Деревья

Владимир Подольский

Факультет компьютерных наук, Высшая Школа Экономики

Деревья

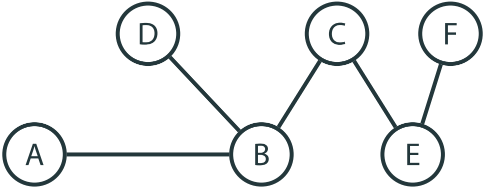
Понятие дерева, примеры

Свойства деревьев

Корневые деревья

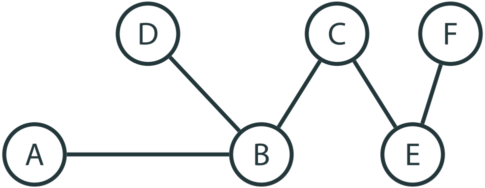
Деревья

- Деревом называется связный граф без простых циклов



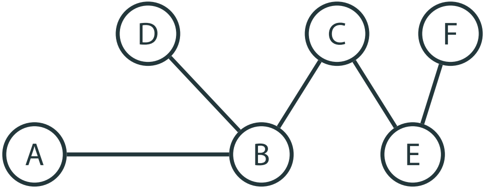
Деревья

- Деревом называется связный граф без простых циклов
- Деревья часто встречаются на практике



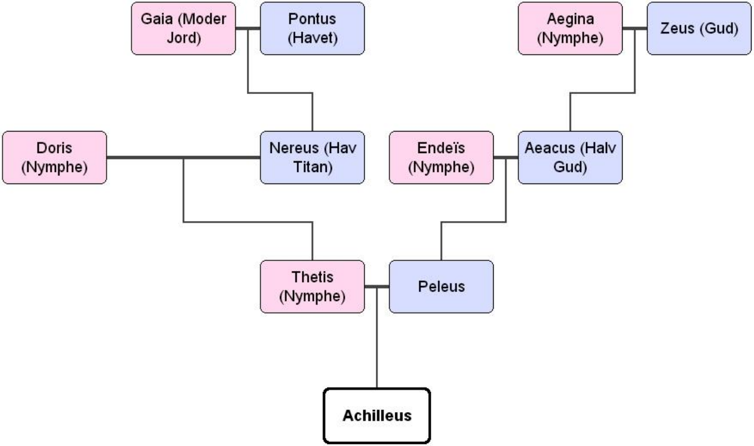
Деревья

- Деревом называется связный граф без простых циклов
- Деревья часто встречаются на практике
- И обладают важными свойствами



Деревья

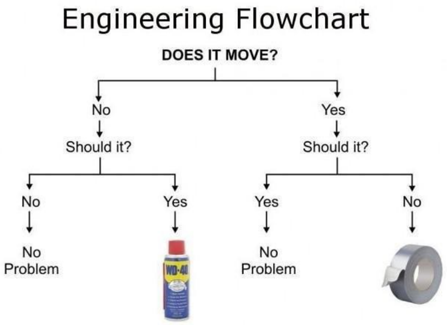
Пример: генеалогическое дерево



wikimedia.org

Деревья

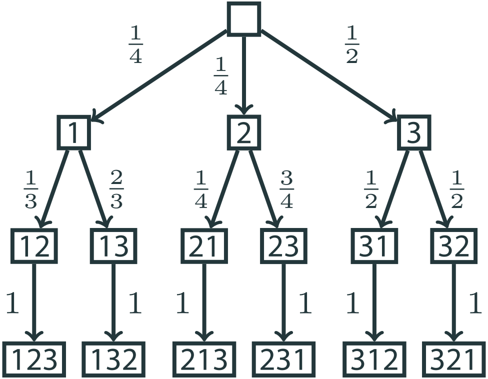
Пример: дерево принятия решений



flickr.com

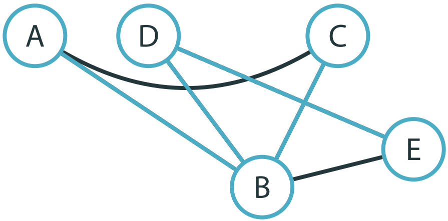
Деревья

Пример: задание вероятностных распределений



Деревья

Пример: обход графа



Определения дерева

Следующие три определения эквивалентны:

(1) Дерево — это связный граф без циклов

Определения дерева

Следующие три определения эквивалентны:

(1) Дерево — это связный граф без циклов

(2) Дерево — это связный граф на n вершинах с $n - 1$ ребром

Определения дерева

Следующие три определения эквивалентны:

(1) Дерево — это связный граф без циклов

(2) Дерево — это связный граф на n вершинах с $n-1$ ребром

(3) Дерево — это граф, в котором между любыми двумя вершинами есть ровно один простой путь

Определения дерева

Следующие три определения эквивалентны:

(1) Дерево — это связный граф без циклов

(2) Дерево — это связный граф на n вершинах с $n-1$ ребром

(3) Дерево — это граф, в котором между любыми двумя вершинами есть ровно один простой путь

Мы докажем цепочку следствий (1) $\to$ (2) $\to$ (3) $\to$ (1)

Эквивалентность определений

- Вспомним рассуждение про число вершин, ребер и компонент связности

Эквивалентность определений

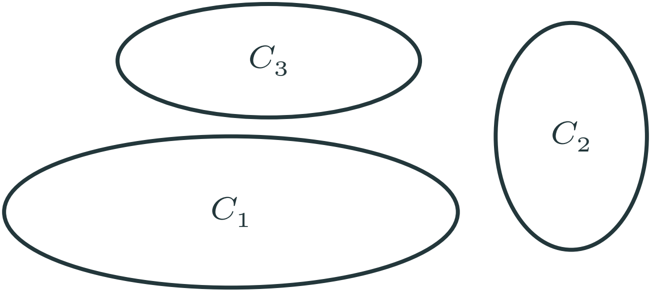
- Вспомним рассуждение про число вершин, ребер и компонент связности
- Мы выкидывали из графа все ребра и возвращали их по одному

Эквивалентность определений

- Вспомним рассуждение про число вершин, ребер и компонент связности
- Мы выкидывали из графа все ребра и возвращали их по одному
- При возвращении одного ребра число компонент связности либо не меняется, либо уменьшается на 1

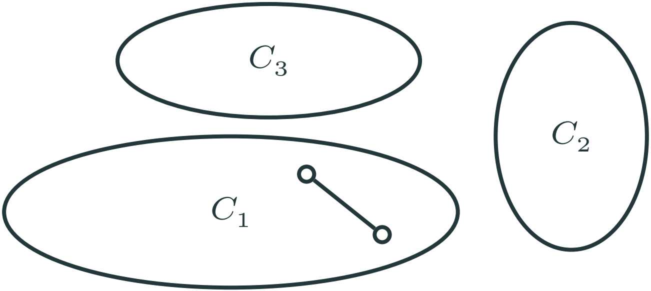
Эквивалентность определений

- Посмотрим на текущие компоненты связности



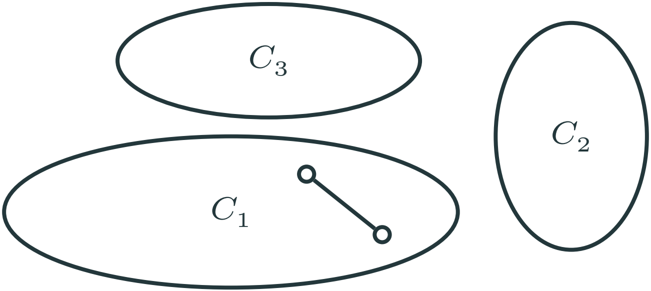
Эквивалентность определений

- Посмотрим на текущие компоненты связности
- Пусть новое ребро соединяет вершины в одной компоненте



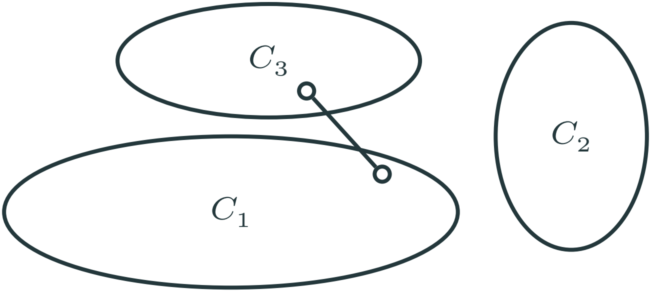
Эквивалентность определений

- Посмотрим на текущие компоненты связности
- Пусть новое ребро соединяет вершины в одной компоненте
- Тогда появляется цикл!



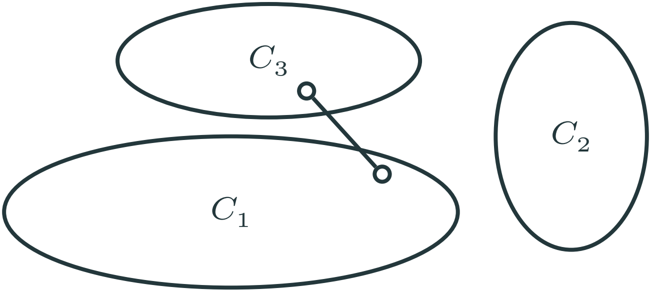
Эквивалентность определений

- Пусть новое ребро соединяет вершины в разных компонентах



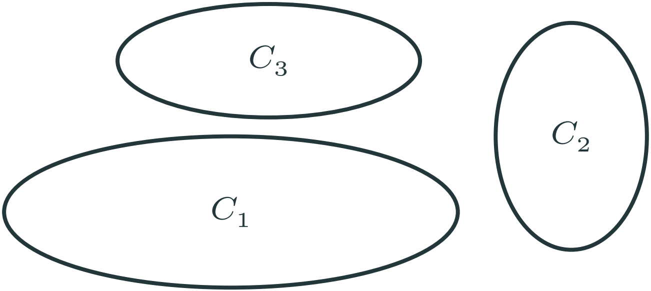
Эквивалентность определений

- Пусть новое ребро соединяет вершины в разных компонентах
- Тогда цикла не появляется



Эквивалентность определений

- Итак, цикл в графе появляется тогда и только тогда, когда возвращаемое ребро соединяет вершины одной компоненты



Эквивалентность определений

Следующие три определения эквивалентны:

(1) Дерево — это связный граф без циклов

(2) Дерево — это связный граф на n вершинах с $n - 1$ ребром

(3) Дерево — это граф, в котором между любыми двумя вершинами есть ровно один простой путь

Мы докажем цепочку следствий (1) $\to$ (2) $\to$ (3) $\to$ (1)

Эквивалентность определений

Следующие три определения эквивалентны:

(1) Дерево — это связный граф без циклов

(2) Дерево — это связный граф на n вершинах с $n-1$ ребром

(3) Дерево — это граф, в котором между любыми двумя вершинами есть ровно один простой путь

Мы докажем цепочку следствий (1) $\rightarrow$ (2) $\rightarrow$ (3) $\rightarrow$ (1)

Перейдем к доказательству

(1) $\to$ (2)

(1) $\to$ (2)

В связном графе без циклов на n вершинах ровно $n-1$ ребро

- Пусть у нас связный граф без циклов

(1) $\to$ (2)

(1) $\to$ (2)

В связном графе без циклов на n вершинах ровно $n-1$ ребро

- Пусть у нас связный граф без циклов
- Удалим все ребра и будем возвращать их по одному

(1) $\to$ (2)

(1) $\to$ (2)

В связном графе без циклов на n вершинах ровно $n-1$ ребро

- Пусть у нас связный граф без циклов
- Удалим все ребра и будем возвращать их по одному
- Поскольку в графе нет циклов, каждое ребро соединяет разные компоненты

(1) $\to$ (2)

(1) $\to$ (2)

В связном графе без циклов на n вершинах ровно $n-1$ ребро

- Пусть у нас связный граф без циклов
- Удалим все ребра и будем возвращать их по одному
- Поскольку в графе нет циклов, каждое ребро соединяет разные компоненты
- Каждый раз компонент становится на одну меньше

(1) $\to$ (2)

(1) $\to$ (2)

В связном графе без циклов на n вершинах ровно $n-1$ ребро

- Пусть у нас связный граф без циклов
- Удалим все ребра и будем возвращать их по одному
- Поскольку в графе нет циклов, каждое ребро соединяет разные компоненты
- Каждый раз компонент становится на одну меньше
- В начале компонент n, в конце — 1

(1) $\to$ (2)

(1) $\to$ (2)

В связном графе без циклов на n вершинах ровно $n - 1$ ребро

- Пусть у нас связный граф без циклов
- Удалим все ребра и будем возвращать их по одному
- Поскольку в графе нет циклов, каждое ребро соединяет разные компоненты
- Каждый раз компонент становится на одну меньше
- В начале компонент n, в конце — 1
- Значит мы добавили ровно $n - 1$ ребро!

Эквивалентность определений

Следующие три определения эквивалентны:

(1) Дерево — это связный граф без циклов

(2) Дерево — это связный граф на n вершинах с $n - 1$ ребром

(3) Дерево — это граф, в котором между любыми двумя вершинами есть ровно один простой путь

Мы докажем цепочку следствий (1) $\to$ (2) $\to$ (3) $\to$ (1)

(2) $\to$ (3)

(2) $\to$ (3)

В связном графе с n вершинами и $n-1$ ребром между любыми двум вершинами есть ровно один простой путь

(2) $\to$ (3)

(2) $\to$ (3)

В связном графе с n вершинами и $n-1$ ребром между любыми двум вершинами есть ровно один простой путь

- Пусть между какими-то вершинами есть два пути

(2) $\to$ (3)

(2) $\to$ (3)

В связном графе с n вершинами и $n-1$ ребром между любыми двум вершинами есть ровно один простой путь

- Пусть между какими-то вершинами есть два пути
- Тогда в графе есть и цикл, в нем k вершин и k ребер

(2) $\to$ (3)

(2) $\to$ (3)

В связном графе с n вершинами и $n-1$ ребром между любыми двум вершинами есть ровно один простой путь

- Пусть между какими-то вершинами есть два пути
- Тогда в графе есть и цикл, в нем k вершин и k ребер
- Удалим все ребра кроме этого цикла и будем возвращать по одному

(2) $\to$ (3)

(2) $\to$ (3)

В связном графе с n вершинами и $n-1$ ребром между любыми двум вершинами есть ровно один простой путь

- До возвращения у нас $n-k+1$ компонента связности

(2) $\to$ (3)

(2) $\to$ (3)

В связном графе с n вершинами и $n-1$ ребром между любыми двум вершинами есть ровно один простой путь

- До возвращения у нас $n-k+1$ компонента связности
- Чтобы получить одну компоненту, нужно добавить хотя бы $n-k$ ребер

(2) $\to$ (3)

(2) $\to$ (3)

В связном графе с n вершинами и $n-1$ ребром между любыми двум вершинами есть ровно один простой путь

- До возвращения у нас $n-k+1$ компонента связности
- Чтобы получить одну компоненту, нужно добавить хотя бы $n-k$ ребер
- Но тогда всего у нас будет $k+(n-k)=n$ ребер!

(2) $\to$ (3)

(2) $\to$ (3)

В связном графе с n вершинами и $n-1$ ребром между любыми двум вершинами есть ровно один простой путь

- До возвращения у нас $n-k+1$ компонента связности
- Чтобы получить одну компоненту, нужно добавить хотя бы $n-k$ ребер
- Но тогда всего у нас будет $k+(n-k)=n$ ребер!
- Противоречие

Эквивалентность определений

Следующие три определения эквивалентны:

(1) Дерево — это связный граф без циклов

(2) Дерево — это связный граф на n вершинах с $n - 1$ ребром

(3) Дерево — это граф, в котором между любыми двумя вершинами есть ровно один простой путь

Мы докажем цепочку следствий (1) $\to$ (2) $\to$ (3) $\to$ (1)

(3) $\to$ (1)

(3) $\to$ (1)

Если в графе между любыми двумя вершинами есть ровно один простой путь, то это связный граф без циклов

$(3) \to (1)$

$(3) \to (1)$

Если в графе между любыми двумя вершинами есть ровно один простой путь, то это связный граф без циклов

- Пусть в графе между любыми двумя вершинами есть ровно один простой путь

(3) $\to$ (1)

(3) $\to$ (1)

Если в графе между любыми двумя вершинами есть ровно один простой путь, то это связный граф без циклов

- Пусть в графе между любыми двумя вершинами есть ровно один простой путь
- Очевидно, он связен

$(3) \to (1)$

$(3) \to (1)$

Если в графе между любыми двумя вершинами есть ровно один простой путь, то это связный граф без циклов

- Пусть в графе между любыми двумя вершинами есть ровно один простой путь
- Очевидно, он связен
- Если бы в графе был цикл, то между двумя вершинами цикла было бы два пути

(3) $\to$ (1)

(3) $\to$ (1)

Если в графе между любыми двумя вершинами есть ровно один простой путь, то это связный граф без циклов

- Пусть в графе между любыми двумя вершинами есть ровно один простой путь
- Очевидно, он связен
- Если бы в графе был цикл, то между двумя вершинами цикла было бы два пути
- Значит циклов нет

Эквивалентность определений

Итак, мы доказали эквивалентность трех определений:

(1) Дерево — это связный граф без циклов

(2) Дерево — это связный граф на n вершинах с $n - 1$ ребром

(3) Дерево — это граф, в котором между любыми двумя вершинами есть ровно один простой путь

Деревья

Понятие дерева, примеры

Свойства деревьев

Корневые деревья

Листья

Утверждение

Если в дереве больше 1 вершины, то в нем есть вершина степени 1

Листья

Утверждение

Если в дереве больше 1 вершины, то в нем есть вершина степени 1

- Такая вершина называется листом

Листья

Утверждение

Если в дереве больше 1 вершины, то в нем есть вершина степени 1

- Такая вершина называется листом
- Пусть в дереве n вершин, тогда в нем $n - 1$ ребро

Листья

Утверждение

Если в дереве больше 1 вершины, то в нем есть вершина степени 1

- Такая вершина называется листом
- Пусть в дереве n вершин, тогда в нем $n - 1$ ребро
- Вспомним соотношение между степенями вершин и числом ребер: $\sum_{v \in V} d(v) = 2|E|$

Листья

Утверждение

Если в дереве больше 1 вершины, то в нем есть вершина степени 1

- Такая вершина называется листом
- Пусть в дереве n вершин, тогда в нем $n-1$ ребро
- Вспомним соотношение между степенями вершин и числом ребер: $\sum_{v \in V} d(v) = 2|E|$
- Значит $\sum_{v \in V} d(v) = 2n - 2$

Листья

Утверждение

Если в дереве больше 1 вершины, то в нем есть вершина степени 1

- В дереве нет вершин степени 0

Листья

Утверждение

Если в дереве больше 1 вершины, то в нем есть вершина степени 1

- В дереве нет вершин степени 0
- Если нет вершин степени 1, то все вершины степени не меньше 2

Листья

Утверждение

Если в дереве больше 1 вершины, то в нем есть вершина степени 1

- В дереве нет вершин степени 0
- Если нет вершин степени 1, то все вершины степени не меньше 2
- Тогда сумма степеней вершин не меньше $2n$

Листья

Утверждение

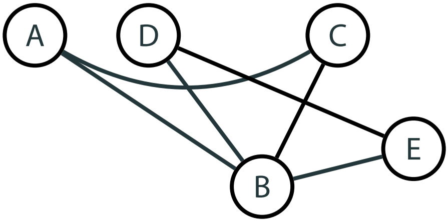
Если в дереве больше 1 вершины, то в нем есть вершина степени 1

- В дереве нет вершин степени 0
- Если нет вершин степени 1, то все вершины степени не меньше 2
- Тогда сумма степеней вершин не меньше $2n$
- Но она равна $2n-2$, противоречие

Остовное дерево

Утверждение

Из всякого связного графа G можно удалить часть ребер так, что останется дерево

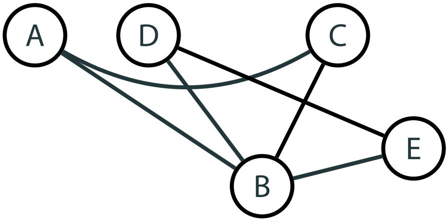


Остовное дерево

Утверждение

Из всякого связного графа G можно удалить часть ребер так, что останется дерево

- Такое дерево называется остовным деревом графа G

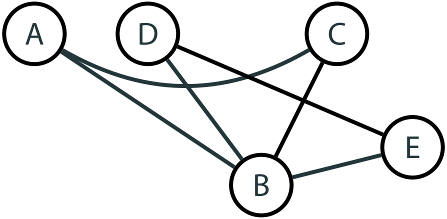


Остовное дерево

Утверждение

Из всякого связного графа G можно удалить часть ребер так, что останется дерево

- Такое дерево называется остовным деревом графа G
- Остовное дерево не единственно

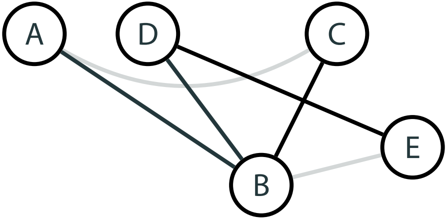


Остовное дерево

Утверждение

Из всякого связного графа G можно удалить часть ребер так, что останется дерево

- Такое дерево называется остовным деревом графа G
- Остовное дерево не единственно

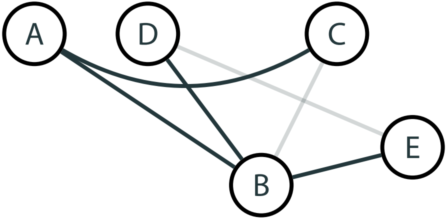


Остовное дерево

Утверждение

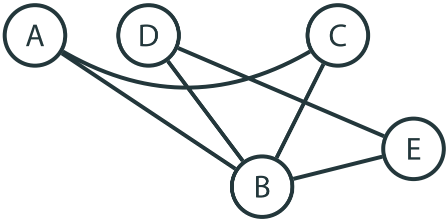
Из всякого связного графа G можно удалить часть ребер так, что останется дерево

- Такое дерево называется остовным деревом графа G
- Остовное дерево не единственно



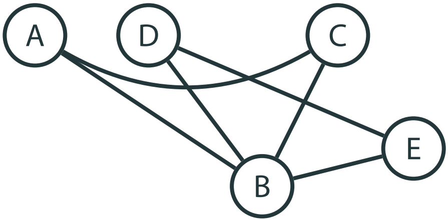
Остовное дерево

- Почему это верно?



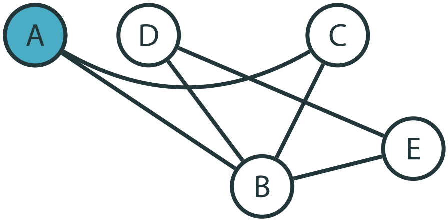
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



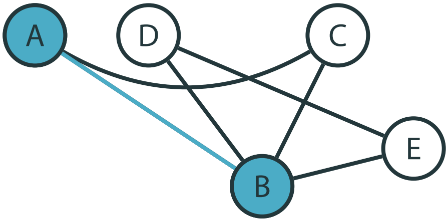
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



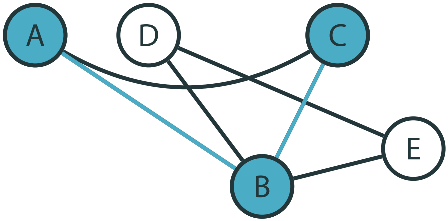
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



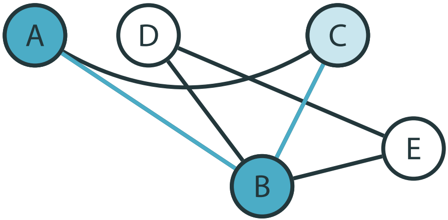
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



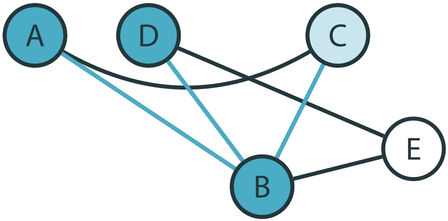
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



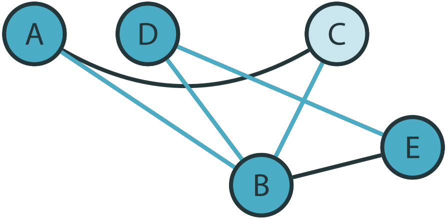
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



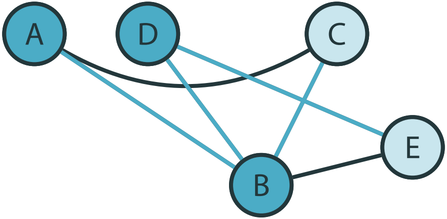
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



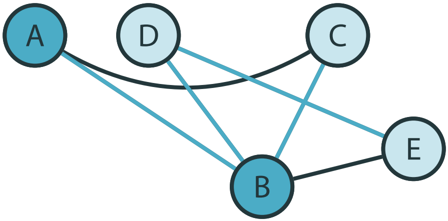
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



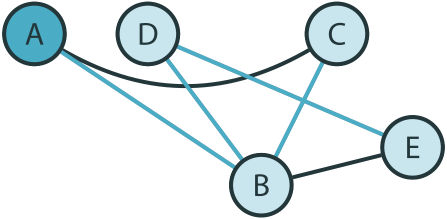
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



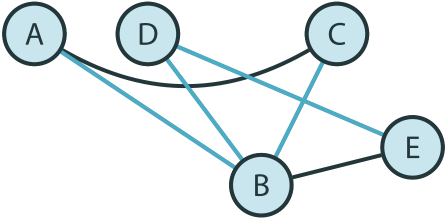
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



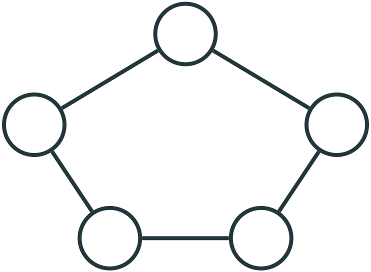
Остовное дерево

- Почему это верно?
- Каждый связный граф можно обойти поиском в глубину или ширину, обход дает остовное дерево



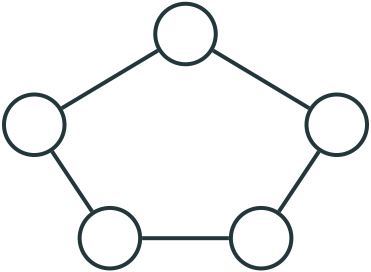
Остовное дерево

- Есть и другое объяснение



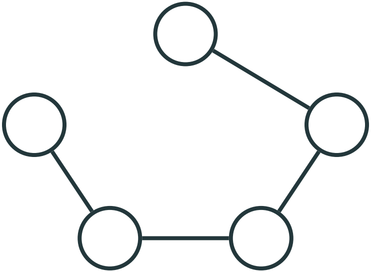
Остовное дерево

- Есть и другое объяснение
- Если в графе есть цикл, удалим любое ребро этого цикла



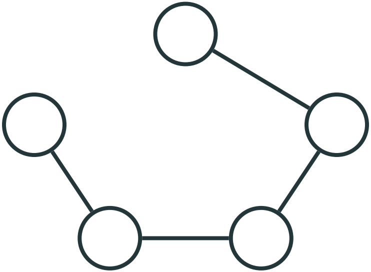
Остовное дерево

- Есть и другое объяснение
- Если в графе есть цикл, удалим любое ребро этого цикла



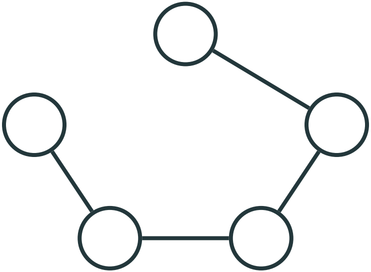
Остовное дерево

- Есть и другое объяснение
- Если в графе есть цикл, удалим любое ребро этого цикла
- Граф останется связным: проход по удаленному ребру можно заменить на обход по оставшемуся циклу



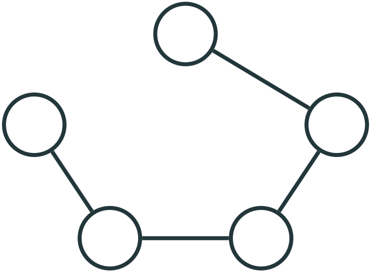
Остовное дерево

- Будем повторять пока в графе есть циклы



Остовное дерево

- Будем повторять пока в графе есть циклы
- В конце получим связный граф без циклов, то есть дерево



Деревья

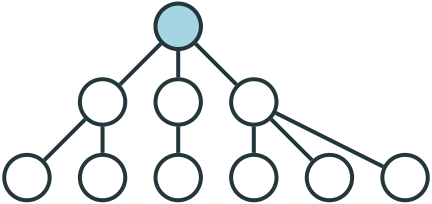
Понятие дерева, примеры

Свойства деревьев

Корневые деревья

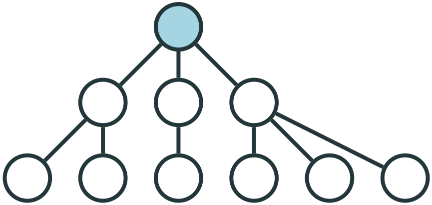
Корневое дерево

- Корневое дерево — дерево с выделенной вершиной



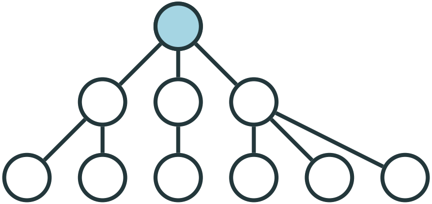
Корневое дерево

- Корневое дерево — дерево с выделенной вершиной
- Выделенную вершину называют корнем



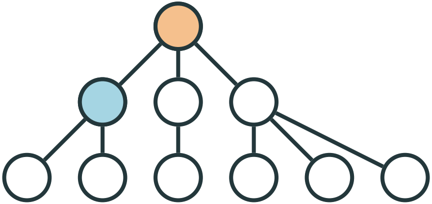
Корневое дерево

- Корневое дерево — дерево с выделенной вершиной
- Выделенную вершину называют корнем
- Обычно корень рисуют вверху, ее соседей ниже, их соседей еще ниже и так далее



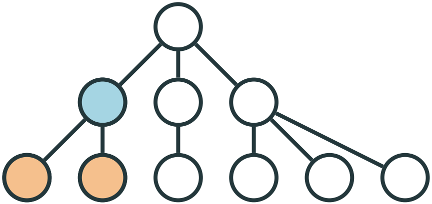
Корневое дерево

- Предком вершины называется ее сосед, лежащий на пути от вершины к корню



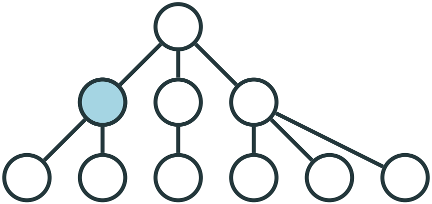
Корневое дерево

- Предком вершины называется ее сосед, лежащий на пути от вершины к корню
- Потомками вершины называются все остальные ее соседи



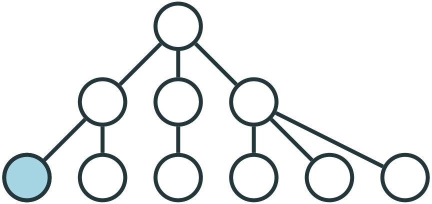
Корневое дерево

- Глубина вершины — длина простого пути из корня в эту вершину



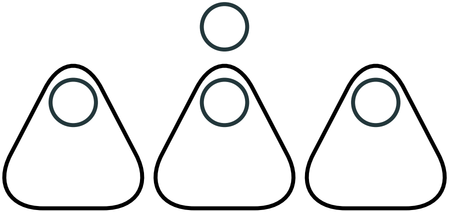
Корневое дерево

- Глубина вершины — длина простого пути из корня в эту вершину
- Глубина дерева — наибольшая глубина его вершин



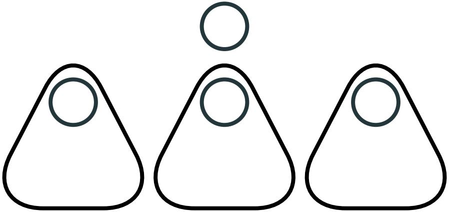
Корневое дерево

- Корневые деревья можно также определять рекурсивно



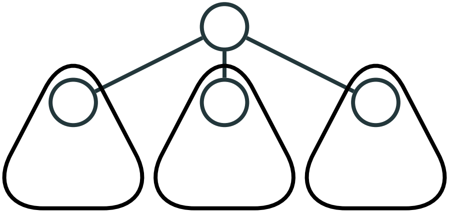
Корневое дерево

- Корневые деревья можно также определять рекурсивно
- Пусть у нас есть вершина v и корневые деревья $T_1, \dots, T_k$



Корневое дерево

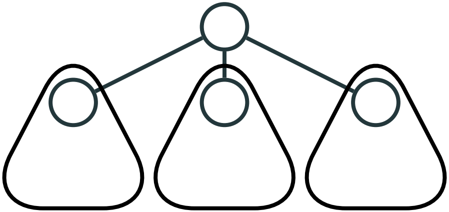
- Тогда мы можем определить новое корневое дерево T: соединим v с корнями всех деревьев $T_1, \ldots, T_k$, объявим v корнем



Корневое дерево

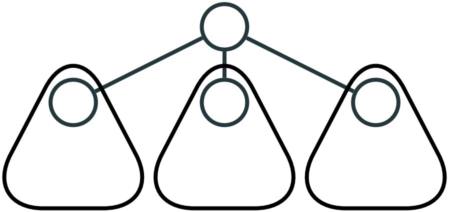
- Тогда мы можем определить новое корневое дерево T: соединим v с корнями всех деревьев $T_1, \ldots, T_k$, объявим v корнем

- Предком корней деревьев $T_1, \ldots, T_k$ объявим v



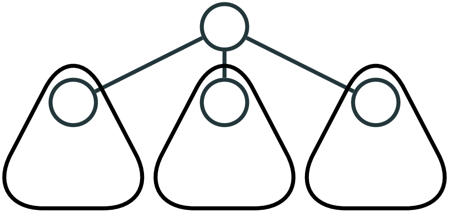
Корневое дерево

- Тогда мы можем определить новое корневое дерево T: соединим v с корнями всех деревьев $T_1, \ldots, T_k$, объявим v корнем
- Предком корней деревьев $T_1, \ldots, T_k$ объявим v
- Потомками v объявим корни деревьев $T_1, \ldots, T_k$



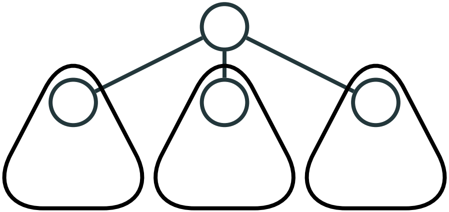
Корневое дерево

- Глубину вершины v объявляем равной 0



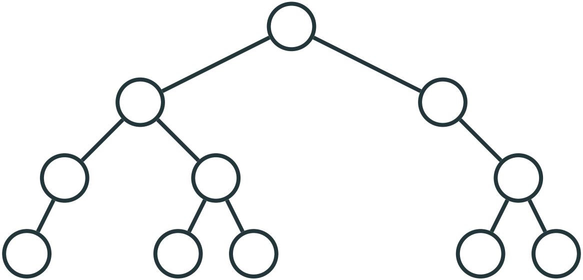
Корневое дерево

- Глубину вершины v объявляем равной 0
- Глубину всех вершин в $T_1, \ldots, T_k$ увеличиваем на 1



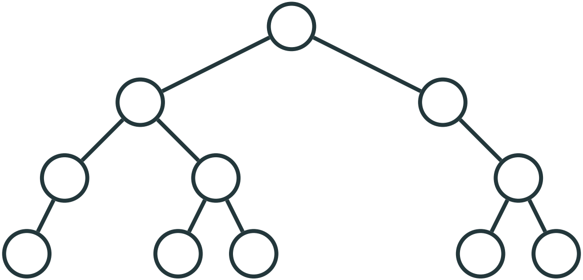
Двоичное дерево

- Корневое дерево называется двоичным, если у каждой вершины не более двух потомком



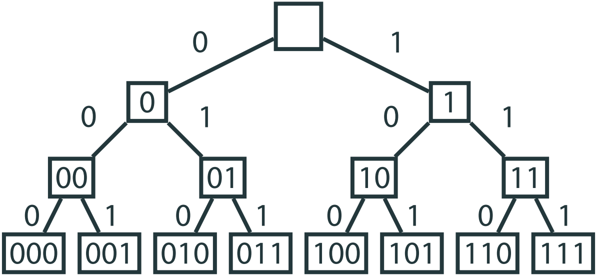
Двоичное дерево

- Корневое дерево называется двоичным, если у каждой вершины не более двух потомком
- В деревьях принятия решений это соответствует вопросам с двумя возможными ответами



Полное двоичное дерево

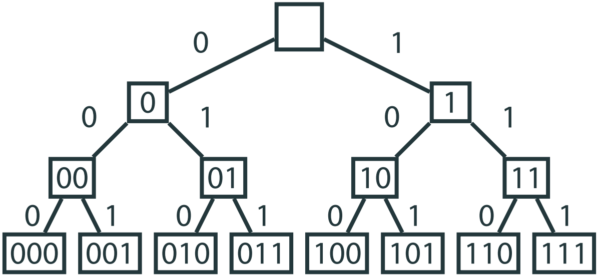
- Полное двоичное дерево глубины n — двоичное дерево, в котором присутствуют все возможные вершины глубины n



Полное двоичное дерево

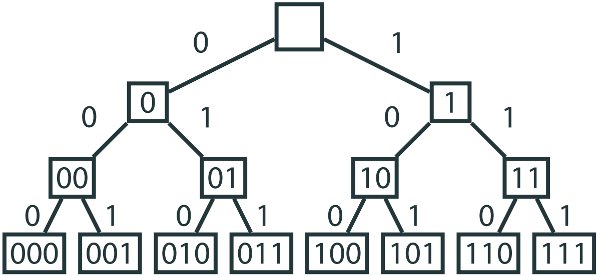
- Полное двоичное дерево глубины n — двоичное дерево, в котором присутствуют все возможные вершины глубины n

- У всех не листьев два потомка



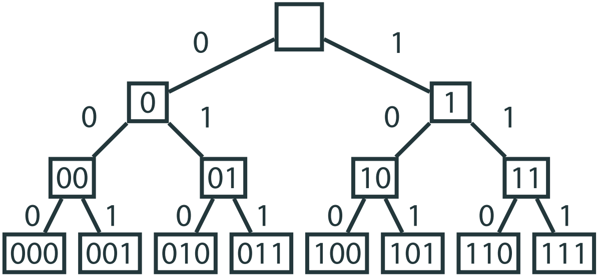
Полное двоичное дерево

- Удобно пометить ребра из каждой вершины к потомкам метками 0 и 1



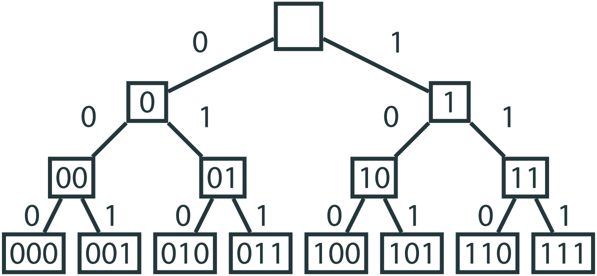
Полное двоичное дерево

- Удобно пометить ребра из каждой вершины к потомкам метками 0 и 1
- Тогда каждая вершина полного двоичного дерева кодируется последовательностью нулей и единиц



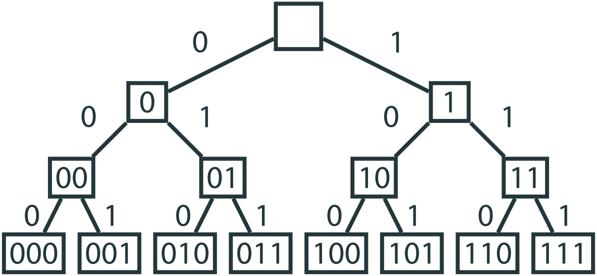
Полное двоичное дерево

- Корень кодируется пустой последовательностью



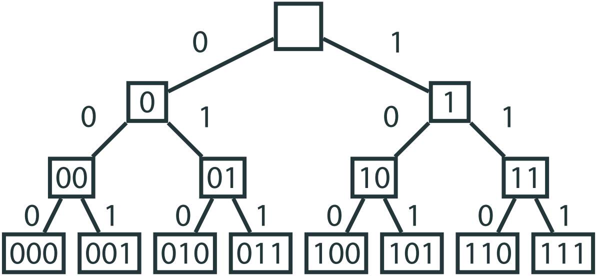
Полное двоичное дерево

- Корень кодируется пустой последовательностью
- Каждая вершина кодируется кодом своего предка с приписанной меткой ребра



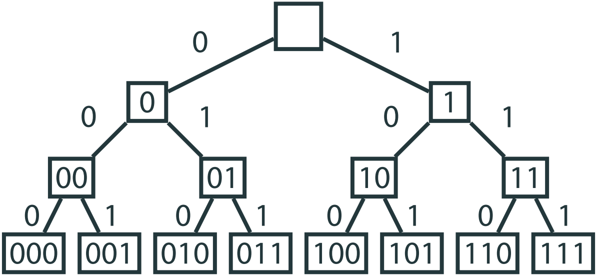
Полное двоичное дерево

- Длина последовательности равна глубине вершины



Полное двоичное дерево

- Длина последовательности равна глубине вершины
- Листья кодируются последовательностями из $\{0, 1\}^n$



Полное двоичное дерево

В полном двоичном дереве глубины n есть:

- 2^k вершин глубины k: столько элементов в $\{0, 1\}^k$

Полное двоичное дерево

В полном двоичном дереве глубины n есть:

- 2^k вершин глубины k: столько элементов в $\{0, 1\}^k$
- 2^n листьев: это вершины глубины n

Полное двоичное дерево

В полном двоичном дереве глубины n есть:

- 2^k вершин глубины k: столько элементов в $\{0,1\}^k$
- 2^n листьев: это вершины глубины n
- $2^{n+1}-1$ вершин: суммируем по всем глубинам, получаем $1+2+\ldots+2^n = 2^{n+1}-1$

Полное двоичное дерево

В полном двоичном дереве глубины n есть:

- 2^k вершин глубины k: столько элементов в $\{0, 1\}^k$
- 2^n листьев: это вершины глубины n
- $2^{n+1} - 1$ вершин: суммируем по всем глубинам, получаем $1 + 2 + \ldots + 2^n = 2^{n+1} - 1$
- $2^{n+1} - 2$ ребер: на одно меньше, чем вершин

Что мы узнали

- Деревья — связные графы без циклов

Что мы узнали

- Деревья — связные графы без циклов
- Встречаются на практике

Что мы узнали

- Деревья — связные графы без циклов
- Встречаются на практике
- Обладают важными свойствами

Что мы узнали

- Деревья — связные графы без циклов
- Встречаются на практике
- Обладают важными свойствами
- Содержатся в каждом связном графе