**Лекция 11**

**МИНИМАЛЬНЫЕ ПОКРЫВАЮЩИЕ ДЕРЕВЬЯ**

Пусть  − связный неориентированный граф, а  − весовая функция, заданная на множестве ребер  и определяющая длину каждого ребра. Считаем, что 

При обозначении ребра используются круглые скобки (а не угловые как для обозначения дуг), подчеркивающие неважность порядка перечисления вершин в ребре.

Связный подграф  графа  являющийся деревом и содержащий все его вершины, называется ***покрывающим деревом***, или ***остовным деревом***.

В общем случае один граф может иметь несколько покрывающих деревьев.

Покрывающее дерево  графа , имеющее минимальную сумму длин его ребер, называется ***минимальным покрывающим деревом***, или ***минимальным остовным деревом***.

В общем случае граф может иметь несколько минимальных покрывающих деревьев.

Наиболее известными алгоритмами построения минимального остовного дерева являются ***алгоритмы Крускала*** (Джозеф Крускал (1928–2010) − американский математик) и ***Прима*** (Роберт Прим (род. 1921) − американский математик).

Построение минимального остовного дерева в этих алгоритмах осуществляется поэтапно. Начинается построение с пустого множества ребер  На каждом этапе множество  пополняется одним ребром, причем так, что множество ребер  всегда остается подмножеством ребер искомого минимального остовного дерева. Задача сводится к поиску необходимого ребра (обычно его называют ***безопасным ребром***) на каждом шаге алгоритма.

Пусть  − множество вершин графа  тогда ***разбиением*** этого множества будем называть семейство множеств   …,   обладающее следующими свойствами:    

Пусть  и  – разбиение множества вершин связного неориентированного графа  Если провести линию, разделяющую множества  и  то эта линия пересечет ребра, концевые вершины которых лежат в разных подмножествах разбиения. Множество ребер, которые пересекла линия, разделяющая подмножества вершин  и  называется ***разрезом*** ***графа***  Для обозначения разреза будем использовать запись . При этом говорят, что ***множество ребер***  ***согласовано с разрезом***  если 

Путь  – множество всех ребер, принадлежащих разрезу  Тогда ребро  имеющее минимальную длину, называется ***легким ребром*** разреза 

**Теорема.** Пусть  − связный неориентированный граф и  – минимальное остовное дерево этого графа. Пусть также множество ребер  согласовано с некоторым разрезом , а – легкое ребро этого разреза. Тогда  является безопасным ребром для множества 

Воспользуемся теоремой для построения минимального остовного дерева для графа, но предварительно сформулируем еще одно утверждение.

Если  – ребро, имеющее минимальную длину в неориентированном связном графе, то оно будет ребром одного из минимальных остовных деревьев. Утверждение является следствием теоремы: достаточно положить 

Алгоритмы Крускала и Прима относятся к классу алгоритмов, называемых ***жадными алгоритмами***. Такое название эти алгоритмы получили за стратегию, заключающуюся в принятии на каждом шаге локально оптимального решения (жадного решения), в предположении, что такая стратегия приведет к конечному оптимальному решению.

Пример построения минимального остовного дерева для связного неориентированного графа. Сначала все ребра графа изображены пунктирной линий, над которой указана длина этого ребра. На рисунке показано пошаговое построение разрезов, начиная с  и заканчивая  По мере того, как вершины и дуги включаются в состав минимального остовного дерева, они меняют свою окраску. На каждом шаге, кроме последнего, выбирается легкое ребро разреза, которое, согласно приведенной выше теореме, является безопасным для множества окрашенных ребер.



**Алгоритм Крускала**

Алгоритм Крускала работает по тому же принципу. Отличие только в порядке объединения вершин в пошагово формирующемся минимальном покрывающем дереве.

Первоначально в алгоритме Крускала неориентированный связный граф  с заданной на его ребрах весовой функцией   разбивается на максимальное количество подграфов, каждый из которых является деревом. Очевидно, что каждое такое дерево будет представлять собой одну вершину графа, а общее количество таких подграфов будет 

Далее из множества  ребер графа  поочередно в порядке возрастания длины выбираются ребра. При этом возможны два случая:

1) концевые вершины лежат в разных подграфах разбиения;

2) обе концевые вершины лежат в одном подграфе разбиения.

В первом случае из двух подграфов, которые можно соединить выбранным ребром, образуется один общий, включающий все вершины и ребра этих подграфов, а также новое связующее ребро. Очевидно, что такое объединение не может образовать циклы, а следовательно, объединенный подграф тоже является деревом.

Во втором случае не выполняется никаких новых построений.

Из условия связности исходного графа  очевидно, что итогом работы такого алгоритма будет дерево, связывающее все вершины графа (т. е. остовное дерево).

Минимальность остовного дерева следует из того, что на каждом его шаге выбиралось безопасное ребро для совокупности ребер двух подграфов.

Пример построения минимального остовного дерева с помощью алгоритма Крускала.

Построение минимального остовного дерева осуществляется по шагам. Номер шага указан для каждого нового построения на рисунке. На первом шаге изображено разбиение исходного графа на максимальное число (равное 9 – количеству вершин в графе) подграфов.

На втором шаге отыскивается ребро исходного графа, имеющее минимальную длину (ребро (1,6) с длиной 1). Вершины, соединенные выбранным ребром, окрашиваются.

На всех последующих шагах из оставшихся ребер каждый раз выбирается ребро с минимальной длиной, но такое, чтобы концевые его вершины находились в изолированных друг от друга подграфах. После выбора ребра подграфы (как описывалось выше, они являются деревьями) становятся для алгоритма единой компонентой.

Следует обратить внимание, что минимальное остовное дерево, построенное в предыдущем примере, отличается от полученного результата. В обоих случаях построено минимальное остовное дерево. Если подсчитать сумму длин ребер деревьев, то в обоих случаях получается 37.



**Алгоритм Прима**

Построение минимального остовного дерева с помощью алгоритма Прима начинается с выбора произвольной вершины исходного неориентированного связного графа. Выбранная вершина окрашивается.

На последующих шагах просматриваются все окрашенные вершины и анализируются все ребра исходного графа, у которых одна концевая вершина окрашена, а другая нет. Среди всех таких ребер выбирается ребро с наименьшей длиной. Неокрашенная вершина этого ребра окрашивается, а само оно добавляется в формируемое минимальное остовное дерево.

Алгоритм заканчивает свою работу, когда все вершины графа станут окрашенными. Сформированное множество выбранных ребер будет составлять искомое минимальное остовное дерево.

Пример применения алгоритма Прима для построения минимального остовного дерева.

Работа алгоритма Прима представлена пошагово (номер шага указывается на рисунке). На первом шаге выбирается стартовая вершина (на рисунке – вершина 8) и окрашивается. На втором шаге среди всех ребер, инцидентных стартовой вершине, отыскивается ребро, имеющее наименьшую длину (на рисунке – ребро (8, 2)). Вторая (неокрашенная) вершина ребра окрашивается, а само ребро вместе с концевыми вершинами включается в будущее минимальное остовное дерево.

На шагах 3–10 алгоритма выбирается по одному ребру с минимальной длиной и одной неокрашенной концевой вершиной. Неокрашенные вершины окрашиваются, выбранные ребра пополняют строящееся минимальное остовное дерево. Цикл построения дерева продолжается до тех пор, пока не будут окрашены все вершины исходного графа.

На последнем, одиннадцатом шаге из выбранных ребер строится минимальное остовное дерево. Несложно подсчитать, что суммарная длина всех ребер сформированного остовного дерева равна 37, что совпадает с результатами, полученными с помощью других алгоритмов.

