Лекция 22 (7)

Мосты, точки сочленения.

Классификация ребер DFS

~ Остовные деревья ~ ?

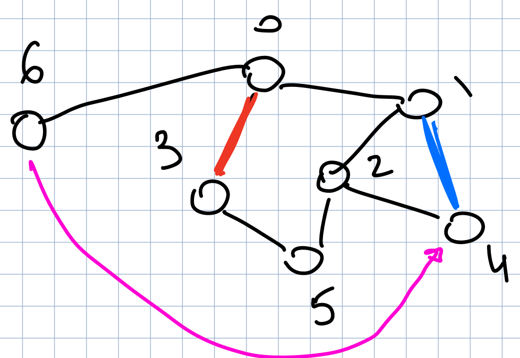
(I) DFS - обход в глубину.

Белая вершина - DFS не пришел

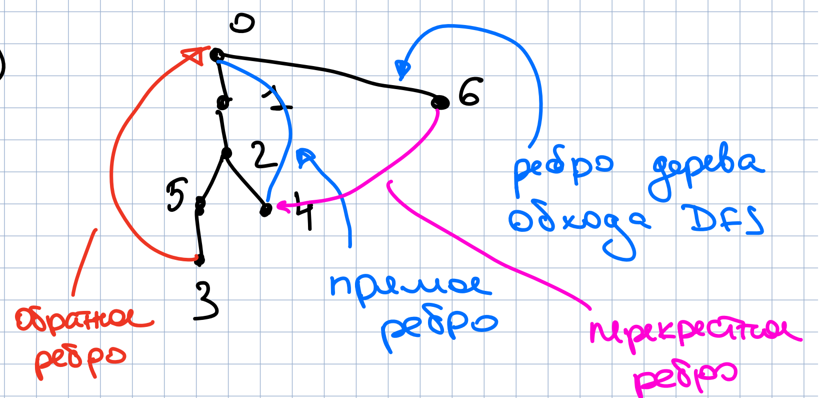
Серая вершина - DFS внутри

Черная вершина - DFS вышел.

<u>Опр</u> Дерево обхода DFS состоит из всех вершин и ребер, по кот-м были совершены переходы DFS.



⇒ DFS(0)



Опр Ребро (v, u) наз-ся обратным, если u является предком вершины v. (в дереве обхода)

Опр Ребро (v, u) наз-ся прямым, если u является потомком вер-ны v. (в дереве обхода)

Опр Ребро (v, u) наз-ся перекрестным, если верш. v и верш. u „лежат в соседних ветках дерева обхода"

Не обратное, не прямое, не ребра обхода

Критерии кл-ции

Обратное ребро → приходим в серую верш.

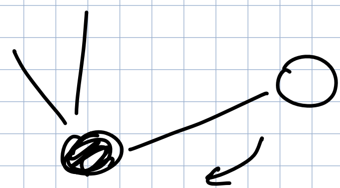
Прямые ребра → приходящие в белые, но не входящие в дерево обх.

Перекрестные → приходят в черную.

Th (Классификация для неор. графа)
В неориентированном графе нет. перекрестных ребер.

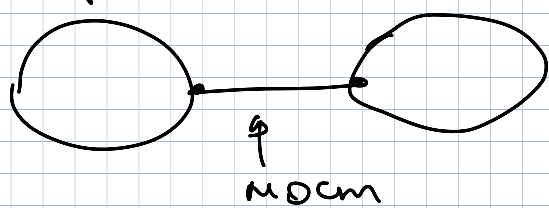
Д-во Перекрестное ребро (v, u), если запускаем

DFS от вершины v, а вершина u - чёрная.
Но если u - чёрная, значит DFS там был, а значит был бы совершен переход по ребру (u, v), а верш u была бы серая.
противоречие ⊠

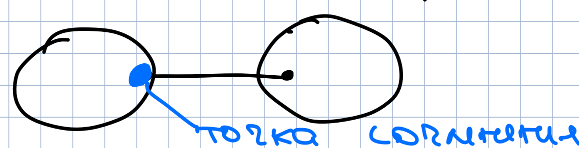


(II)

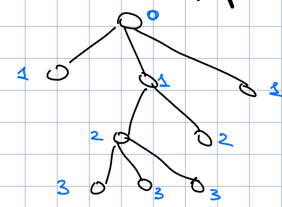
Опр Мостом в связном неорграфе наз-ся ребро (u, v) при удалении кот-го неориентированный граф становится несвязным.



Опр Точкой сочленения в связном неорграфе наз-ся вершина, при удалении кот-й граф становится несвязным.



III. **Опр** Глубиной вершины в дерево обхода DFS наз-ся кол-во ребер, которое необх. пройти от корня до вершины.



IV. тривиальный алгоритм поиска моста

- → Рассматриваем ребро (удаляем) $O(M)$
- → проверяем на связность $O(N+M)$ → DFS

$O(M(N+M)) \sim O(M^2)$

V. Кто может быть мостом?

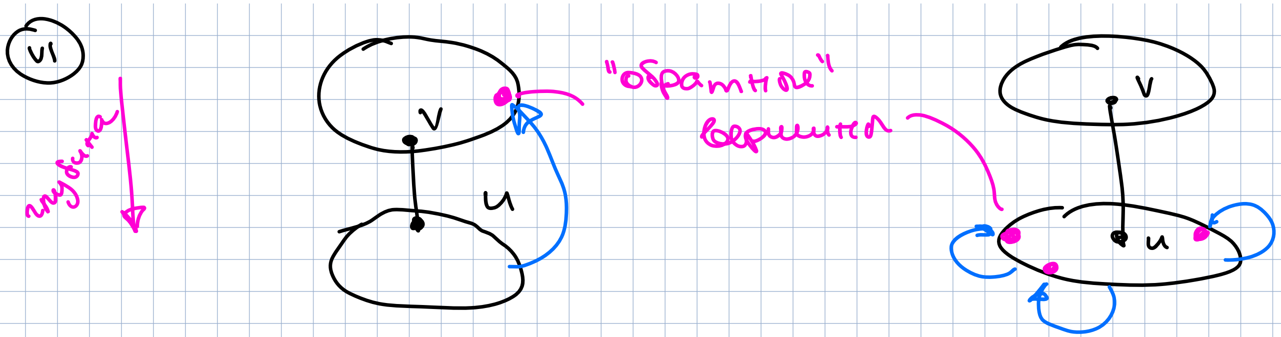
1. Перекрестные? их нет в неорграфе
2. Обратные? образуют цикл $\Rightarrow$ их удаление ничего не изменит.
3. Прямые и обычные ребра могут быть мостами

↑ буду наз-ть их прямыми.

Внимание! Приходит в белые верш.

Можно улучшить трив. алгоритм до $O(\underbrace{N}_{\text{по ребрам дерева обхода}}(\underbrace{N+M}_{\text{DFS связность}}))$

VI



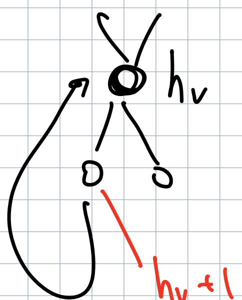
Если есть такое ребро (обратное), то (v,u) не мост.

Если все обратные ребра не

Глубина обратных вершин не меньше глубина u.

Обозначим: h_v — глубина вершины v

d_v — „циклическая" глубина верш. v
„обратная"



Есть ребро $v \to u$

$$d_v = \min \begin{cases} h_v \\ d_u, & v \to u \text{ прямое} \\ h_u, & v \to u \text{ обратное} \end{cases}$$

Если $h_v < d_u \Rightarrow v-u$ мост

VII Алгоритм:

корень дерева

DFS (G, v, parent = -1)

used[v] = TRUE

если ↳ parent = -1 ⇒ d[v] = h[v] = 0

иначе ↳ d[v] = h[v] = h[parent] + 1

ch = G.GetChildren(v)

for i = 0.. ch.size():

u = ch[i] (v → u) – ребро

if used[u] = true // ребро обратное

↳ d[v] = min(d[v], h[u])

else // прямое ребро

↳ DFS(G, u, v) parent

d[v] = min(d[v], d[u])

if h[v] < d[u]

↳ нашли мост v→u !!!

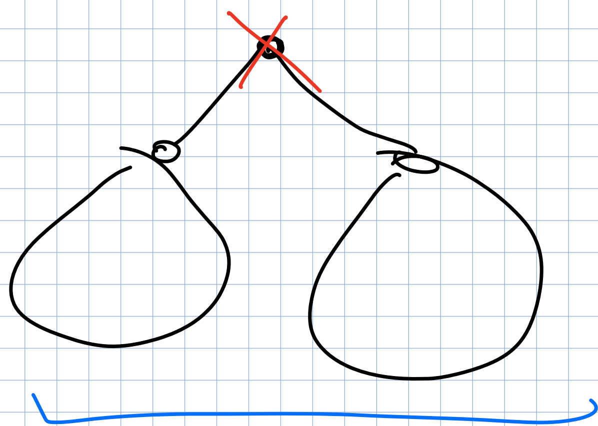
Асимптотика: O(N+m)

VIII Алгоритм точек сочленения

v - точка сочленения

- v - корень дерева обхода
  - ⇓ степень корня 1
    - да: не точка сочленения
    - нет: точка сочленения
- не корень
  - ⇓ часть моста

if h[v] ≤ d[u] and parent != -1
↳ v - точка сочленения.



отдельный случай.

IX Применение: