влияния» (в различных значениях) определённого общего предка в филогенетическом дереве. В нашем случае алгоритм позволяет определить роль того или иного предка в эволюции исходных белков.

# Описание алгоритма

Центральностью (англ. betweenness centrality) вершины v графа G = (V, E) называется величина

$$C_B(v) = \sum_{s \neq t \neq v \in V} \frac{\sigma_{st}(v)}{\sigma_{st}},$$

где $\sigma_{st}$ – число различных кратчайших путей в графе G от вершины s к вершине t, а $\sigma_{st}(v)$ – число таких путей, проходящих через вершину v.

Для реализации алгоритма мы используем немного модернизированный алгоритм Поиска в ширину. Этот алгоритм находит кратчайшее расстояние между вершинами графа, нам удобно, что бы он еще и запоминал вершины, через которые проходят кратчайшие расстояния.

Алгоритм вычисления центральности вершин будет доработан и лучше документирован, в ближайшем будущем.

# Алгоритм Дейкстры

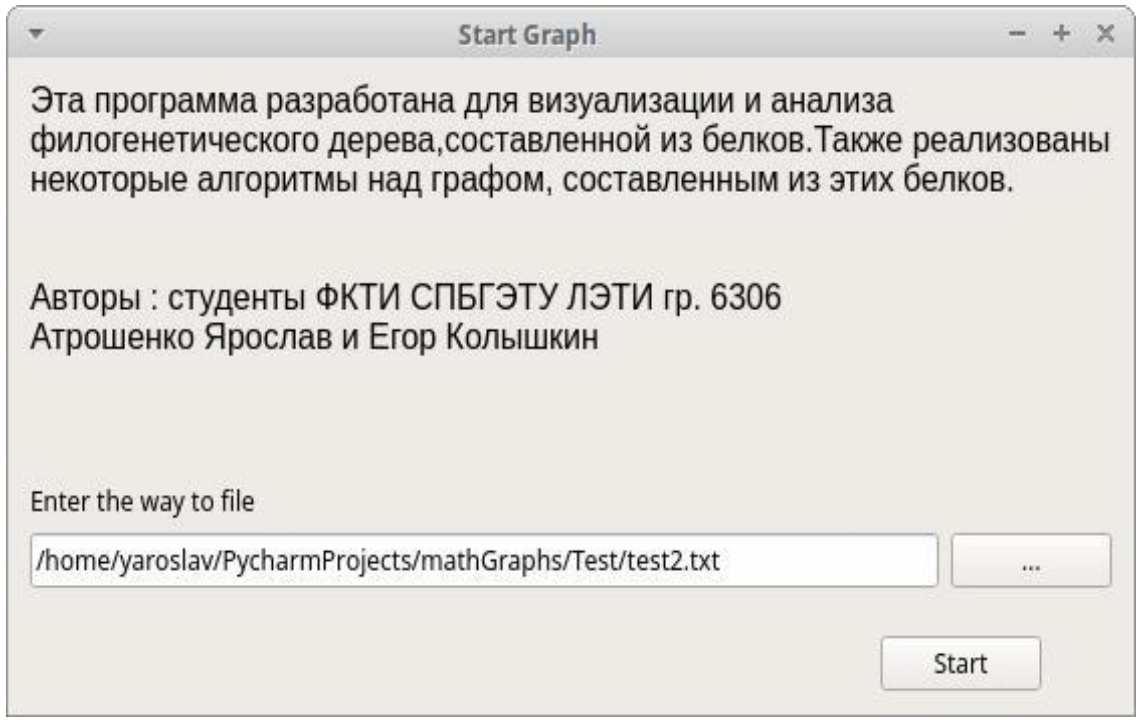
С помощью данного алгоритма строится эволюционная цепочка. Алгоритм вычисляет кратчайшее расстояние от одной вершины графа до всех остальных.

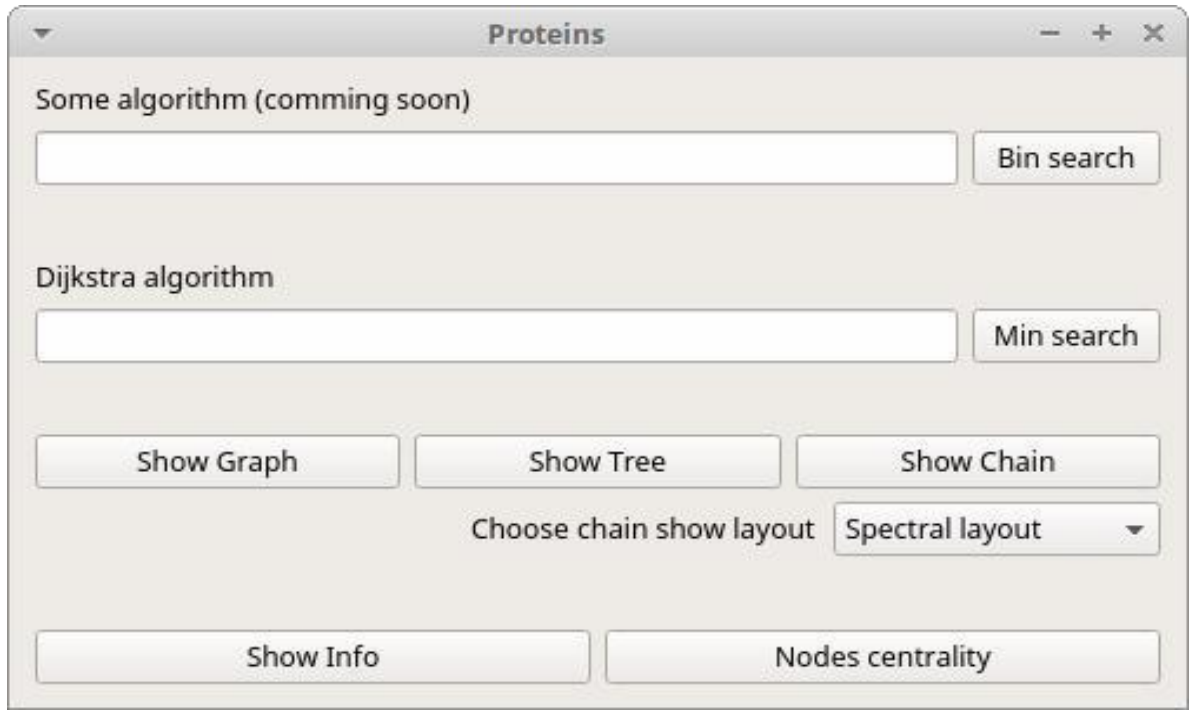
Каждой вершине сопоставим метку — минимальное известное расстояние от этой вершины до a. Алгоритм работает пошагово — на каждом шаге он «посещает» одну вершину и пытается уменьшать метки. Работа алгоритма завершается, когда все вершины посещены.

**Инициализация**. Метка самой вершины ***a*** полагается равной 0, метки остальных вершин — бесконечности. Это отражает то, что расстояния от ***a*** до других вершин пока неизвестны. Все вершины графа помечаются как непосещённые.

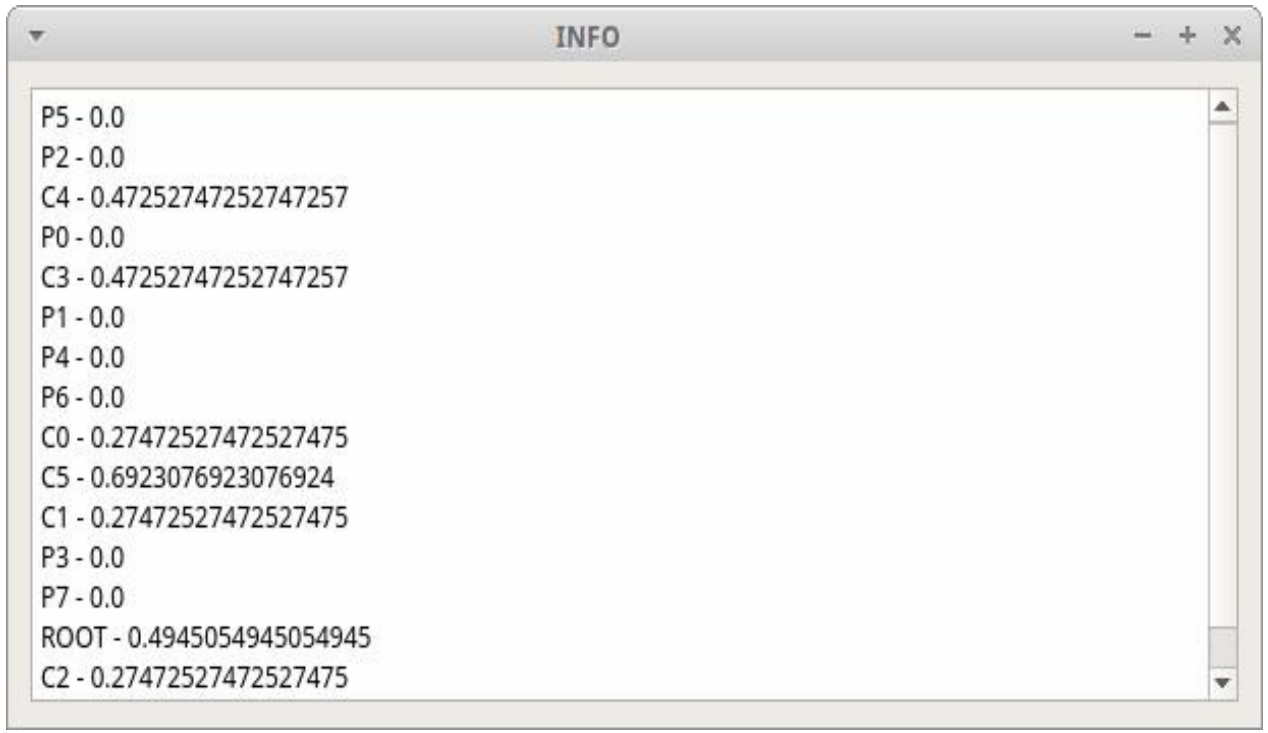
**Шаг алгоритма**. Если все вершины посещены, алгоритм завершается. В противном случае, из ещё не посещённых вершин выбирается вершина ***u***, имеющая минимальную метку. Мы рассматриваем всевозможные маршруты, в которых ***u*** является предпоследним пунктом. Вершины, в которые ведут рёбра из ***u***, назовём *соседями* этой вершины. Для каждого соседа вершины ***u***, кроме отмеченных как посещённые, рассмотрим новую длину пути, равную сумме значений текущей метки ***u*** и длины ребра, соединяющего ***u*** с этим соседом. Если полученное значение длины меньше значения метки соседа, заменим значение метки полученным значением длины. Рассмотрев всех соседей, пометим вершину ***u*** как посещённую и повторим шаг алгоритма.

# Скриншоты работы программы



Центральность вершин




Описание белков

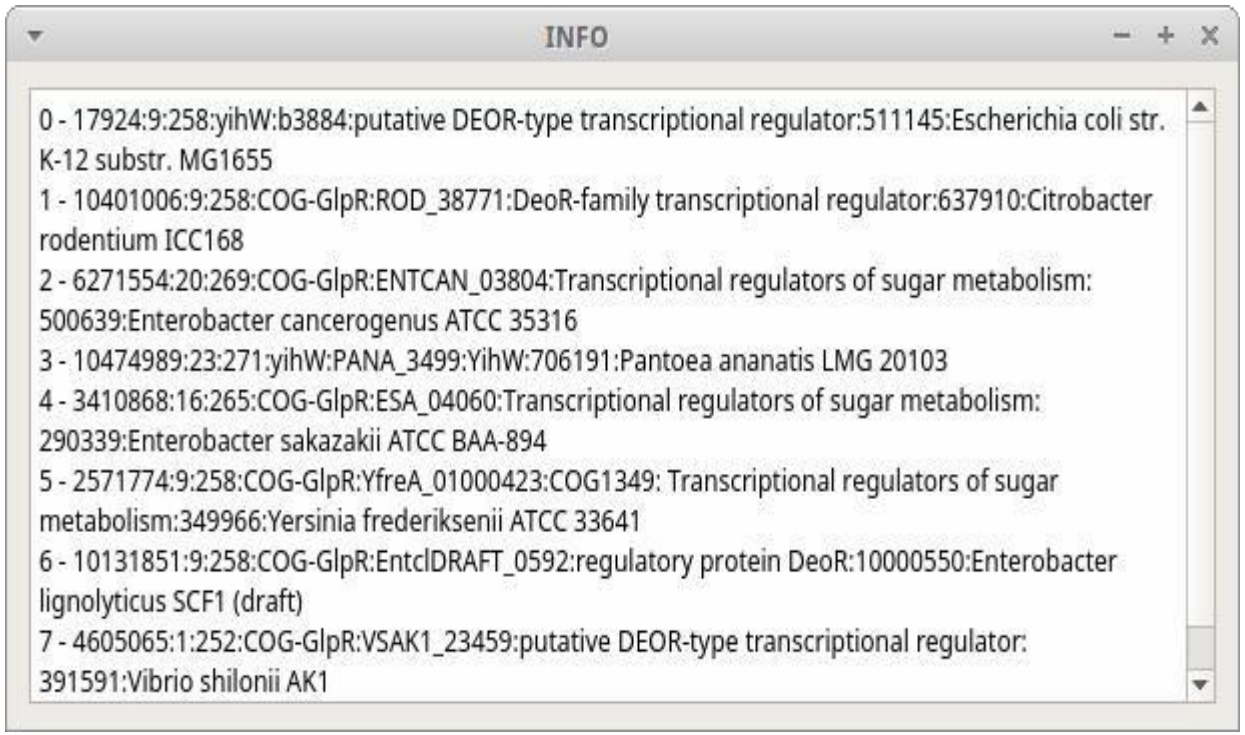


Figure 1

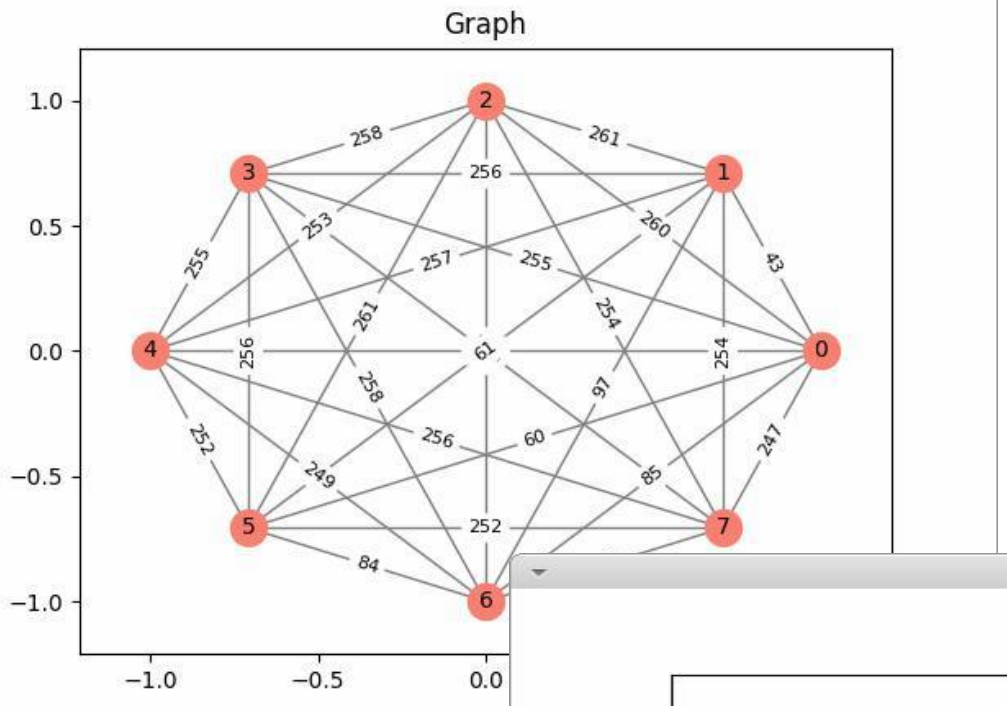
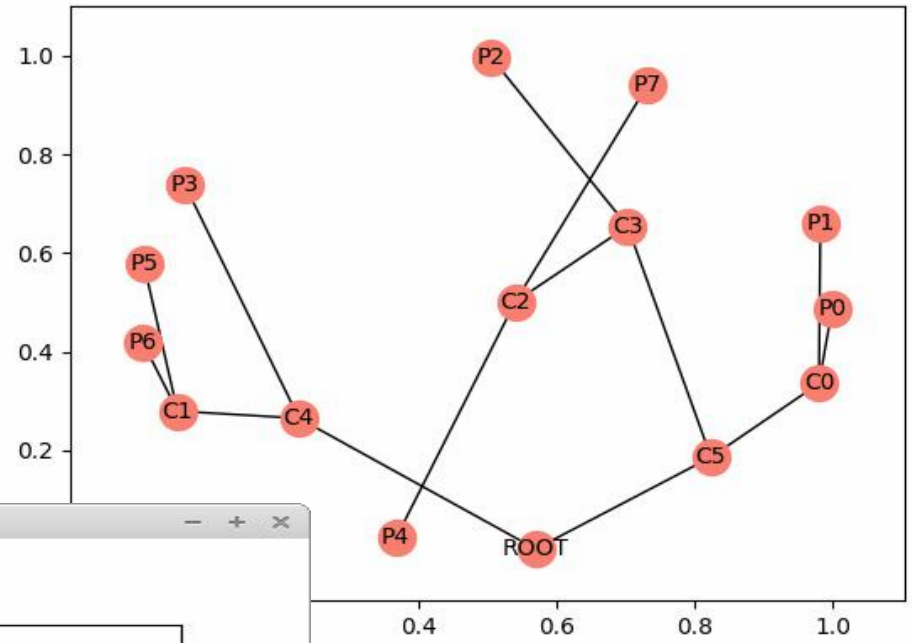
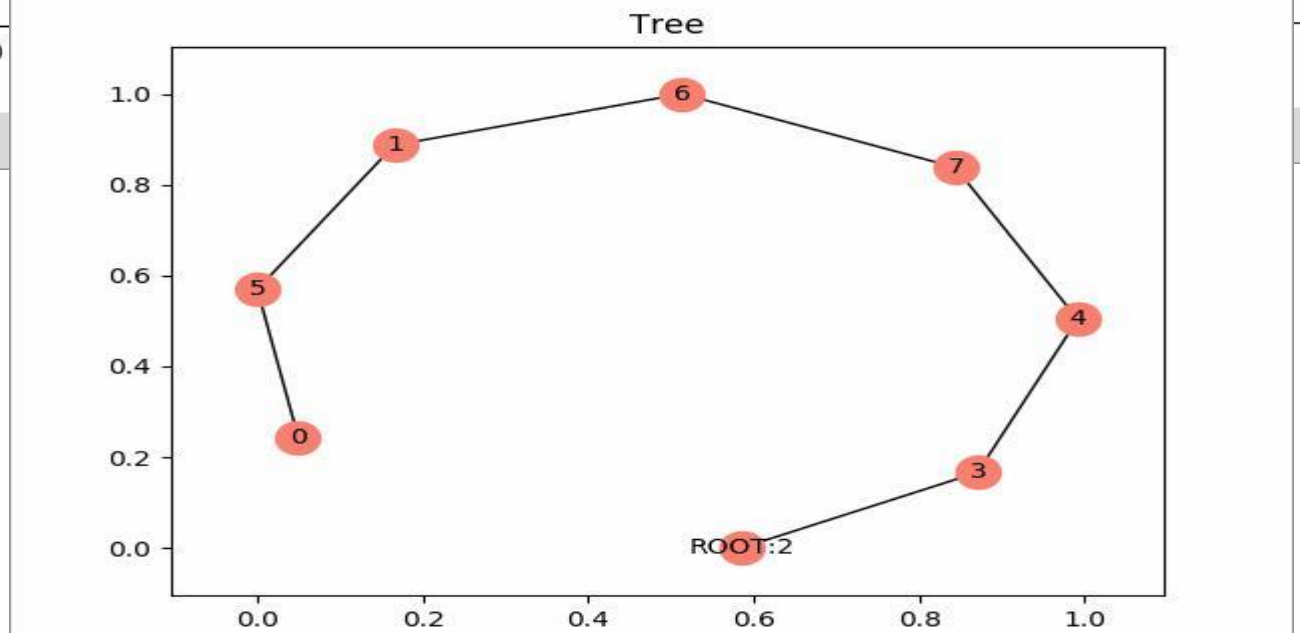



Figure 1




x=0.831774 y=0.983374

Figure 1



# Выводы

Мы познакомились с новой для нас областью, биоинформатикой, с методами хранения, анализа и визуализации графов. Применили полученные знания на практике, реализовав алгоритмы: UPGMA, алгоритм вычисления централности вершин, алгоритм Дейкстры. Разработали и реализовали алгоритм построения эволюционной цепочки. Существенно улучшили свои навыки в программировании. Постепенно улучшая нашу программу, можно создать инструмент для генетического и эволюционного анализа самых разных организмов.

# Использованные технологии