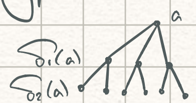
Немного воспоминаний:

BFS:
- верш. активна $\Leftrightarrow$ вся её окр-ть исследована
- $O(n+m)$
- очередь
- применение:
  - выявл-е комп. связн-ти
  - кратч. расст-е
  - каркас

1959г.

Мур: – обход графа: исследуем окр-ть a



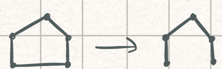
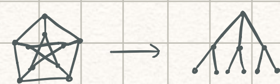
$G=(V,E)$, $|V|=n$, k-регулярн. (*) $n \leq 1+k+k(k-1)+\ldots+k(k-1)^{d-1}$, где $d=dim(G)$

Q (Мура): Существуют ли графы, на ктр достигается оценка (*)

A (80е): 1) Графа Мура с $d=3$ $(d\geq 3)$ не существует

2) Граф Мура с $d=2$ существует и должен иметь $k=2,3,5$ или 57

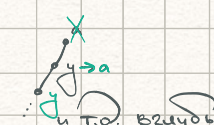
Опр Граф, на ктр достигается (*) наз. **графом Мура**

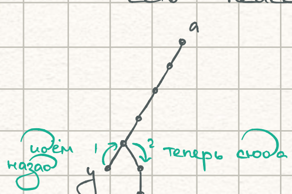
Ex Графы Мура:
- $k=2$ C_5: 
- $k=3$ Граф Петерсона: 
- $k=5$ граф Хоффмана-Синглтона (рисовать сложно)
- $k=57$ в 2020 показали, что НЕ сущ. такой граф Мура

6. Поиск в глубину (DFS)

Стратегия: идём вперёд (глубже) как можно дальше

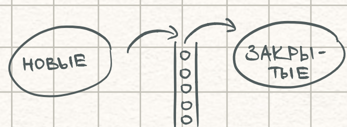
Тактика:
- у активной вершины иссл-ся только одно ребро
- активной становится последняя посещ. вершина, у ктр есть неисследованные рёбра





$d(a,y) \leq ecc(a)$

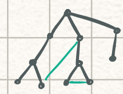
Структура данных:



Трудоёмкость: $O(m+n)$

Опр DFS-**дерево** — каркас, получ. поиском в глубину в связн. графе. Стартовая вершина a – его корень.

Опр Все рёбра BFS/DFS-дерева наз. **прямыми**.
Все остальные рёбра G наз. **обратными**.



Тут — пр.
— обр.

Опр Обратное ребро **продольное**, если одна из его вершин явл-ся предком другой,
иначе оно **поперечное**

Алгоритм DFS:

```
Procedure DFS(a)
 посетить вершину a
 a => S
while S ≠ ∅ do
  x = top(S)
  if имеется неисследов-е ребро (x,y)
   then исследовать ребро (x,y)
    if вершина y новая
      then посетить вершину y
        y => S
  else удалить x из S
```

рекурсивный:

```
For x∈V do  Dnum(x) := 0
  c := 0
For x∈V do
    if Dnum(x) = 0  then DFSR(x)
Procedure DSFR(x)
  c := c + 1
  Dnum(x) := c
For y∈V do
  if Dnum(y) = 0 then DSFR(y)
```

с построением каркаса:

```
пометить все вершины как новые
For a∈V do
  if a новая then DFST(a)
Procedure DFST(a)
 F(a) := a
 открыть вершину a
 x := a
 while x открытая do
  if имеется неисслед-е (x,y)
   then иссл-ть (x,y)
     if y новая
      then F(y) := x
           открыть y
           x := y
   else закрыть x
        x := F(x)
```

Теорема
G - связн., T - его DFS-дерево => любое обр. ребро отн-но этого T будет продольным.

Д-во: a - стартовая. Посл-ть вершин в стеке образует путь с началом в a, а все его рёбра ∈ T, т.к. каждый раз добавляя новую y в стек, к дереву добавляется ребро (x, y), где x наход. в стеке под y, а удаление вершин ничего не рушит.
Пусть (x, y) - обратное и x оказалась в стеке раньше y.
Когда y помещается в стек, вершина a там ещё есть. y новая и ∈ окр-ти x.
=> x лежит на пути м/у a и y ∈ T. => x - предок y в T => (x, y) - продольн.

Дальше идет презентация Иры ("10_16_pres"). Формулировки оттуда:

Лемма
Старт. вершина a явл-ся шарниром графа <=> в его DFS-дереве T deg(a) > 1

Опр Если в дереве имеется ориентированный путь из x в y, то x - **предок** для y,
а y - **потомок** для x.

Каждая вершина в дереве явл-ся и предком, и потомком самой себя.
Предок/потомок, отличный от самой вершины, наз. **собственным** предком/потомком

Опр Если в дереве есть ориентир. ребро (x, y), то y наз. **сыном** x,
а x - **отцом** y.

Теорема
Пусть T - DFS-дерево графа G с корнем a.
Вершина x ≠ a явл-ся шарниром G <=> у x ∃ такой сын y в T, что ни один потомок y не соединён ребром ни с одним собств. предком x.