Лабораторная работа № 7

“Исследование алгоритмов определения минимального остовного дерева”

Задание:

1. Разработать проект для исследования алгоритмов определения минимального остовного дерева (МОД) в соответствии с вариантом (таблица 1).
2. Создать класс для исследования алгоритмов построения МОД со следующими полями, свойствами, и методами:
   - число вершин графа n,
   - число ребер графа m,
   - метод задания XY-координат вершин графа,
   - матрица длин ребер полносвязного графа,
   - метод генерации случайного связного графа на n вершинах и m ребрах,
   - матрица длин ребер сгенерированного графа,
   - матрица смежности графа (или/и любая другая структура для представления графа),
   - очередь по приоритетам длин ребер сгенерированного графа (простейший вариант) - возвращает из оставшихся ребер ребро минимальной длины,
   - метод, выводящий информацию о графе - число вершин, число ребер, матрица длин ребер, матрица смежности, матрица списка ребер и т.д.
   - метод проверки связности графа,
   - метод определения МОД алгоритмом Краскала (метод возвращает список ребер МОД),
   - метод проверки ацикличности графа,
   - метод определения безопасности вставляемого ребра для алгоритма Краскала,
   - метод определения МОД алгоритмом Прима (метод возвращает список ребер МОД),
   - метод определения безопасности вставляемого ребра для алгоритма Прима,
   - метод определения длины МОД на основании списка ребер МОД,
   - метод, выводящий информацию о МОД - число вершин, список ребер МОД, список длин ребер МОД, длина МОД.
3. Разработать интерфейс проекта, *позволяющий*:
   - задавать размерность графа;
   - осуществлять выбор алгоритма определения МОД для исследования;
   - задавать координаты вершин графа;
   - выводить и редактировать координаты вершин графа;

- задавать количество ребер графа X (в процентах от максимально возможного числа ребер – $X=m/m_{\max}*100\%$) (граф должен оставаться связным);
- выводить информацию о графе;
- выводить информацию о построенном МОД (список ребер и длину МОД);
- осуществлять вывод информации о результатах исследования алгоритмов (значения показателя качества работы алгоритмов).

4. Создать проект, реализующую алгоритмы определения МОД в соответствии с вариантом. В проекте *предусмотреть*:
   - реализацию интерфейса по п. 2;
   - сравнительную оценку исследуемых алгоритмов;
5. Для формирования координат вершин графа использовать датчик равномерно распределенных чисел *rand*().
6. Результатом работы алгоритма определения МОД является список вершин и длина остовного дерева.

<u>Письменный отчет</u> по лабораторной работе должен содержать:

1. Титульный лист. (Название лабораторной работы. Фамилия, имя, отчество, номер группы исполнителя, дата сдачи.)
2. Математическую постановку задачи.
3. Выполняемый вариант и его содержание.
4. Диаграмму классов.
5. Исходные тексты разработанных программ (*обязательны комментарии*).
6. Исследование программной реализации алгоритмов, содержащее следующие материалы, таблицы и графики:

- *примеры пошаговой работы* исследуемых алгоритмов Краскала и Прима для небольшой размерности задачи (n = 7), также для исходного графа привести матрицы смежностей, длин ребер, списка ребер, списка длин ребер, инцидентности (в соответствии с вариантом – таблица 2);
- сведенную в *таблицу* зависимость количества операций (*Op*) (или времени выполнения Т) исследуемых алгоритмов от размерности задачи n при различных X;

Пример таблицы:

Таблица 1 Алгоритм Краскала, количество ребер X=50%

n	50	…	500
T, ms		…	

- *графики* зависимости показателя качества (количества операций или времени выполнения) исследуемых алгоритмов от размерности задачи n при различных X.

7. Количество операций (или время счета) исследуемого алгоритма определяется как математическое ожидание *для числа испытаний не менее 100*.

8. Привести результаты тестирования проекта (вывод информации о графе, вывод информации о МОД, построенном различными алгоритмами для n = 7).

9. *Выводы* по лабораторной работе (в выводах провести сравнительную характеристику исследованных алгоритмов определения кратчайшего остовного дерева).

Варианты задач по лабораторной работе:

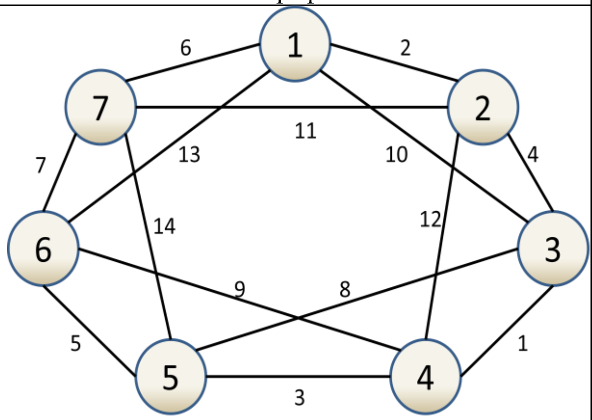
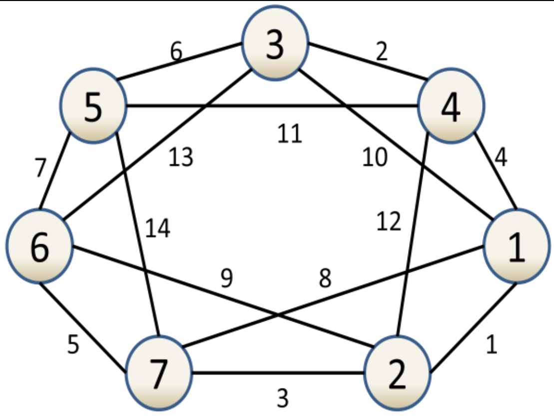
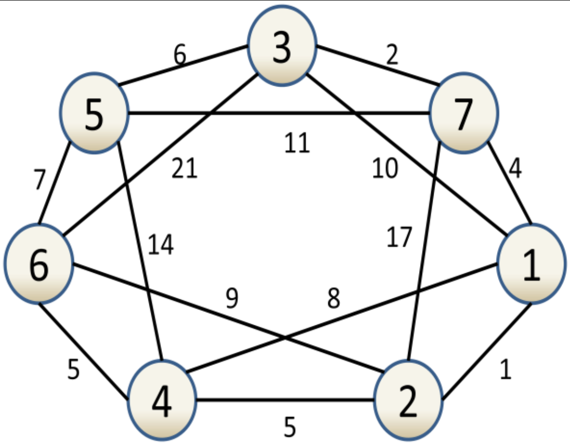
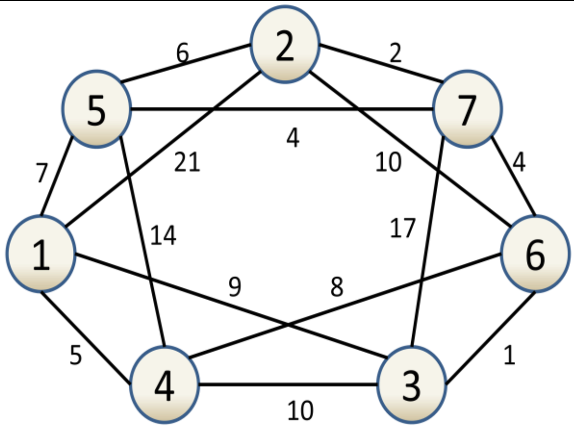
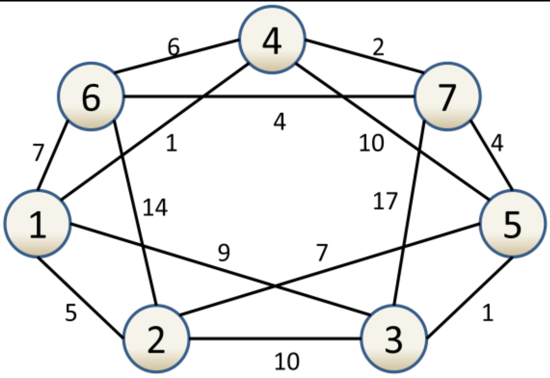
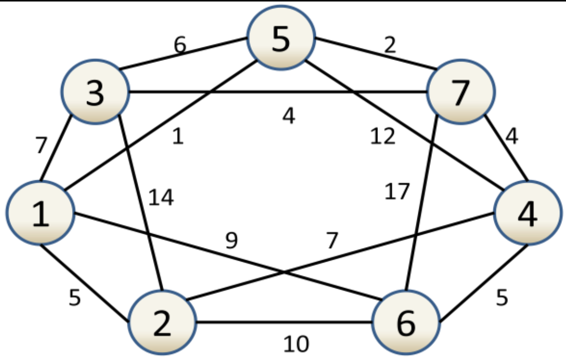
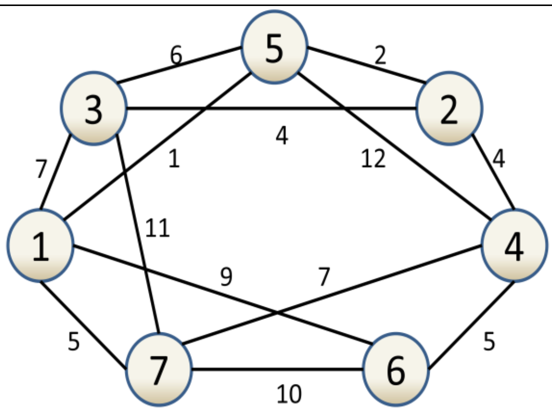
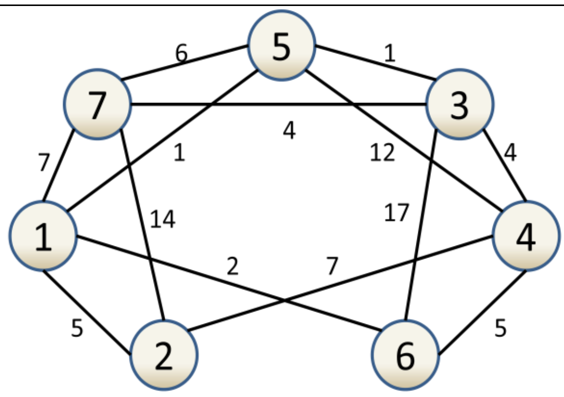
1. Алгоритм определения МОД Краскала.
2. Алгоритм определения МОД Прима.

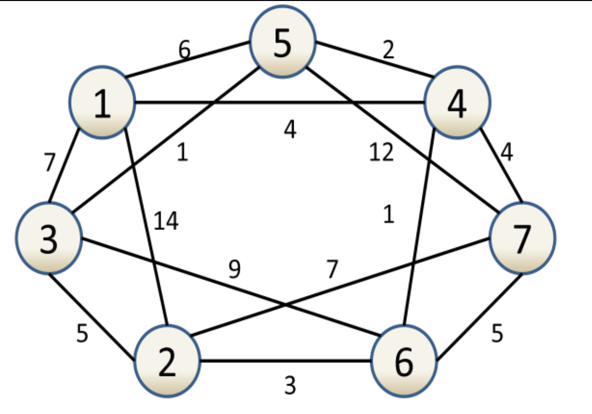
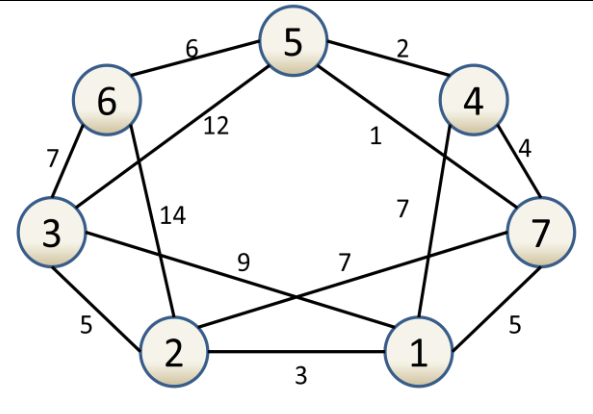
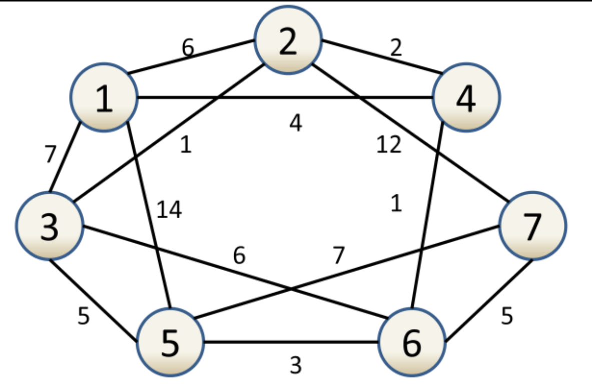
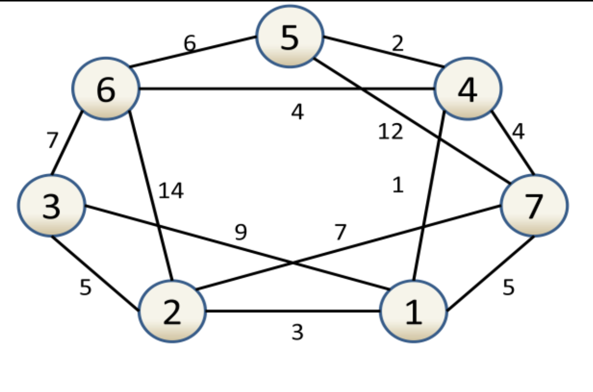
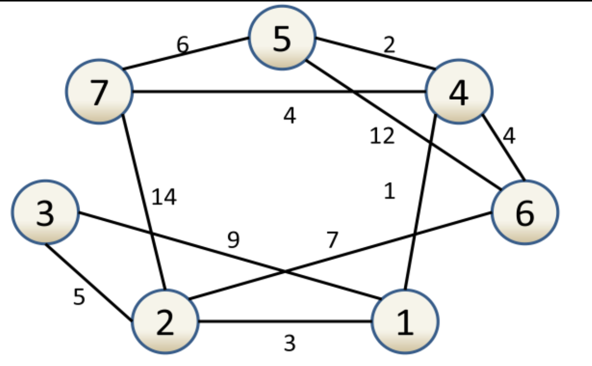
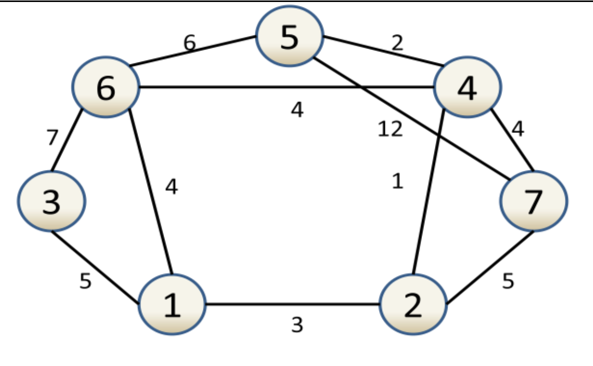
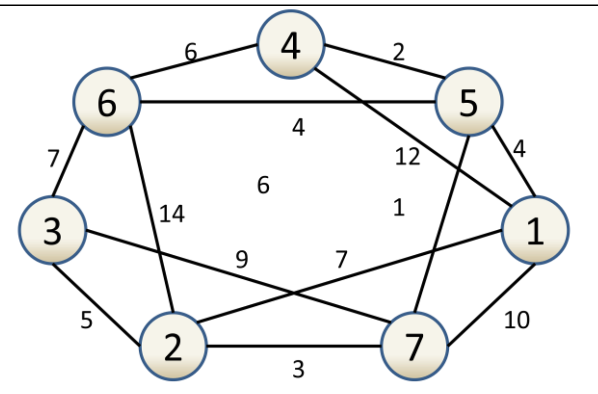
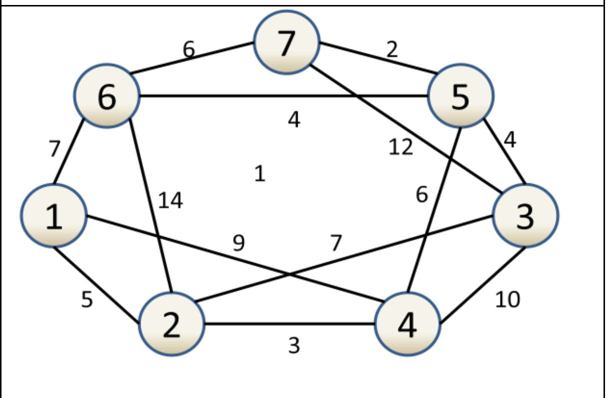
Таблица 1. – Варианты заданий

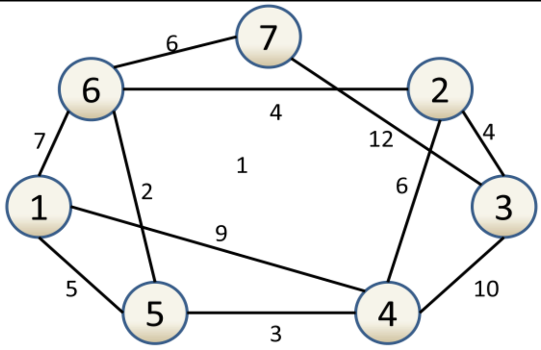
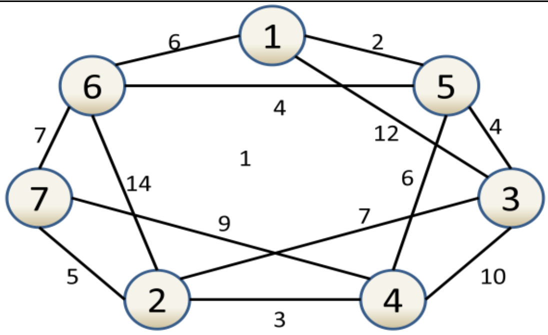
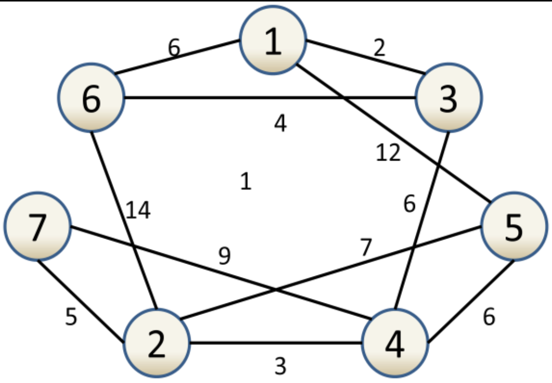
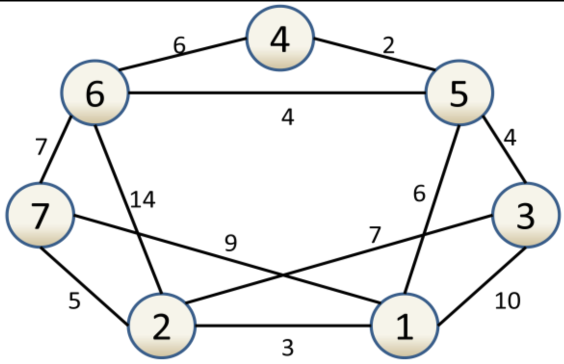
Вариант	*Координаты X, Y*	*X, %*	*Варианты задач*	*Вариант*	*Координаты X, Y*	*X, %*	*Варианты задач*
1	100, 100	50, 60, 70	1, 2	11	200, 600	50, 60, 70	1, 2
2	100, 200	60, 70, 80	1, 2	12	200, 800	60, 70, 80	1, 2
3	100, 300	50, 60, 70	1, 2	13	400, 800	50, 60, 70	1, 2
4	100, 500	55, 65, 75	1, 2	14	800, 400	55, 65, 75	1, 2
5	200, 100	60, 70, 80	1, 2	15	500, 1000	60, 70, 80	1, 2
6	300, 100	50, 60, 70	1, 2	16	1000, 500	50, 60, 70	1, 2
7	400, 100	55, 65, 75	1, 2	17	1000, 800	55, 65, 75	1, 2
8	500, 100	65, 75, 85	1, 2	18	1000, 900	65, 75, 85	1, 2
9	200, 200	50, 60, 70	1, 2	19	1000, 700	50, 60, 70	1, 2
10	200, 400	55, 65, 75	1, 2	20	600, 1000	55, 65, 75	1, 2

X, Y = (100, 200) – координаты вершин графа распределены в прямоугольнике 100 (по оси OX) на 200 (по оси OY).

Таблица 2. – Варианты заданий для пошаговой работы

№	Граф	№	Граф
1		2	
3		4	
5		6	
7		8	

9		10	
11		12	
13		14	
15		16	

17		18	
19		20	

<u>Литература</u>

1. Майника Э. Алгоритмы оптимизации на сетях и графах.-М.:Мир, 1981.-323с.(с.23-29).
2. Зубов В.С. Справочник программиста. Базовые методы решения графовых задач и сортировки.-М.: Филинъ,1999.-256с.(с.163-165).
3. Гудман С., Хидетниеми С. Введение в разработку и анализ алгоритмов.-М.:Мир, 1981.-(с.181-196).
4. Филлипс Д., Гарсиа-Диас А. Методы анализа сетей.-М.:Мир, 1984.-496с.(с.103-с.108).
5. Бондарев В.М., Рублинецкий В.И., Качко Е.Г. Основы программирования.-Харьков:Фолио;1998. –368с.(с.125-с.130).
6. Лекции по теории графов/Емеличев В.А. и др..-М.:Наука, 1990.-384с.(с.59-с.63).

Контрольные вопросы

1. Каково определение графа ?
2. Какими свойствами обладает обыкновенный граф ?
3. Какие вершины называются смежными
4. Какая часть графа называется подграфом ?
5. Какая часть графа называется суграфом ?
6. Каково определение степени вершины графа ?
7. Каково определение вектора степеней вершин графа ?
8. Как задается маршрут в неорграфе ?
9. Какова длина маршрута в неорграфе ?
10. Какой маршрута называется цепью ?
11. Какой маршрута называется простой цепью ?
12. Какой маршрута называется циклическим ?
13. Каково определение гамильтонова цикла ?
14. Каково определение эйлерова цикла ?
15. Каково определение матрицы смежностей графа?
16. Каково определение матрицы инцидентности графа?
17. Как определяется граф с помощью матрицы списка ребер?
18. Как определяется граф с помощью списка ребер ?
19. Как определяется граф с помощью списка смежных вершин?

20. Какие операции могут быть выполнены над абстрактным графом ?
21. В чем состоит задача поиска кратчайшего остовного (покрывающего) дерева?
22. В чем состоит алгоритм генерации случайного связного графа на n вершинах и m ребрах?
23. В чем состоит алгоритм Краскала решения задачи поиска КОД?
24. В чем состоит алгоритм проверки безопасности очередного вставляемого ребра в алгоритме Краскала?
25. Какова сложность алгоритм Краскала?
26. В чем состоит алгоритм Прима решения задачи поиска КОД?
27 В чем состоит алгоритм проверки безопасности очередного вставляемого ребра в алгоритме Прима?
28. Какова сложность алгоритм Прима ?
29. В чем состоит алгоритм проверки связности графа?
30. Каково определение очереди с приоритетом?

Краткий теоретический материал

Задача поиска минимального остовного (покрывающего) дерева

Пусть дан связный неориентированный граф G=(X, U) (X – множество вершин графа, U – множество ребер графа), для которого заданы веса (стоимости) всех ребер, представляемые вещественными числами.

Остовным (покрывающим – spanning tree) деревом S=(X, T) (T - множество ребер дерева) графа G называется неориентированное дерево, содержащее все узлы графа. Вес (стоимость) остовного дерева S=(X, T) графа G определяется как сумма весов ребер, входящих в остовное дерево.

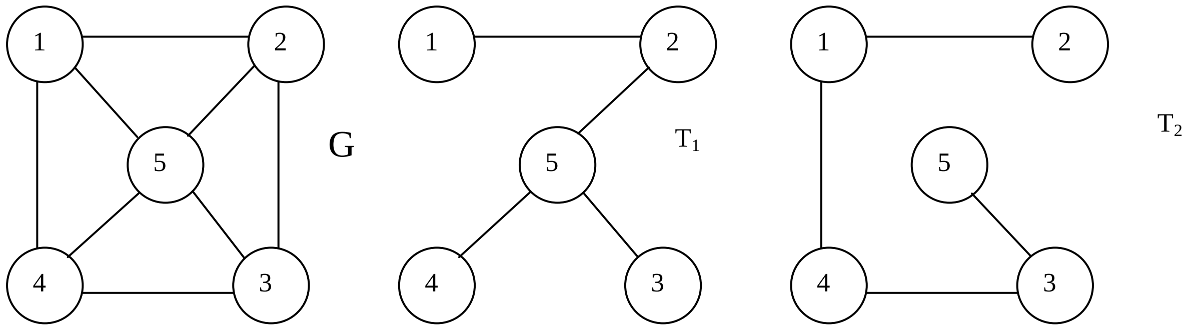


Рисунок 1. Исходный взвешенный граф G и два его остовных дерева T_1 и T_2.

Задача построения минимального остовного дерева (МОД) состоит в отыскании остовного дерева минимального веса в графе G=(X, U).

Так как в соответствии с теоремой Кэли полносвязный граф с n вершинами имеет n^{n-2} различных остовных деревьев, то алгоритм полного перебора для нахождения остовного дерева минимального веса будет иметь экспоненциальную временную сложность, то есть $O(n^{n-2})$. Однако задача построения МОД считается одной из самых легких оптимизационных задач на графах, так как точное решение этой задачи можно получить с помощью одного из “*жадных*” полиномиальных алгоритмов – алгоритма Краскала или алгоритма Прима.

1. Алгоритм Краскала построения МОД

Алгоритма Краскала основан на следующей лемме:

Пусть G=(X, U) – связный неориентированный граф и S=(X, T) – остовное дерево для него. Тогда а) для любых двух узлов x_1, $x_2 \in X$ путь между x_1и x_2 в S единственен; б) если к S добавить ребро из (U - T), то возникнет ровно один цикл.

Алгоритма Краскала предусматривает:

1) последовательный выбор ребер исходного графа с наименьшим весом;

2) ребра, образующие в процессе построения циклы, в остовное дерево не включаются;

3) процес построения дерева продолжается, пока не будут выбраны m=n-1 (m – число ребер в остовном дереве; n – число вершин исходного графа G) ребер.

Рассмотрим пример построения МОД с помощью алгоритма Краскала.

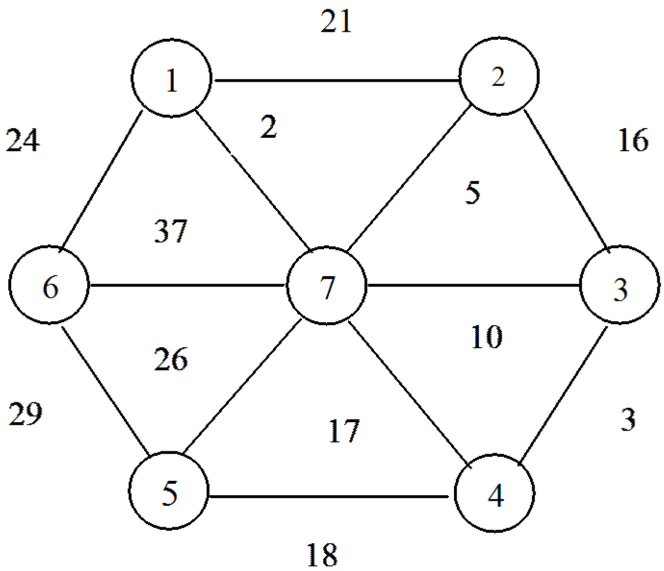


Рисунок 2. Исходный взвешенный граф G типа “*колесо*”.

Отсортируем ребра графа G в порядке возрастания их весов (Таблица 1). (На самом деле ребра не сортируются, а хранятся в виде сортирующего дерева, 2-3-дерева или какой-нибудь другой приемлемой структуры данных, пока они не потребуются. Кроме того, число ребер, выбираемых для построения остовного дерева равно $\|T\|$, что меньше $\|U\|$, то есть множество ребер U никогда не упорядочивается полностью.)

Таблица 1 – Ребра графа *G* в порядке возрастания их весов (рис. 2)

№	*Ребро*	*Стоимость*
1	(x_1, x_7)	2
2	(x_3, x_4)	3
3	(x_2, x_7)	5
4	(x_3, x_7)	10
5	(x_2, x_3)	16
6	(x_4, x_7)	17
7	(x_4, x_5)	18
8	(x_1, x_2)	21
9	(x_1, x_6)	24
10	(x_5, x_7)	26
11	(x_5, x_6)	29
12	(x_6, x_7)	37

Таблица 2 – Процесс построения МОД на графе *G* (рис. 2)

Шаг	*Ребро*	*Действие*	*Связные подграфы*
0	---	---	1,2,3,4,5,6,7
1	(1, 7)	Добавить	{1,7},2,3,4,5,6
2	(3, 4)	Добавить	{1,7},2,{3,4},5,6
3	(2, 7)	Добавить	{1,2,7},{3,4},5,6
4	(3, 7)	Добавить	{1,2,3,4,7},5,6
5	(2, 3)	Отвергнуть	{1,2,3,4,7},5,6
6	(4, 7)	Отвергнуть	{1,2,3,4,7},5,6
7	(4, 5)	Добавить	{1,2,3,4,5,7},6
8	(1, 2)	Отвергнуть	{1,2,3,4,5,7},6
9	(1, 6)	Добавить	{1,2,3,4,5,6,7}

Первоначально множество связных подграфов графа *G* содержит каждую вершину графа в виде одноэлементного множества, то есть мощность множества равна числу вершин в графе *G*. Наименьший вес имеет ребро (1, 7), следовательно, оно добавляется к дереву и одноэлементные множества {1} и {7} сливаются в множество {1, 7}. На четвертом шаге добавляется ребро (3, 7), после чего только вершины 5 и 6 еще не вошли в остовное дерево. На пятом шаге следующее по минимальности ребро (2, 3) не включается в остовное дерево, так как его включение приведет к образованию цикла. После 9-го шага множество связных подграфов содержит все 7 вершин исходного графа *G*, а остовное дерево содержит m=6 ребер, то есть алгоритм Краскала закончил свою работу. Минимальное остовное дерево для графа *G* изображено на рис. 3, вес построенного остовного дерева минимален и равен 62.

Сложность алгоритма Краскала - $O(k\cdot\log k)$, где k – количество ребер в данном графе. Алгоритм Краскала лучше работает с графами, слабо насыщенными ребрами.

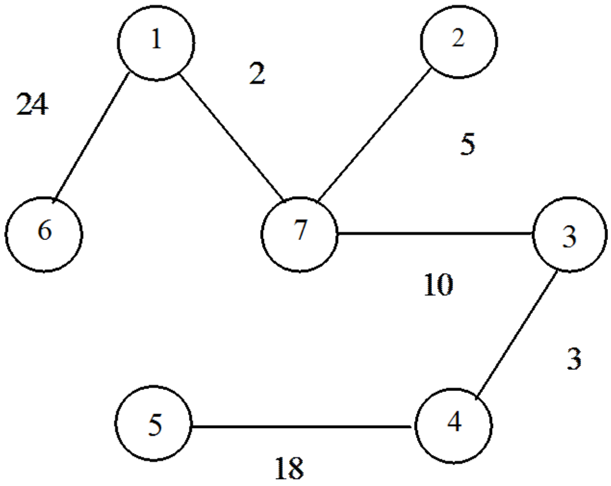


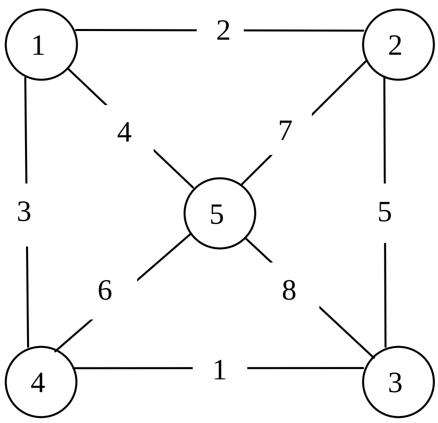
Рисунок 3. Минимальное остовное дерево веса 62 для графа G.

2. Алгоритм Прима построения МОД

Алгоритм Прима состоит в следующем:

1. Из ребер графа G выбираем ребро минимального веса, пусть это будет ребро e_1 между вершинами a и b, то есть $e_1= (a, b)$, и строим на нем дерево T_1, полагая, что множество вершин этого дерева – $X\ T_1=\{a, b\}$,а множество ребер $UT_1=\{e_1\}$.
2. Если число ребер дерева $i<n-1$, то среди ребер, соединяющих вершины T_i с вершинами графа G, не входящими в T_i, выбираем ребро e_{i+1} минимального веса. Строим дерево T_{i+1} , присоединяя к T_i ребро e_{i+1} вместе с его не входящими в T_i концами.

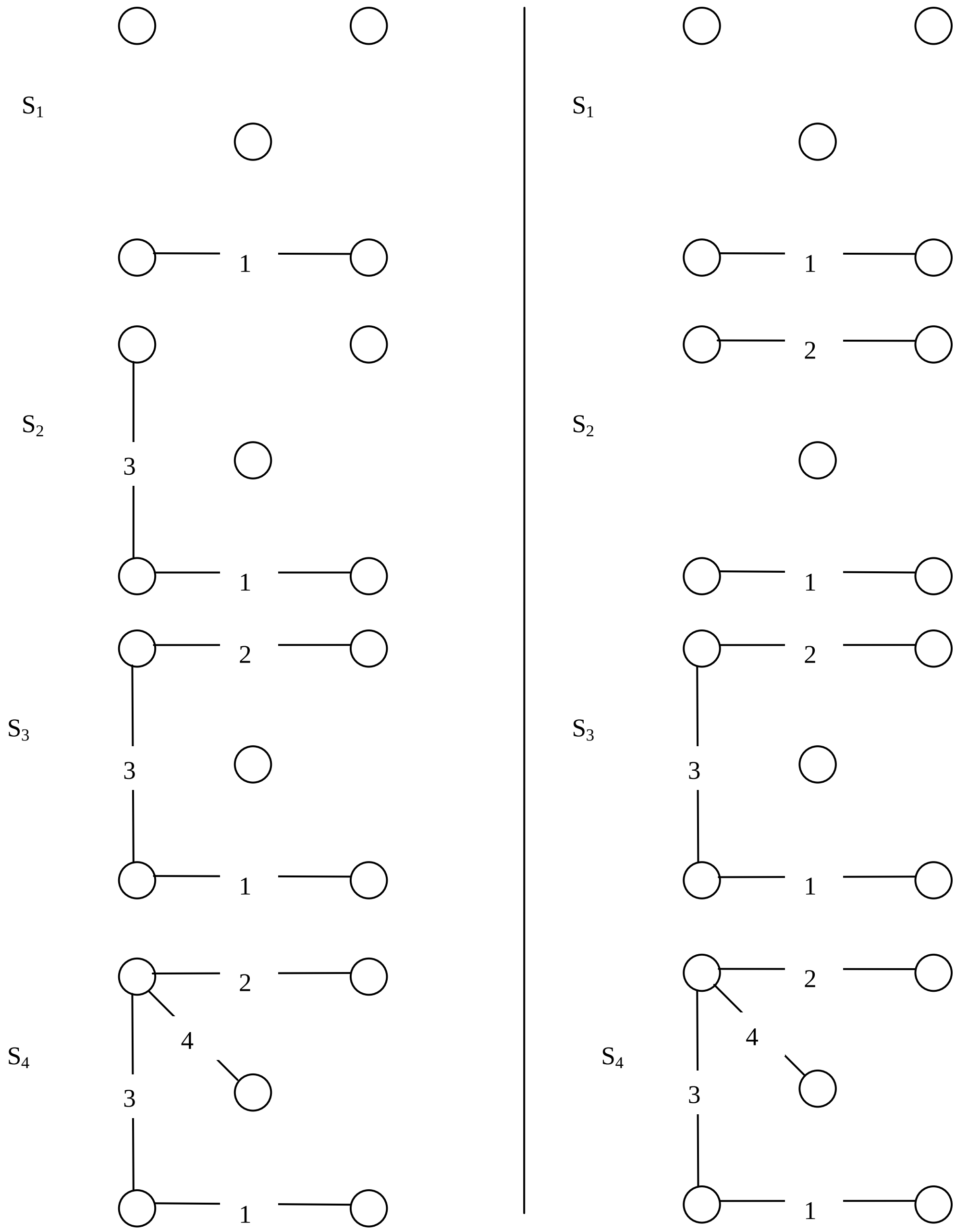
Пример:



- исходный граф G.

Последовательность деревьев в алгоритме Прима

Последовательность деревьев в алгоритме Краскала



Алгоритм Прима *отличается* от алгоритма Краскала только тем, что на каждом этапе строится не просто ациклический граф, а дерево. То есть в алгоритме Прима ребро присоединяется всякий раз к одной и той же компоненте леса, а в алгоритме Краскала ребро может присоединяться к каждой компоненте.

Сложность алгоритма Прима - $O(n^2)$. Алгоритм Прима недостаточно эффективен в применении к “редким” графам, то есть к графам, слабо насыщенным ребрами.