

Доказательство корректности:

Пусть $s(x_i)$ – начало промежутка x_i , $e(x_i)$ – окончание.

Пусть y_1, \dots, y_n – решение, найденное жадным алгоритмом, и z_1, \dots, z_m – оптимальное решение.

Доказательство корректности:

Пусть $s(x_i)$ – начало промежутка x_i , $e(x_i)$ – окончание.

Пусть y_1, \dots, y_n – решение, найденное жадным алгоритмом, и z_1, \dots, z_m – оптимальное решение.

Лемма

Для любого k , $e(y_k) \leq e(z_k)$.

Доказательство корректности:

Пусть $s(x_i)$ – начало промежутка x_i , $e(x_i)$ – окончание.

Пусть y_1, \dots, y_n – решение, найденное жадным алгоритмом, и z_1, \dots, z_m – оптимальное решение.

Лемма

Для любого k , $e(y_k) \leq e(z_k)$.

Доказательство.

База индукции: жадный алгоритм выбирает y_1 так, что $e(y_1)$ – минимально.

Шаг индукции: поскольку $e(y_{k-1}) \leq e(z_{k-1})$, то z_k является допустимым для продолжения y_1, \dots, y_{k-1} . y_k – элемент с минимальным e из всех допустимых, следовательно, $e(y_k) \leq e(z_k)$. □

Доказательство корректности:

Пусть $s(x_i)$ – начало промежутка x_i , $e(x_i)$ – окончание.

Пусть y_1, \dots, y_n – решение, найденное жадным алгоритмом, и z_1, \dots, z_m – оптимальное решение.

Лемма

Для любого k , $e(y_k) \leq e(z_k)$.

Доказательство.

База индукции: жадный алгоритм выбирает y_1 так, что $e(y_1)$ – минимально.

Шаг индукции: поскольку $e(y_{k-1}) \leq e(z_{k-1})$, то z_k является допустимым для продолжения y_1, \dots, y_{k-1} . y_k – элемент с минимальным e из всех допустимых, следовательно, $e(y_k) \leq e(z_k)$. □

Лемма

$n \geq m$.

Доказательство корректности:

Пусть $s(x_i)$ – начало промежутка x_i , $e(x_i)$ – окончание.

Пусть y_1, \dots, y_n – решение, найденное жадным алгоритмом, и z_1, \dots, z_m – оптимальное решение.

Лемма

Для любого k , $e(y_k) \leq e(z_k)$.

Доказательство.

База индукции: жадный алгоритм выбирает y_1 так, что $e(y_1)$ – минимально.

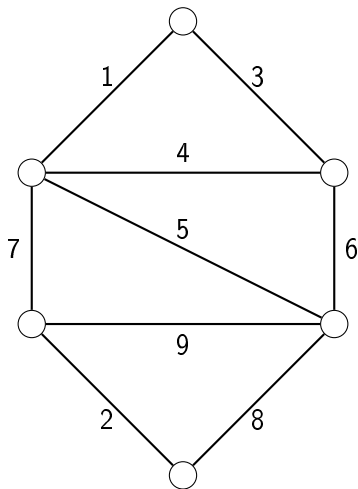
Шаг индукции: поскольку $e(y_{k-1}) \leq e(z_{k-1})$, то z_k является допустимым для продолжения y_1, \dots, y_{k-1} . y_k – элемент с минимальным e из всех допустимых, следовательно, $e(y_k) \leq e(z_k)$. □

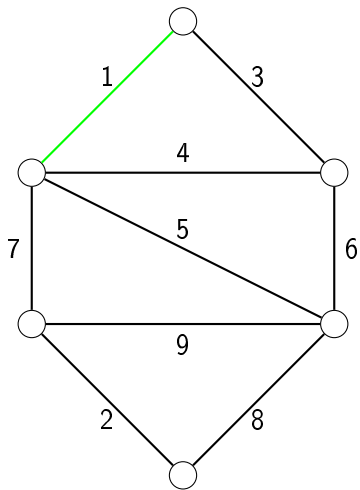
Лемма

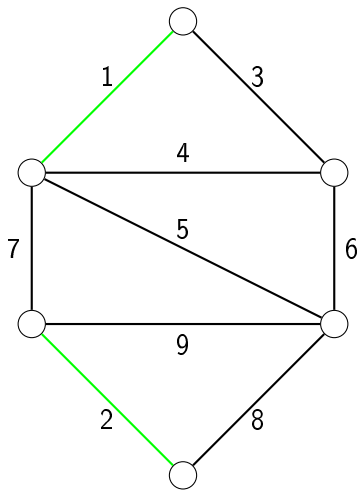
$n \geq m$.

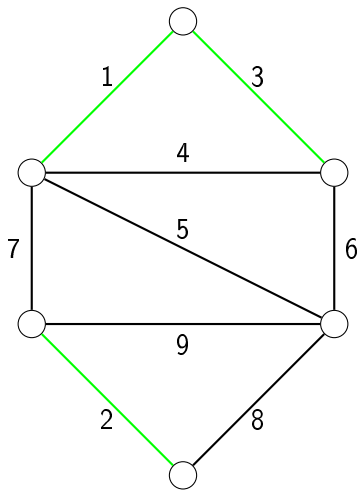
Доказательство.

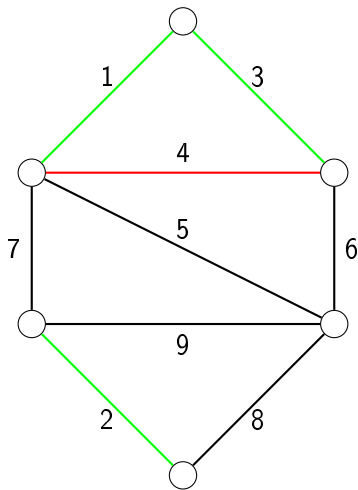
Пусть $m > n$. Поскольку $e(y_n) < e(z_n)$, то z_n допустим для y_1, \dots, y_n . Но тогда жадный алгоритм включил бы его в эту последовательность.

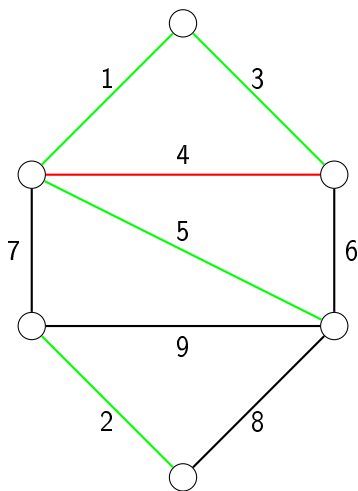


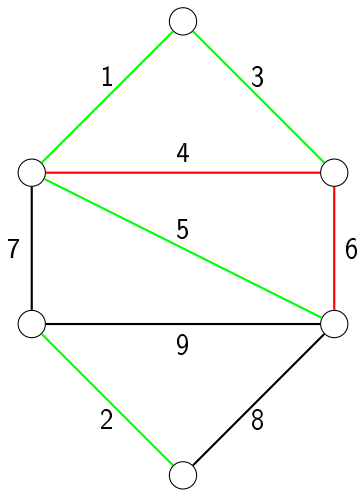


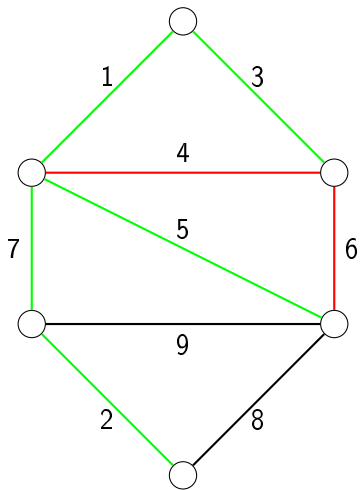


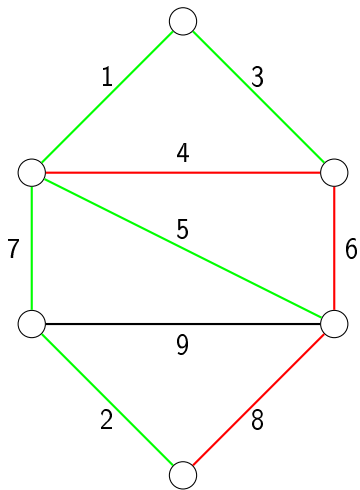


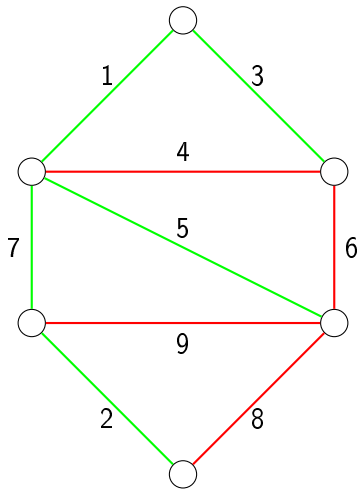












Теорема

На каждом шаге алгоритма Краскала, последовательность ребер e_1, \dots, e_k является подмножеством минимальное остовного дерева.

Теорема

На каждом шаге алгоритма Краскала, последовательность ребер e_1, \dots, e_k является подмножеством минимальное остовного дерева.

Доказательство.



Теорема

На каждом шаге алгоритма Краскала, последовательность ребер e_1, \dots, e_k является подмножеством минимальное остовного дерева.

Доказательство.

База индукции. Очевидно для $k = 0$ и пустой последовательности.



Теорема

На каждом шаге алгоритма Краскала, последовательность ребер e_1, \dots, e_k является подмножеством минимальное остовного дерева.

Доказательство.

База индукции. Очевидно для $k = 0$ и пустой последовательности.

Шаг индукции. Пусть $F = \{e_1, \dots, e_{k-1}\}$. По предположению индукции, существует минимальное остовное дерево T , содержащее F . Если T содержит e_k , то шаг индукции выполняется.



Теорема

На каждом шаге алгоритма Краскала, последовательность ребер e_1, \dots, e_k является подмножеством минимальное остовного дерева.

Доказательство.

База индукции. Очевидно для $k = 0$ и пустой последовательности.

Шаг индукции. Пусть $F = \{e_1, \dots, e_{k-1}\}$. По предположению индукции, существует минимальное остовное дерево T , содержащее F . Если T содержит e_k , то шаг индукции выполняется.

Если нет, то $T + e_k$ содержит цикл C . Этот цикл содержит некое ребро p , такое что p не входит в F (в противном случае, $F + e_k$ содержит цикл, и алгоритм Краскала не мог бы выбрать e_k как продолжение F).



Теорема

На каждом шаге алгоритма Краскала, последовательность ребер e_1, \dots, e_k является подмножеством минимальное остовного дерева.

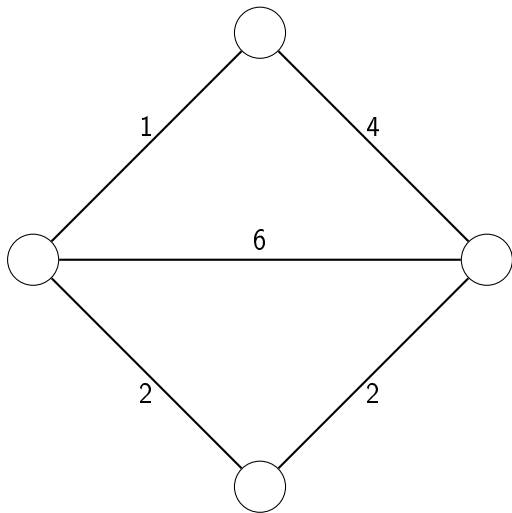
Доказательство.

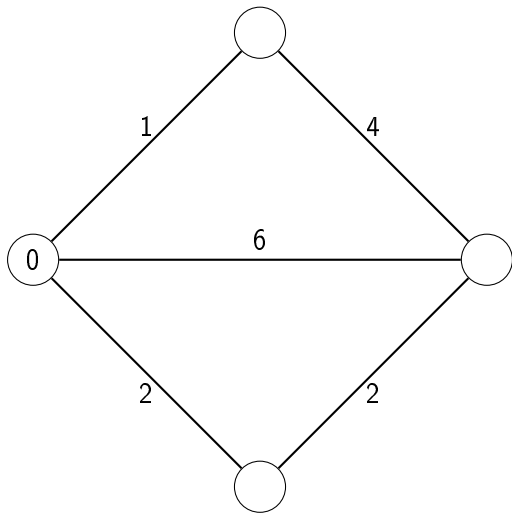
База индукции. Очевидно для $k = 0$ и пустой последовательности.

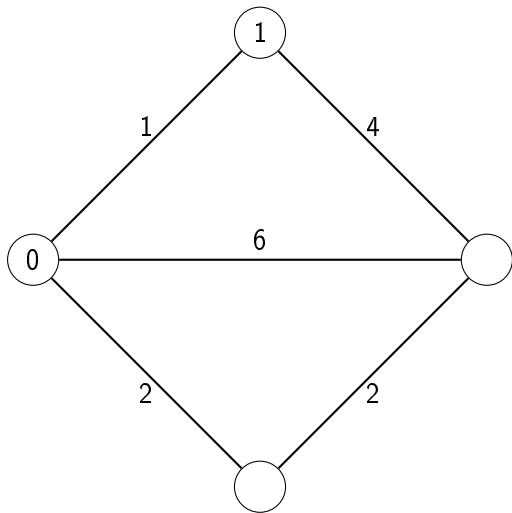
Шаг индукции. Пусть $F = \{e_1, \dots, e_{k-1}\}$. По предположению индукции, существует минимальное остовное дерево T , содержащее F . Если T содержит e_k , то шаг индукции выполняется.

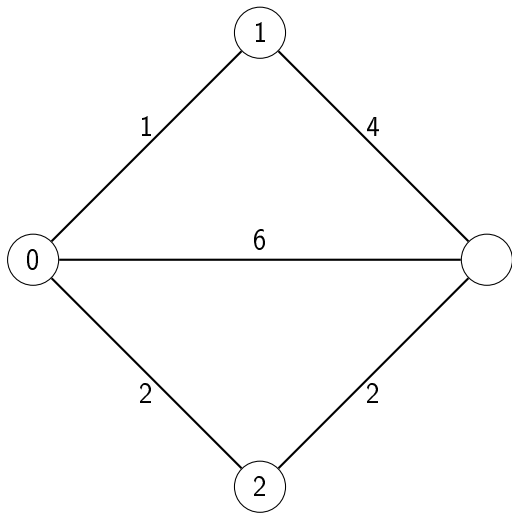
Если нет, то $T + e_k$ содержит цикл C . Этот цикл содержит некое ребро p , такое что p не входит в F (в противном случае, $F + e_k$ содержит цикл, и алгоритм Краскала не мог бы выбрать e_k как продолжение F).

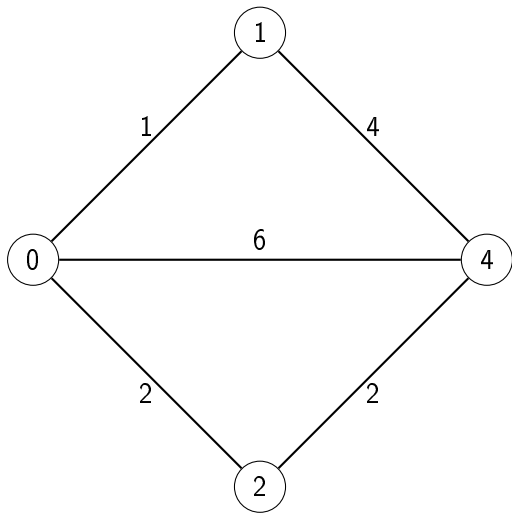
Тогда $T - p + e_k$ является деревом. Учтем, что $w(e_k) \leq w(p)$, поскольку жадный алгоритм выбрал e_k , а не p . Следовательно, $w(T - p + e_k) \leq w(T)$, но T — оптимально, следовательно, $T - p + e_k$ также оптимально. □

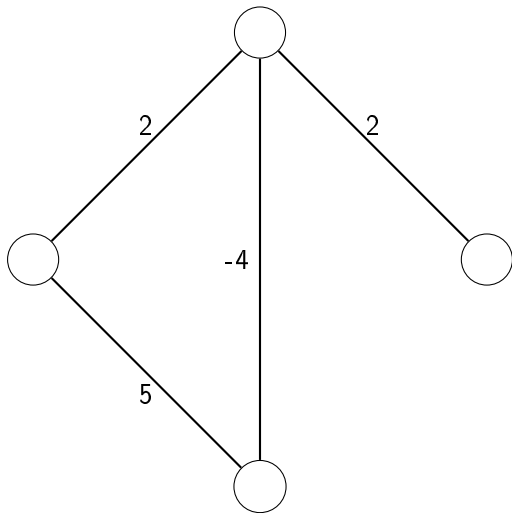


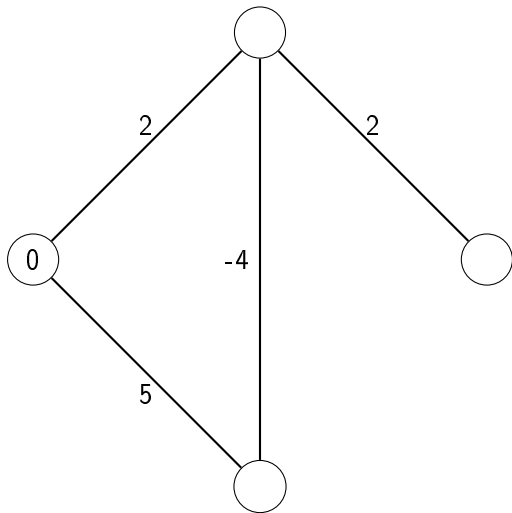


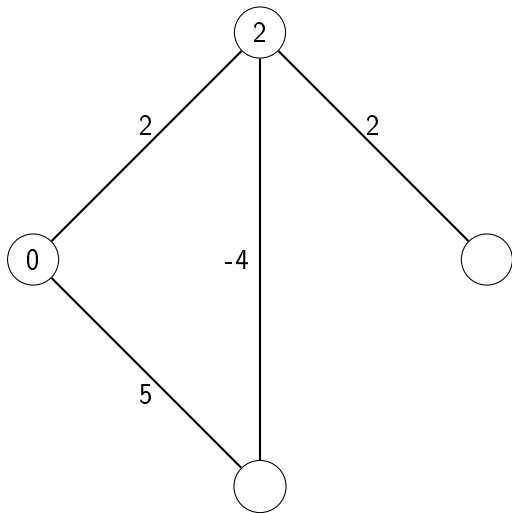


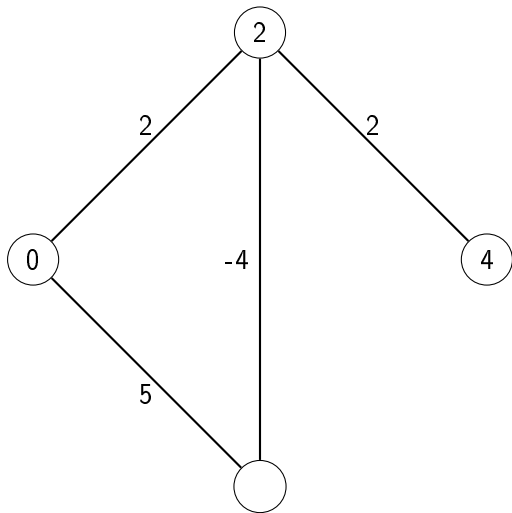


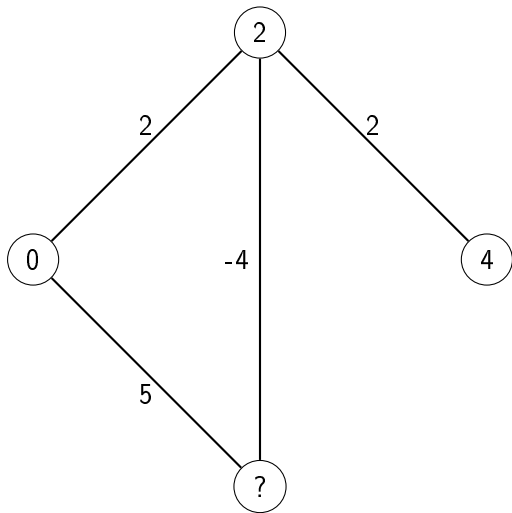












Обозначим `visited` через S , `visited[v].PathLength` через $p(v)$, `start` через v_0 .

Обозначим `visited` через S , `visited[v].PathLength` через $p(v)$, `start` через v_0 .

Теорема

Пусть все веса в графе неотрицательны. Тогда на каждом шаге алгоритма для всех v из S , $p(v)$ является длиной кратчайшего пути из v_0 в v .

Обозначим `visited` через S , `visited[v].PathLength` через $p(v)$, `start` через v_0 .

Теорема

Пусть все веса в графе неотрицательны. Тогда на каждом шаге алгоритма для всех v из S , $p(v)$ является длиной кратчайшего пути из v_0 в v .

Доказательство.

Обозначим `visited` через S , `visited[v].PathLength` через $p(v)$, `start` через v_0 .

Теорема

Пусть все веса в графе неотрицательны. Тогда на каждом шаге алгоритма для всех v из S , $p(v)$ является длиной кратчайшего пути из v_0 в v .

Доказательство.

Индукция по количеству вершин в S .

Обозначим `visited` через S , `visited[v].PathLength` через $p(v)$, `start` через v_0 .

Теорема

Пусть все веса в графе неотрицательны. Тогда на каждом шаге алгоритма для всех v из S , $p(v)$ является длиной кратчайшего пути из v_0 в v .

Доказательство.

Индукция по количеству вершин в S .

База индукции: очевидно для v_0 .

Обозначим `visited` через S , `visited[v].PathLength` через $p(v)$, `start` через v_0 .

Теорема

Пусть все веса в графе неотрицательны. Тогда на каждом шаге алгоритма для всех v из S , $p(v)$ является длиной кратчайшего пути из v_0 в v .

Доказательство.

Индукция по количеству вершин в S .

База индукции: очевидно для v_0 .

Шаг индукции. Пусть v – вершина, которую алгоритм добавляет в S по ребру (u, v) . Пусть P_u – путь, найденный алгоритмом из v_0 в u , P_v – аналогичный путь для v . По предположению индукции P_u является кратчайшим путем из v_0 в u . По выбору ребра (u, v) , P_v является кратчайшим путем из v_0 в v из тех, что проходят только через вершины из S .

Обозначим $visited$ через S , $visited[v].PathLength$ через $p(v)$, $start$ через v_0 .

Теорема

Пусть все веса в графе неотрицательны. Тогда на каждом шаге алгоритма для всех v из S , $p(v)$ является длиной кратчайшего пути из v_0 в v .

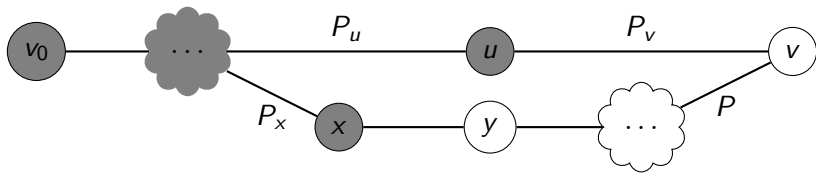
Доказательство.

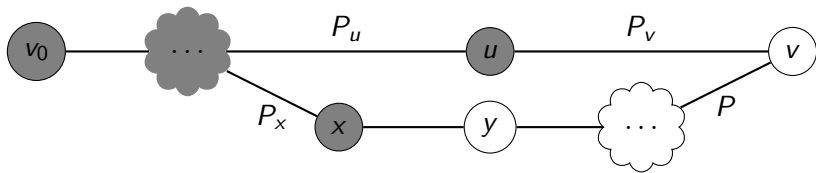
Индукция по количеству вершин в S .

База индукции: очевидно для v_0 .

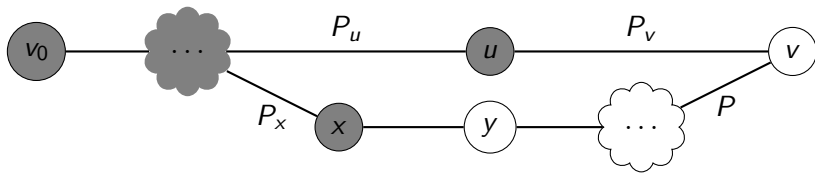
Шаг индукции. Пусть v – вершина, которую алгоритм добавляет в S по ребру (u, v) . Пусть P_u – путь, найденный алгоритмом из v_0 в u , P_v – аналогичный путь для v . По предположению индукции P_u является кратчайшим путем из v_0 в u . По выбору ребра (u, v) , P_v является кратчайшим путем из v_0 в v из тех, что проходят только через вершины из S .

Предположим, что P_v не является кратчайшим. Следовательно, существует другой путь P с меньшей длиной. Поскольку P_v – кратчайший из путей, которые состоят только из вершин S , в P должна быть вершина не из S . Обозначим первую такую вершину как u , предшествующую ей – как x .

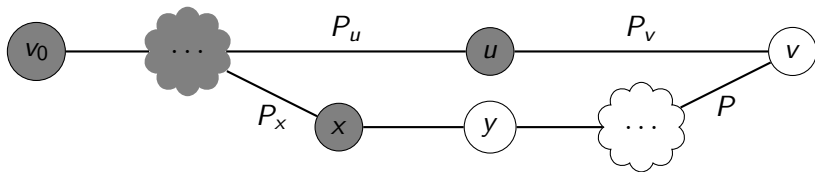




$$\mathcal{L}(P_v)$$

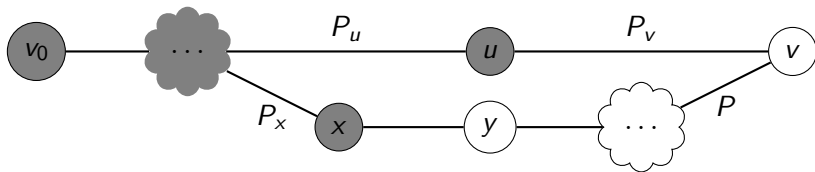


$$\mathcal{L}(P_v) = \mathcal{L}(P_u) + w(u, v)$$



$$\mathcal{L}(P_v) = \mathcal{L}(P_u) + w(u, v) \stackrel{1}{\leq} \mathcal{L}(P_x) + w(x, y)$$

1) т.к. для добавления выбрана v , а не y



$$\mathcal{L}(P_v) = \mathcal{L}(P_u) + w(u, v) \stackrel{1}{\leq} \mathcal{L}(P_x) + w(x, y) \stackrel{2}{\leq} \mathcal{L}(P)$$

- 1) т.к. для добавления выбрана v , а не y
- 2) т.к. все веса неотрицательны