Тема:
Построение минимальных стягивающих деревьев

Сергей Витальевич Рыбин

svrybin@etu.ru

СПбГЭТУ «ЛЭТИ», кафедра «Алгоритмической математики»

21 июня 2023 г.

Основные понятия

Основные понятия

1. Связный граф, не содержащий циклов, называют **деревом**. Произвольный граф, не содержащий циклов, называют **ациклическим**, или **лесом**.

Основные понятия

1. Связный граф, не содержащий циклов, называют **деревом**. Произвольный граф, не содержащий циклов, называют **ациклическим**, или **лесом**.
2. Граф $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ называют **нагруженным**, или **взвешенным**, если для каждого ребра (v_i, v_j) определена его длина (или вес) w_{ij}.

Основные понятия

1. Связный граф, не содержащий циклов, называют **деревом**. Произвольный граф, не содержащий циклов, называют **ациклическим**, или **лесом**.
2. Граф $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ называют **нагруженным**, или **взвешенным**, если для каждого ребра (v_i, v_j) определена его длина (или вес) w_{ij}.
3. Матрицу $\{w_{ij}\}$ называют **матрицей весов ребер**.

Основные понятия

1. Связный граф, не содержащий циклов, называют **деревом**. Произвольный граф, не содержащий циклов, называют **ациклическим**, или **лесом**.
2. Граф $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ называют **нагруженным**, или **взвешенным**, если для каждого ребра (v_i, v_j) определена его длина (или вес) w_{ij}.
3. Матрицу $\{w_{ij}\}$ называют **матрицей весов ребер**.
4. Веса несуществующих ребер полагают равными ∞ или 0 в зависимости от приложений.

Основные понятия

1. Связный граф, не содержащий циклов, называют **деревом**. Произвольный граф, не содержащий циклов, называют **ациклическим**, или **лесом**.
2. Граф $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ называют **нагруженным**, или **взвешенным**, если для каждого ребра (v_i, v_j) определена его длина (или вес) w_{ij}.
3. Матрицу $\{w_{ij}\}$ называют **матрицей весов ребер**.
4. Веса несуществующих ребер полагают равными ∞ или 0 в зависимости от приложений.
5. Пусть $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ — связный граф. Дерево, являющееся подграфом $\boldsymbol{G}$ и содержащее все его вершины, называют деревом, **покрывающим** граф (стягивающим деревом, каркасом, остовом, остовым деревом).

Основные понятия

1. Связный граф, не содержащий циклов, называют **деревом**. Произвольный граф, не содержащий циклов, называют **ациклическим**, или **лесом**.
2. Граф $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ называют **нагруженным**, или **взвешенным**, если для каждого ребра (v_i, v_j) определена его длина (или вес) w_{ij}.
3. Матрицу $\{w_{ij}\}$ называют **матрицей весов ребер**.
4. Веса несуществующих ребер полагают равными ∞ или 0 в зависимости от приложений.
5. Пусть $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ — связный граф. Дерево, являющееся подграфом $\boldsymbol{G}$ и содержащее все его вершины, называют деревом, **покрывающим** граф (стягивающим деревом, каркасом, остовом, остовым деревом).

Постановка задачи

Основные понятия

1. Связный граф, не содержащий циклов, называют **деревом**. Произвольный граф, не содержащий циклов, называют **ациклическим**, или **лесом**.
2. Граф $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ называют **нагруженным**, или **взвешенным**, если для каждого ребра (v_i, v_j) определена его длина (или вес) w_{ij}.
3. Матрицу $\{w_{ij}\}$ называют **матрицей весов ребер**.
4. Веса несуществующих ребер полагают равными ∞ или 0 в зависимости от приложений.
5. Пусть $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ — связный граф. Дерево, являющееся подграфом $\boldsymbol{G}$ и содержащее все его вершины, называют деревом, **покрывающим** граф (стягивающим деревом, каркасом, остовом, остовым деревом).

Постановка задачи

В связном нагруженном графе нужно из множества стягивающих деревьев найти дерево, у которого сумма длин ребер минимальна.

Алгоритм 2.0. Прима или ближайшего соседа

// *Построение дерева* $\boldsymbol{T} = (\boldsymbol{E_T}, \boldsymbol{V_T})$ *начинаем с произвольной вершины* $\boldsymbol{s}$.

$\boldsymbol{V_T} \leftarrow \{s\}$

while существуют вершины не принадлежащие дереву $\boldsymbol{T}$ **do**

// *из ребер, соединяющих вершины* $\boldsymbol{T}$ *с вершинами графа* $\boldsymbol{G}$, *выбираем ребро минимального веса*

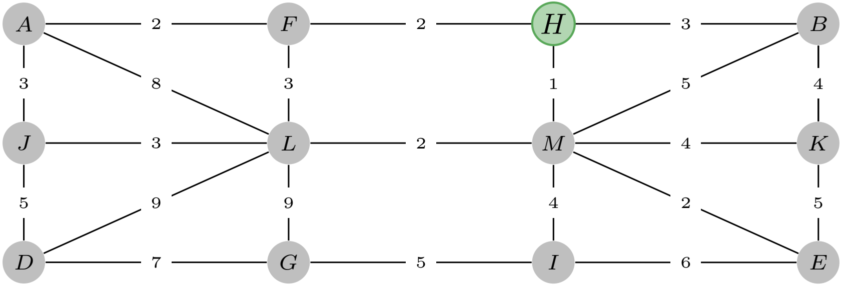
$e \leftarrow \arg\min \left\{ w_{i,j} \mid (v_i, v_j) \in \boldsymbol{E} \mid v_i \in \boldsymbol{T}, v_j \notin \boldsymbol{T} \right\}$

// *включаем его в дерево вместе с его инцидентной вершиной, не входящей в* $\boldsymbol{T}$.

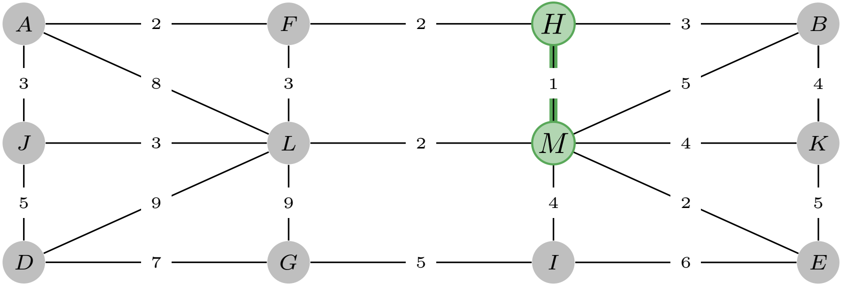
$\boldsymbol{E_T} \leftarrow \boldsymbol{E_T} \cup \{e\}$, $\boldsymbol{V_T} \leftarrow \boldsymbol{V_T} \cup \{v_j\}$

end while

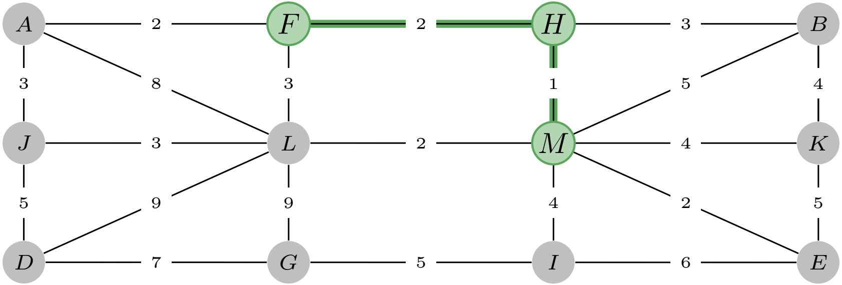
Пример работы алгоритма Прима



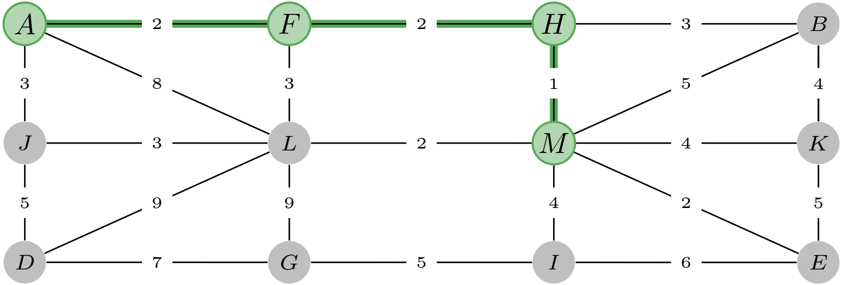
Пример работы алгоритма Прима



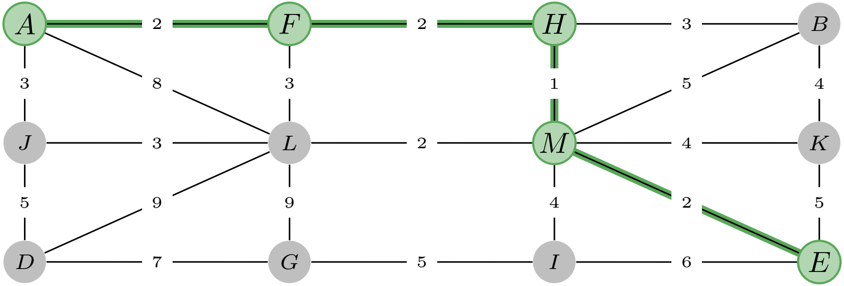
Пример работы алгоритма Прима



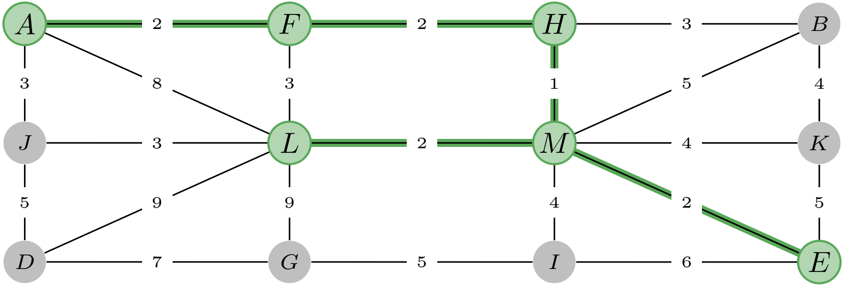
Пример работы алгоритма Прима



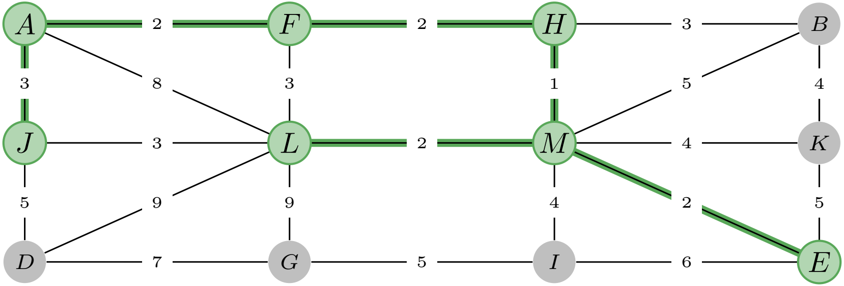
Пример работы алгоритма Прима



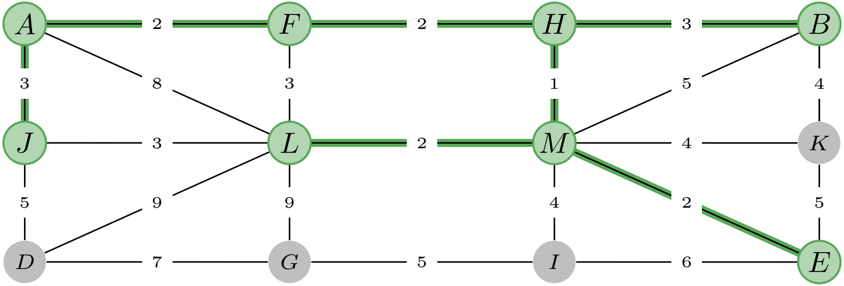
Пример работы алгоритма Прима



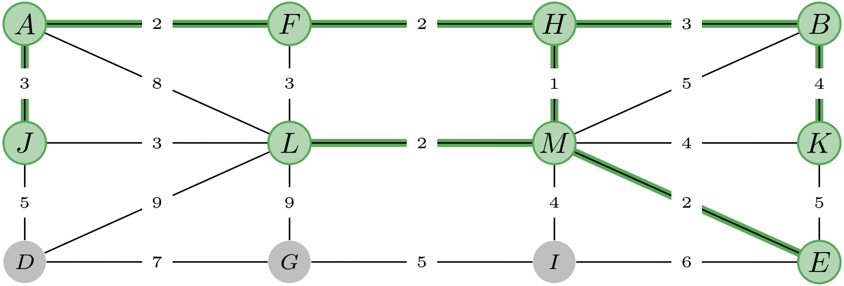
Пример работы алгоритма Прима



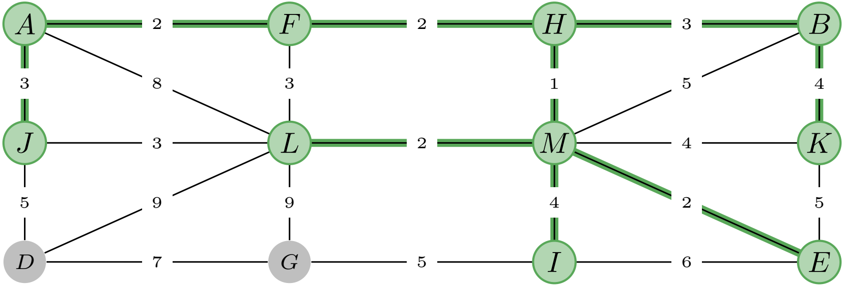
Пример работы алгоритма Прима



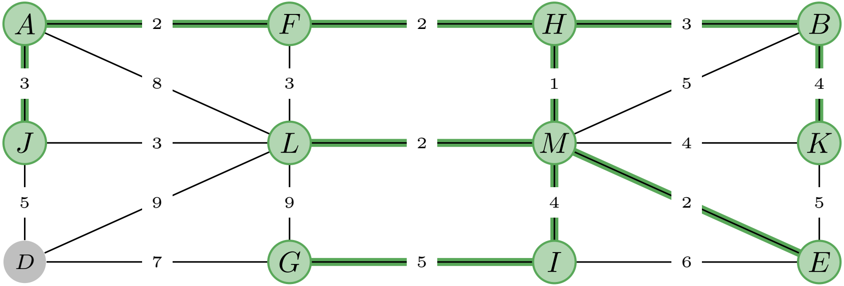
Пример работы алгоритма Прима



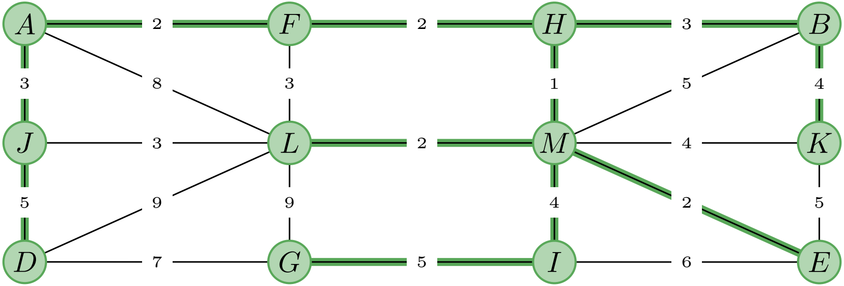
Пример работы алгоритма Прима



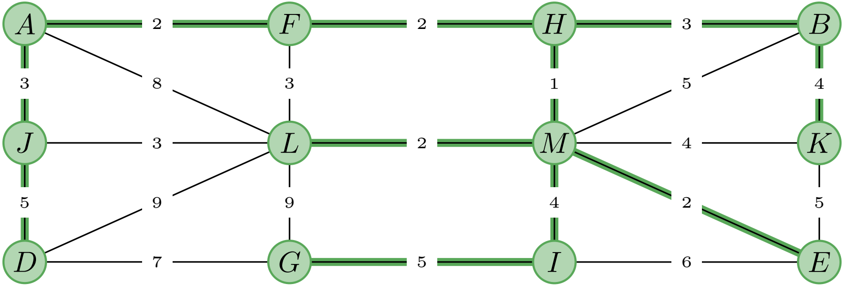
Пример работы алгоритма Прима



Пример работы алгоритма Прима

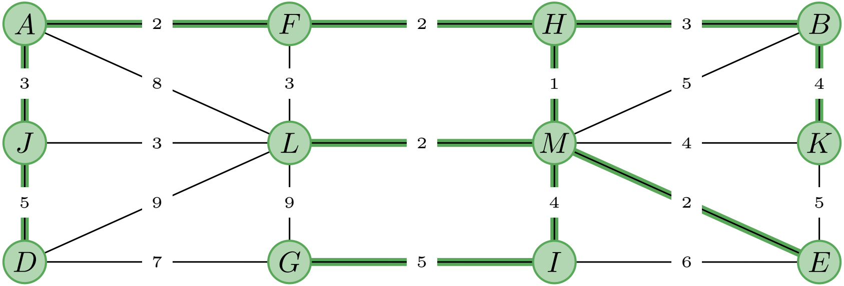


Пример работы алгоритма Прима



1. Последовательность включаемых ребер:

$$HM(1) \to HF(2) \to FA(2) \to ME(2) \to ML(2) \to AJ(3) \to HB(3) \to BK(4) \to MI(4) \to IG(5) \to JD(5)$$

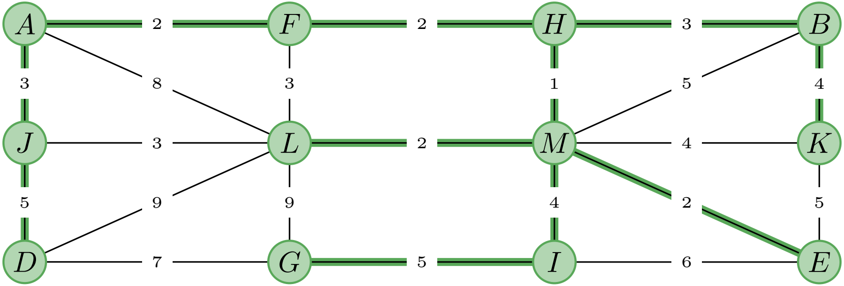


1. Последовательность включаемых ребер:

$$HM(1) \to HF(2) \to FA(2) \to ME(2) \to ML(2) \to AJ(3) \to HB(3) \to BK(4) \to MI(4) \to IG(5) \to JD(5)$$

2. Вес построенного дерева равен 33.

Пример работы алгоритма Прима



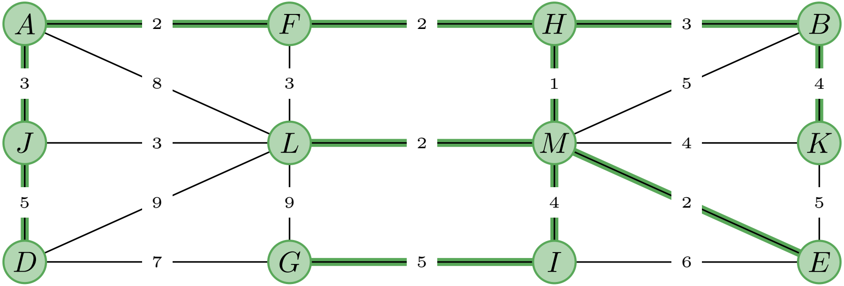
1. Последовательность включаемых ребер:

$$HM(1) \to HF(2) \to FA(2) \to ME(2) \to ML(2) \to AJ(3) \to HB(3) \to BK(4) \to MI(4) \to IG(5) \to JD(5)$$

2. Вес построенного дерева равен 33.

3. Если начать построение дерева с другой вершины, последовательность включаемых ребер и сам каркас могут получиться другими, однако вес построенного дерева останется неизменным и является инвариантом алгоритма.

Пример работы алгоритма Прима



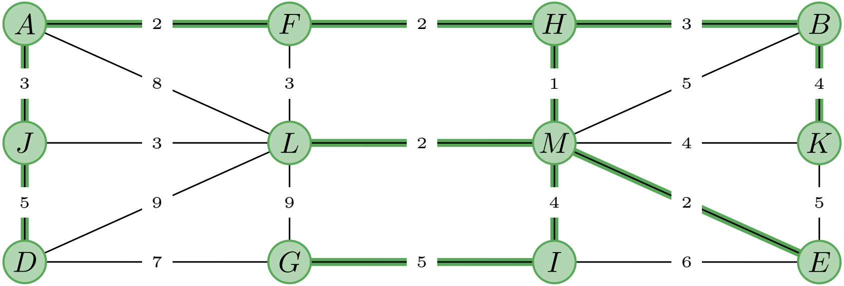
1. Последовательность включаемых ребер:

$$HM(1) \to HF(2) \to FA(2) \to ME(2) \to ML(2) \to AJ(3) \to HB(3) \to BK(4) \to MI(4) \to IG(5) \to JD(5)$$

2. Вес построенного дерева равен 33.
3. Если начать построение дерева с другой вершины, последовательность включаемых ребер и сам каркас могут получиться другими, однако вес построенного дерева останется неизменным и является инвариантом алгоритма.
4. Если начать, например, работу алгоритма с вершины D. Тогда последовательность включаемых ребер:

$$DJ(5) \to JA(3) \to AF(2) \to FH(2) \to HM(1) \to ME(2) \to ML(2) \to HB(3) \to BK(4) \to MI(4) \to IG(5)$$

Пример работы алгоритма Прима



1. Последовательность включаемых ребер:

$$HM(1) \to HF(2) \to FA(2) \to ME(2) \to ML(2) \to AJ(3) \to HB(3) \to BK(4) \to MI(4) \to IG(5) \to JD(5)$$

2. Вес построенного дерева равен 33.
3. Если начать построение дерева с другой вершины, последовательность включаемых ребер и сам каркас могут получиться другими, однако вес построенного дерева останется неизменным и является инвариантом алгоритма.
4. Если начать, например, работу алгоритма с вершины D. Тогда последовательность включаемых ребер:

$$DJ(5) \to JA(3) \to AF(2) \to FH(2) \to HM(1) \to ME(2) \to ML(2) \to HB(3) \to BK(4) \to MI(4) \to IG(5)$$

5. Вес построенного дерева также равен 33.

Алгоритм 4.0. Модифицированный алгоритм Краскала

// *Упорядочиваем все ребра графа* $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ *по возрастанию (неубыванию) весов*

$L = \{e_1, \dots, e_k \mid e_i \in \boldsymbol{E}\}$, $\boldsymbol{T} \leftarrow \emptyset$

// *Раскрашиваем вершины графа в разные цвета*

$clr \leftarrow 1$

for each $u \in \boldsymbol{V}$ **do**

  $color(u) \leftarrow clr,\ clr \leftarrow clr + 1$

end for

// *Основной алгоритм*

for each $(v, u) \in L \mid color(u) \neq color(v)$ **do**

  $\boldsymbol{T} \leftarrow \boldsymbol{G}' \cup \{(v, u)\}$

  $c_{max} = \max\{color(u), color(v)\}$, $c_{min} = \min\{color(u), color(v)\}$

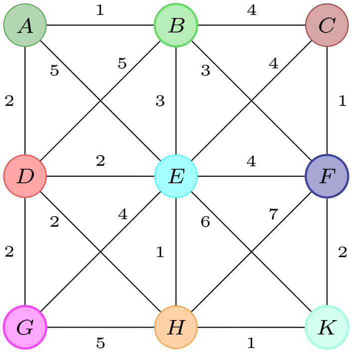
  for each $v_i \in \boldsymbol{V} \mid color(v_i) = c_{max}$ **do**

    $color(v_i) = c_{min}$

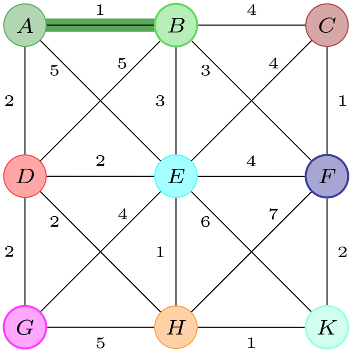
  end for

end for

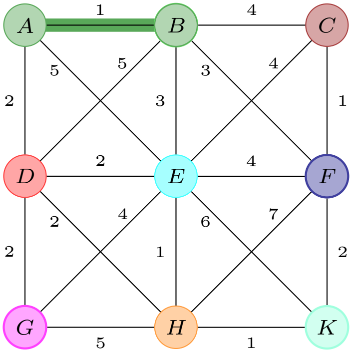
Пример работы алгоритма Краскала



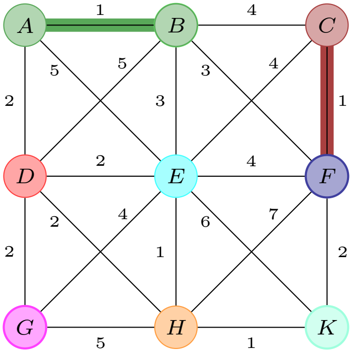
Пример работы алгоритма Краскала



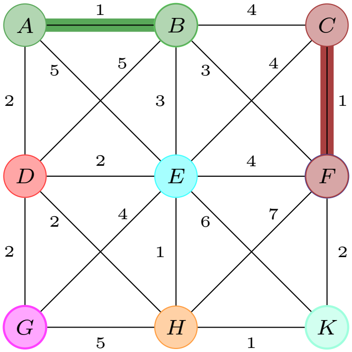
Пример работы алгоритма Краскала



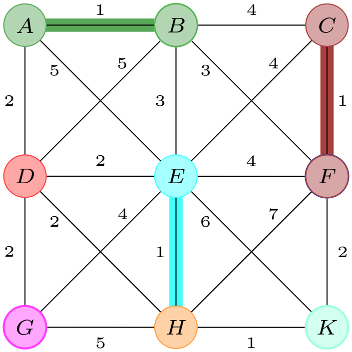
Пример работы алгоритма Краскала



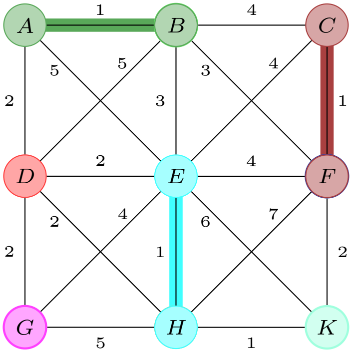
Пример работы алгоритма Краскала



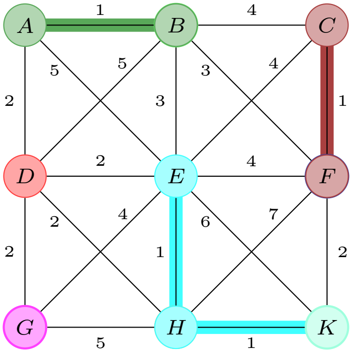
Пример работы алгоритма Краскала



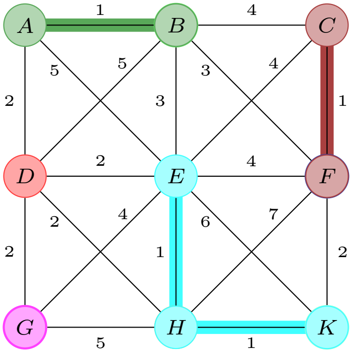
Пример работы алгоритма Краскала



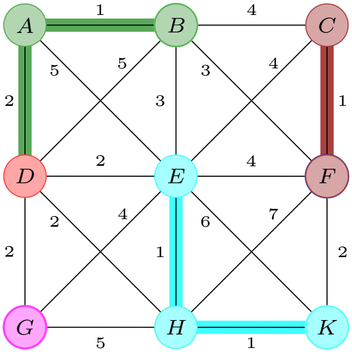
Пример работы алгоритма Краскала



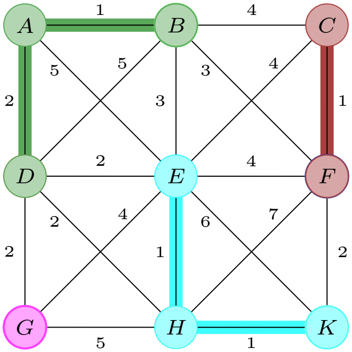
Пример работы алгоритма Краскала



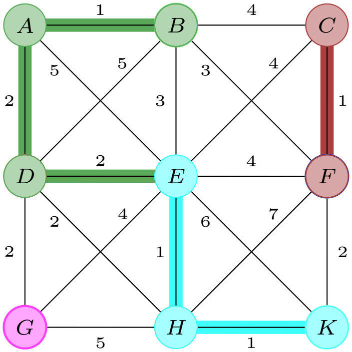
Пример работы алгоритма Краскала



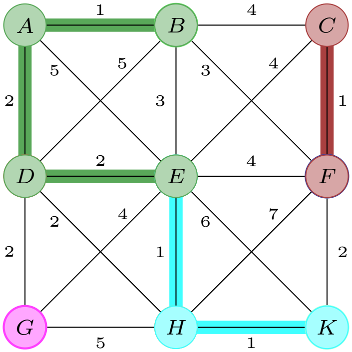
Пример работы алгоритма Краскала



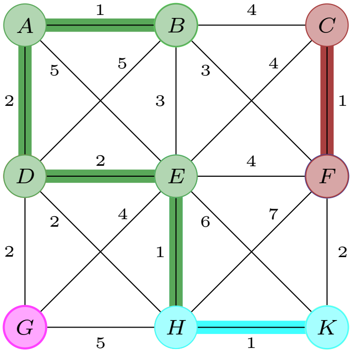
Пример работы алгоритма Краскала



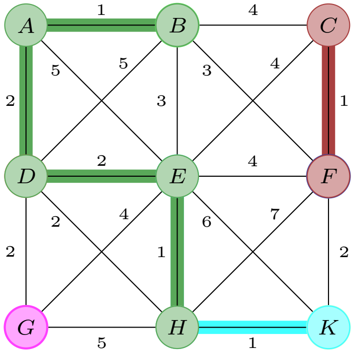
Пример работы алгоритма Краскала



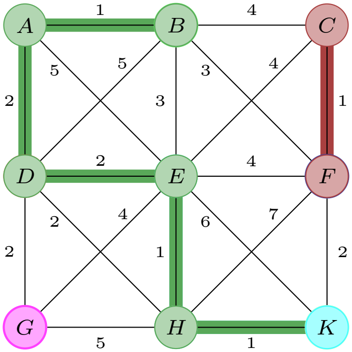
Пример работы алгоритма Краскала



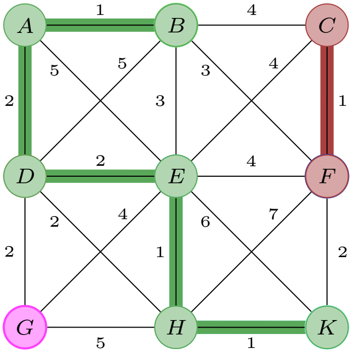
Пример работы алгоритма Краскала



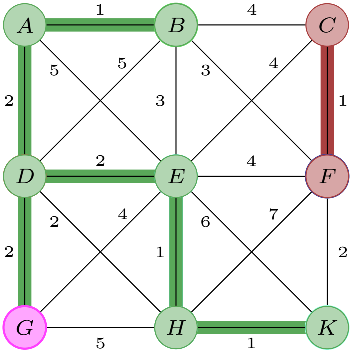
Пример работы алгоритма Краскала



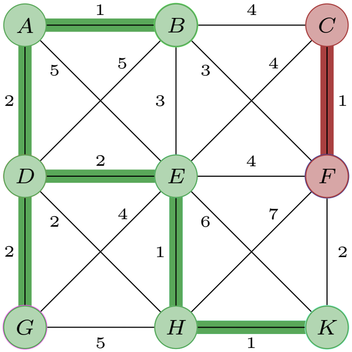
Пример работы алгоритма Краскала



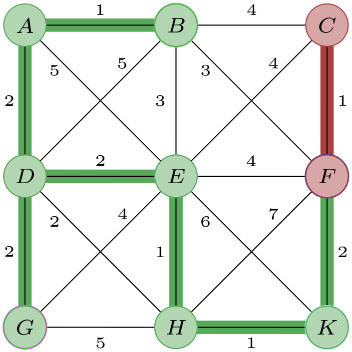
Пример работы алгоритма Краскала



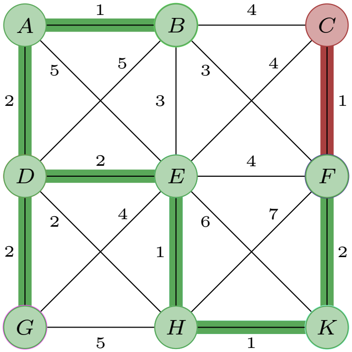
Пример работы алгоритма Краскала



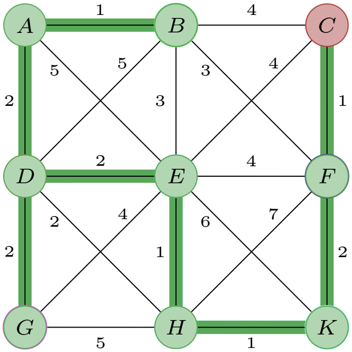
Пример работы алгоритма Краскала



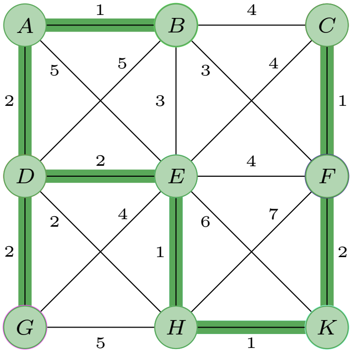
Пример работы алгоритма Краскала



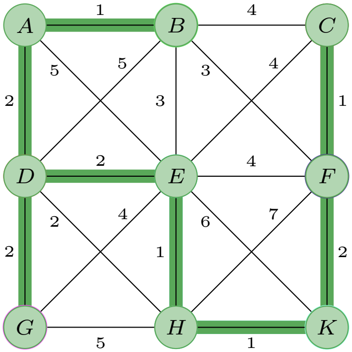
Пример работы алгоритма Краскала



Пример работы алгоритма Краскала



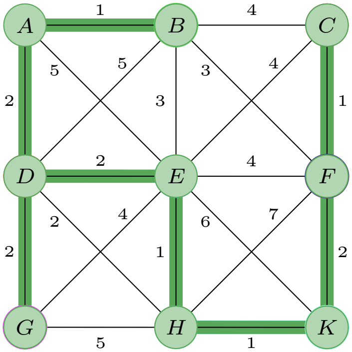
Пример работы алгоритма Краскала



1. Последовательность рассматриваемых алгоритмом ребер, упорядоченных по неубыванию весов:

$$AB(1) \to CF(1) \to EH(1) \to HK(1) \to AD(2) \to DE(2) \to DG(2) \to DH(2) \to FK(2)$$

Пример работы алгоритма Краскала

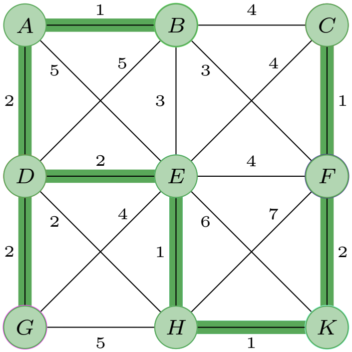


1. Последовательность рассматриваемых алгоритмом ребер, упорядоченных по неубыванию весов:

$$AB(1) \to CF(1) \to EH(1) \to HK(1) \to AD(2) \to DE(2) \to DG(2) \to DH(2) \to FK(2)$$

2. Ребра одного веса упорядочены в лексикографическом порядке (можно было взять любой другой порядок).

Пример работы алгоритма Краскала

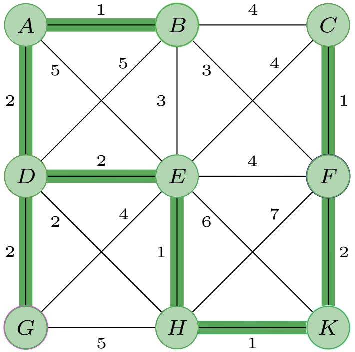


1. Последовательность рассматриваемых алгоритмом ребер, упорядоченных по неубыванию весов:

$$AB(1) \to CF(1) \to EH(1) \to HK(1) \to AD(2) \to DE(2) \to DG(2) \to DH(2) \to FK(2)$$

2. Ребра одного веса упорядочены в лексикографическом порядке (можно было взять любой другой порядок).
3. Вес построенного дерева равен 12.

Пример работы алгоритма Краскала

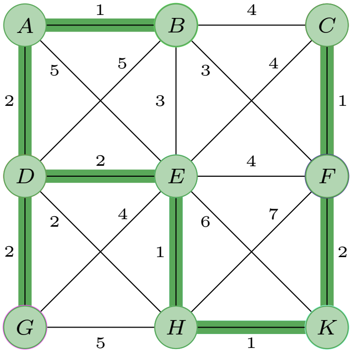


1. Последовательность рассматриваемых алгоритмом ребер, упорядоченных по неубыванию весов:

$$AB(1) \to CF(1) \to EH(1) \to HK(1) \to AD(2) \to DE(2) \to DG(2) \to DH(2) \to FK(2)$$

2. Ребра одного веса упорядочены в лексикографическом порядке (можно было взять любой другой порядок).
3. Вес построенного дерева равен 12.
4. Видно, что дерево построено, когда все вершины графа раскрашены в один цвет.

Пример работы алгоритма Краскала



1. Последовательность рассматриваемых алгоритмом ребер, упорядоченных по неубыванию весов:

$$AB(1) \to CF(1) \to EH(1) \to HK(1) \to AD(2) \to DE(2) \to DG(2) \to DH(2) \to FK(2)$$

2. Ребра одного веса упорядочены в лексикографическом порядке (можно было взять любой другой порядок).
3. Вес построенного дерева равен 12.
4. Видно, что дерево построено, когда все вершины графа раскрашены в один цвет.
5. Этот признак можно использовать для проверки окончания цикла в алгоритме Краскала.