5. Деревья

Опр. Неориентированное дерево - связный неограф без циклов.
Опр. Произвольный неограф без циклов называется лесом.

Свойства деревьев:

$G(X,U)$ - неориентированные дерево
$|X| = n, |U| = m$

1. $m = n - 1$
2. Если $x_i, x_j \in X$, то их соединяет единственная простая цепь
   - существование цепи следует из связности
   - единственность из отсутствия циклов
3. Если $x_i, x_j \in X$ не смежны, то введение в дерево ребра $(x_i x_j)$ даёт граф, содержащий ровно один цикл
4. Всякое неориентированное дерево содержит по крайней мере 2 концевые вершины
5. **Теорема Кайли.** Число различных деревьев, которые можно построить на n вершинах равно 2^{n-2}

Опр. Орграф $G(X,U)$ называют ориентированным деревом(ордеревом, корневым деревом), если выполняются следующие условия:

- существует ровно одна вершина (корень), не имеющая предшествующих ей: $p^+(x_1) = 0$
- для всех остальных вершин $p^+(x_i) = 1, \forall i \neq 1$

Опр. Висячие вершины дерева называются листьями.
Опр. Путь из корня в лист называется ветвью.
Опр. Длина наибольшей ветви называется высотой дерева.
Опр. Расстояние (число рёбер) от корня до вершины называется уровнем этой вершины.
Опр. Все вершины одного уровня называются ярусом.
Если из ордерева удалить корень, то оно распадётся на k деревьев $\{T_1, T_2, \ldots, T_k\}$. На этом множестве деревьев можно задать отношение порядка. Если рекурсивно из этих поддеревьев снова удалить корни задать порядок и продолжить этот процесс, пока все поддеревья не станут вершинами, то получим упорядоченное множество всех вершин дерева. Это позволяет использовать деревья для описания иерархий объектов.
Примеры:

1. Разбор математических выражений
2. Файловая система
3. Описание сложных программных систем

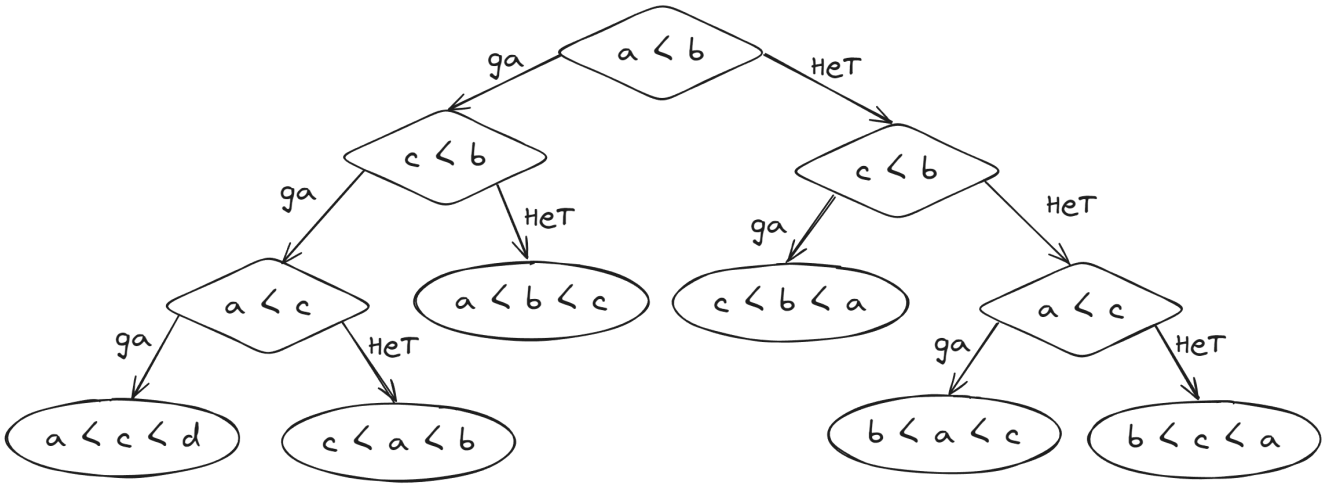
Опр. Если полустепень исхода каждой отличной от листа вершины равна 2 и все листья дерева располагаются в одном ярусе, то дерево называется полным бинарным деревом.
Используя индукцию по высоте, можно доказать, что число листьев полного бинарного дерева высоты h равно 2^h:

1. $h = 0, n = 2^0 = 1$
2. $h = 1, n = 2^1 = 2$
3. $h = k, n = 2^k$

4. Чтобы получить дерево с высотой $h = k + 1$, каждому листу дерева высоты $h = k$ добавить 2 листа, тогда на $k + 1$ уровне получится $2 \cdot 2^k = 2^{k+1}$ листьев

Теорема. Произвольное бинарное дерево с n листьями имеет высоту не меньше $\log_2 n$

$\{a_1, a_3, \ldots, a_n\}$ - последовательность из n элементов. Нужно ввести на ней отношение порядка - это задача сортировки.
В общем случае придётся рассмотреть $n!$ перестановок. Все сравнения приведут к построению бинарного дерева (дерева решений).



Дерево решений имеет $n!$ вершин, тогда сортировка даёт пути от корня к каждому листу. Число операций пропорционально высоте дерева - $\log_2(n!)$
Приближение для $n!$ $\approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$
Тогда получим минимально возможную сложность для алгоритма классической сортировки $O(n \log n)$

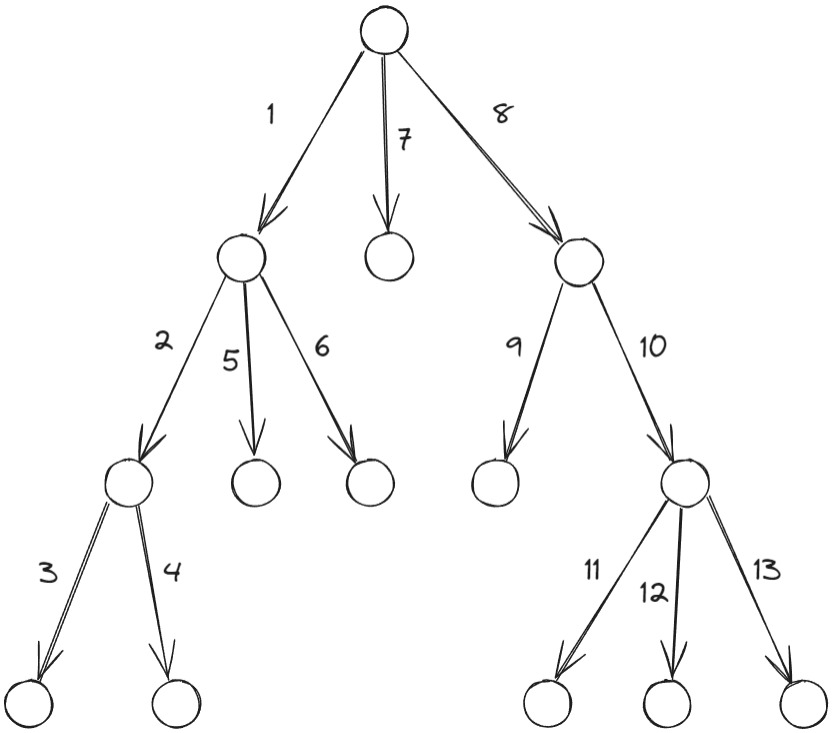
В общем случае ордерево - это связный, но не сильно связный орграф. Компонентами связности ордерева являются поддеревья на множестве вершин, образующие путь из корня в некоторый лист. - ???

Поиск в глубину

i - номер яруса
k - номер ребра
0) $i = 0, k = 0$

1. Выбираем вершину, смежную текущей из яруса $i = i + 1$ по непомеченному ребру, новую вершину объявляем текущей, пройденное ребро помечаем числом $k = k + 1$
2. Повторяем пункт 1 для текущей вершины
3. Если текущая вершина является листом, то возвращаемся в смежную ей вершину ярусом выше $i = i - 1$ и переходим к пункту 1
4. Если для текущей вершины нет ни одного инцидентного ей непомеченного ребра, возвращаемся в смежную ей вершину ярусом выше $i = i - 1$
5. Алгоритм останавливается, когда помечены все рёбра и $i = 0$

Пример:



Поиск в глубину - обход дерева по ярусам

- Оба алгоритма имеют одинаковую вычислительную сложность при обходе всех вершин
- Если требуется найти конкретную вершину/ребро, то:
  - Поиск в глубину применяется для "широких" деревьев
  - Поиск в ширину применяется для "узких" деревьев

Аналогичные подходы можно применять также для неориентированных деревьев