Семинар 5

Графы. Способы задания графов. Поиск в ширину. Нахождение компонент связности в неориентированном графе.

Граф G(V, E)

1) Рассмотрим неориентированный граф с n вершинами и без параллельных ребер. Будем считать, что граф является связным, то есть существует как одно целое. Какое минимальное и максимальное число ребер соответственно может иметь такой граф?

а) $n-1$ и $n(n-1)/2$

б) $n-1$ и n^2

в) n и 2^n

г) n и n^n

Способы представления графа

Способы задания:

1) Матрица смежности $A = (a_{ij})$

$$a_{ij} = \begin{cases} 1, \text{если есть ребро } (v_i, v_j) \\ 0, \text{в противном случае} \end{cases}$$

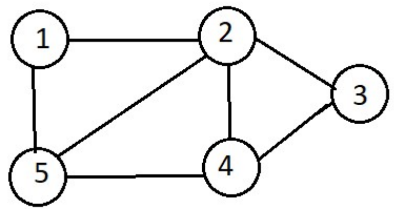
хранение (пространственная сложность) ?

2) Списки смежности: любой вершине v соответствет список смежных с ней вершин.?

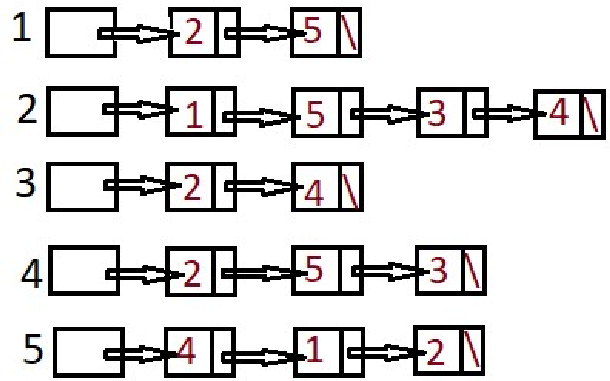
(1)

	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

Пример:

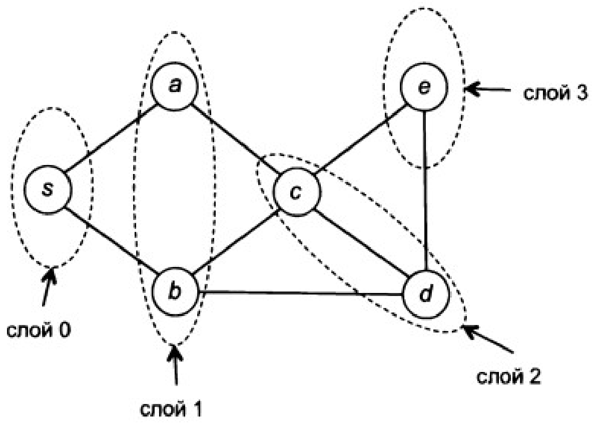


2)



Какова сумма длин всех списков (неор/ор)?
Какой объем памяти необходим для хранения списков?
Сколько операций потребуется, чтобы найти число вершин, смежных с данной (неор/ор)?

Поиск в ширину



Поиск в ширину обнаруживает вершины слоями.
Вершины i-го слоя являются соседями вершин *(i-1)*-го слоя, которые не появлялись ни в одном более раннем слое.

ТЕСТОВОЕ ЗАДАНИЕ 8.1

Рассмотрим неориентированный граф с $n \geq 2$ вершинами. Каково соответственно минимальное и максимальное число разных слоев, которые этот граф мог бы иметь?

а) 1 и $n - 1$

б) 2 и $n - 1$

в) 1 и n

г) 2 и n

Алгоритм поиска в ширину и примеры

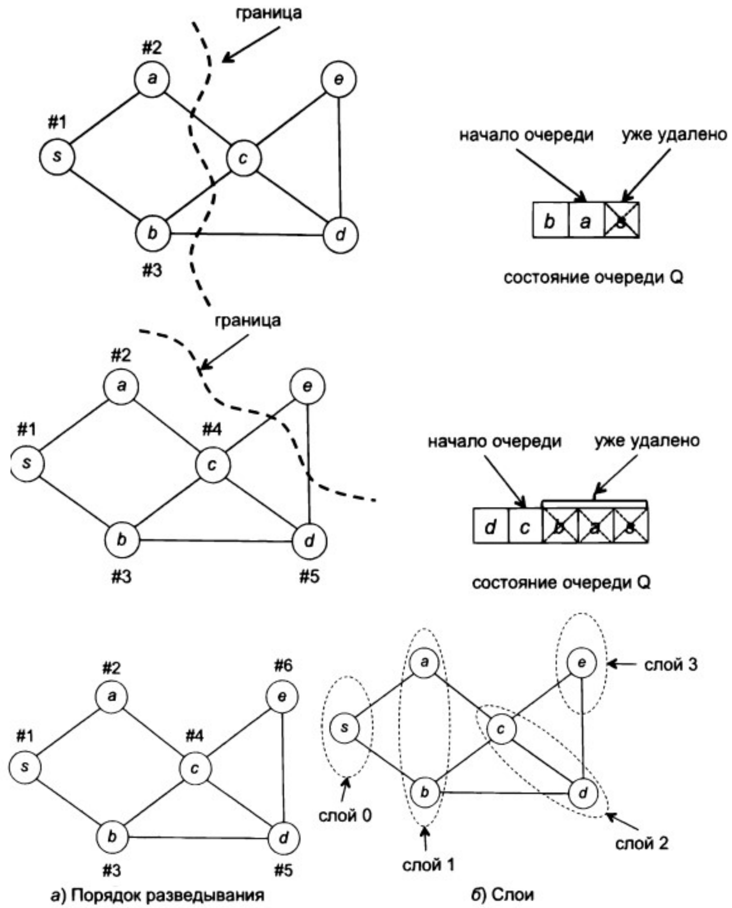
BFS

Вход: граф $G = (V, E)$, представленный в виде списков смежности, и вершина $s \in V$.

Постусловие: вершина достижима из s тогда и только тогда, когда она помечена как «разведанная».

---

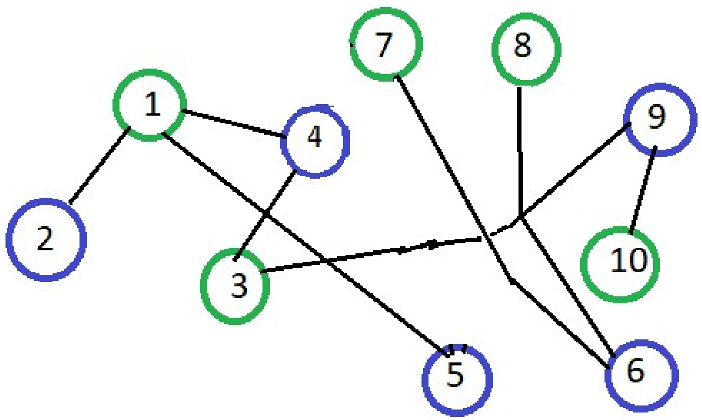
1) пометить s как разведанную вершину, все остальные как неразведанные
2) Q := очередь, инициализированная вершиной s
3) **while** Q не является пустой **do**
4) удалить вершину из начала Q, назвать ее v
5) **for** каждое ребро (v, w) в списке смежности v **do**
6) **if** w не разведана **then**
7) пометить w как разведанную
8) добавить w в конец Q



Время выполнения?
Каждая вершина добавляется в очередь ровно1 раз, каждое ребро обрабатывается ровно 1 раз.

Задача: проверить, является ли граф двудольным.

Произведём серию поисков в ширину. Т.е. будем запускать поиск в ширину из каждой неразведанной вершины. Ту вершину, из которой мы начинаем идти, мы помещаем в первую долю. В процессе поиска в ширину, если мы идём в какую-то новую вершину, то мы помещаем её в долю, отличную от доли текущей вершину. Если же мы пытаемся пройти по ребру в вершину, которая уже посещена, то мы проверяем, чтобы эта вершина и текущая вершина находились в разных долях. В противном случае граф двудольным не является. По окончании работы алгоритма мы либо обнаружим, что граф не двудолен, либо найдём разбиение вершин графа на две доли.



Определение длины кратчайшего пути *dist(s, v)* от *s* до любой достижимой вершины

BFS

Вход: граф $G = (V, E)$, представленный в виде списков смежности, и вершина $s \in V$.

Постусловие: вершина достижима из s тогда и только тогда, когда она помечена как «разведанная».

---

1) пометить s как разведанную вершину, все остальные как неразведанные
2) Q := очередь, инициализированная вершиной s
3) **while** Q не является пустой **do**
4) удалить вершину из начала Q, назвать ее v
5) **for** каждое ребро (v, w) в списке смежности v **do**
6) **if** w не разведана **then**
7) пометить w как разведанную
8) добавить w в конец Q

ДОПОЛНЕННЫЙ ПОИСК В ШИРИНУ (AUGMENTED-BFS)

Вход: граф $G = (V, E)$, представленный в виде списков смежности, и вершина $s \in V$.

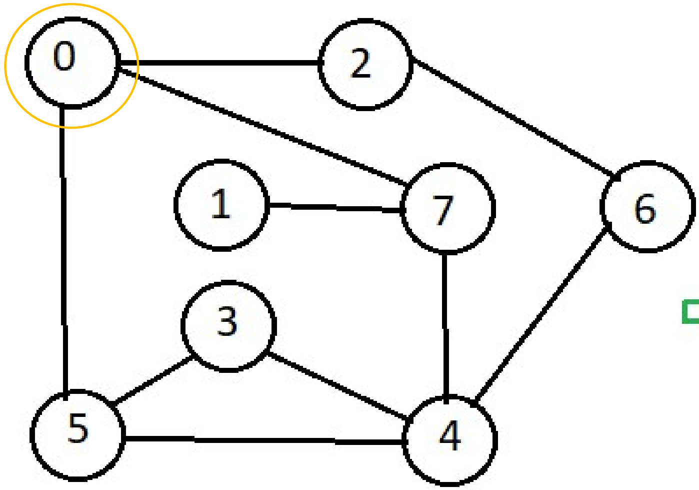
Постусловие: для каждой вершины $v \in V$ значение $l(v)$ равно истинному расстоянию кратчайшего пути $dist(s, v)$.

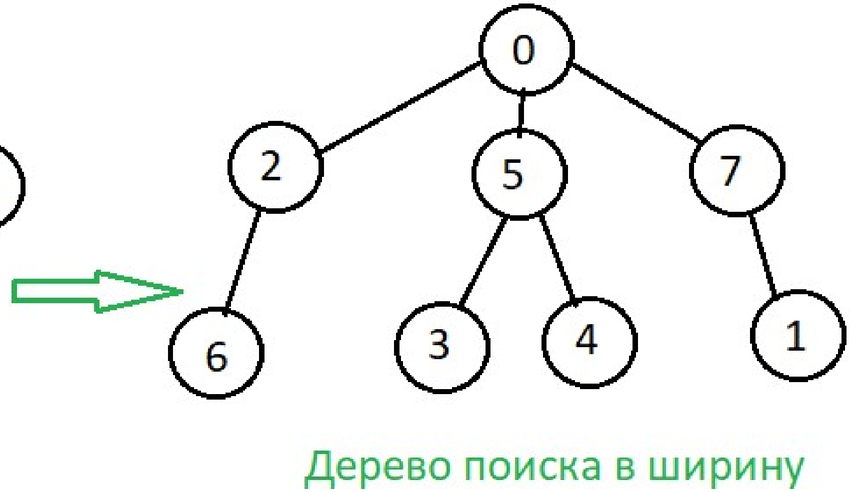
---

1) пометить s как разведанную вершину, все остальные как неразведанные
2) $l(s) := 0$, $l(v) := +\infty$ для каждой $v \neq s$
3) Q := очередь, инициализированная вершиной s
4) **while** Q не является пустой **do**
5) удалить вершину из начала Q, назвать ее v
6) **for** каждое ребро (v, w) в списке смежности вершины v **do**
7) **if** w не разведана **then**
8) пометить w как разведанную
9) $l(w) := l(v) + 1$
10) добавить w в конец Q

Итог : $\forall\ dist(s, v) - min$ расстояние $(dist(s, v) = i,\ v\ - \text{в}\ i - \text{ом слое})$

Кратчайший путь (по количеству пройденных ребер)





В кратчайшем пути до каждой вершины количество предшествующих равно количеству предков в дереве.

Еще задачи с применением поиска в ширину

- Нахождение кратчайшего пути в **0-1-графе** (т.е. графе взвешенном, но с весами равными только 0 либо 1)

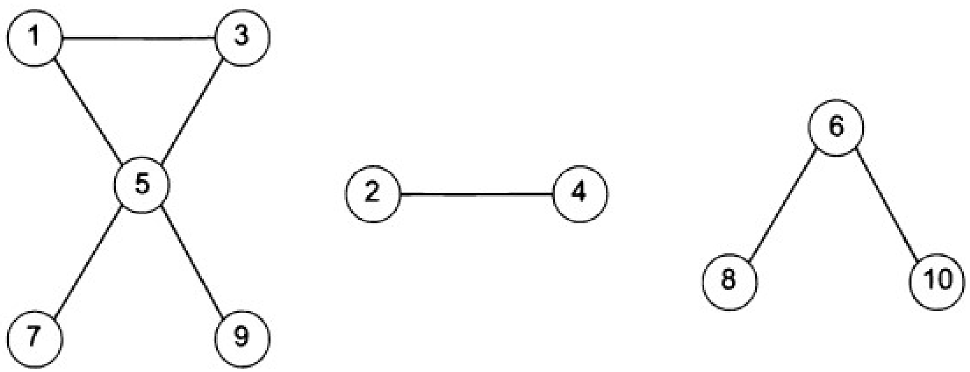
- Нахождение **кратчайшего цикла** в ориентированном невзвешенном графе.

- Найти все ребра, лежащие на каком-либо пути между a и b.

- Нахождение связных компонент.

Нахождение компонент связности в неориентированном графе (CC - Connectivity Component)

Компонента связности – это максимальное подмножество $S \subseteq V$ вершин, так что существует путь из любой вершины из S в любую другую вершину из S.



Граф из 10 вершин. 3 компоненты связности.

Нахождение компонент связности в неориентированном графе (продолжение)

Рассмотрим неориентированный граф с n вершинами и m ребрами. Какое соответственно минимальное и максимальное число связных компонент этот граф может иметь?

а) 1 и $n - 1$

б) 1 и n

в) 1 и $\max\{m, n\}$

г) 2 и $\max\{m, n\}$

BFS

Вход: граф $G = (V, E)$, представленный в виде списков смежности, и вершина $s \in V$.

Постусловие: вершина достижима из s тогда и только тогда, когда она помечена как «разведанная».

---

1) пометить s как разведанную вершину, все остальные как неразведанные
2) Q := очередь, инициализированная вершиной s
3) **while** Q не является пустой **do**
4) удалить вершину из начала Q, назвать ее v
5) **for** каждое ребро (v, w) в списке смежности v **do**
6) **if** w не разведана **then**
7) пометить w как разведанную
8) добавить w в конец Q

UCC

Вход: неориентированный граф $G = (V, E)$, представленный в виде списков смежности, где $V = \{1,2,3, ..., n\}$.

Постусловие: для каждой $u, v \in V$, $cc(u) = cc(v)$ тогда и только тогда, когда u, v находятся в одной и той же связной компоненте.

пометить все вершины как неразведанные

$numCC := 0$

for i := от 1 до n **do** // перебрать все вершины

if i не разведана **then** // избежать избыточности

$numCC := numCC + 1$ // новая компонента

Пометить i как разведанную

// вызвать алгоритм BFS, начиная с i (строки 2-8)

Q := очередь, инициализированная значением i

while Q не является пустой **do**

удалить вершину из начала Q, назвать ее v

$\boxed{cc(v) := numCC}$

for каждая (v, w) в списке смежности вершины v **do**

if w не разведана **then**

пометить w как разведанную,

добавить w в конец Q

Временная сложность?

Задача поиска «узких мест» в графе

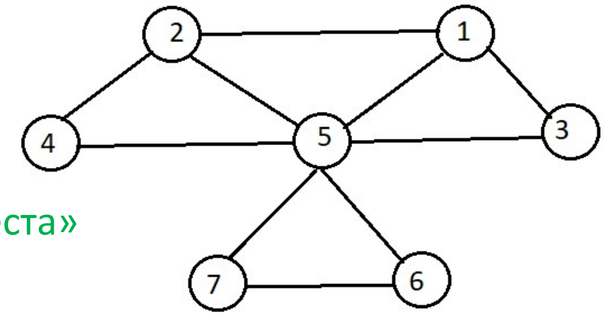
$v \in V$ называется <u>точкой сочленения</u>, если граф с вершинами $V - v$ имеет больше компонент связности, чем G.

Ребро e называется <u>мостом,</u> если его удаление увеличивает число компонент связности.

Точки сочленения и мосты – «узкие места» графа.

Назвать узкие места в дереве.

Пример:



Указать «узкие места»

Как находить «узкие места» с помощью выделения компонент связности?