

Нахождение минимального остовного дерева Алгоритм Прима

Баринова Екатерина, 5 курс ФВЭ

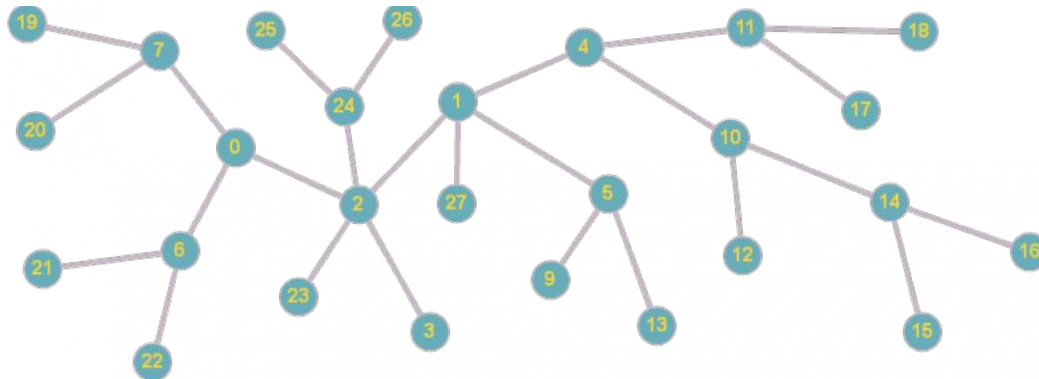


Содержание

1. Понятие графа и неориентированного графа
2. Нахождение минимального остовного дерева. Постановка задачи
3. Свойства минимальных остовов
4. Алгоритм Прима
5. Иллюстрация алгоритма Прима
6. Реализация: алгоритмы за $O(nm)$, $O(n^2 + m \log n)$

Понятие графа и неориентированного графа


- **Граф**, или **неориентированный граф** G — это упорядоченная пара $G=(V,E)$, где V — это непустое множество **вершин** или **узлов**, а E — множество пар (в случае неориентированного графа — неупорядоченных) вершин, называемых **рёбрами**. V и E обычно считаются конечными множествами.





Понятие графа и неориентированного графа

- Вершины и рёбра графа называются также **элементами** графа, число вершин в графе $|V|$ — **порядком**, число рёбер $|E|$ — **размером** графа.
- Вершины u и v называются **концевыми** вершинами (или просто **концами**) ребра $e=\{u,v\}$. Ребро, в свою очередь, **соединяет** эти вершины. Две концевые вершины одного и того же ребра называются **соседними**.
- Два ребра называются **смежными**, если они имеют общую концевую вершину.
- Два ребра называются **кратными**, если множества их концевых вершин совпадают.
- Ребро называется **петлёй**, если его концы совпадают, то есть $e=\{v,v\}$.
- Граф без петель и кратных рёбер называется **простым**.
- **Степенью** $\deg V$ вершины V называют количество инцидентных ей рёбер (при этом петли считают дважды).



Нахождение минимального остовного дерева. Постановка задачи

Дан взвешенный неориентированный граф G с n вершинами и рёбрами. Требуется найти такое поддерево этого графа, которое бы соединяло все его вершины, и при этом обладало наименьшим возможным весом (т.е. суммой весов рёбер). Поддерево — это набор рёбер, соединяющих все вершины, причём из любой вершины можно добраться до любой другой ровно одним простым путём.

Такое поддерево называется минимальным остовным деревом или просто **минимальным остовом**. Легко понять, что любой остов обязательно будет содержать $n-1$ ребро.



Свойства минимальных остовов

- **Максимальный** остов также можно искать алгоритмом Прима.
- Минимальный остов **единственен**, если веса всех рёбер различны. В противном случае, может существовать несколько минимальных остовов.
- Минимальный остов также является остовом, **минимальным по произведению** всех рёбер (предполагается, что все веса положительны).
- Минимальный остов является остовом с минимальным весом **самого тяжёлого ребра**. Яснее всего это утверждение понятно, если рассмотреть работу алгоритма Крускала.
- **Критерий минимальности** остова: остов является минимальным тогда и только тогда, когда для любого ребра, не принадлежащего остову, цикл, образуемый этим ребром при добавлении к остову, не содержит рёбер тяжелее этого ребра.



Алгоритм Прима

Роберт Прим (Robert Prim), 1957 г. Войтек Ярник (Vojtěch Jarník), 1930 г.

- Искомый минимальный остов строится постепенно, добавлением в него рёбер по одному. Изначально остов полагается состоящим из единственной вершины.
- Выбирается ребро минимального веса, исходящее из этой вершины, и добавляется в минимальный остов.
- После этого остов содержит уже две вершины, и теперь ищется и добавляется ребро минимального веса, имеющее один конец в одной из двух выбранных вершин, а другой — наоборот, во всех остальных, кроме этих двух.
- Этот процесс повторяется до тех пор, пока остов не станет содержать все вершины.
- В итоге будет построен остов, являющийся минимальным. Если граф был изначально не связан, то остов найден не будет (количество выбранных рёбер останется меньше).

Иллюстрация алгоритма Прима

Исходный взвешенный граф. Числа возле ребер показывают их веса, которые можно рассматривать как расстояния между вершинами.

Множество выбранных вершин: $\{\}$

Множество невыбранных вершин:
 $\{A, B, C, D, E, F, G\}$

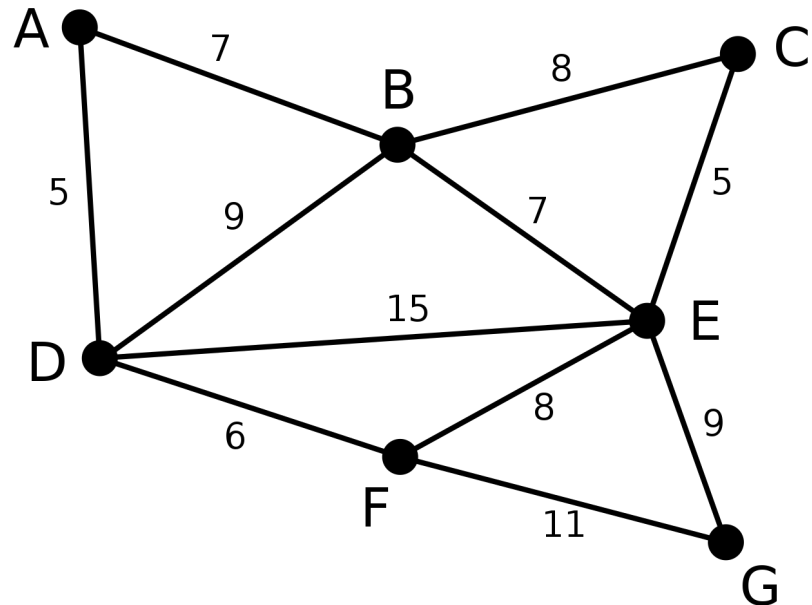


Иллюстрация алгоритма Прима

В качестве начальной произвольно выбирается вершина **D**. Каждая из вершин **A**, **B**, **E** и **F** соединена с **D** единственным ребром. Вершина **A** — ближайшая к **D**, и выбирается как вторая вершина вместе с ребром **AD**.

Множество выбранных вершин: {D}

Множество невыбранных вершин: {A,B,C,D,E,F,G}

Ребро (u, v):

- (D,A) = 5
- (D,B) = 9
- (D,E) = 15
- (D,F) = 6

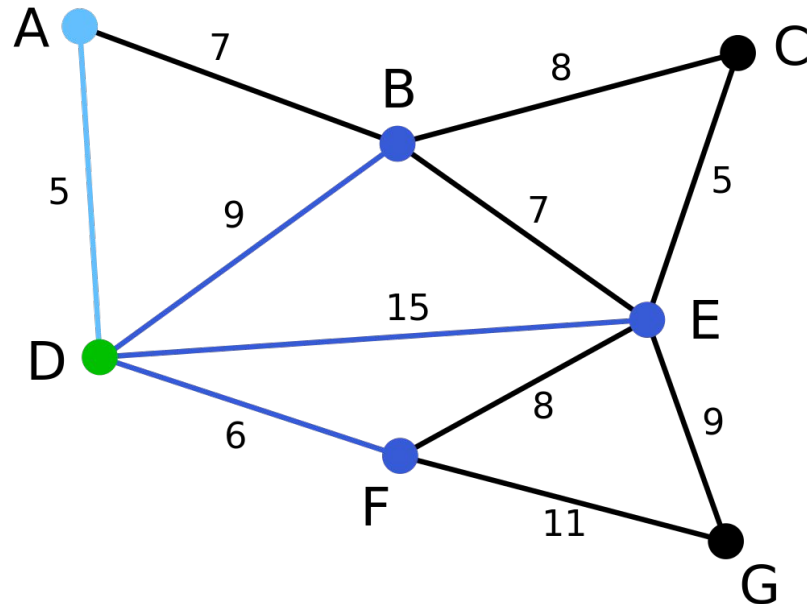


Иллюстрация алгоритма Прима

Следующая вершина — ближайшая к любой из выбранных вершин **D** или **A**. **B** удалена от **D** на 9 и от **A** — на 7. Расстояние до **E** равно 15, а до **F** — 6. **F** является ближайшей вершиной, поэтому она включается в дерево **F** вместе с ребром **DF**.

Множество выбранных вершин: {A,D}

Множество невыбранных вершин: {B,C,E,F,G}

Ребро (u, v):

- (D,B) = 9
- (D,E) = 15
- (D,F) = 6
- (A,B) = 7

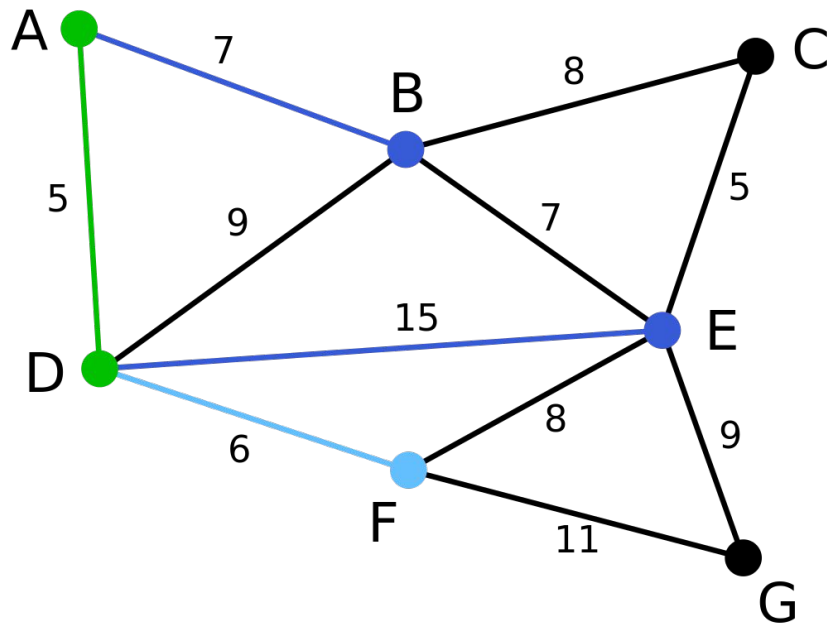


Иллюстрация алгоритма Прима

Аналогичным образом выбирается вершина **В**, удаленная от **А** на 7.

Множество выбранных вершин: {A,D,F}

Множество невыбранных вершин: {B,C,E,G}

Ребро (u, v):

(D,B) = 9

(D,E) = 15

(A,B) = 7 **✓**

(F,E) = 8

(F,G) = 11

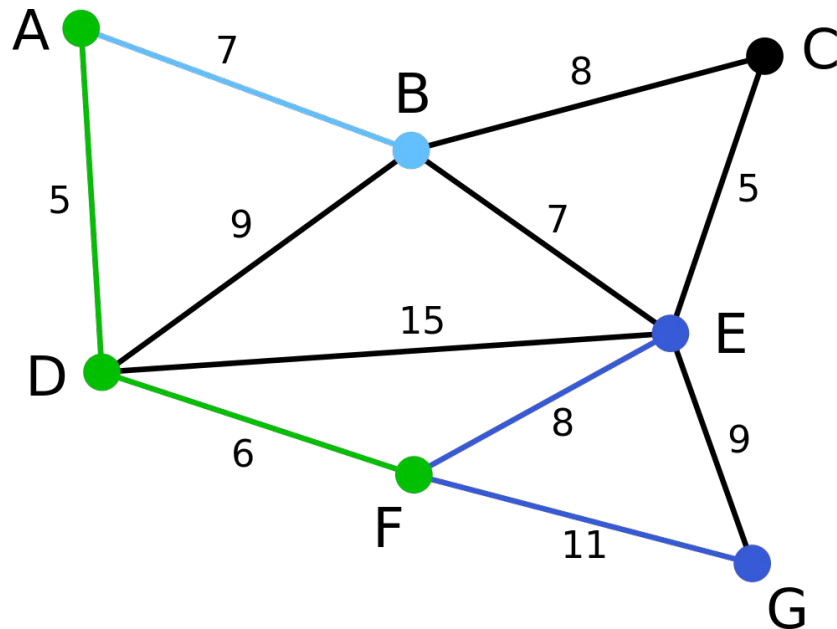


Иллюстрация алгоритма Прима

В этом случае есть возможность выбрать либо **С**, либо **Е**, либо **Г**. Судалена от **В** на 8, **Е** удалена от **В** на 7, а **Г** удалена от **В** на 11. **Е** — ближайшая вершина, поэтому выбирается **Е** и ребро **ВЕ**.

Множество выбранных вершин: {A,B,D,F}

Множество невыбранных вершин: {C,E,G}

Ребро (u, v):

(B,C) = 8

(B,E) = 7

(D,B) = 9 цикл

(D,E) = 15

(F,E) = 8

(F,G) = 11

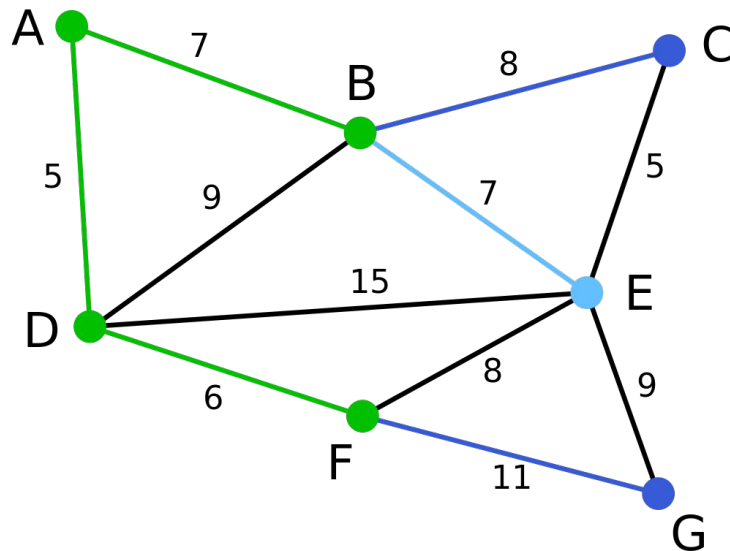


Иллюстрация алгоритма Прима

Здесь доступны только вершины **С** и **G**. Расстояние от **Е** до **С** равно 5, а до **G** — 9. Выбирается вершина **С** и ребро **ЕС**.

Множество выбранных вершин: {A,B,D,E,F}

Множество невыбранных вершин: {C,G}

Ребро (u, v):

(B,C) = 8

(D,B) = 9 цикл

(D,E) = 15 цикл

(E,C) = 5 **V**

(E,G) = 9

(F,E) = 8 цикл

(F,G) = 11

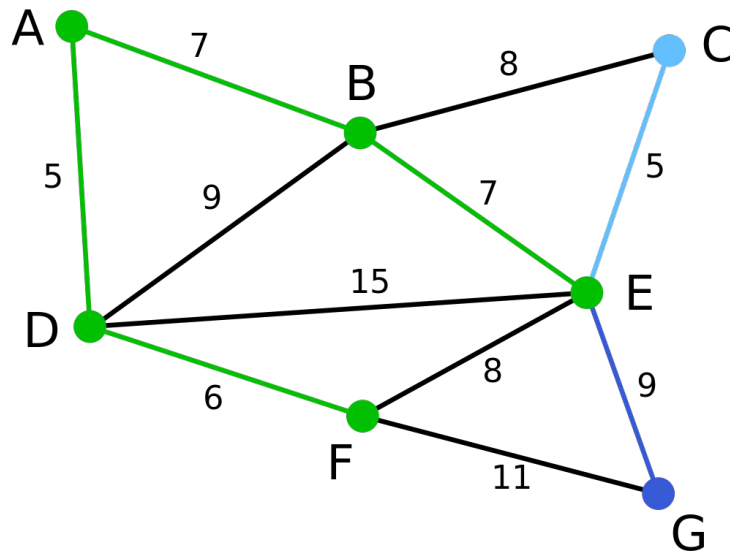


Иллюстрация алгоритма Прима

Единственная оставшаяся вершина — **G**. Расстояние от **F** до неё равно 11, от **E** — 9. **E** ближе, поэтому выбирается вершина **G** и ребро **EG**.

Множество выбранных вершин: {A,B,C,D,E,F}

Множество невыбранных вершин: {G}

Ребро (**u**, **v**):

(B,C) = 8 цикл

(D,B) = 9 цикл

(D,E) = 15 цикл

(E,G) = 9 **V**

(F,E) = 8 цикл

(F,G) = 11

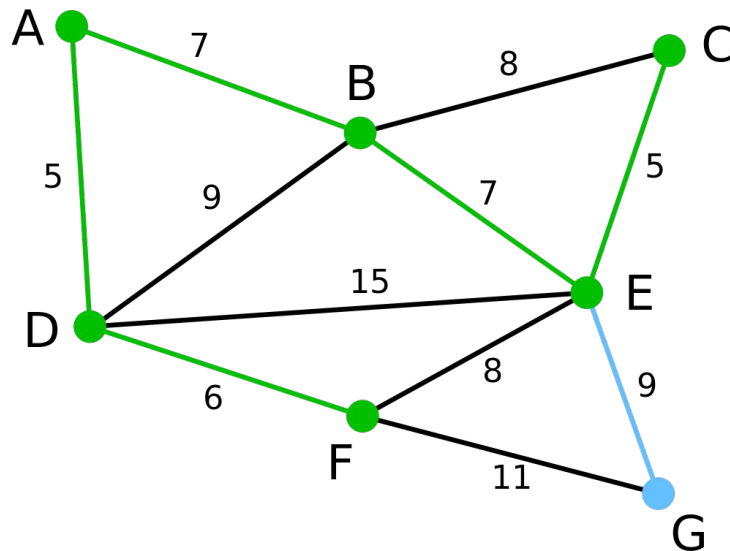


Иллюстрация алгоритма Прима

Выбраны все вершины, минимальное остовное дерево построено (выделено зеленым). В этом случае его вес равен 39.

Множество выбранных вершин: {A,B,C,D,E,F,G}

Множество невыбранных вершин: {}

Ребро (u, v):

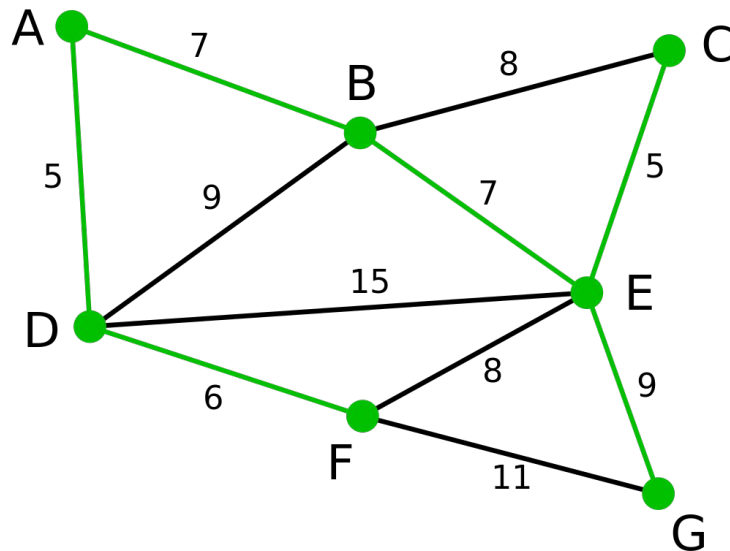
B,C) = 8 цикл

(D,B) = 9 цикл

(D,E) = 15 цикл

(F,E) = 8 цикл

(F,G) = 11 цикл





Реализация: алгоритмы за $O(nm)$, $O(n^2 + m \log n)$

Если искать каждый раз ребро простым просмотром среди всех возможных вариантов, то асимптотически будет требоваться просмотр $O(m)$ рёбер, чтобы найти среди всех допустимых ребро с наименьшим весом. Суммарная асимптотика алгоритма составит в таком случае $O(nm)$, что в худшем случае есть $O(n^3)$, — слишком медленный алгоритм.

Этот алгоритм можно улучшить, если просматривать каждый раз не все рёбра, а только по одному ребру из каждой уже выбранной вершины. Для этого, например, можно отсортировать рёбра из каждой вершины в порядке возрастания весов, и хранить указатель на первое допустимое ребро. Тогда, если пересчитывать эти указатели при каждом добавлении ребра в остов, суммарная асимптотика алгоритма будет $O(n^2 + m)$, но предварительно потребуется выполнить сортировку всех рёбер за $O(m \log n)$, что в худшем случае (для плотных графов) даёт асимптотику $O(n^2 \log n)$.