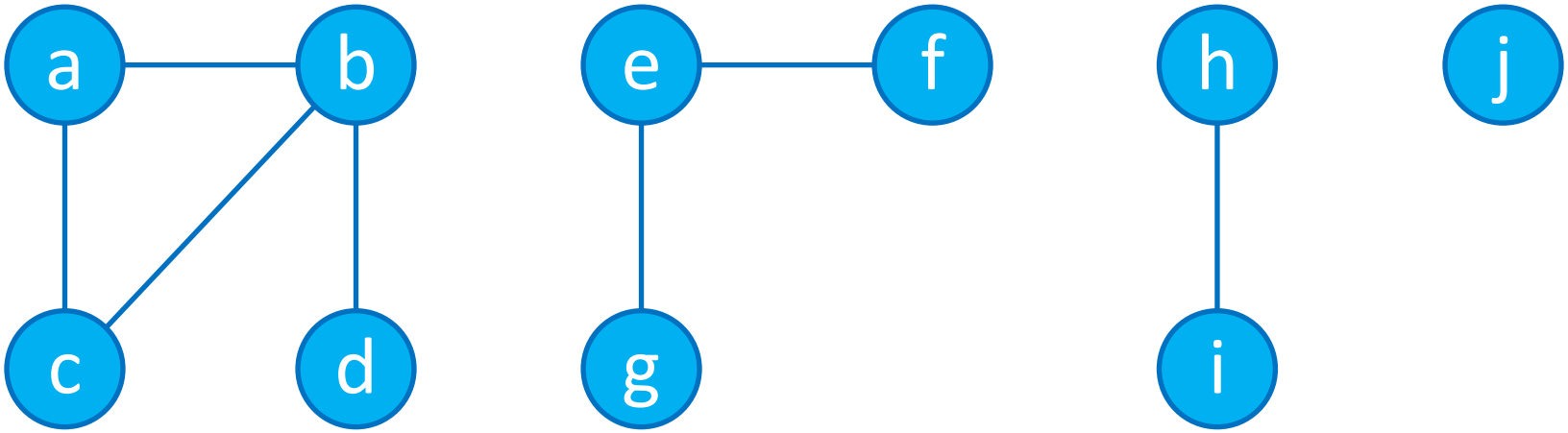
Структуры данных для непересекающихся множеств

Практикум 3 курс
Осень 2015

Поиск связанных компонент

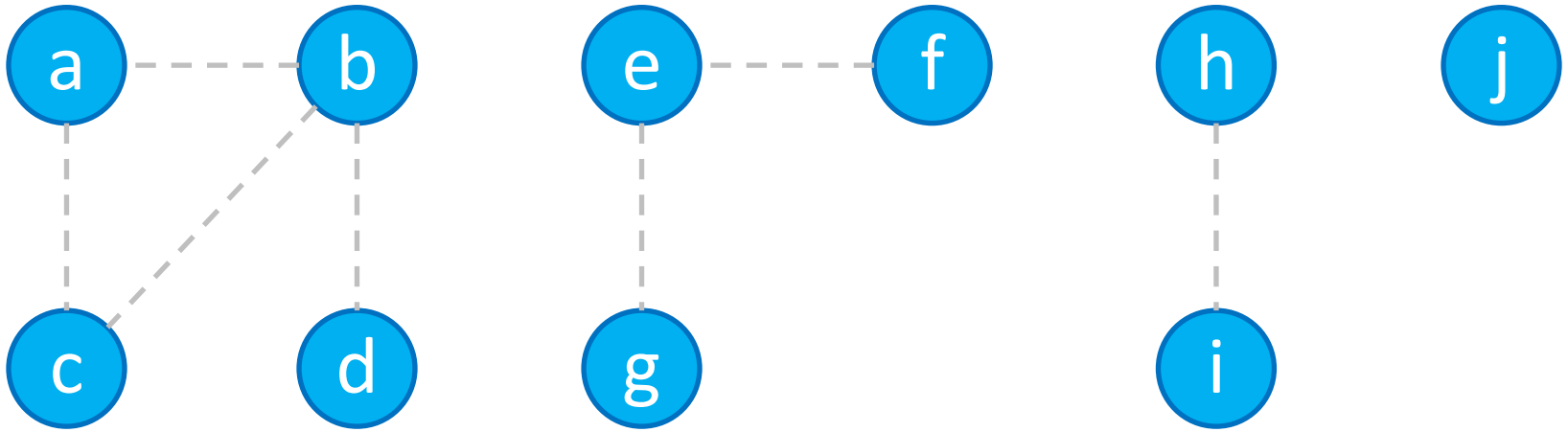


$$\{a, b, c, d\}, \{e, f, g\}, \{h, i\}, \{j\}$$

Поиск связанных компонент

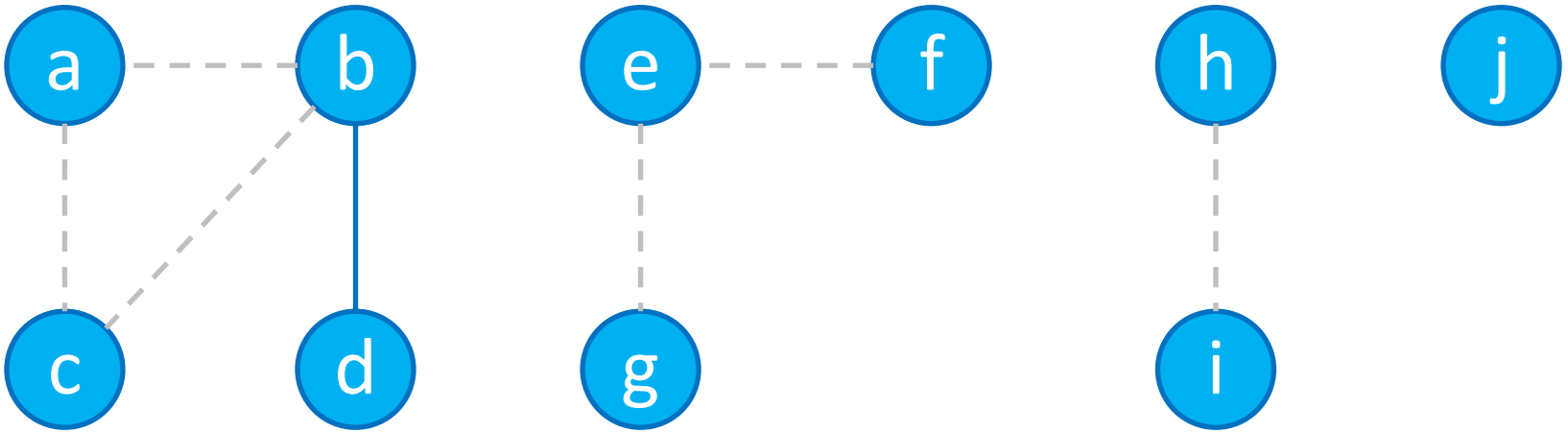
```
Connected_Components(G)
        foreach v∈V[G]
                Make_Set(v)
        foreach (u,v) ∈E[G]
                if Find_Set(u)≠Find_Set(v) then
                        Union(u,v)

Same_Component(u, v)
        if Find_Set(u) = Find_Set(v) then
                return True
        else
                return False
```

Поиск связанных компонент



Старт $\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}, \{h\}, \{i\}, \{j\}$

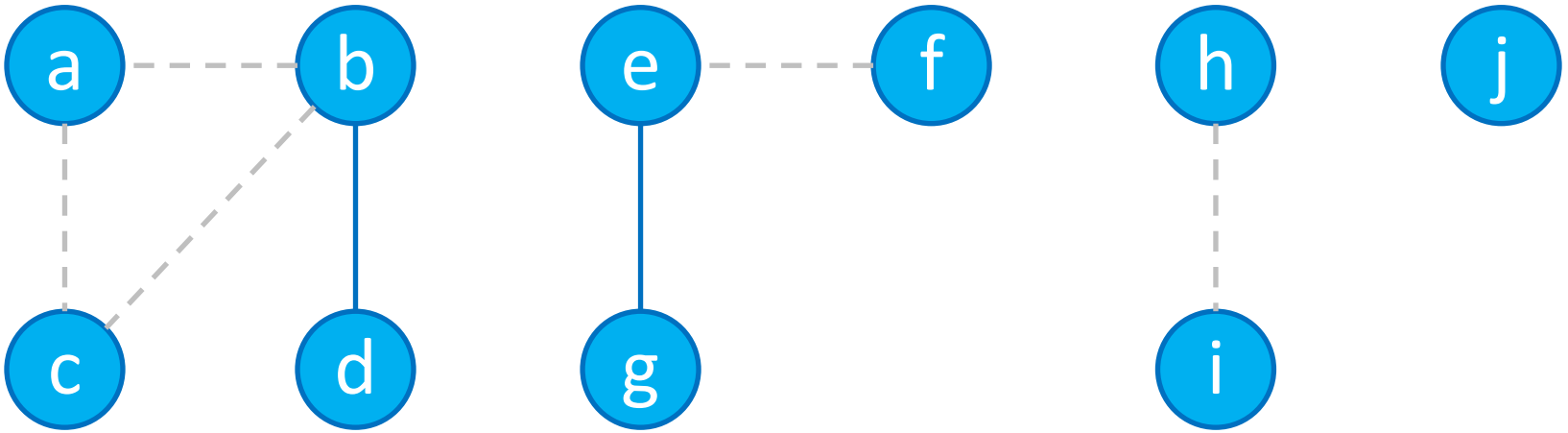
Поиск связанных компонент



Старт $\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}, \{h\}, \{i\}, \{j\}$

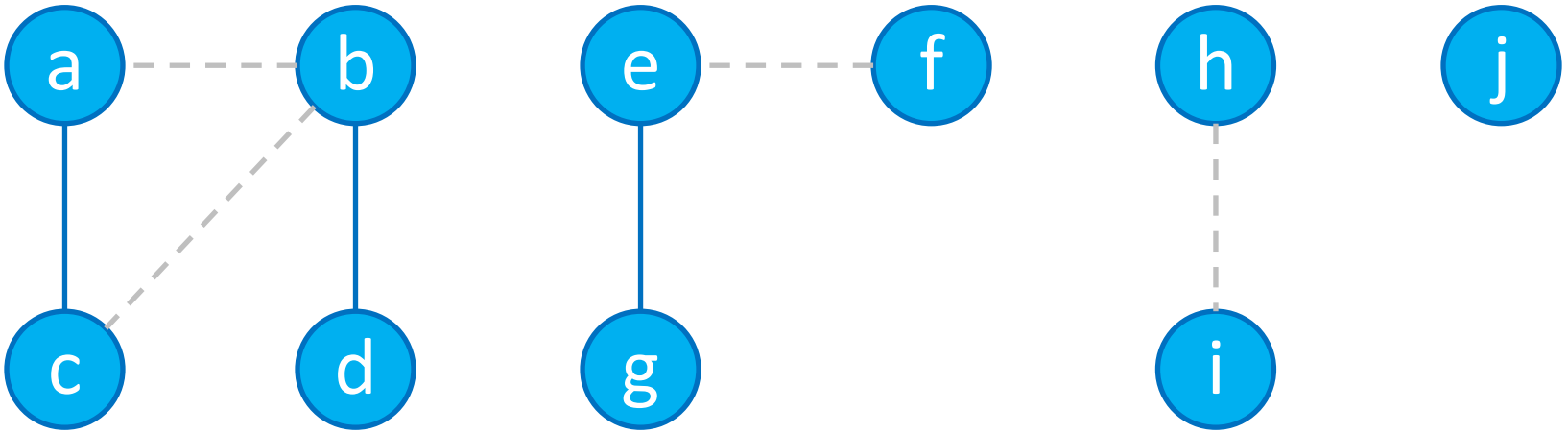
(b,d) $\{\text{a}\}, \{\text{b, d}\}, \{\text{c}\}, \{e\}, \{f\}, \{g\}, \{h\}, \{i\}, \{j\}$

Поиск связанных компонент



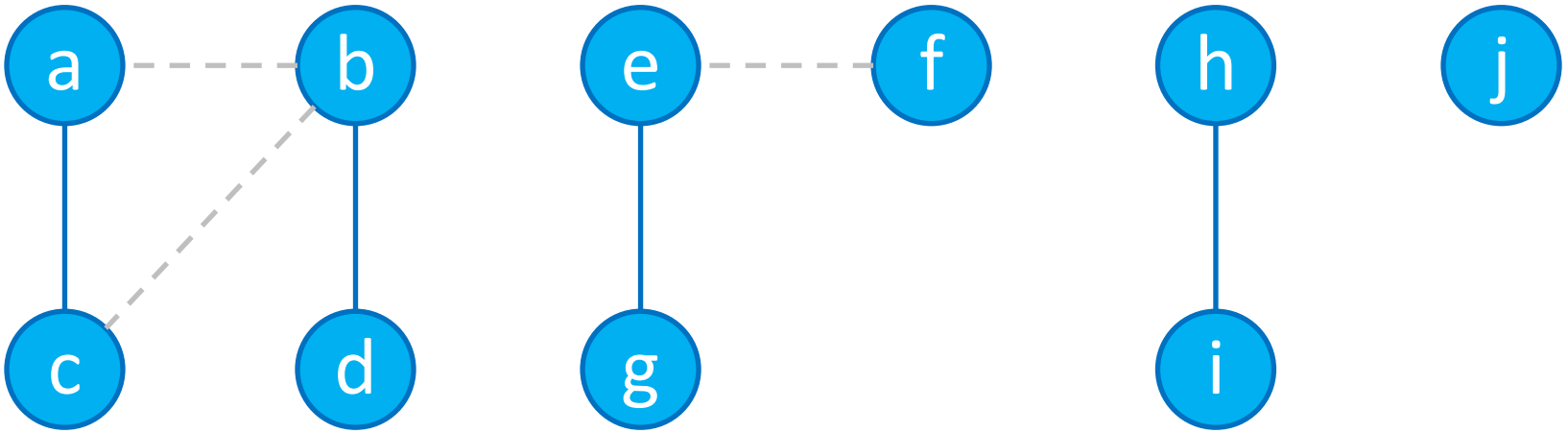
Старт	$\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}, \{h\}, \{i\}, \{j\}$
(b,d)	$\{\mathrm{a}\}, \{\mathrm{b}, \mathrm{d}\}, \{\mathrm{c}\}, \{e\}, \{f\}, \{g\}, \{h\}, \{i\}, \{j\}$
(e,g)	$\{\mathrm{a}\}, \{\mathrm{b}, \mathrm{d}\}, \{\mathrm{c}\}, \{e, g\}, \{f\}, \{h\}, \{i\}, \{j\}$

Поиск связанных компонент



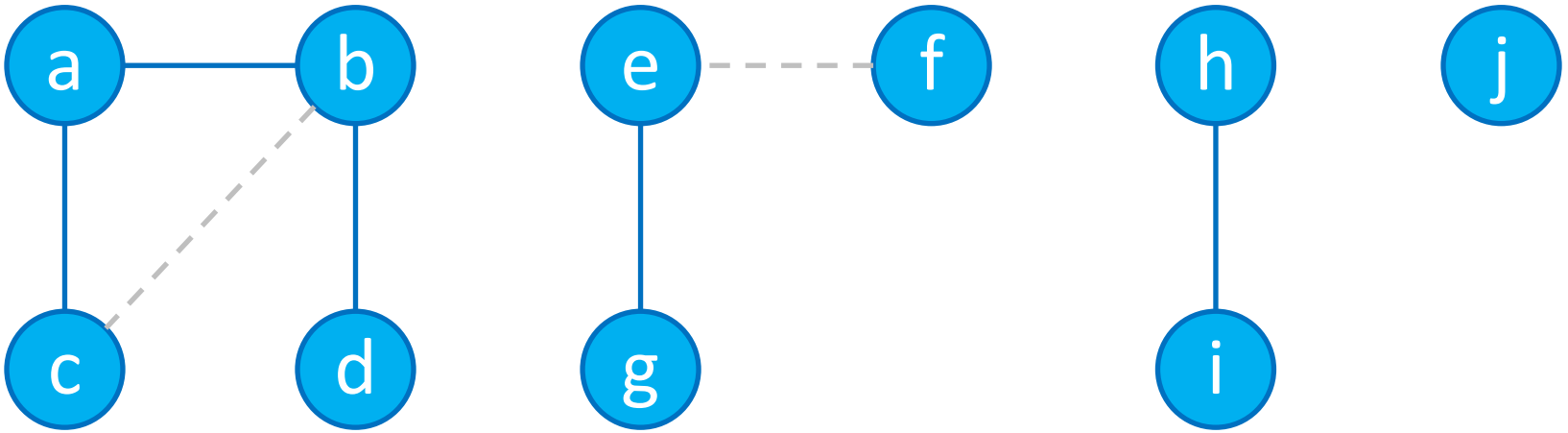
Старт	$\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}, \{h\}, \{i\}, \{j\}$
(b,d)	$\{\mathrm{a}\}, \{\mathrm{b}, \mathrm{d}\}, \{\mathrm{c}\}, \{e\}, \{f\}, \{g\}, \{h\}, \{i\}, \{j\}$
(e,g)	$\{\mathrm{a}\}, \{\mathrm{b}, \mathrm{d}\}, \{\mathrm{c}\}, \{e, g\}, \{f\}, \{h\}, \{i\}, \{j\}$
(a,c)	$\{\mathrm{a}, c\}, \{\mathrm{b}, \mathrm{d}\}, \{e, g\}, \{f\}, \{h\}, \{i\}, \{j\}$

Поиск связанных компонент



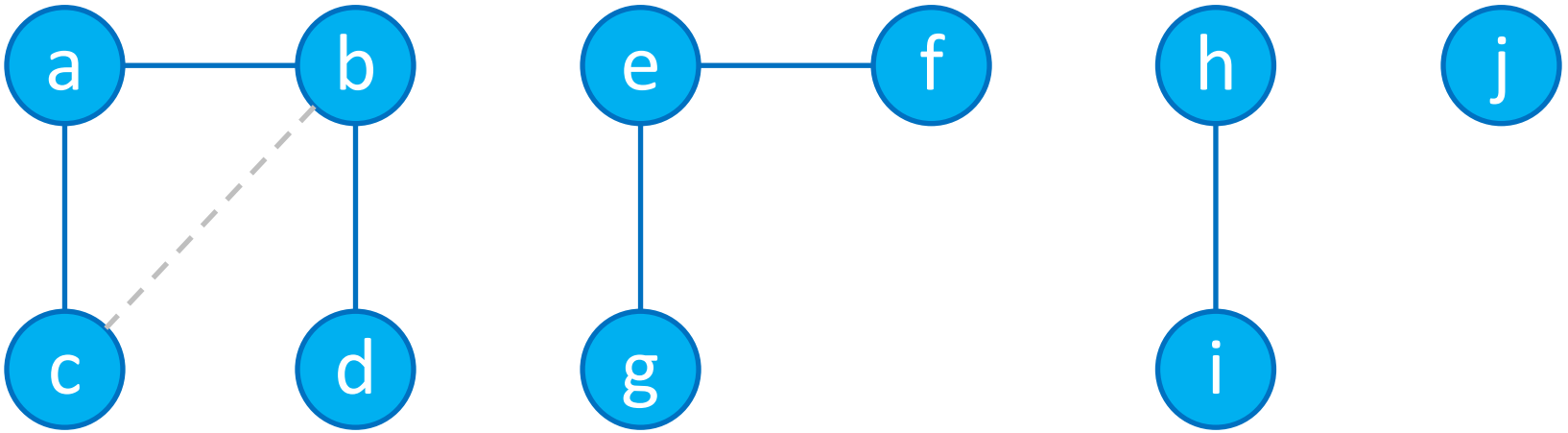
Старт	$\{a\},\{b\},\{c\},\{d\},\{e\},\{f\},\{g\},\{h\},\{i\},\{j\}$
(b,d)	$\{\mathrm{a}\},\{\mathrm{b},\mathrm{d}\},\{\mathrm{c}\},\{e\},\{f\},\{g\},\{h\},\{i\},\{j\}$
(e,g)	$\{\mathrm{a}\},\{\mathrm{b},\mathrm{d}\},\{\mathrm{c}\},\{e,g\},\{f\},\{h\},\{i\},\{j\}$
(a,c)	$\{\mathrm{a},c\},\{\mathrm{b},\mathrm{d}\},\{e,g\},\{f\},\{h\},\{i\},\{j\}$
(h,i)	$\{\mathrm{a},c\},\{\mathrm{b},\mathrm{d}\},\{e,g\},\{f\},\{h,i\},\{j\}$

Поиск связанных компонент



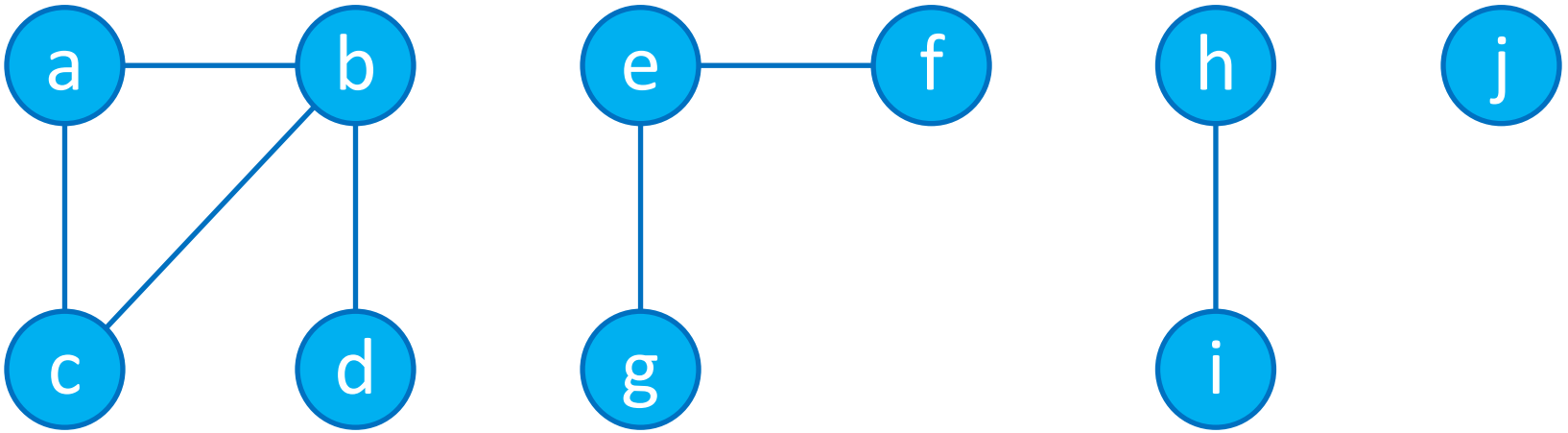
Старт	$\{a\},\{b\},\{c\},\{d\},\{e\},\{f\},\{g\},\{h\},\{i\},\{j\}$
(b,d)	$\{\mathrm{a}\},\{\mathrm{b},\mathrm{d}\},\{\mathrm{c}\},\{e\},\{f\},\{g\},\{h\},\{i\},\{j\}$
(e,g)	$\{\mathrm{a}\},\{\mathrm{b},\mathrm{d}\},\{\mathrm{c}\},\{e,g\},\{f\},\{h\},\{i\},\{j\}$
(a,c)	$\{\mathrm{a},c\},\{\mathrm{b},\mathrm{d}\},\{e,g\},\{f\},\{h\},\{i\},\{j\}$
(h,i)	$\{\mathrm{a},c\},\{b,d\},\{e,g\},\{f\},\{h,i\},\{j\}$
(a,b)	$\{a,b,c,d\},\{e,g\},\{f\},\{h,i\},\{j\}$

Поиск связанных компонент



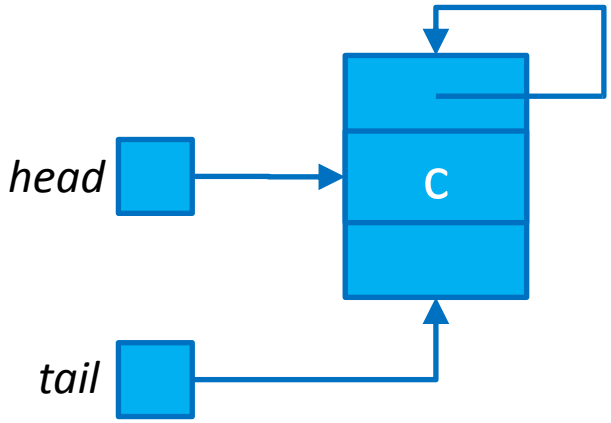
Старт	$\{a\},\{b\},\{c\},\{d\},\{e\},\{f\},\{g\},\{h\},\{i\},\{j\}$
(b,d)	$\{\mathrm{a}\},\{\mathrm{b},\mathrm{d}\},\{\mathrm{c}\},\{e\},\{f\},\{g\},\{h\},\{i\},\{j\}$
(e,g)	$\{\mathrm{a}\},\{\mathrm{b},\mathrm{d}\},\{\mathrm{c}\},\{e,g\},\{f\},\{h\},\{i\},\{j\}$
(a,c)	$\{\mathrm{a},c\},\{\mathrm{b},\mathrm{d}\},\{e,g\},\{f\},\{h\},\{i\},\{j\}$
(h,i)	$\{\mathrm{a},c\},\{b,d\},\{e,g\},\{f\},\{h,i\},\{j\}$
(a,b)	$\{a,b,c,d\},\{e,g\},\{f\},\{h,i\},\{j\}$
(e,f)	$\{a,b,c,d\},\{e,f,g\},\{h,i\},\{j\}$

Поиск связанных компонент



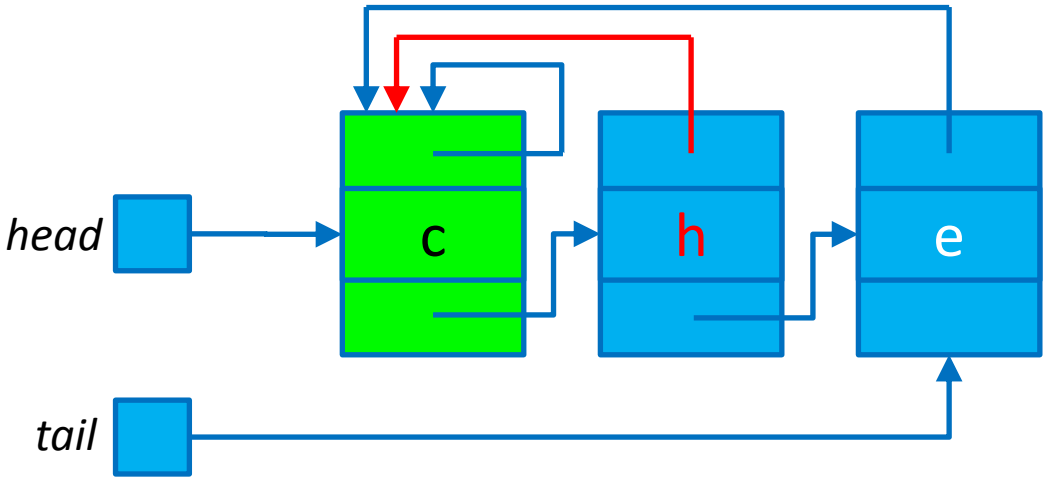
Старт	$\{a\},\{b\},\{c\},\{d\},\{e\},\{f\},\{g\},\{h\},\{i\},\{j\}$
(b,d)	$\{\mathrm{a}\},\{\mathrm{b},\mathrm{d}\},\{\mathrm{c}\},\{e\},\{f\},\{g\},\{h\},\{i\},\{j\}$
(e,g)	$\{\mathrm{a}\},\{\mathrm{b},\mathrm{d}\},\{\mathrm{c}\},\{e,g\},\{f\},\{h\},\{i\},\{j\}$
(a,c)	$\{\mathrm{a},c\},\{\mathrm{b},\mathrm{d}\},\{e,g\},\{f\},\{h\},\{i\},\{j\}$
(h,i)	$\{\mathrm{a},c\},\{b,d\},\{e,g\},\{f\},\{h,i\},\{j\}$
(a,b)	$\{a,b,c,d\},\{e,g\},\{f\},\{h,i\},\{j\}$
(e,f)	$\{a,b,c,d\},\{e,f,g\},\{h,i\},\{j\}$
(b,c)	$\{a,b,c,d\},\{e,f,g\},\{h,i\},\{j\}$

Реализация на основе связанных списков



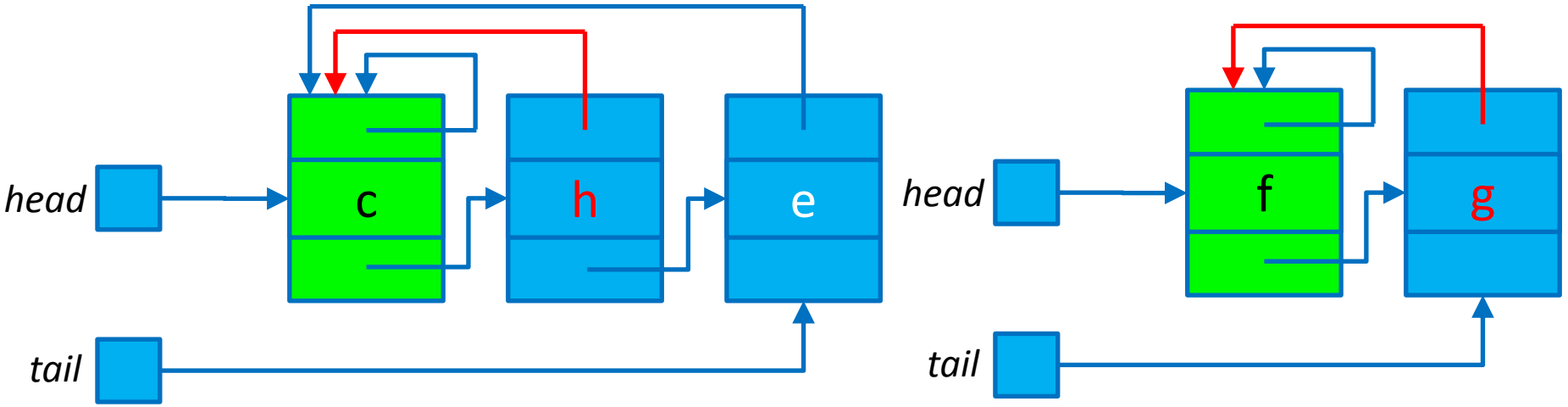

```
Make_Set(a)
Сложность = O(1)
```

Реализация на основе связанных списков



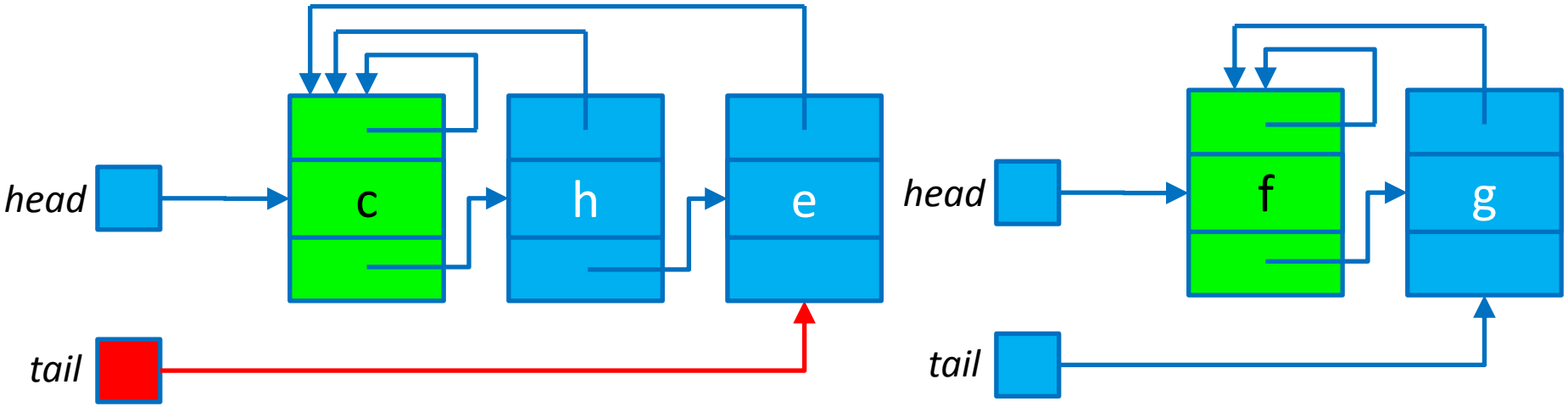

```
Find_Set(h)
Сложность = O(1)
```

Реализация на основе связанных списков



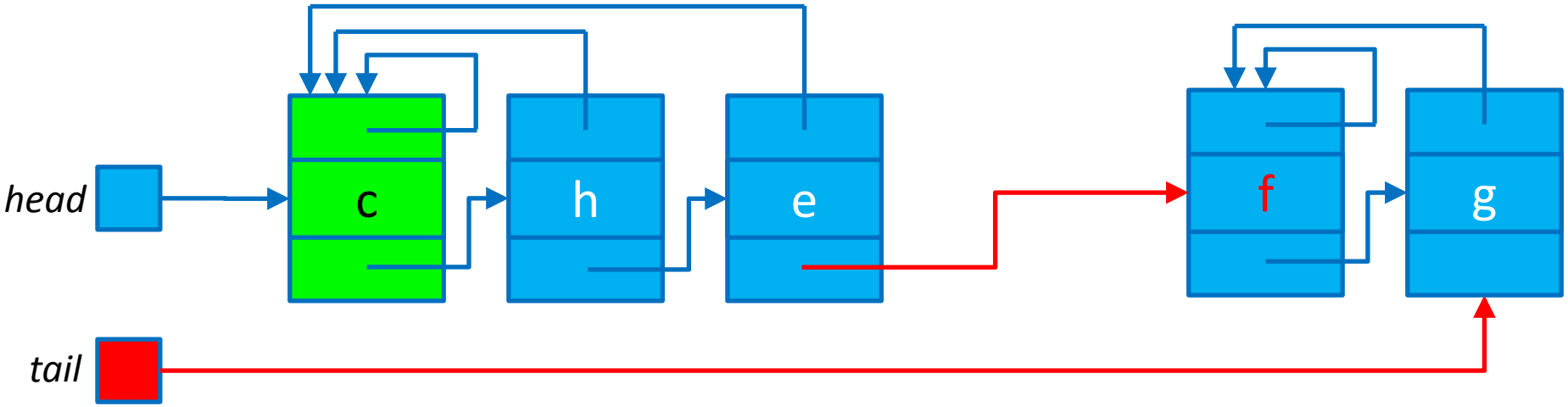
`Union(h,g)`

Реализация на основе связанных списков



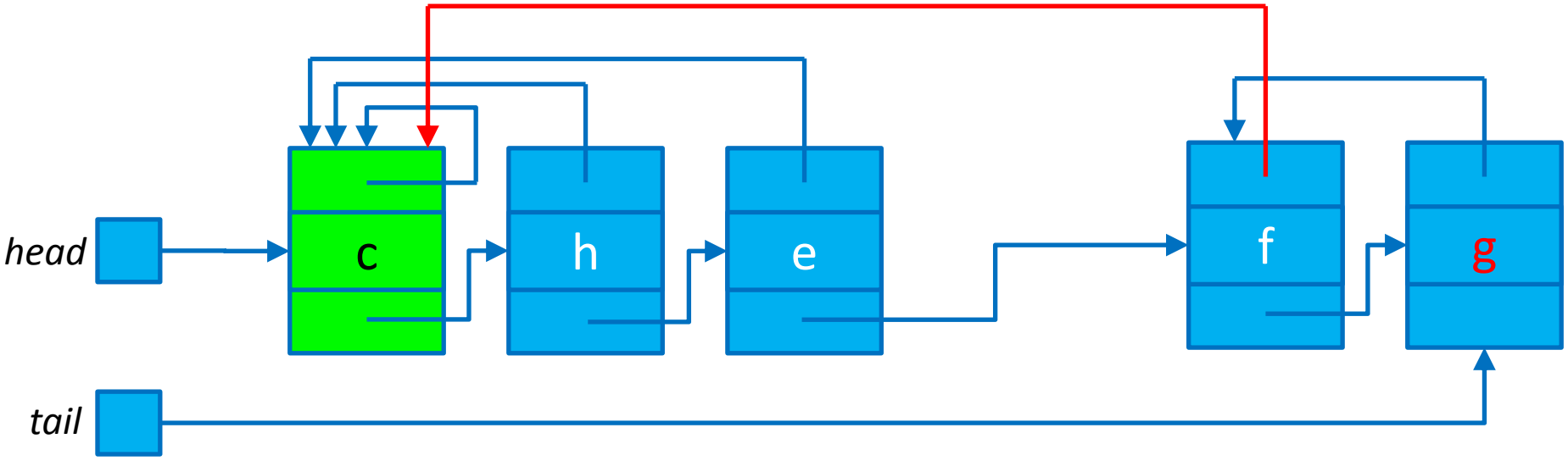
`Union(h,g)`

Реализация на основе связанных списков



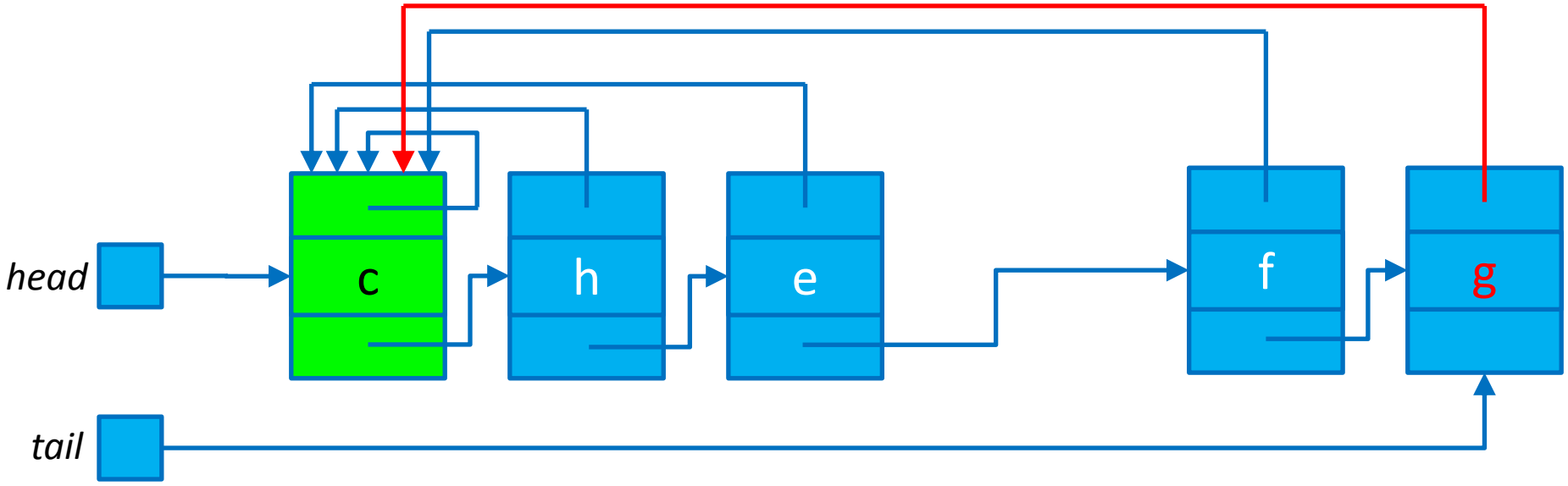
`Union(h,g)`

Реализация на основе связанных списков



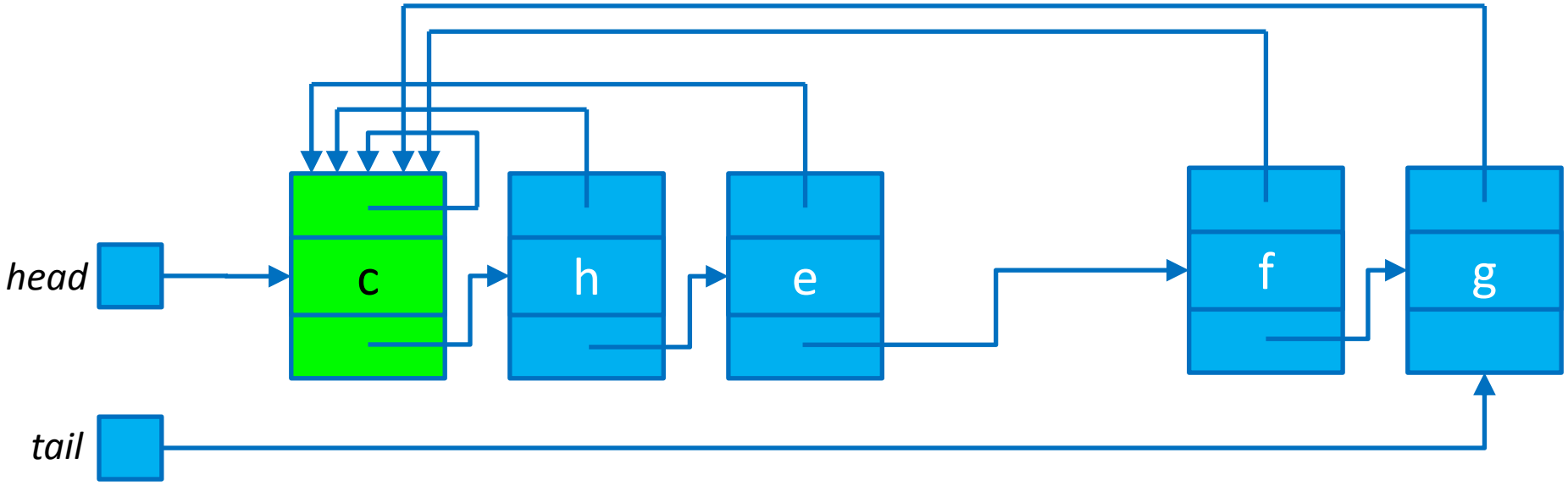
`Union(h,g)`

Реализация на основе связанных списков




```
Union(h,g)
```

Реализация на основе связанных списков

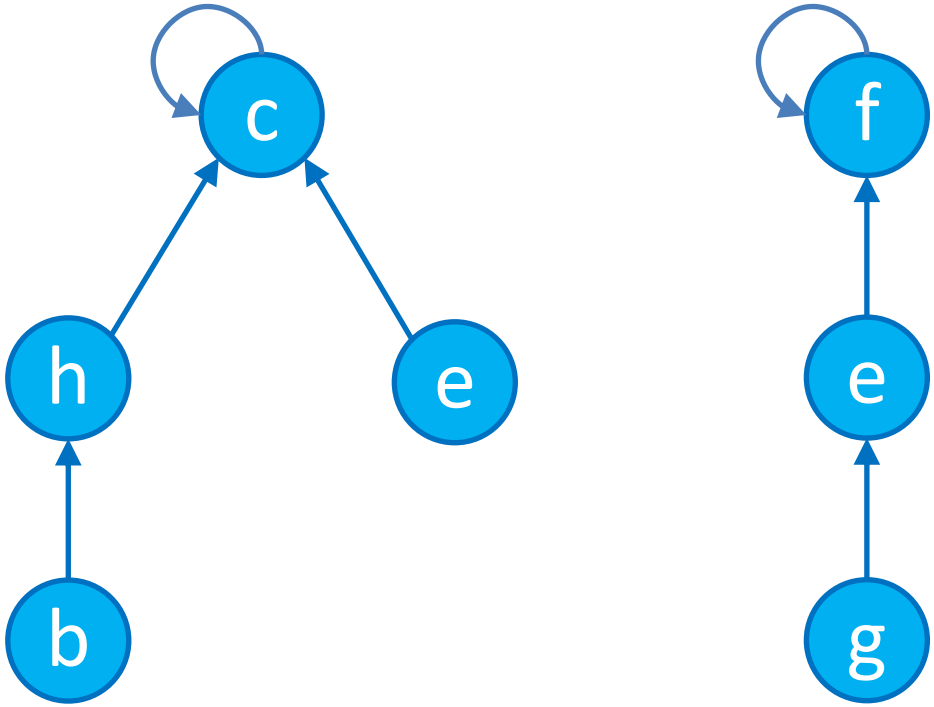



```
Union(h,g)
Сложность = O(n)
```

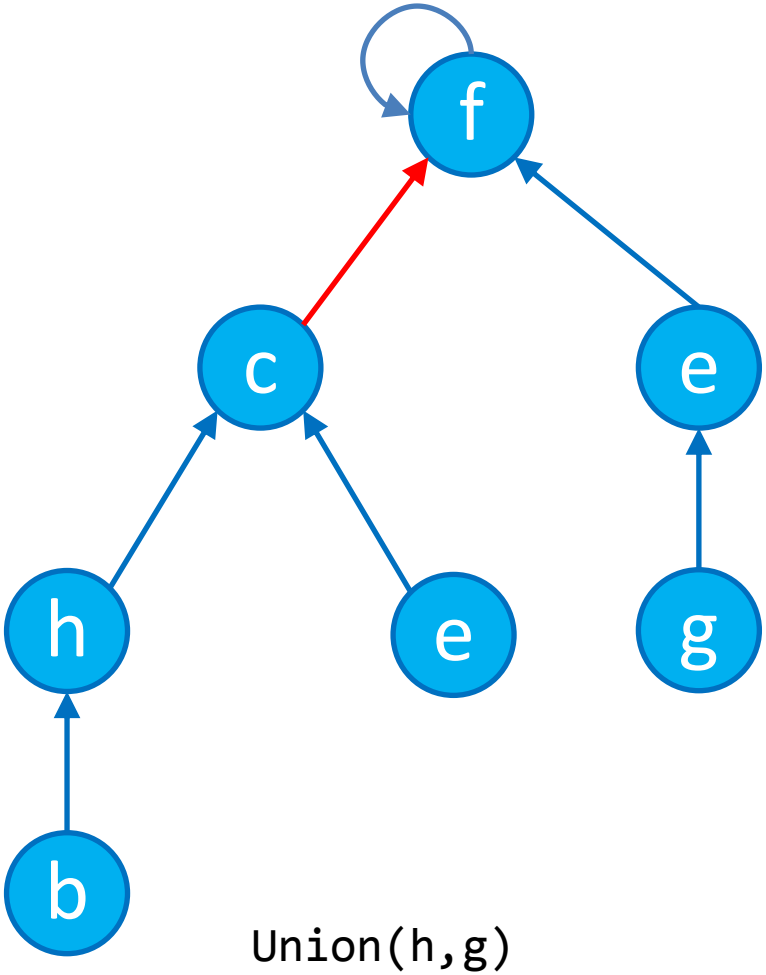
Весовая эвристика

- Для списков поддерживаем значение их длины
- Присоединяем более короткий список к более длинному
- Последовательность из m операций Make_Set, Union и Find_Set, n из которых составляют операции Make_Set требует для выполнения O(m+nlog n) времени

Лес непересекающихся множеств



Лес непересекающихся множеств



Union(h,g)

Объединение по рангу и сжатие пути

- Аналог весовой эвристики
- Идея: корень с меньшим количеством узлов должен указывать на корень с большим количеством узлов
- Ранг корня – верхняя граница высоты узла (длина максимального пути от листовых вершин до корня)
- Сжатие пути – перебрасывание ссылок на родителя при поиске канонического представителя

Объединение по рангу и сжатие пути

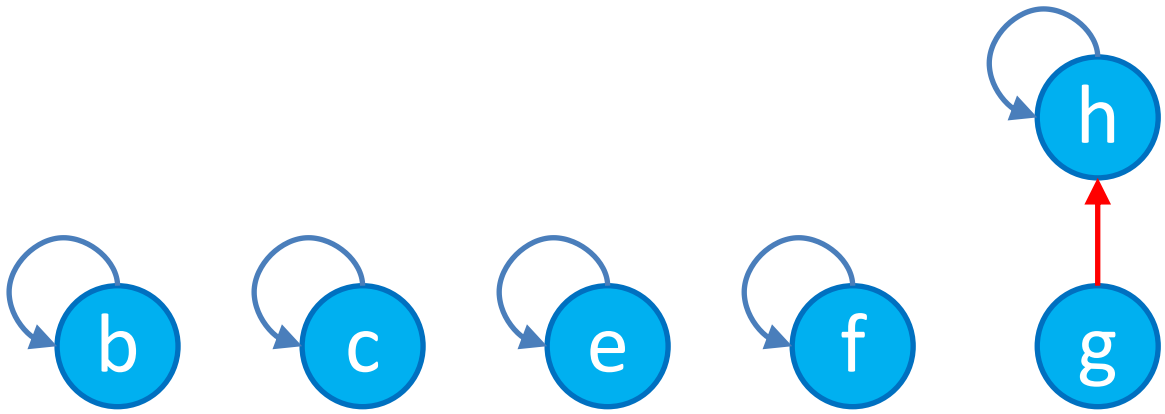
```
Make_Set(x)
        p[x] := x
        rank[x] := 0

Union(x, y)
        Link(Find_Set(x), Find_Set(y))

Link(x, y)
        if rank[x] > rank[y] then
                p[y] := x
        else
                p[x] := y
                if rank[x] = rank[y] then
                        rank[y]++

Find_Set(y)
        if x≠p[x] then
                p[x] := Find_Set(p[x])
        return p[x]
```

Объединение по рангу и сжатие пути

```
Make_Set(b)
Make_Set(c)
Make_Set(e)
Make_Set(f)
Make_Set(g)
Make_Set(h)
```

Объединение по рангу и сжатие пути

```
Make_Set(b)
Make_Set(c)
Make_Set(e)
Make_Set(f)
Make_Set(g)
Make_Set(h)
Union(g,h)
```

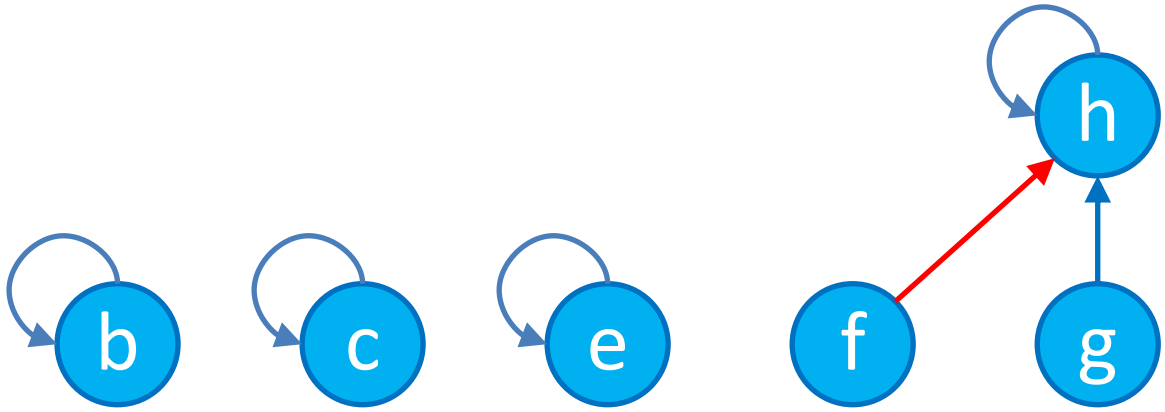


r = 1

r = 0

Объединение по рангу и сжатие пути

```
Make_Set(b)
Make_Set(c)
Make_Set(e)
Make_Set(f)
Make_Set(g)
Make_Set(h)
Union(g,h)
Union(f,g)
```

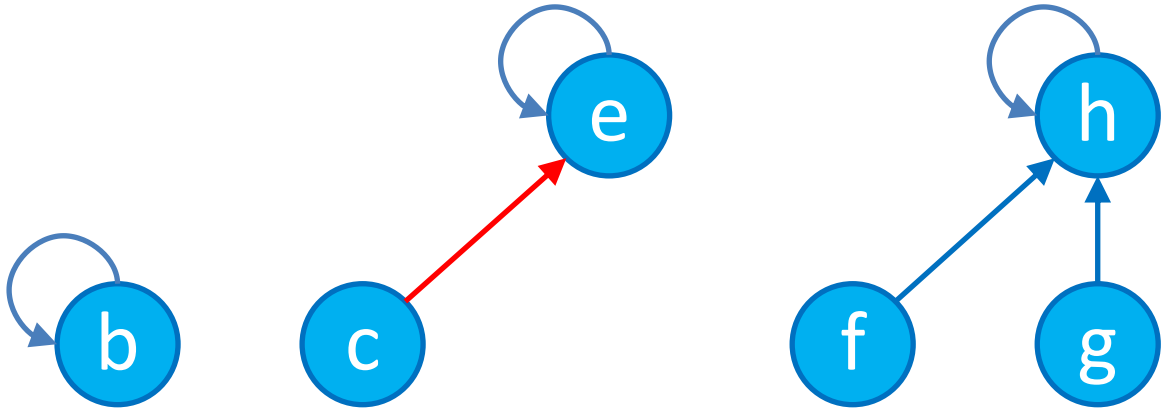


r = 1

r = 0

Объединение по рангу и сжатие пути

```
Make_Set(b)
Make_Set(c)
Make_Set(e)
Make_Set(f)
Make_Set(g)
Make_Set(h)
Union(g,h)
Union(f,g)
Union(c,e)
```

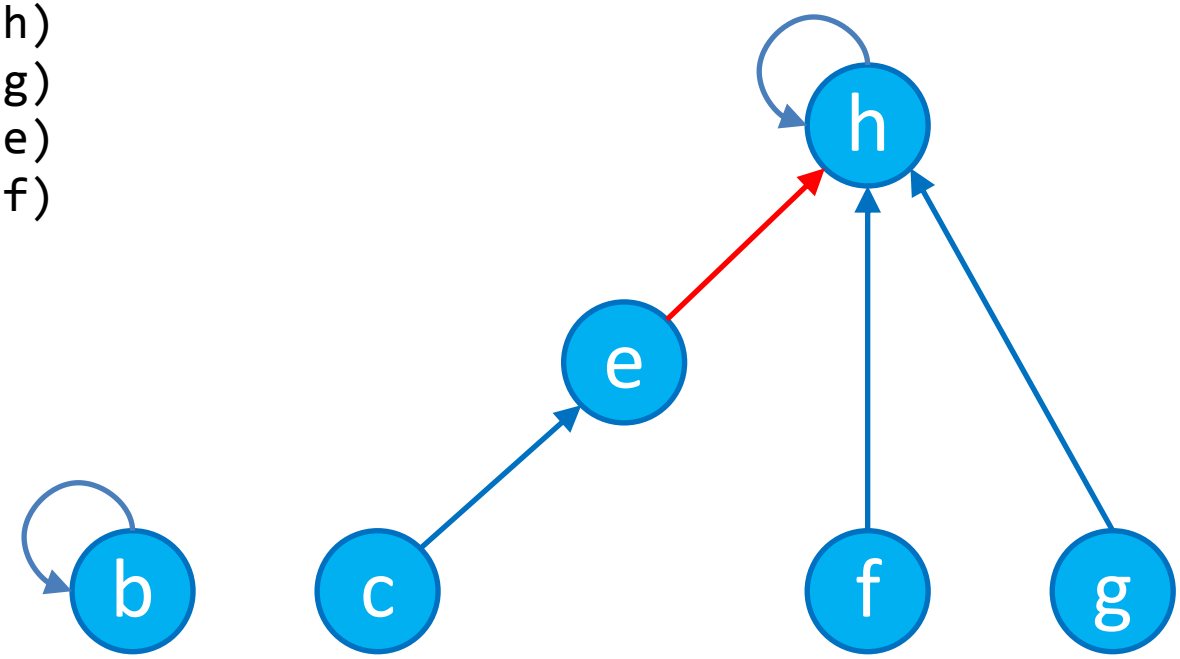


r = 1

r = 0

Объединение по рангу и сжатие пути

```
Make_Set(b)
Make_Set(c)
Make_Set(e)
Make_Set(f)
Make_Set(g)
Make_Set(h)
Union(g,h)
Union(f,g)
Union(c,e)
Union(c,f)
```



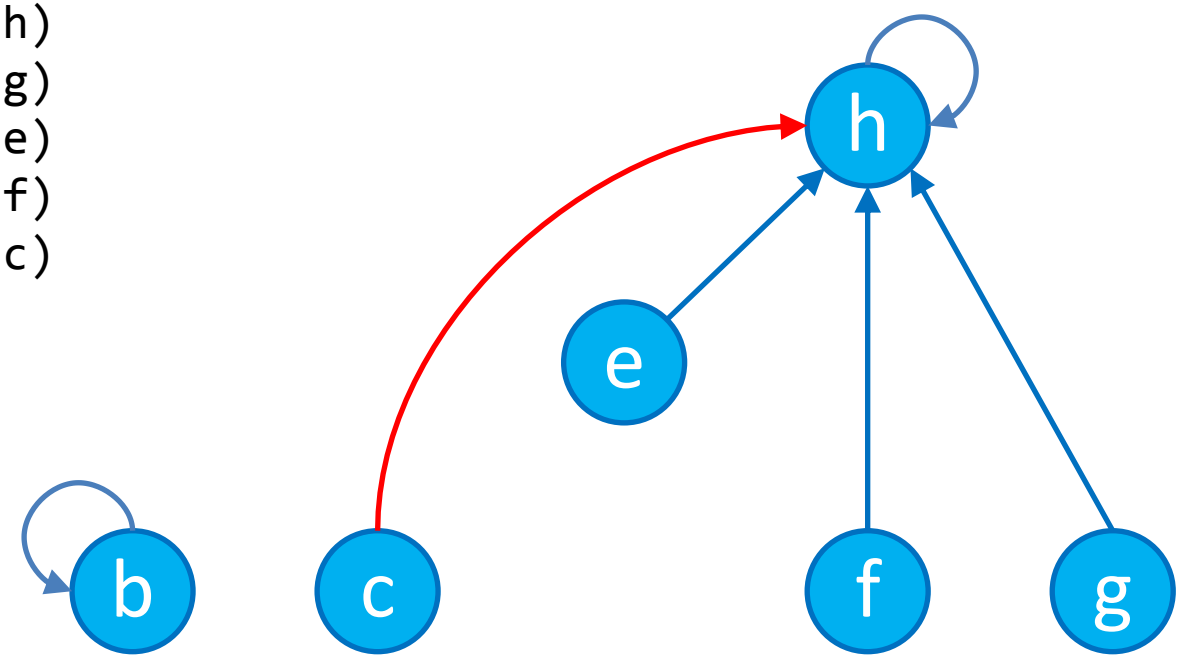
r = 2

r = 1

r = 0

Объединение по рангу и сжатие пути

```
Make_Set(b)
Make_Set(c)
Make_Set(e)
Make_Set(f)
Make_Set(g)
Make_Set(h)
Union(g,h)
Union(f,g)
Union(c,e)
Union(c,f)
Union(b,c)
```



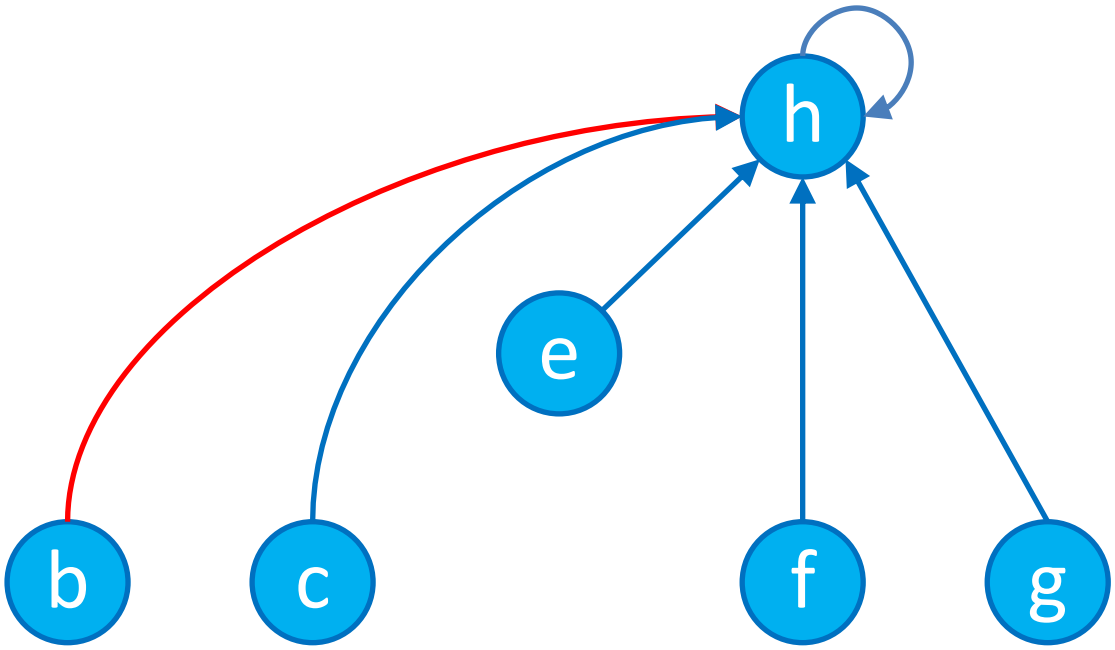
r = 2

r = 1

r = 0

Объединение по рангу и сжатие пути

```
Make_Set(b)
Make_Set(c)
Make_Set(e)
Make_Set(f)
Make_Set(g)
Make_Set(h)
Union(g,h)
Union(f,g)
Union(c,e)
Union(c,f)
Union(b,c)
```



r = 2

r = 1

r = 0