Деревья. Задачи о кратчайших расстояниях на графах. Основные алгоритмы для решения задач о кратчайших расстояниях.

Связный граф G(V,E), не имеющий циклов, называется *деревом*.

ТЕОРЕМА (основные свойства деревьев):

Пусть граф G(V,E) имеет n вершин. Тогда следующие утверждения эквивалентны:

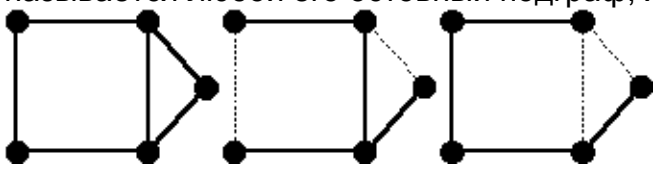
1. G является деревом;
2. G не содержит циклов и имеет n-1 рёбер;
3. G связен и имеет n-1 рёбер;
4. G связен, но удаление любого ребра нарушает связность;
5. две вершины графа G соединены ровно одним путём;
6. G не имеет циклов, но добавление любого ребра порождает ровно один цикл**.**

Ориентированное дерево представляет собой ориентированный граф без циклов, в котором полустепень захода каждой вершины (за исключением одной, например x_1) не больше 1, а полустепень захода вершины x_1 (называемой также корнем) равна нулю.

Вершину v ордерева называют **потомком** вершины u, если ∃ путь из u в v. В этом же случае вершину u называют **предком** вершины v. Вершину, не имеющую потомков, называют **листом**.

Высота ордерева – это наибольшая длина пути из корня в лист. **Глубина d(v) вершины** ордерева – длина пути из корня в эту вершину. **Высота вершины h(v)** – наибольшая длина пути из данной вершины в лист. **Уровень вершины** – это разница между высотой дерева и глубиной данной вершины. Ордерево называют **бинарным**, если полустепень исхода любой его вершины не превосходит 2.

Пусть задан неориентированный граф. **Остовным деревом (остовом)** связного графа называется любой его остовный подграф, являющийся деревом.



Граф и два его остовных дерева (удаленные ребра показаны пунктиром).

Задачи о кратчайших расстояниях на графах:

1. Построение минимального остовного дерева (кратчайшей связывающей сети) – соединение всех узлов сети с помощью путей наименьшей длины.
2. Задача о нахождении дерева кратчайших расстояний – нахождение кратчайшего пути из одной вершины в любую другую.

ЗАДАЧА 1(пример прикладной задачи): необходимо проложить линии коммуникаций (дороги, линии связи, электропередач и т.п.) между *n* заданными "точечными" объектами, при условии, что, во-первых, известны "расстояния" между каждой парой объектов (это может быть геометрическое расстояние или стоимость прокладки коммуникаций между ними), и, во-вторых, объекты могут быть связаны как непосредственно, так и с участием произвольного количества промежуточных объектов.

При допущении, что разветвления возможны только в этих же *n* объектах, задача сводится к нахождению кратчайшего остовного дерева (SST - shortest spanning tree, или MST - minimal spanning tree) во взвешенном графе, вершины которого соответствуют заданным объектам, а веса ребер равны "расстояниям" между ними.

Тогда формулировка задачи принимает вид: **найти кратчайшее остовное дерево(минимальный покрывающий остов) взвешенного графа.**

Пусть дан неориентированный связный граф со взвешенными ребрами. Вес ребра (x_i,x_j) обозначим c_{ij}. Из всех остовов графа необходимо найти один, у которого сумма весов на ребрах наименьшая.

Стоимость остовного дерева вычисляется как сумма стоимостей всех рёбер, входящих в это дерево.

Алгоритм Прима

Определим расстояние между произвольной вершиной v взвешенного графа G=(V,E) и некоторым его подграфом $G' \subseteq G$ как минимальный вес ребра, одним из концов которого является v, а другой лежит в G':

$d(v,G')=\min_{(v,w)\in E(G),w\in G'} Weight(v,w)$.

1. SST '=<граф, состоящий из одной произвольной вершины графа G>;
2. если |E(SST ')|=n-1, то SST=SST '; КОНЕЦ;
3. среди множества I вершин графа G, не входящих в SST ', но инцидентных хотя бы одной вершине из SST' ($I=\{u \mid u\in V(G), u\notin SST', (u,v)\in E(G), v\in SST'\}$), найти вершину $w\in I$, расстояние которой до SST' минимально: $d(w,SST')=\min_{v\in I} d(v,SST')$;
4. добавить ребро (w,u), на котором достигается минимальное расстояние d(w,SST'), в SST ';
5. перейти на шаг 2.

ЗАДАЧА 2(пример прикладной задачи): необходимо добраться на самолете из города A в город B при условии, что между ними нет прямого авиационного сообщения, затратив при этом минимальные средства (при условии, что заданы возможные промежуточные аэропорты, для каждой пары аэропортов известно, существует ли между ними прямой маршрут, и если да, то известна минимальная стоимость перелета по этому маршруту).

Решение:

построим взвешенный граф (орграф - если бывают "несимметричные" маршруты), вершины которого соответствуют всем возможным аэропортам, ребра (дуги) - прямым маршрутам между ними, а веса ребер (дуг) равны стоимости перелета (очевидно, неотрицательной) между соответствующими аэропортами. Задача сводится к нахождению в графе (орграфе) пути минимального веса между вершинами, соответствующими A и B.

Тогда задача имеет следующую формулировку: найти один из путей минимальной суммарной длины между двумя заданными вершинами взвешенного орграфа с неотрицательными весами дуг. (Построить дерево кратчайших расстояний).

Алгоритм Дейкстры (в случае неотрицательных весов.)

Дано: непустой взвешенный граф G=(V,E) с неотрицательными весами дуг. Требуется найти кратчайший путь от s к t ($s\neq t$).

Инициализация:

- всем вершинам vi приписывается метка - вещественное число: $d(s)=0$, $d(v_i)=\infty$ для всех $v_i\neq s$;
- метки всех вершин, кроме s, считаются временными, метка s - постоянной;
- вершина s объявляется текущей (c:=s);
- все дуги считаются непомеченными.

Основная часть:

1. для всех вершин u_j, инцидентных текущей вершине c, метки которых являются временными, пересчитываем эти метки по формуле:
   $d(u_j):=\min\{d(u_j), d(c)+Weight(c,u_j)\}$ (*)
   где (c,u_j) - дуга, соединяющая вершины c и u_j, а $Weight(c,u_j)$ - ее вес;
2. если метки всех вершин являются постоянными либо равны ∞, то путь не существует; ВЫХОД("нет решения");
3. иначе находим среди вершин с временными метками (среди всех таких вершин, а не только тех, чьи метки изменились в результате последнего выполнения шага (1)!) вершину x с минимальной меткой, объявляем ее метку постоянной, помечаем дугу (c',x), такую, что d(x) = d(c')+Weight(c',x), где c'=c либо c' - вершина, бывшая текущей на одном из предыдущих шагов (c'=c, если на шаге (1) при u_j=x реализовалась вторая часть формулы (*)), и делаем эту вершину текущей (c:=x);
4. если c=t, то найден путь длины d(t), который можно восстановить следующим образом: это тот путь между s и t, который состоит только из помеченных на шаге (3) дуг (можно доказать, что он существует и единственен); ВЫХОД("решение найдено");
5. иначе переходим на шаг (1).