Автоматизация конструкторско-технологического проектирования

Лекция 04

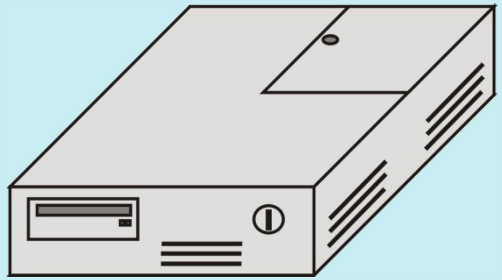
Декомпозиция,
Конструктивные алгоритмы,
алгоритм Кернигана-Лина, Фидуччи-Маттеуса

Разбиение

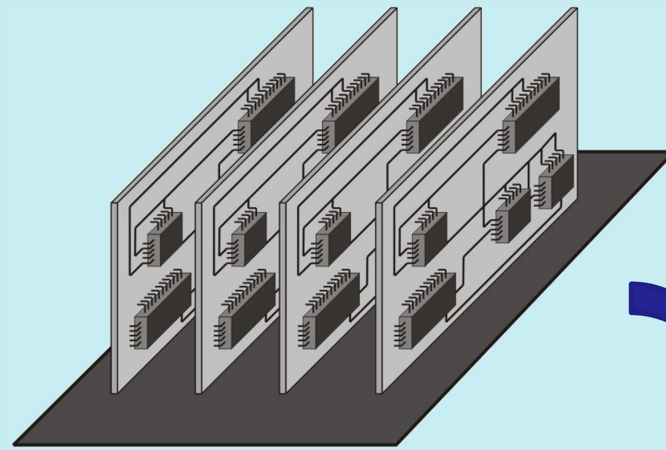
Используется для разделения системы на меньшие подсистемы

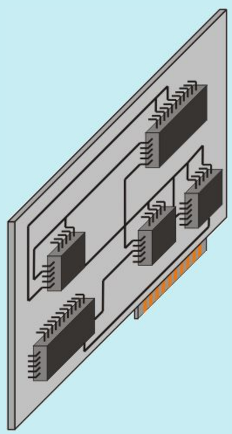
- Обычно производится иерархически
- Система бьется до тех пор, пока размер каждой подсистемы не будет удовлетворительным
- Подсистемы могут проектироваться независимо
- Минимизируется количество связей между частями
  - меньше требования к интерфейсу
  - сигнал между подсистемами имеет большую задержку

Уровни иерархии системы



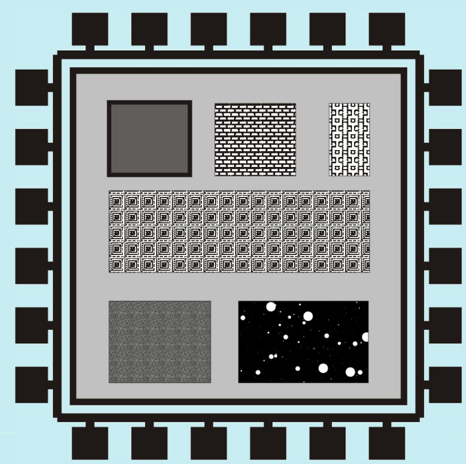
1. Уровень системы:
множество печатных плат



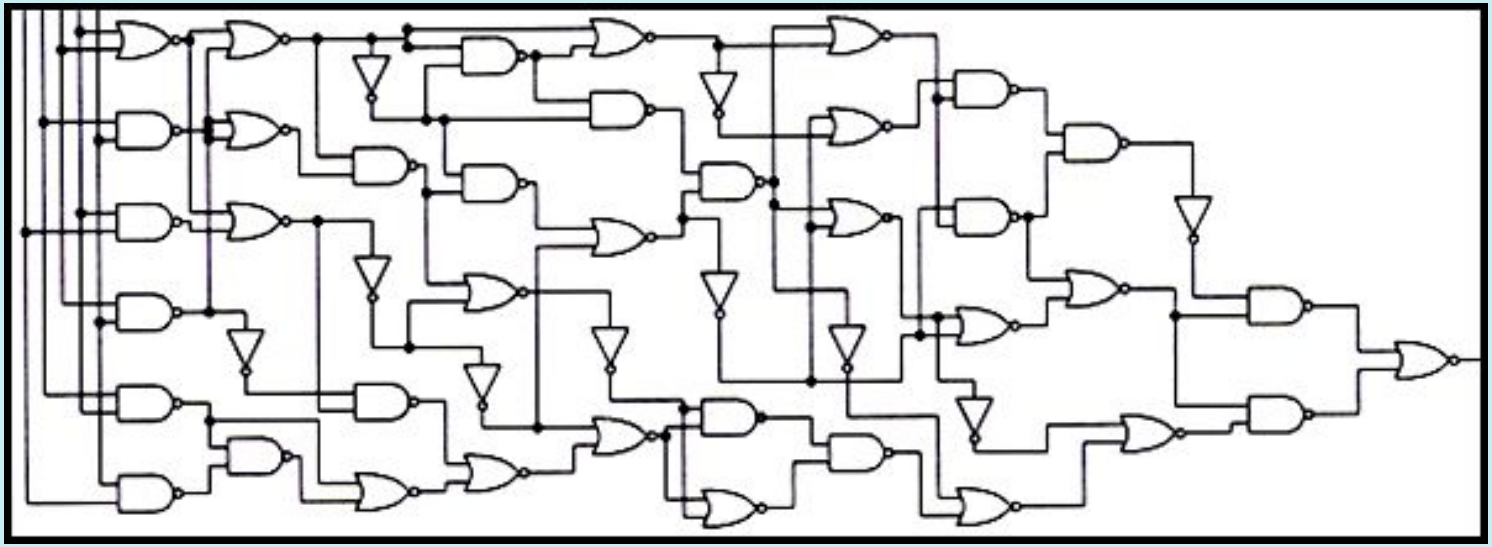


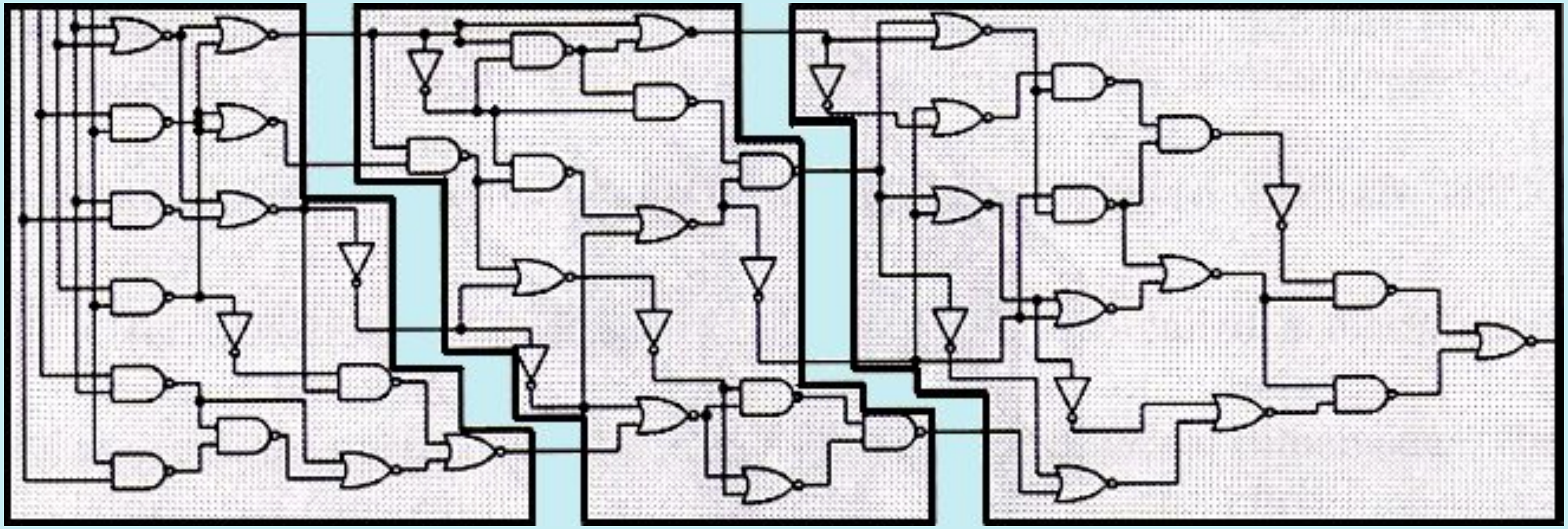
2. Уровень платы:
подсхемы реализуются в виде микросхем

3. Уровень микросхемы:
разбивается на блоки

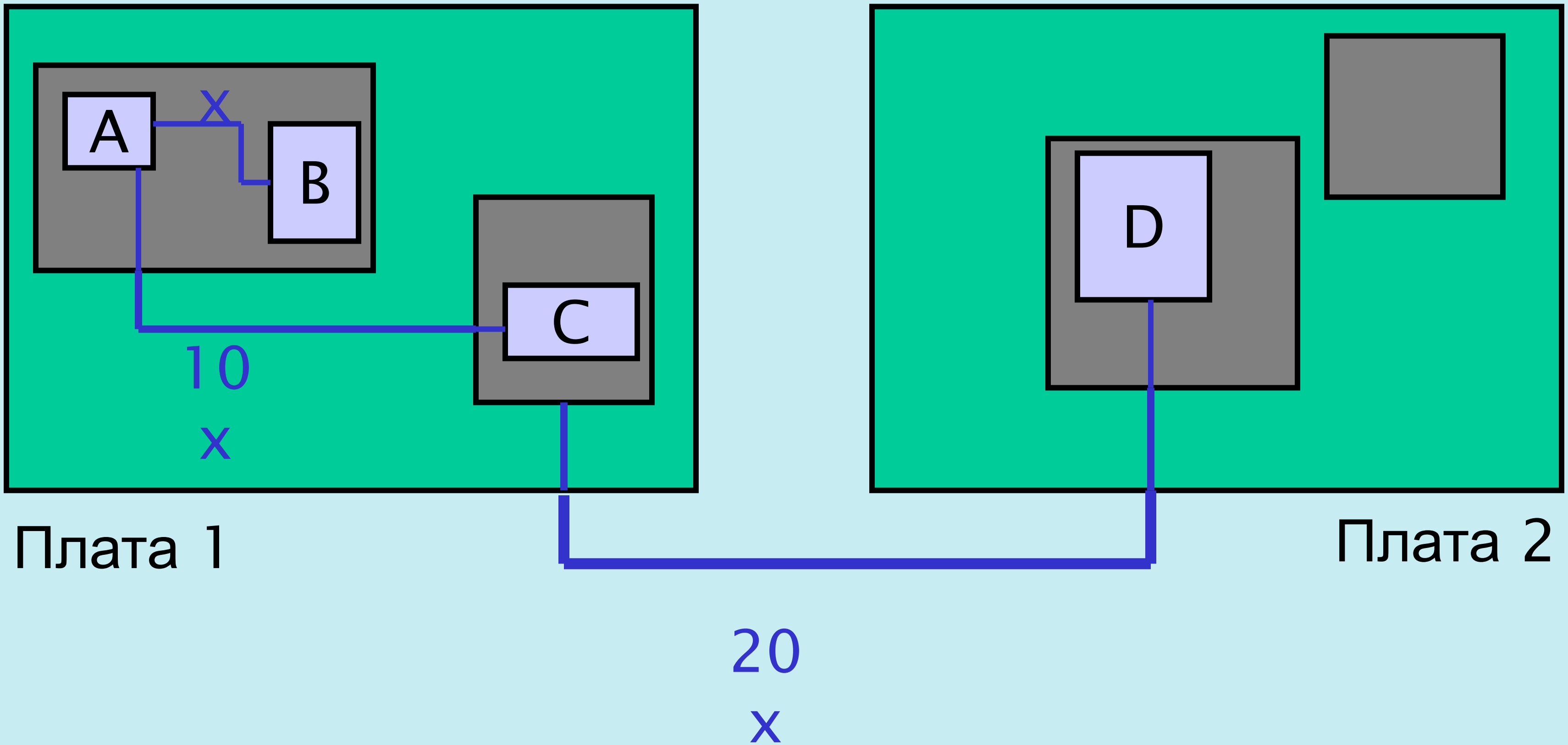


Разбиение схемы





Задержки сигнала



Быстродействие повышается при хорошем разбиении на верхних уровнях проектирования

Почему важна декомпозиция

Стратегия «разделяй и властвуй»

- **Эффективна для решения очень сложных задач**

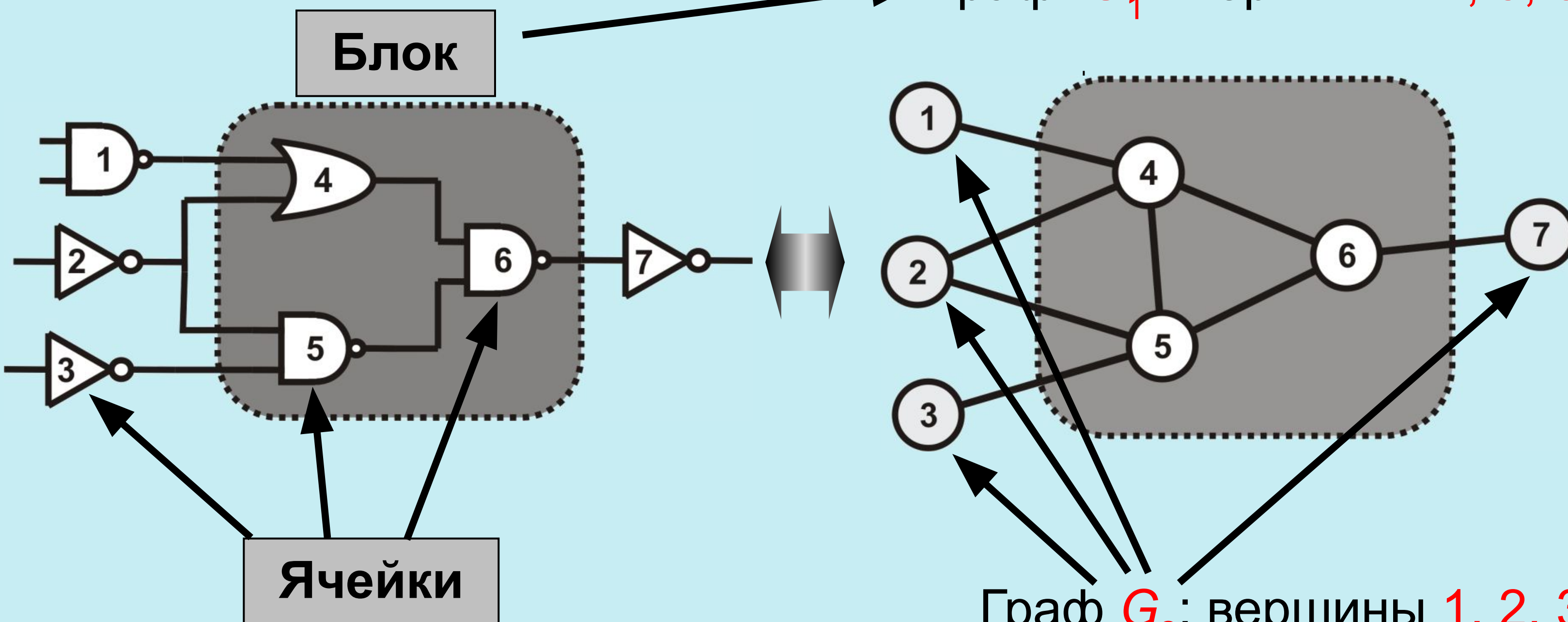
  ***Области применения*: дихотомическое размещение, генерация тестовых последовательностей, …**

- **На системном уровне разбивается на многочиповые схемы**
- **Влияет на задержку сигнала и быстродействие системы**
- **Применяется при параллельном моделировании схем**
- **Разбиение больших схем на набор ПЛИС или микроконтроллеров**
- **Разработка параллельных алгоритмов САПР**

  - разбиение задачи и распределение нагрузки

- **В современных схемах определяет локальные и глобальные межсоединеня → <u>прямо влияет на быстродействие</u>**

Необходимые определения



Граф G_1: вершины 4, 5, 6

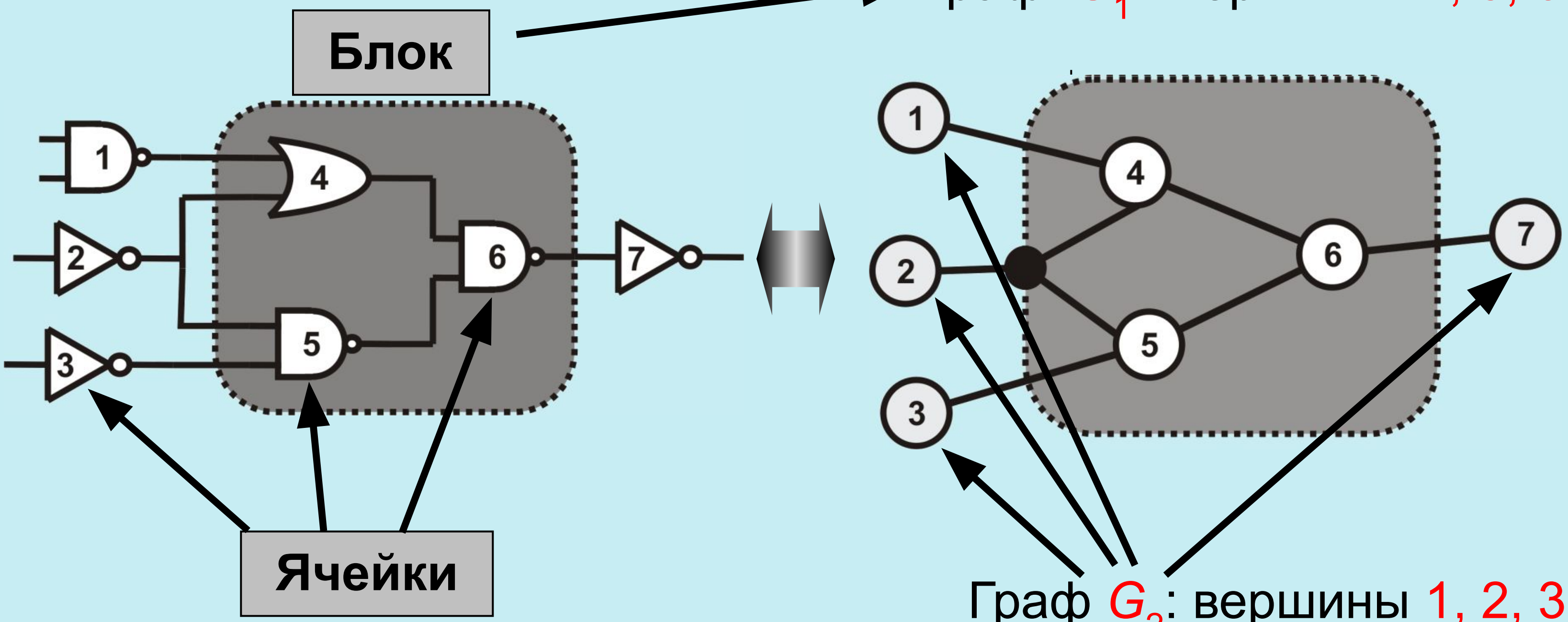
Граф G_2: вершины 1, 2, 3, 7

Ребра, попадающие под сечение:

(1,4), (2,4), (2,5), (3,5), (6,7)

Необходимые определения

Граф G_1: вершины 4, 5, 6



Граф G_2: вершины 1, 2, 3, 7

Ребра, попадающие под сечение:

(1,4), (2,4,5), (3,5), (6,7)

Необходимые определения

Виды задач

Декомпозиция: разбиение больших схем на несколько меньших сверху-вниз

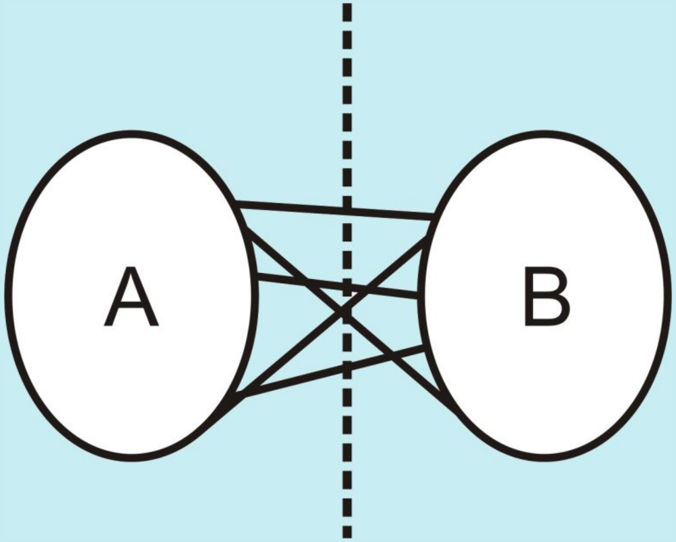
- Кластеризация: группирование небольших элементов в большие группы(кластеры) снизу-вверх
- Покрытие тестами / приведение к технологии:

  Кластеризация, где каждый кластер имеет специальную структуру (может быть реализован ячейкой из библиотеки)
- Разделение на k частей
- Половинное деление: разбиение на 2 части
- Бисекция: разбиение на 2 части одинакового размера

Формулировка задачи

Разновидности половинного деления

Цель: необходимо минимизировать количество связей между блоками c(A, B)



- *Минимальный разрез*: min c(A,B)
- *Минимальная бисекция*: min c(A,B) дополнительно |A|= |B|
- *Минимальный относительный разрез*: $\min \dfrac{c(A,B)}{|A|\cdot|B|}$

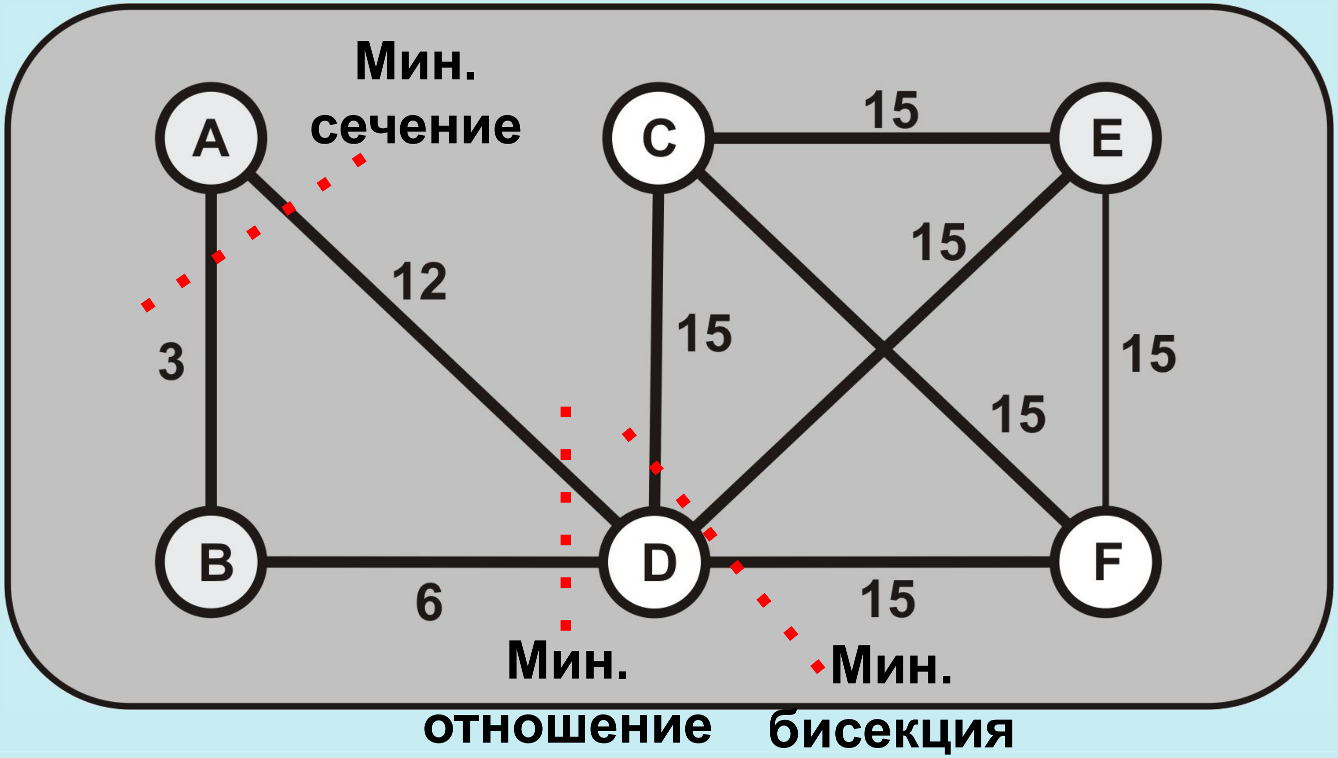
Пример разбиения

Стоимость минимального сечения = 15

Стоимость минимальной бисекции = 45

Стоимость деления с мин. отношения = 18

Минимизация отношения помогает находить натуральные кластеры



Критерии и ограничения

Критерии:

- ❑ минимальная связанность блоков
- ❑ максимальная связанность блоков
- ❑ равномерная связанность блоков
- ❑ функциональный признак
- ❑ удовлетворение ограничениям

Ограничения:

- ❑ количество блоков
- ❑ размер блоков
- ❑ число выводов блока

Классификация алгоритмов

Классы алгоритмов декомпозиции

Алгоритмы декомпозиции

Конструктивные

- Формируют разбиение из исходных данных и ограничений
- Предоставляют начальное разбиение другим алгоритмам

Итерационные

- Улучшают начальное разбиение
- Наиболее распространены из-за эффективности

Классификация алгоритмов

Классы алгоритмов декомпозиции

Алгоритмы декомпозиции

Детерминистические

- При одинаковых исходных данных дают одинаковое решение

Вероятностные

- Производят различные решения при каждом запуске

Классификация алгоритмов

Алгоритмы декомпозиции

Перемещения групп

Керниган-Лин
Голдберг-Бурштейн
Фидуччи-Маттеус
Относительное разбиение

Моделирование процессов

Моделирование отжига
Моделирование эволюции

Оптимизации быстродействия

Лоулер
Левитт
Тюнер

Другие

Metric allocation

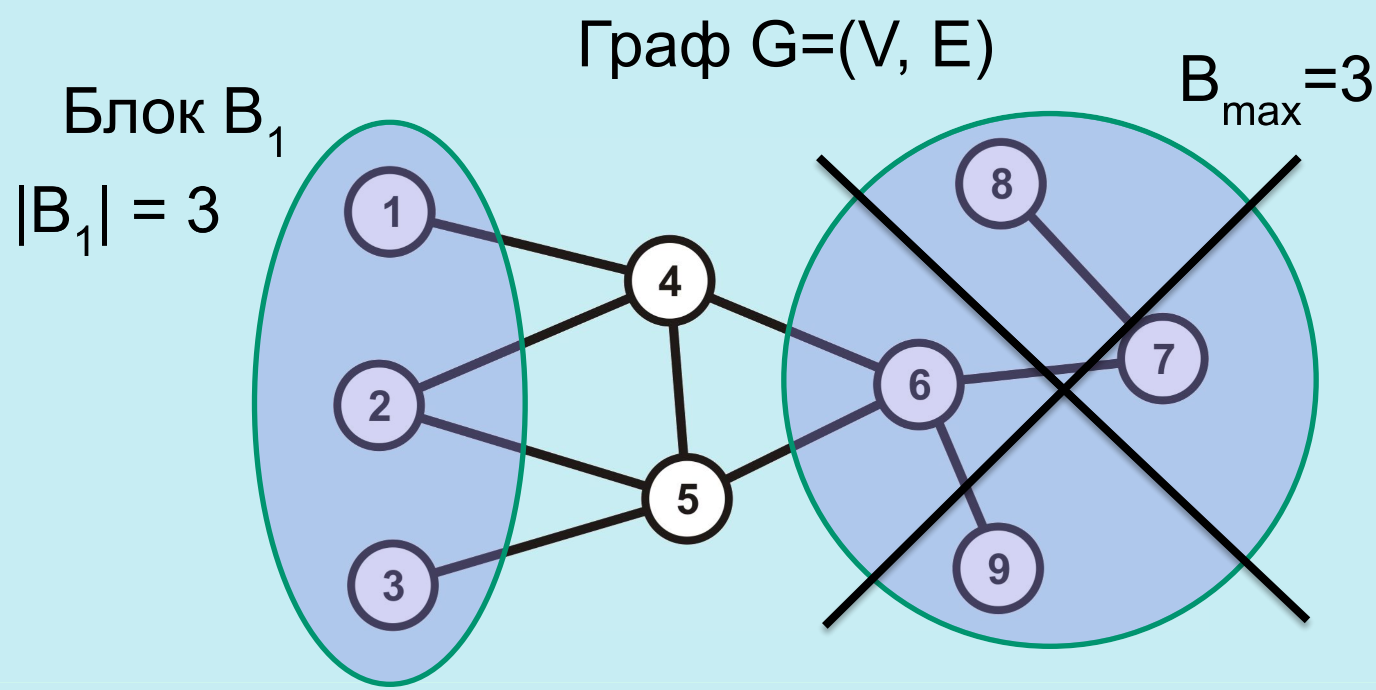
Базовый алгоритм кластеризации

Входные данные:

- Граф G=(V, E)
- Матрица смежности C
- Максимальный размер блока B_{max}

Обозначения:

- Bi – i-й блок
- |Bi| - число элементов в блоке



Базовый алгоритм кластеризации

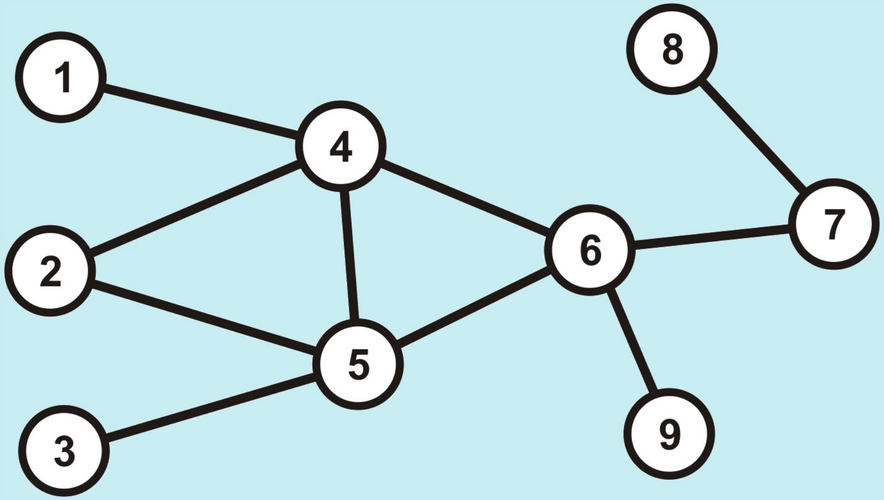
Алгоритм:

i=1
Пока в E есть элементы
  Если $|Bi|=0$
    Найти элемент с наибольшим количеством связей e_{max}
    Переместить e_{max} из E в Bi
  Иначе
    Если $|B_i| < B_{max}$
      Найти элемент e_{max}, наиболее связанный с B_i
      Переместить e_{max} из E в B_i
    Иначе
      i=i+1

Базовый алгоритм кластеризации

Пример:

bmax=3

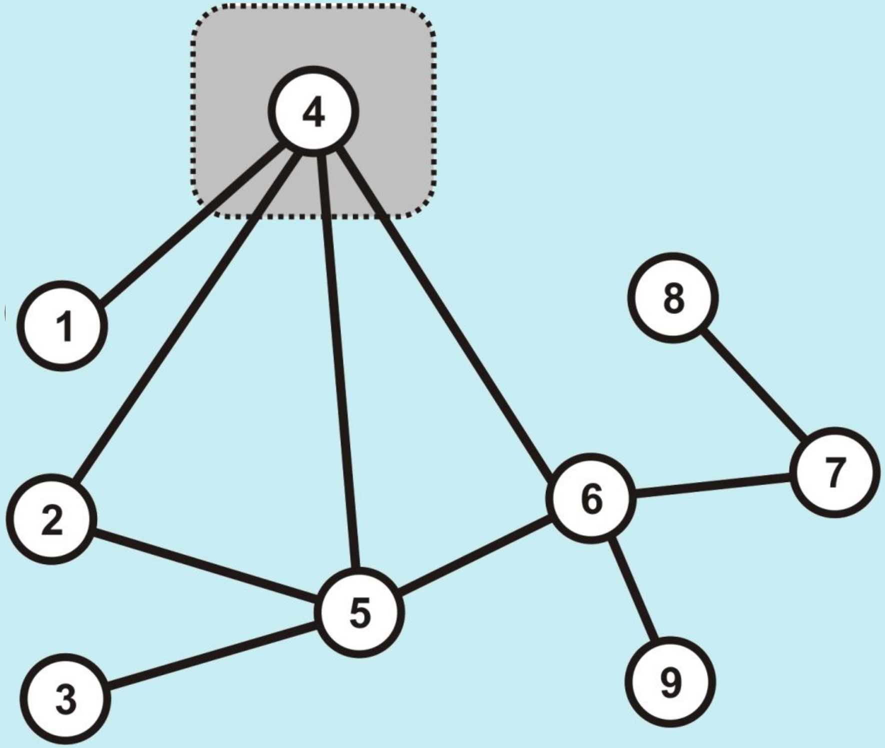


Матрица смежности

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

Базовый алгоритм кластеризации

Пример:



Матрица смежности

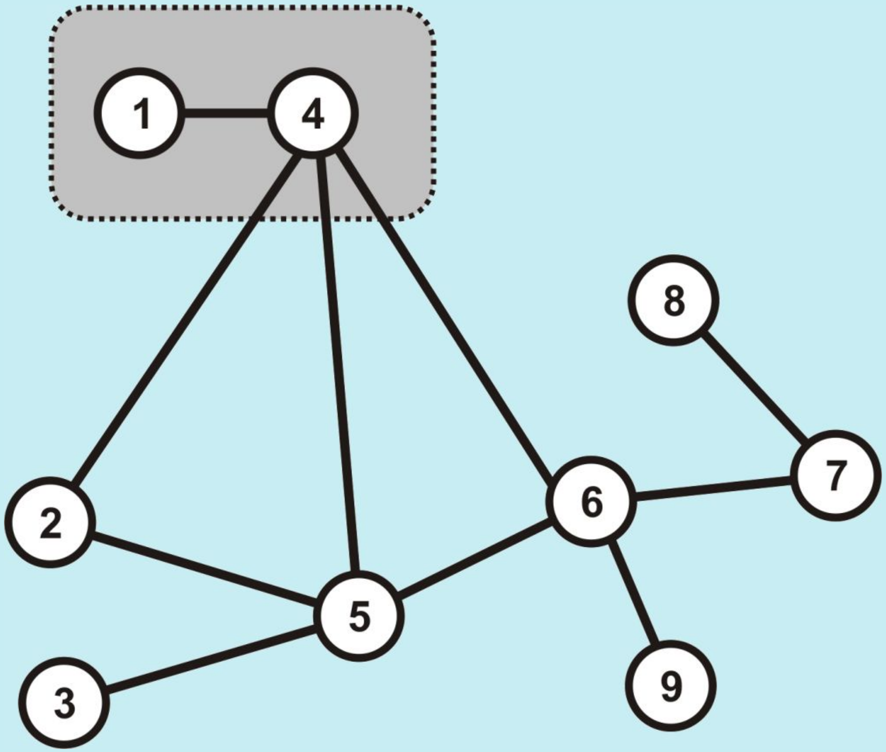
	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

i=1
Набор первого элемента в блок B_1
Наиболее связаны $\{e_4, e_5, e_6\}$
Лексикографически выбираем e_4

$$B_1=\{e_4\}$$

Базовый алгоритм кластеризации

Пример:



Матрица смежности

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

B_1={e_1, e_4 }

i=1

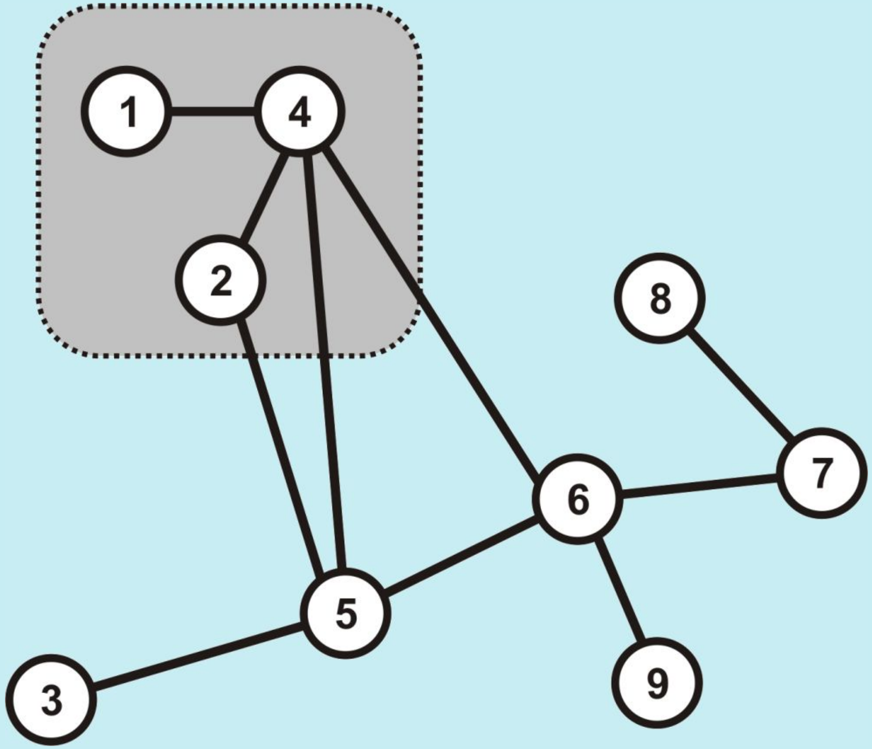
Поиск очередного элемента в B_1

Наиболее связаны с B_1 {e_1, e_2, e_5 , e_6}

Лексикографически выбираем e_1

Базовый алгоритм кластеризации

Пример:



Матрица смежности

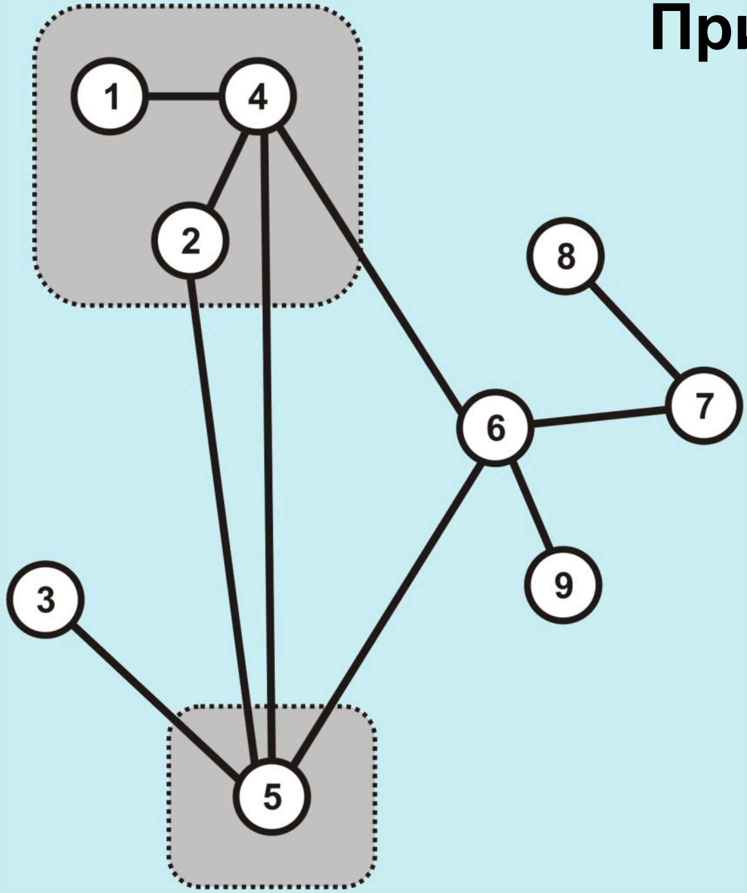
	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

B_1={e_1, e_2, e_4}

B_1 сформирован
Поиск очередного элемента в B_1
Наиболее связаны с B_1 {e_2, e_5 , e_6}
Лексикографически выбираем e_2

Базовый алгоритм кластеризации

Пример:



Матрица смежности

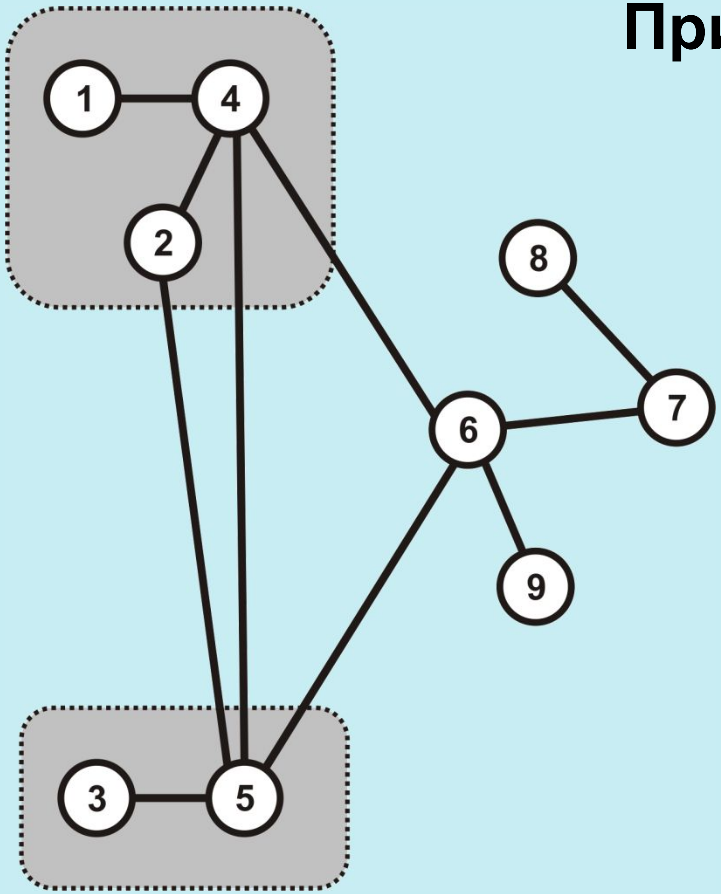
	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

i=2
Набор первого элемента в блок B_2
Наиболее связаны $\{e_5, e_6\}$
Лексикографически выбираем e_5

$B_1=\{e_1, e_2, e_4\}$
$B_2=\{e_5\}$

Базовый алгоритм кластеризации

Пример:



Матрица смежности

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	**1**	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	**1**	1	0	**1**	0	0	0
e_6	0	0	0	1	**1**	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

i=2

Поиск очередного элемента в B_2

Наиболее связаны с B_2 $\{e_3, e_6\}$

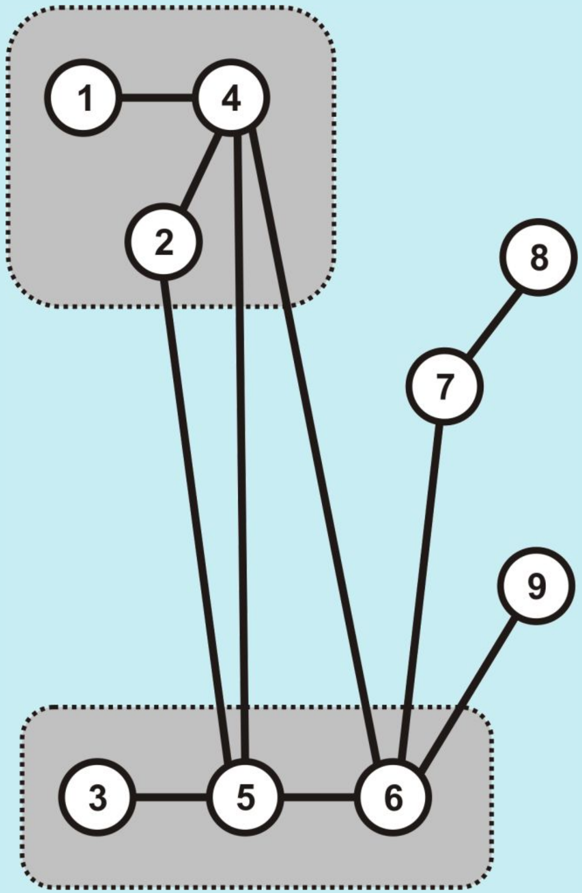
Лексикографически выбираем e_3

$B_1=\{e_1, e_2, e_4\}$

$B_2=\{e_5, e_3\}$

Базовый алгоритм кластеризации

Пример:



Матрица смежности

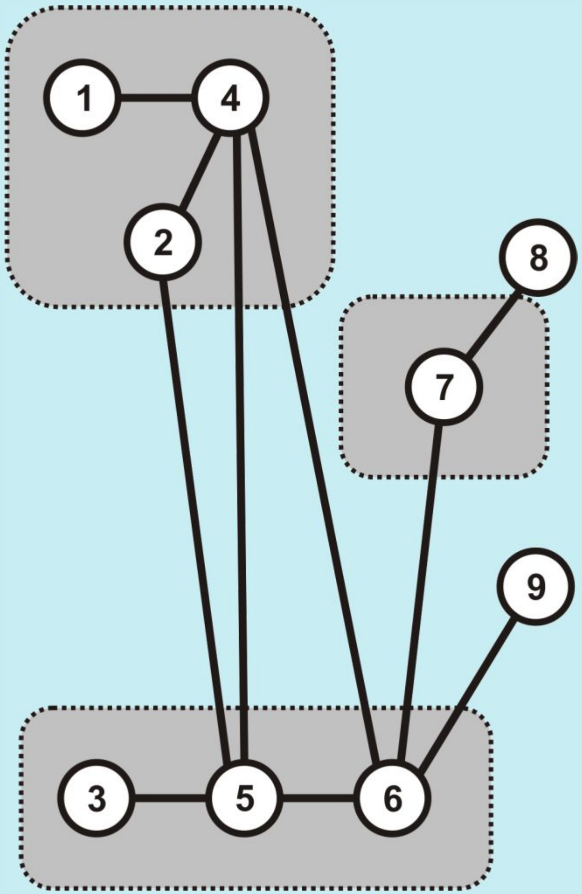
	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

B_2 сформирован
Поиск очередного элемента в B_2
Наиболее связаны с B_2 {e_6}
Выбираем e_6

B_1={e_1, e_2, e_4 }
B_2={e_5, e_3, e_6 }

Базовый алгоритм кластеризации

Пример:



Матрица смежности

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

i=3

Набор первого элемента в блок B_3

Наиболее связаны $\{e_7\}$

Выбираем e_7

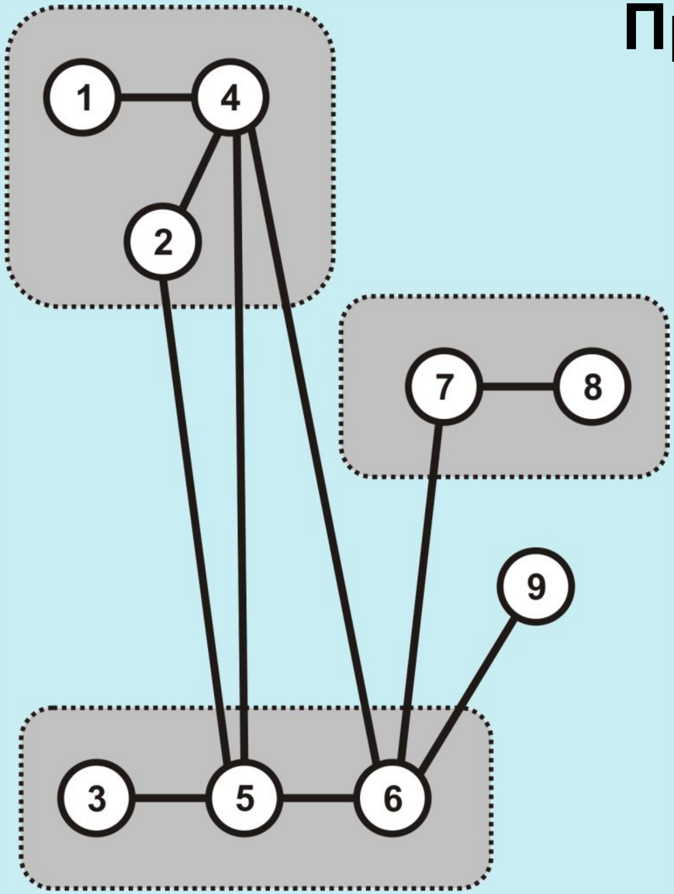
$B_1=\{e_1, e_2, e_4\}$

$B_2=\{e_5, e_3, e_6\}$

$B_2=\{e_7\}$

Базовый алгоритм кластеризации

Пример:



Матрица смежности

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

i=3

Поиск очередного элемента в B_3

Наиболее связаны с B_3 {e_8}

Выбираем e_8

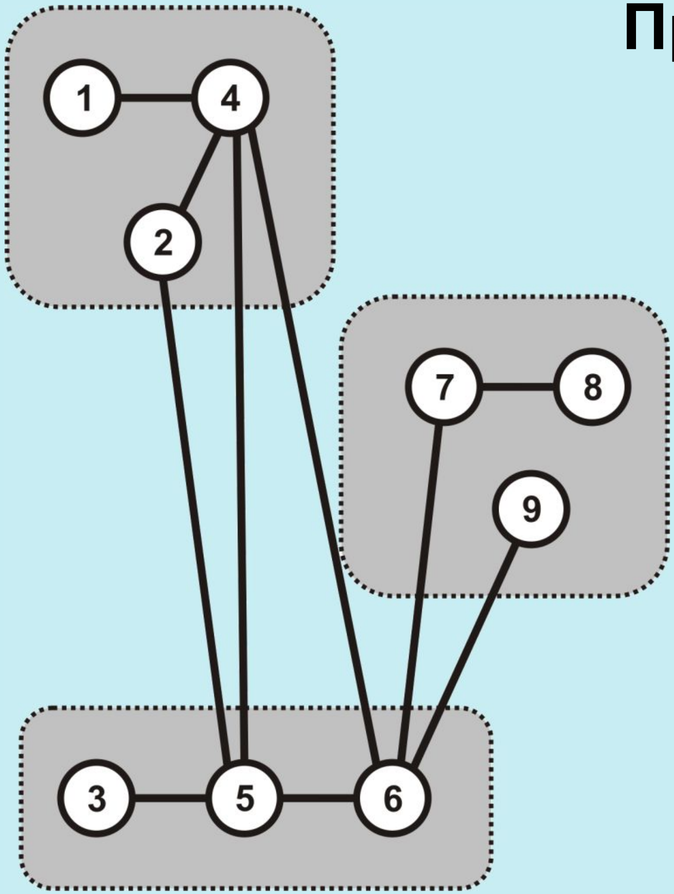
B_1={e_1, e_2, e_4 }

B_2={e_5, e_3, e_6 }

B_2={e_7, e_3}

Базовый алгоритм кластеризации

Пример:



Матрица смежности

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9
e_1	0	0	0	1	0	0	0	0	0
e_2	0	0	0	1	1	0	0	0	0
e_3	0	0	0	0	1	0	0	0	0
e_4	1	1	0	0	1	1	0	0	0
e_5	0	1	1	1	0	1	0	0	0
e_6	0	0	0	1	1	0	1	0	1
e_7	0	0	0	0	0	1	0	1	0
e_8	0	0	0	0	0	0	1	0	0
e_9	0	0	0	0	0	1	0	0	0

B_3 сформирован
Поиск очередного элемента в B_3
Наиболее связаны с B_3 {e_9}
Выбираем e_9

B_1={e_1, e_2, e_4 }
B_2={e_5, e_3, e_6 }
B_2={e_7, e_3, e_9}

Базовый алгоритм кластеризации

Способы повышения эффективности:

- После переноса вершины в кластер обновлять матрицу смежности (не учитывать связность с кластерами)
- Учитывать не только связность с кластером, но и с другими элементами
- «Просматривать» работу алгоритма на несколько ходов вперед и выбирать наилучший вариант
- При просмотре вперед использовать параллельные вычисления

Алгоритм Кодреса

Идеи алгоритма

- Блоки формируются последовательно
- Первый элемент блока выбирается по набольшей связности с периферией
- Очередной элемент блока выбирается по наибольшей связанности с элементами внутри текущего блока и по наименьшей связанности с остальными
- Блок наполняется элементами как можно плотнее, даже в ущерб связности

Алгоритм Кодреса

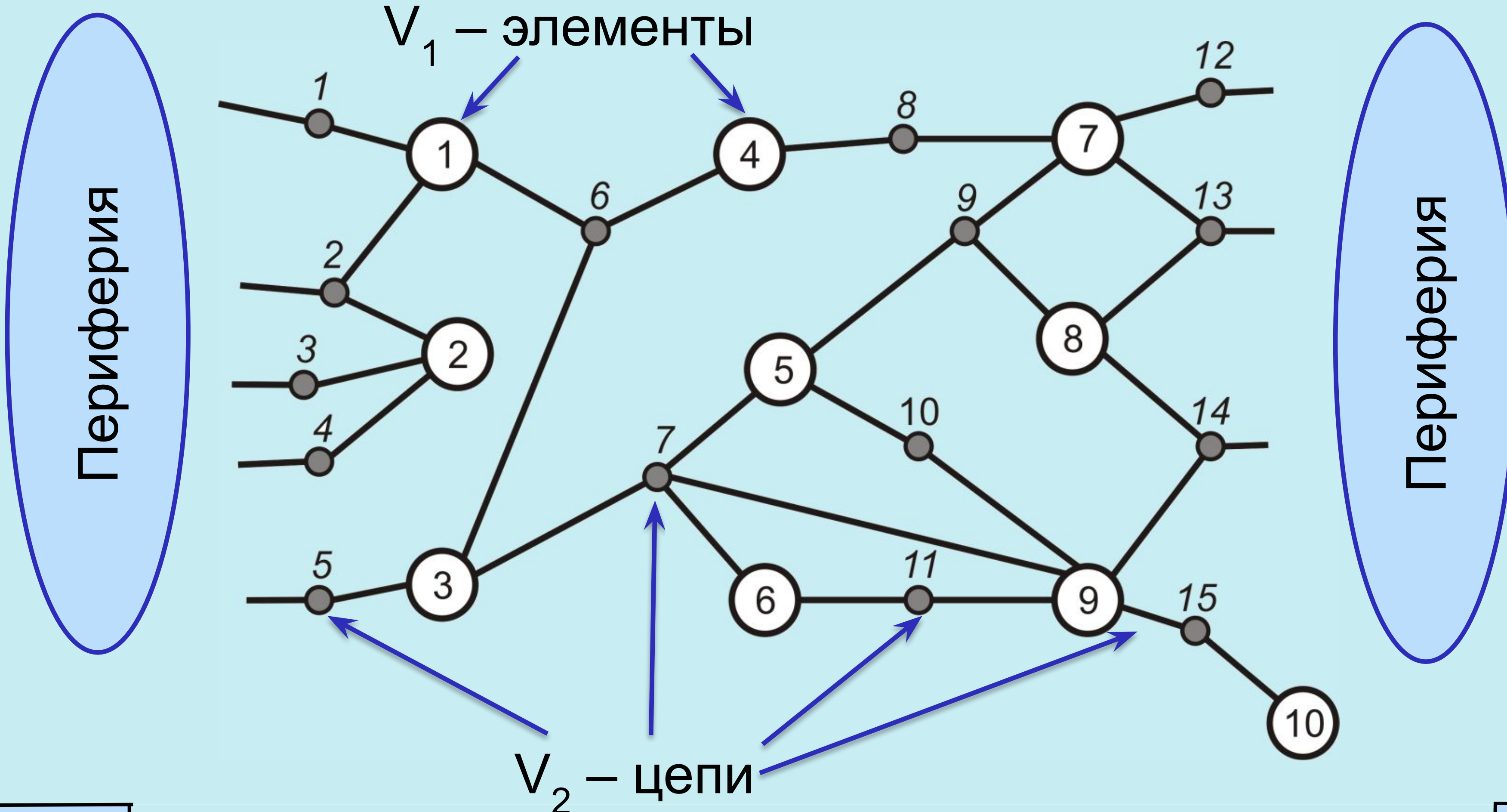
Входные данные:

Граф Кенига $G = \left(V_1 \cup V_2, E\right)$

Ограничения:

S_{max} – Макс. площадь блока

T_{max} – Макс. количество выводов



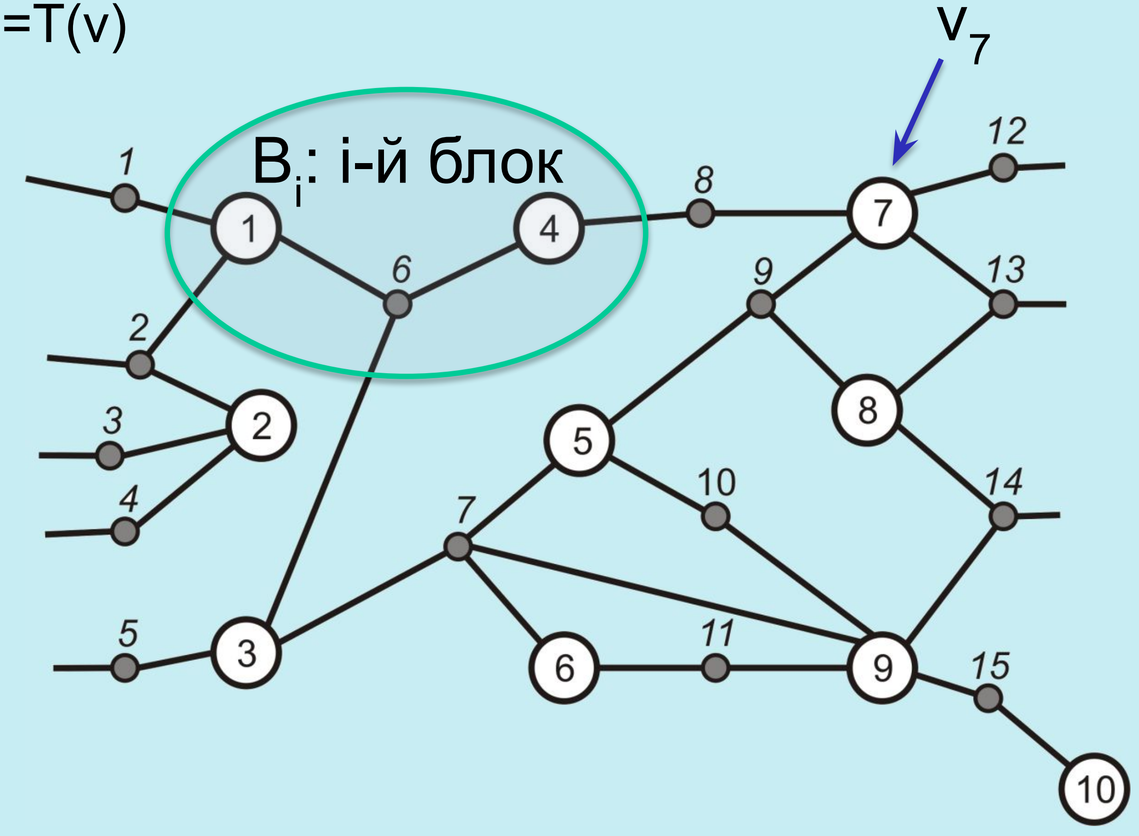
Алгоритм Кодреса

Условные обозначения

Количество элементов в блоке: $|B_i|=2$

$S(v)$ – площадь блока
$T(v)$ – число внешних выводов
Примем для простоты $S(v)=T(v)$

$S(B_i)=5$
$T(B_i)=4$

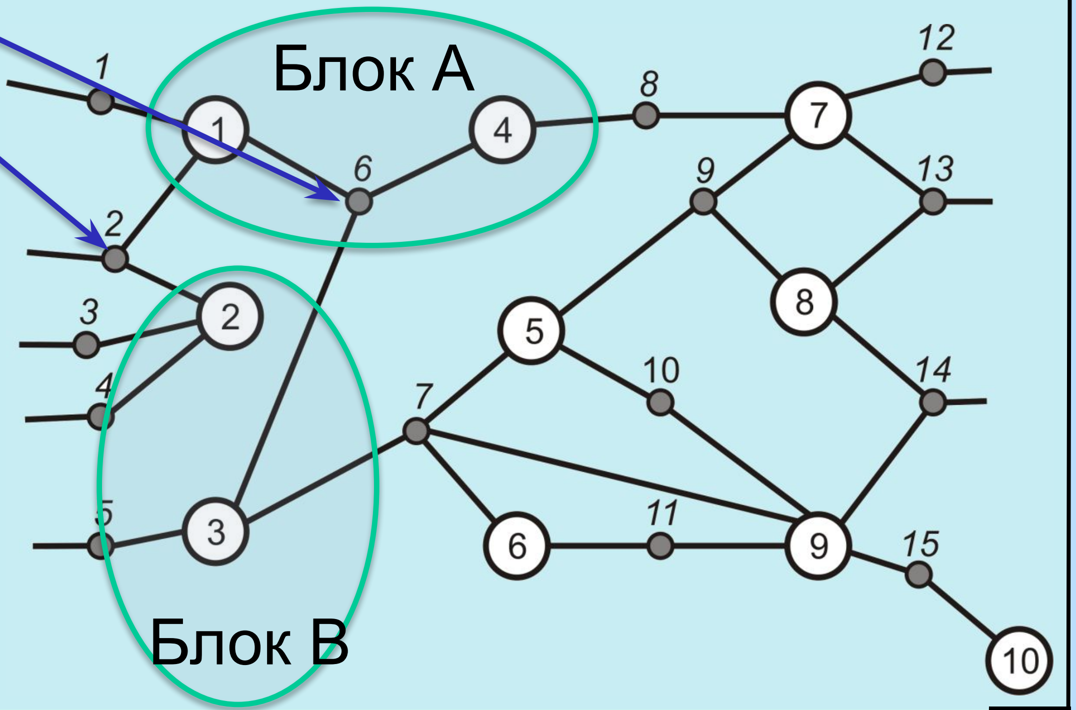


Алгоритм Кодреса

Условные обозначения

- Конъюнкция А и В: con(A,B) - число общих цепей у А и В
- Дизъюнкция А и В: dis(A,B) - суммарное число цепей без повторений за вычетом конъюнкции А и В

Пример: $con(A,B) = |\{1,2,6,8\}| \cap |\{2 \div 7\}| = |\{2,6\}| = 2$



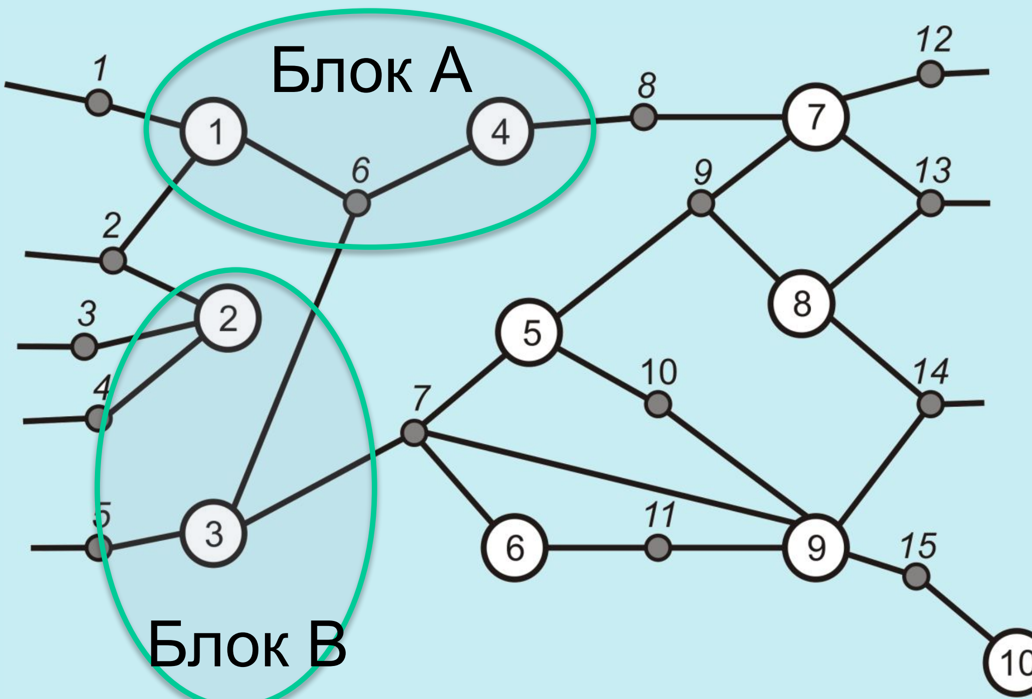
Алгоритм Кодреса

Условные обозначения

- Конъюнкция А и В: con(A,B) - число общих цепей у А и В
- Дизъюнкция А и В: dis(A,B) - суммарное число цепей без повторений за вычетом конъюнкции А и В

Пример: dis(A,B) = |{1÷8}| -con(A,B)= 8-2 = 6

con(A,B)=2



Алгоритм Кодреса

Алгоритм

1. i=1; $A=A_0$

2. Выбрать а из А так, что $con(a, A) \to \max$
Если есть равные, то $con(a, A \setminus a) \to \min$
Если есть равные, то лексикографически.

3. Переместить а из А в B_i
Если А пустое, то КОНЕЦ
Иначе
Выбрать а из А так, что $con(B_i, a) \to \max$
Если есть равные, то $dis(P_i, a) \to \min$
Если есть равные, то лексикографически.

4. Проверить ограничения на блок:

$$\begin{cases} S(B_i \cup a) \le S_{\max} \\ T(B_i \cup a) \le T_{\max} \end{cases}$$

Если выполняется, то перейти к п. 3
Иначе проверить остальные элементы в лексикографическом порядке
Если
удовлетворяющий условиям найден, то добавить в блок, п. 2
i=i+1

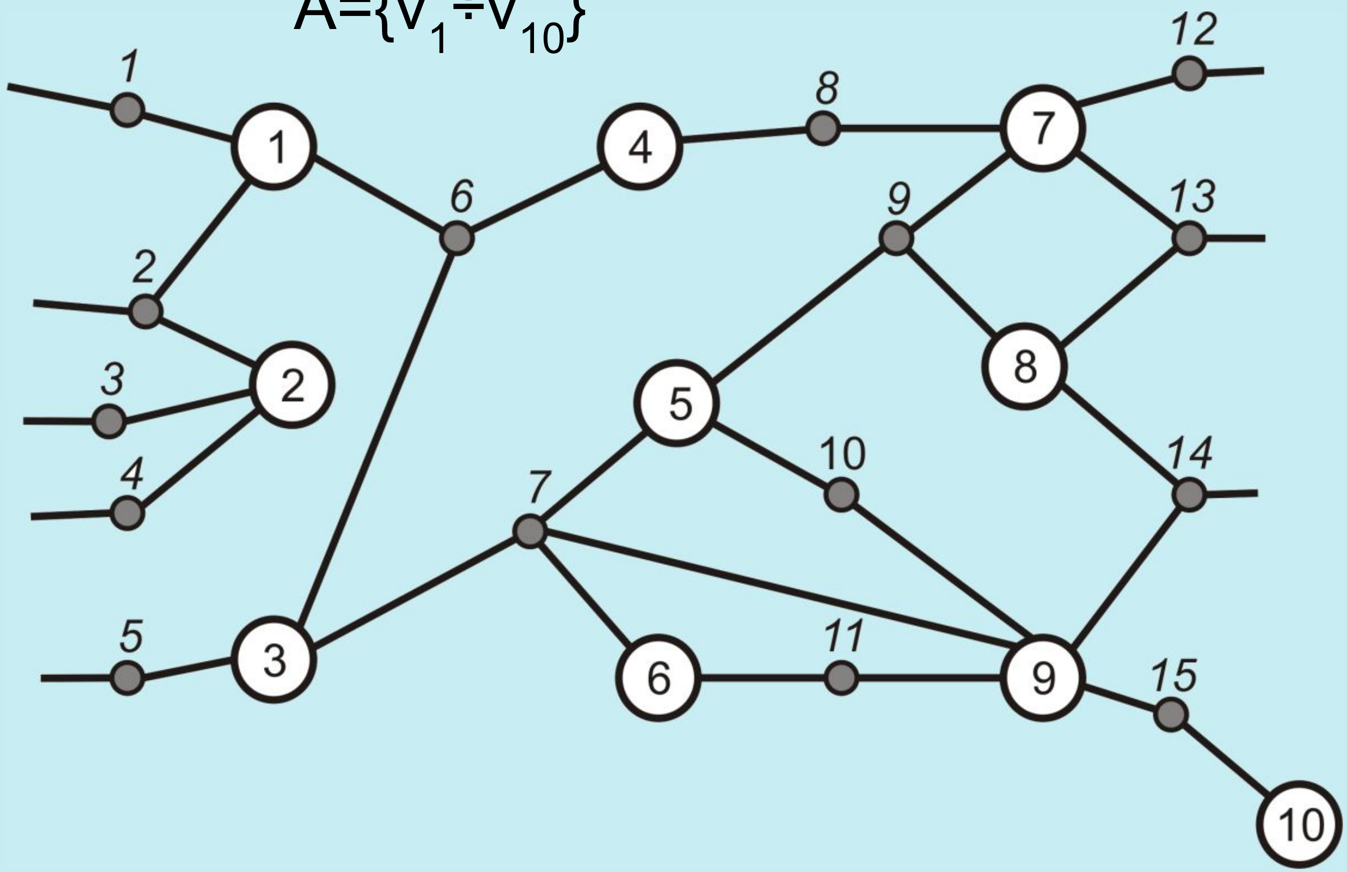
Алгоритм Кодреса

Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

$S(v_1) = 3$
$S(v_2) = 3$
$S(v_3) = 3$
$S(v_4) = 2$
$S(v_5) = 3$
$S(v_6) = 2$
$S(v_7) = 4$
$S(v_8) = 3$
$S(v_9) = 5$
$S(v_{10}) = 1$



Алгоритм Кодреса

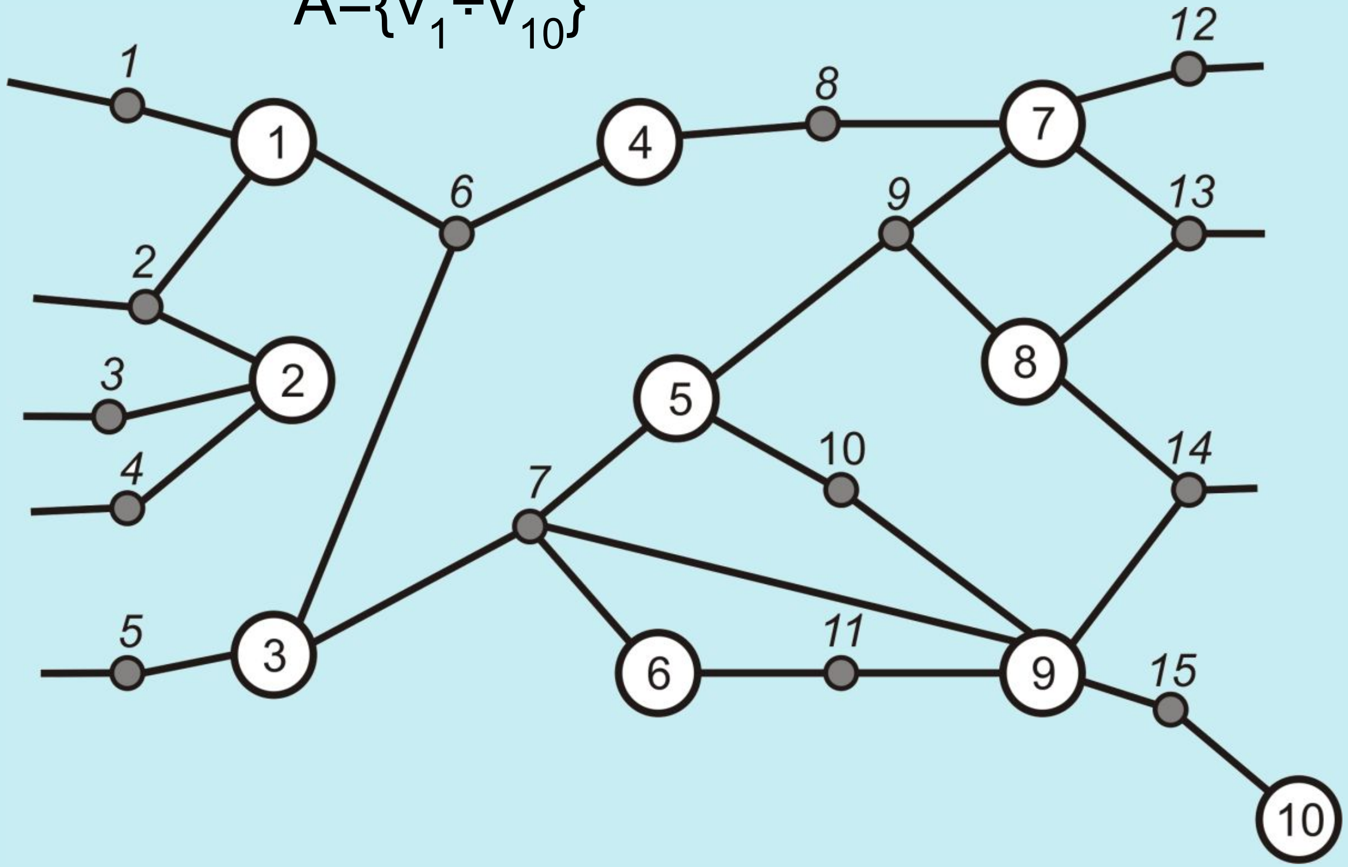
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

Выполнение п. 1

$i=1$



$B_1=\{\}$

Алгоритм Кодреса

Алгоритм

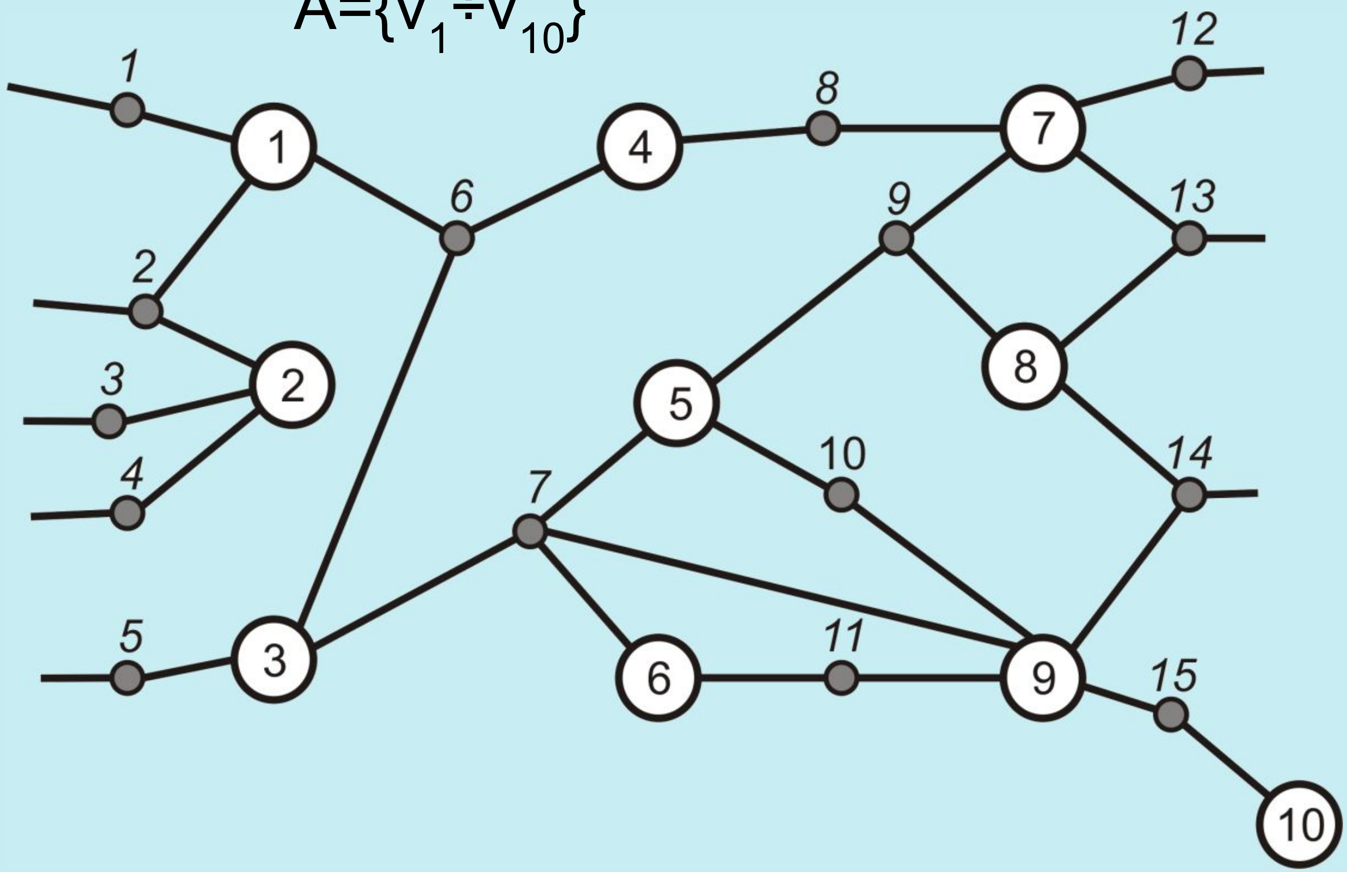
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

Выполнение п. 2

$con(a, A) \to \max$

$a=v_2$



$B_1=\{\}$

#a	1	2	3	4	5	6	7	8	9	10
con(a,A)	2	3	1	0	0	0	2	2	2	0

Алгоритм Кодреса

Алгоритм

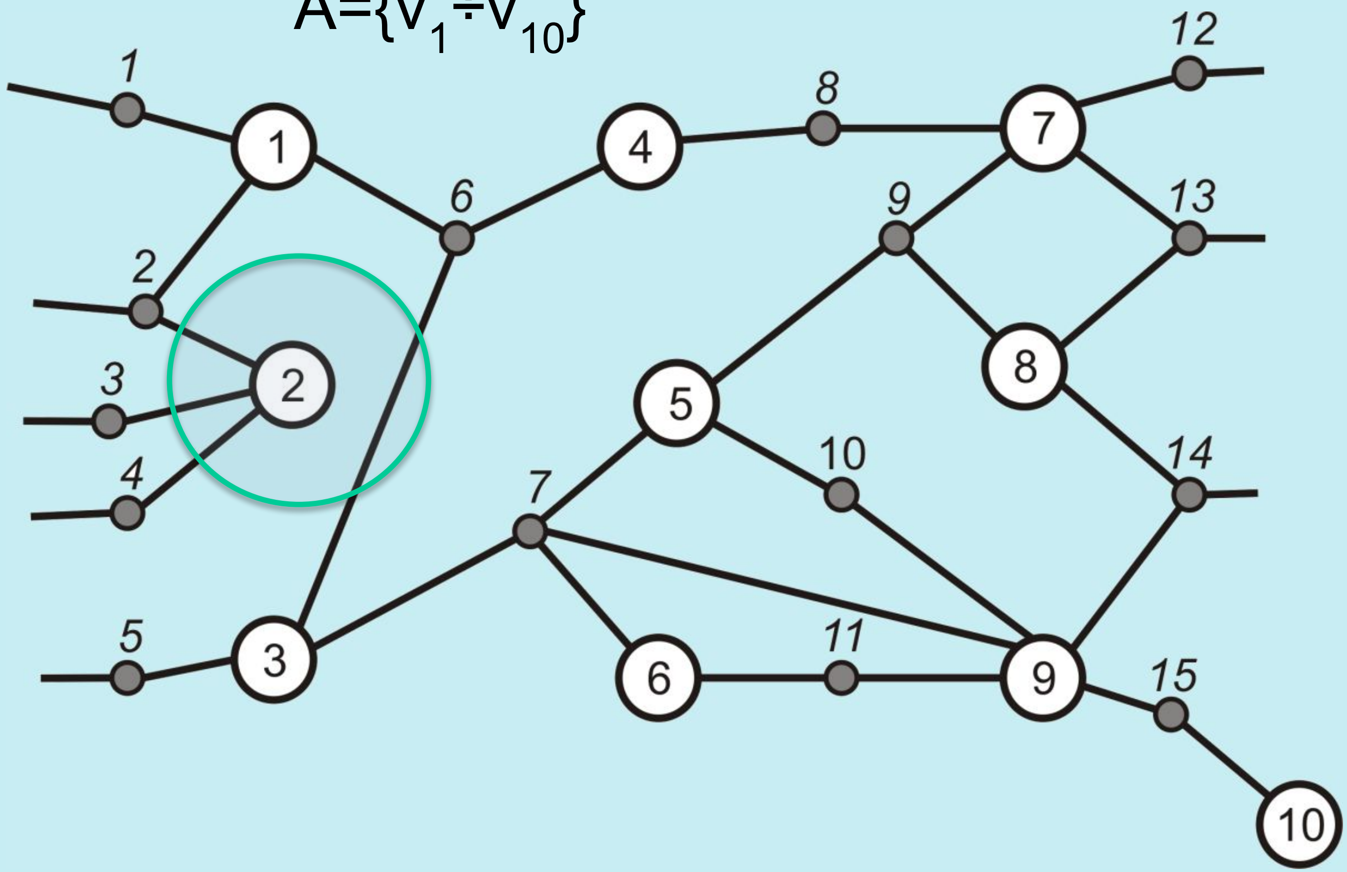
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

Выполнение п. 3

$con(a, B_1) \rightarrow \max$

$a=v_1$



$B_1=\{v_2\}$

#a	1	2	3	4	5	6	7	8	9	10
con(a,B_1)	1	x	0	0	0	0	0	0	0	0

Алгоритм Кодреса

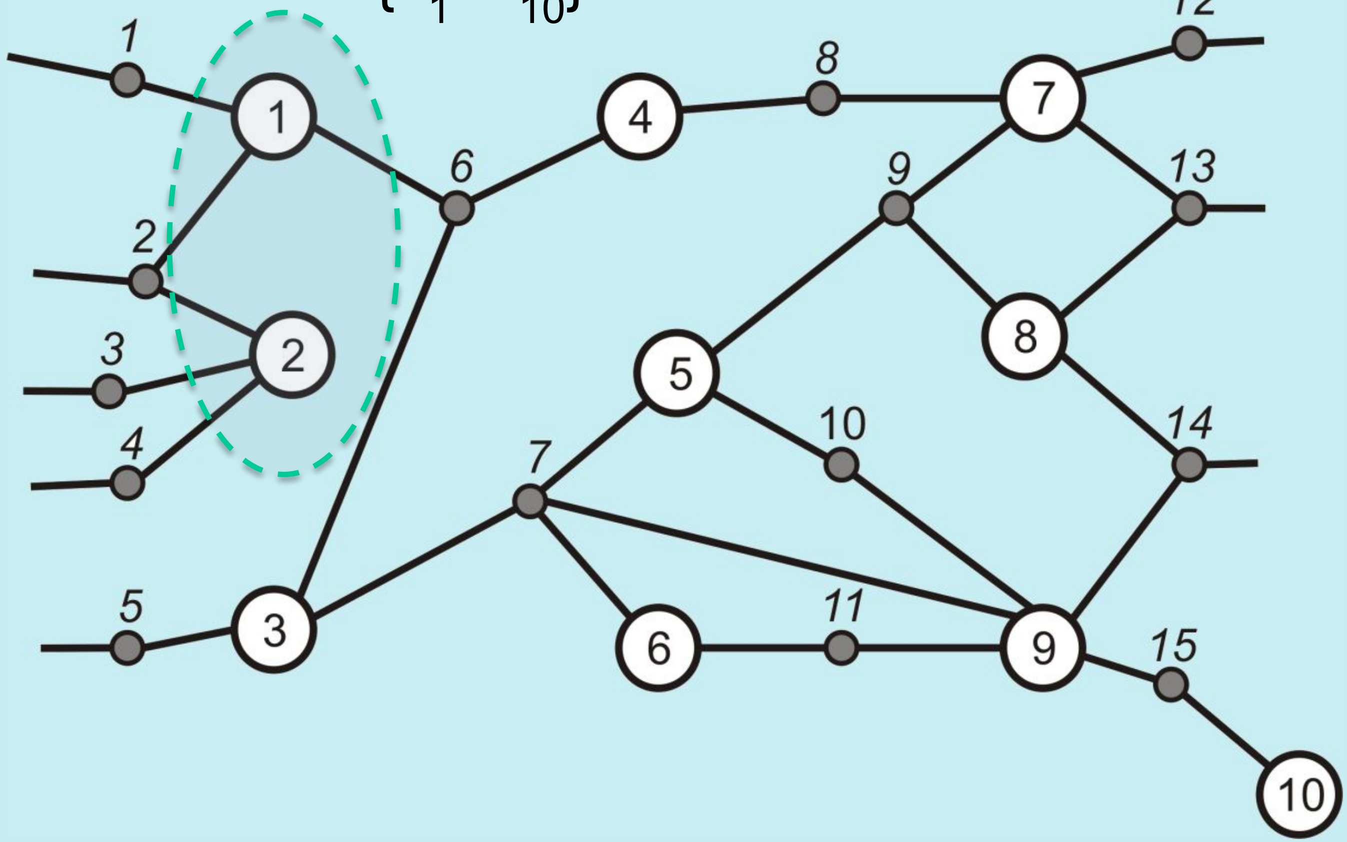
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_1 \cup v_1) = 6 \\ T(B_1 \cup v_1) = 5 \end{cases}$$



$B_1=\{v_2\}$

Алгоритм Кодреса

Алгоритм

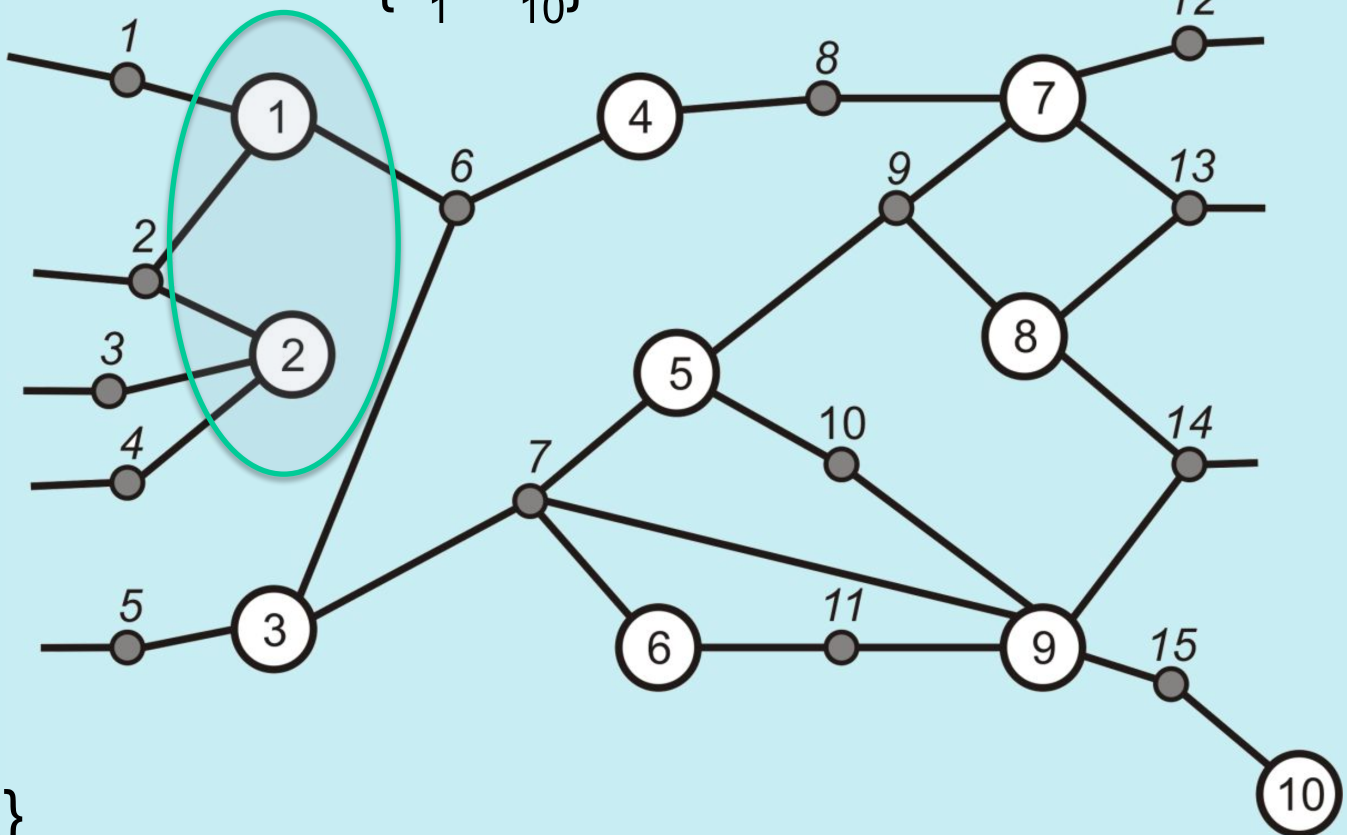
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

Выполнение п. 3

$con(a, B_1) \to \max$

$a=\{v_3, v_4\}$



$B_1=\{v_1, v_2\}$

#a	1	2	3	4	5	6	7	8	9	10
$con(a,B_1)$	x	x	1	1	0	0	0	0	0	0

Алгоритм Кодреса

Алгоритм

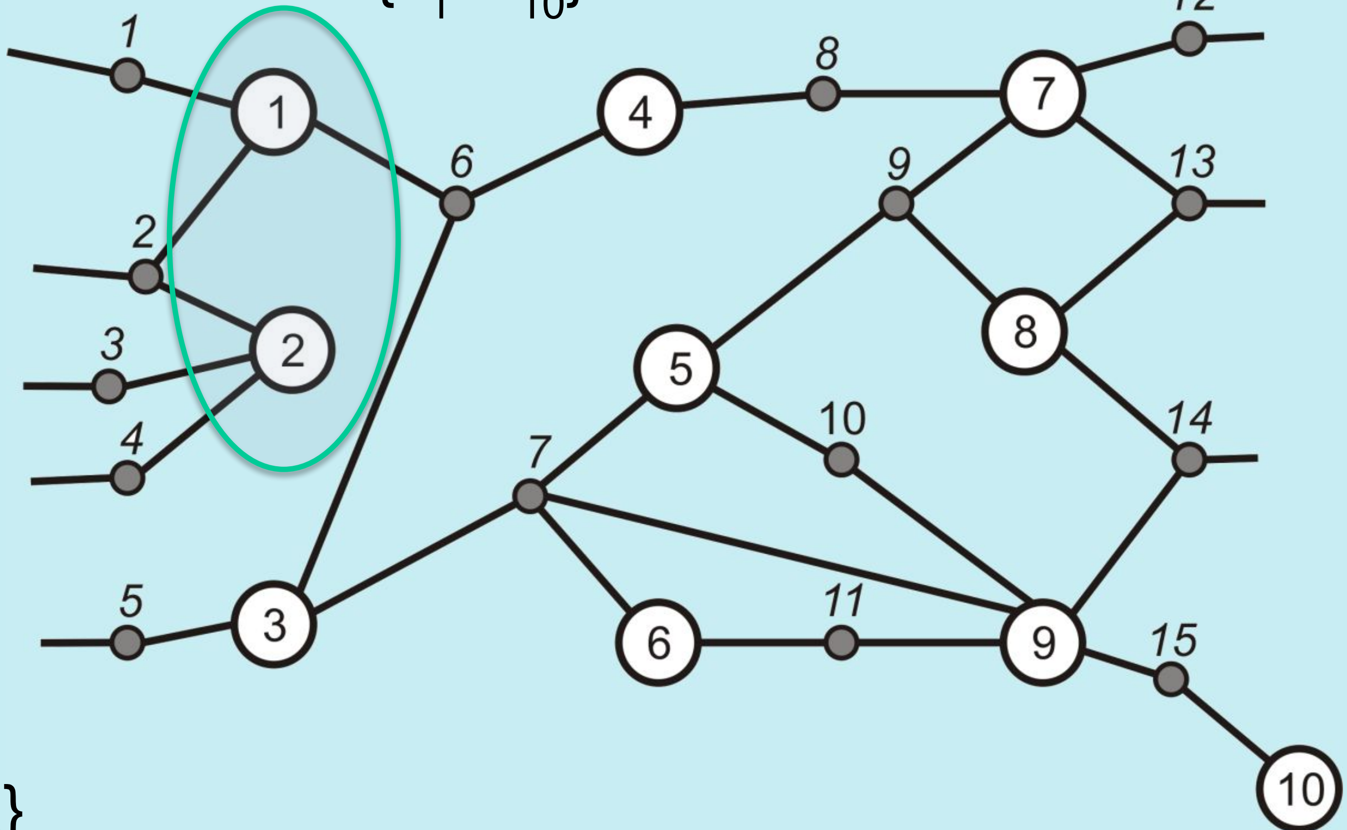
$S_{max} = 10$
$T_{max} = 8$

Выполнение п. 3

$dis(a, B_1) \to \min$

$a = v_4$

$A = \{v_1 \div v_{10}\}$



$B_1 = \{v_1, v_2\}$

#a	1	2	3	4	5	6	7	8	9	10
dis(a,B_1)	x	x	6	5	x	x	x	x	x	x

Алгоритм Кодреса

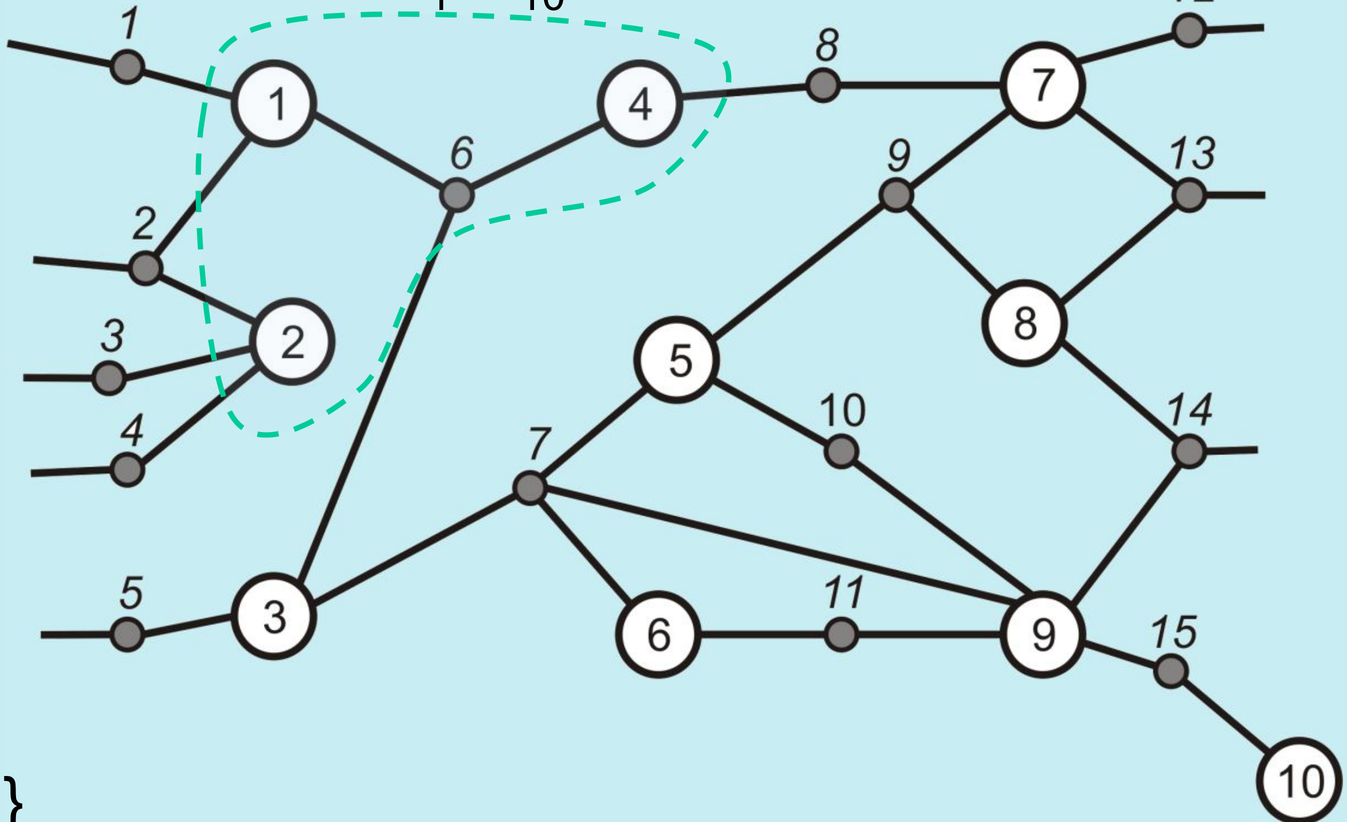
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_1 \cup v_1) = 8 & + \\ T(B_1 \cup v_1) = 6 & + \end{cases}$$



$B_1=\{v_1, v_2\}$

Алгоритм Кодреса

Алгоритм

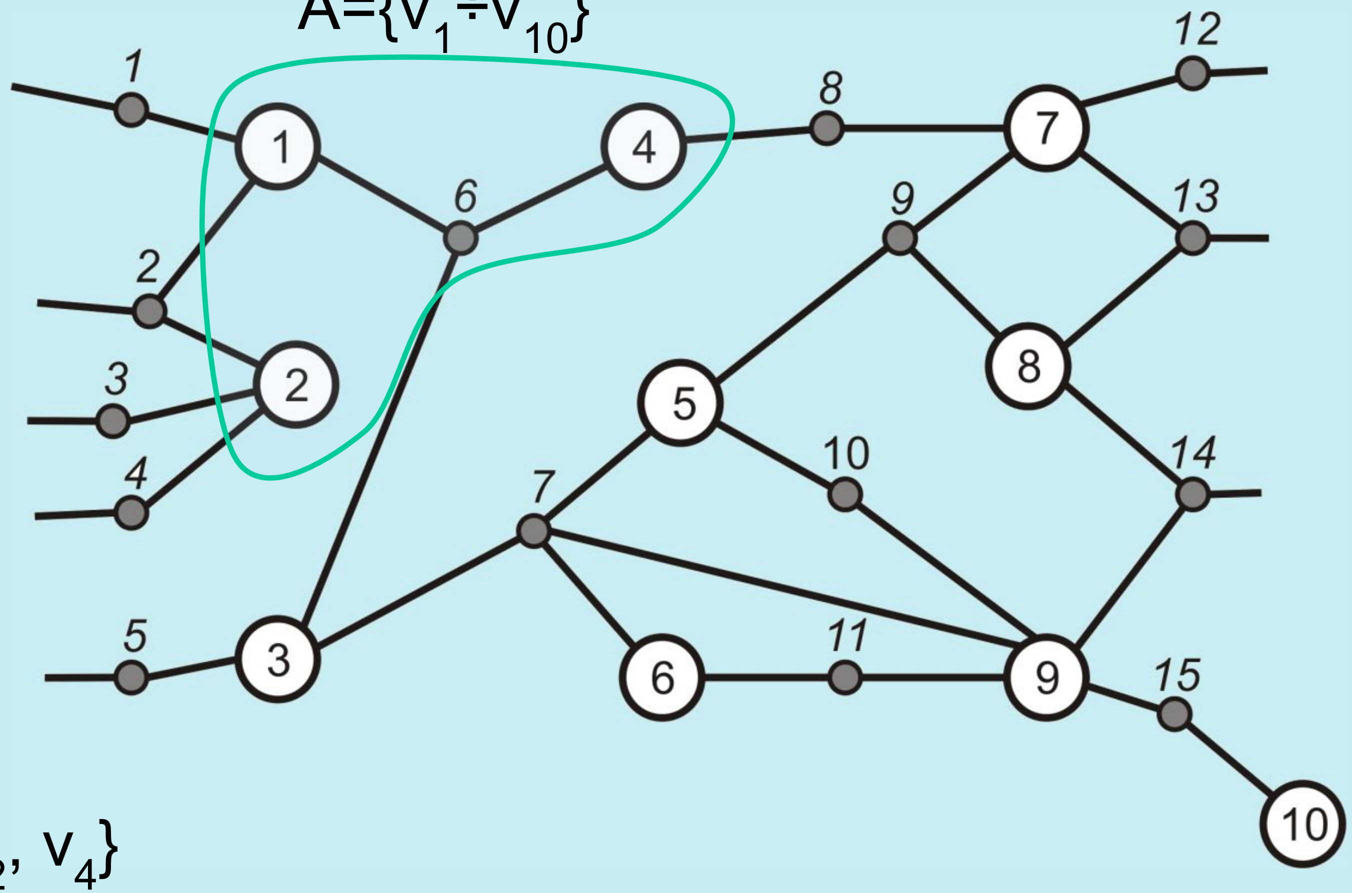
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

Выполнение п. 3

$con(a, B_1) \to \max$

$a=\{v_3, v_7\}$



$B_1=\{v_1, v_2, v_4\}$

#a	1	2	3	4	5	6	7	8	9	10
con(a,B$_1$)	x	x	1	x	0	0	1	0	0	0

Алгоритм Кодреса

Алгоритм

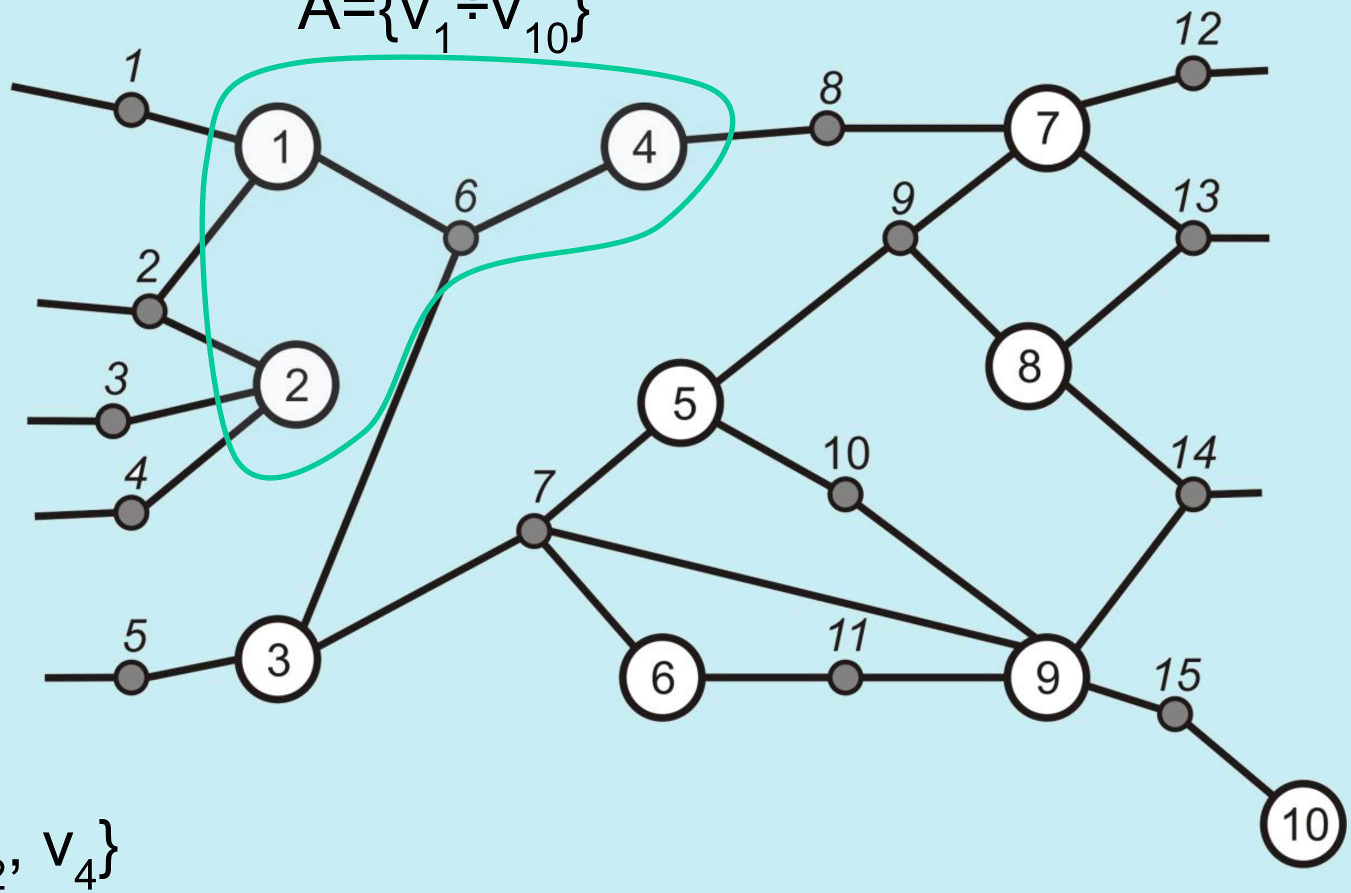
$A=\{v_1 \div v_{10}\}$

$S_{max} = 10$
$T_{max} = 8$

Выполнение п. 3

$dis(a, B_1) \rightarrow \min$

$a=v_3$



$B_1=\{v_1, v_2, v_4\}$

#a	1	2	3	4	5	6	7	8	9	10
con(a,B_1)	x	x	7	x	x	x	8	x	x	x

Алгоритм Кодреса

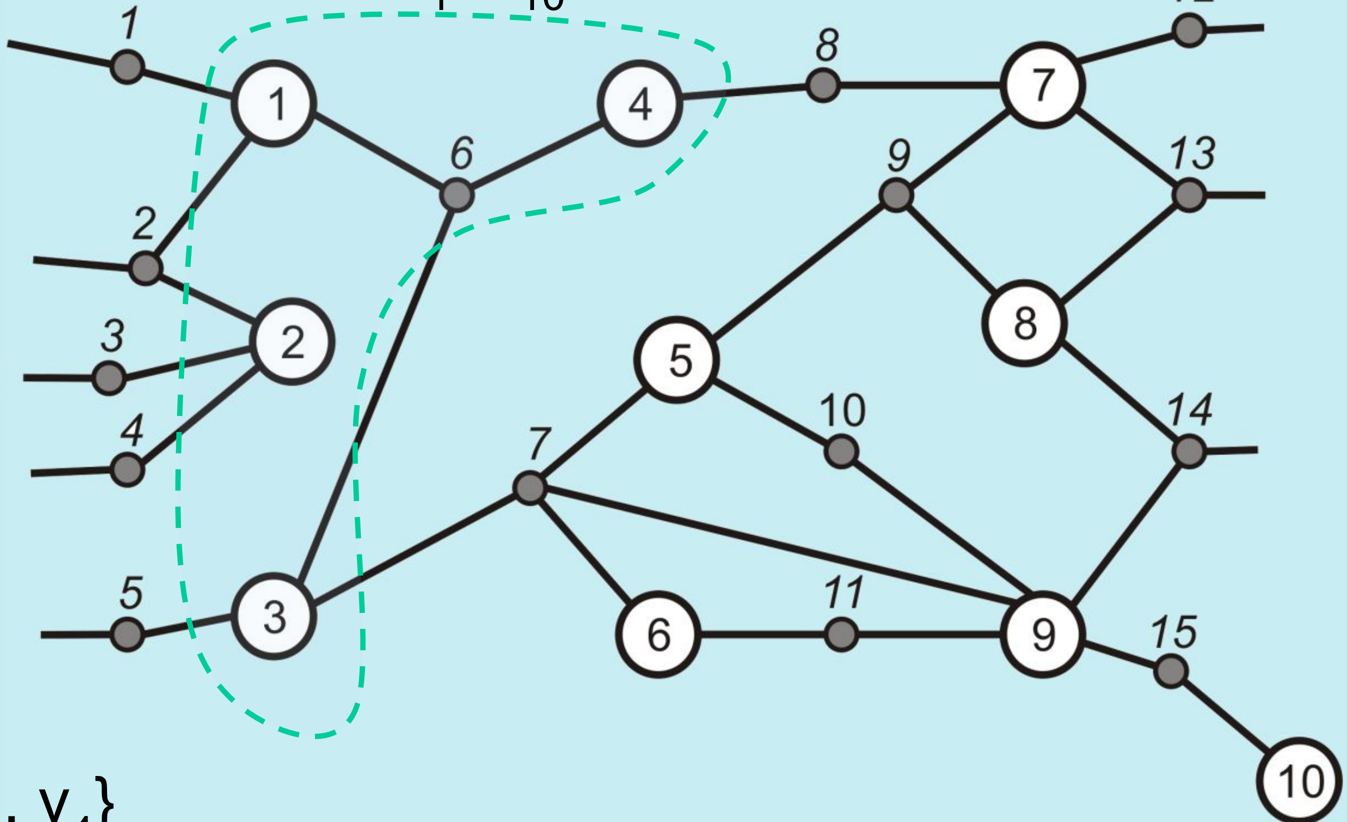
$S_{max} = 10$
$T_{max} = 8$

Выполнение п. 4

$$\begin{cases} S(B_1 \cup v_3) = 11 \\ \end{cases}$$

Алгоритм

$A = \{v_1 \div v_{10}\}$



$B_1 = \{v_1, v_2, v_4\}$

Алгоритм Кодреса

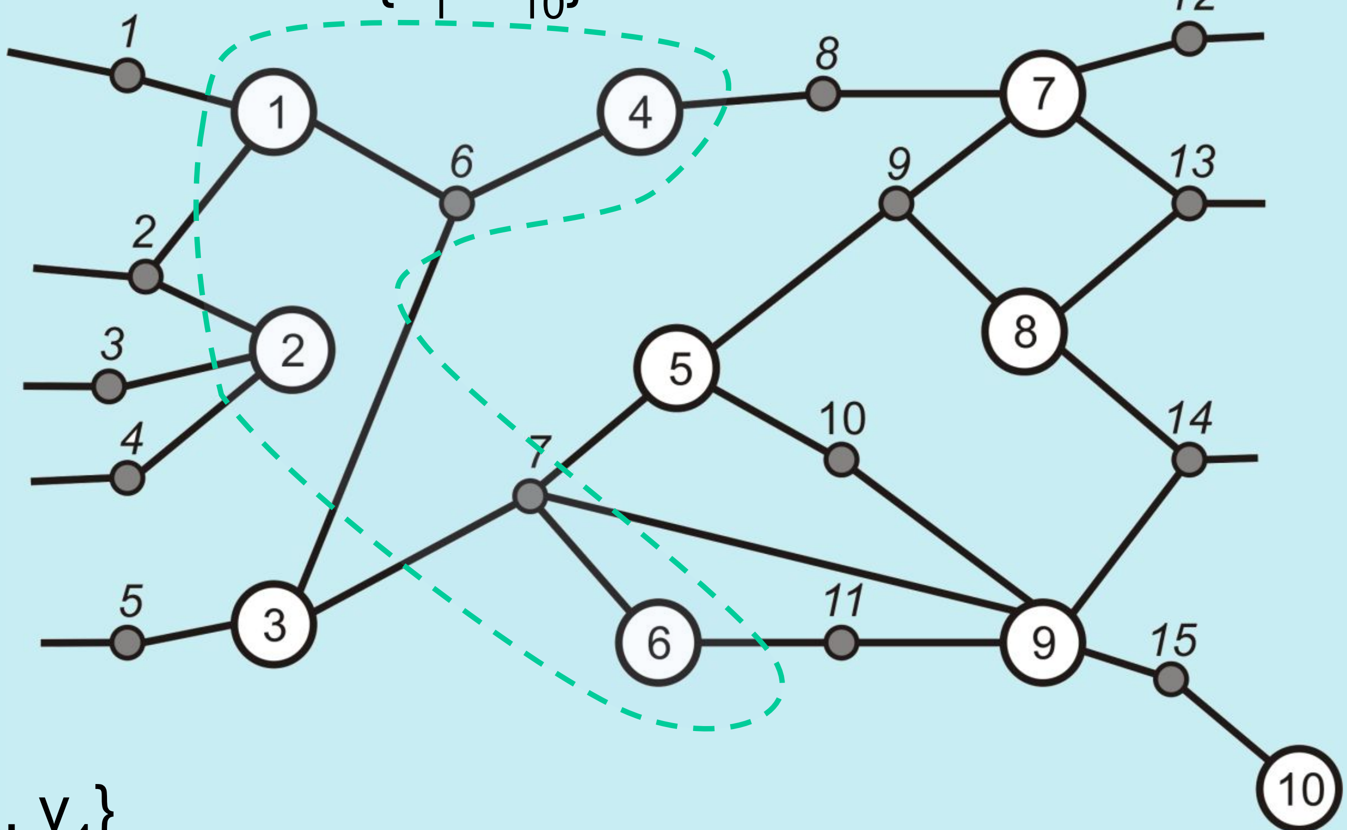
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_1 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_1 \cup v_6) = 10 \\ T(B_1 \cup v_6) = 8 \end{cases}$$



$B_1=\{v_1, v_2, v_4\}$

Алгоритм Кодреса

Алгоритм

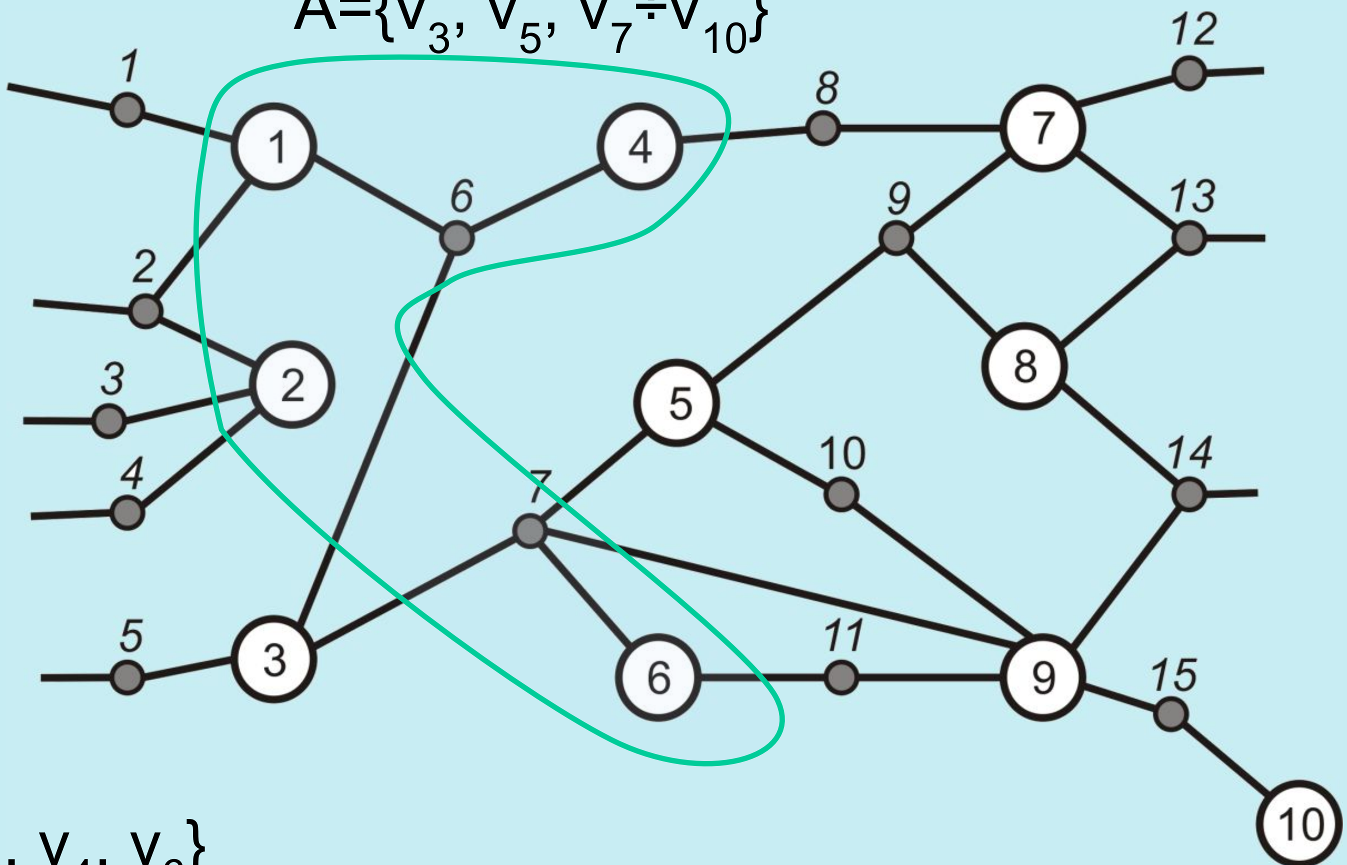
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_3, v_5, v_7 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_1 \cup v_6) = 10 \\ T(B_1 \cup v_6) = 8 \end{cases}$$

i=2



$B_1=\{v_1, v_2, v_4, v_6\}$

Алгоритм Кодреса

$S_{max} = 10$
$T_{max} = 8$

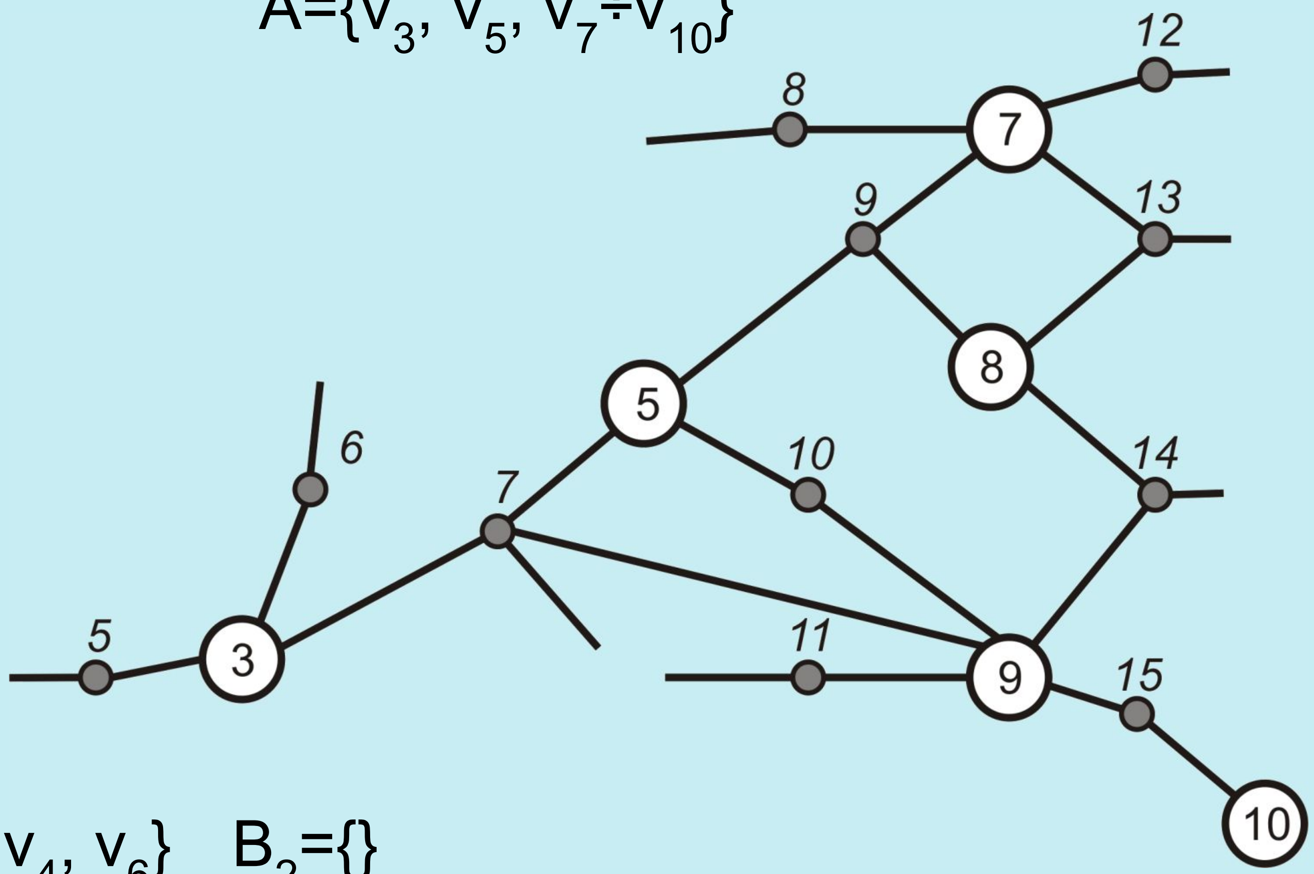
Выполнение п. 2

$con(a, A) \to \max$

$a=\{v_3, v_7, v_9\}$

Алгоритм

$A=\{v_3, v_5, v_7 \div v_{10}\}$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{\}$

#a	3	5	7	8	9	10
con(a,A)	3	1	3	2	3	0

Алгоритм Кодреса

Алгоритм

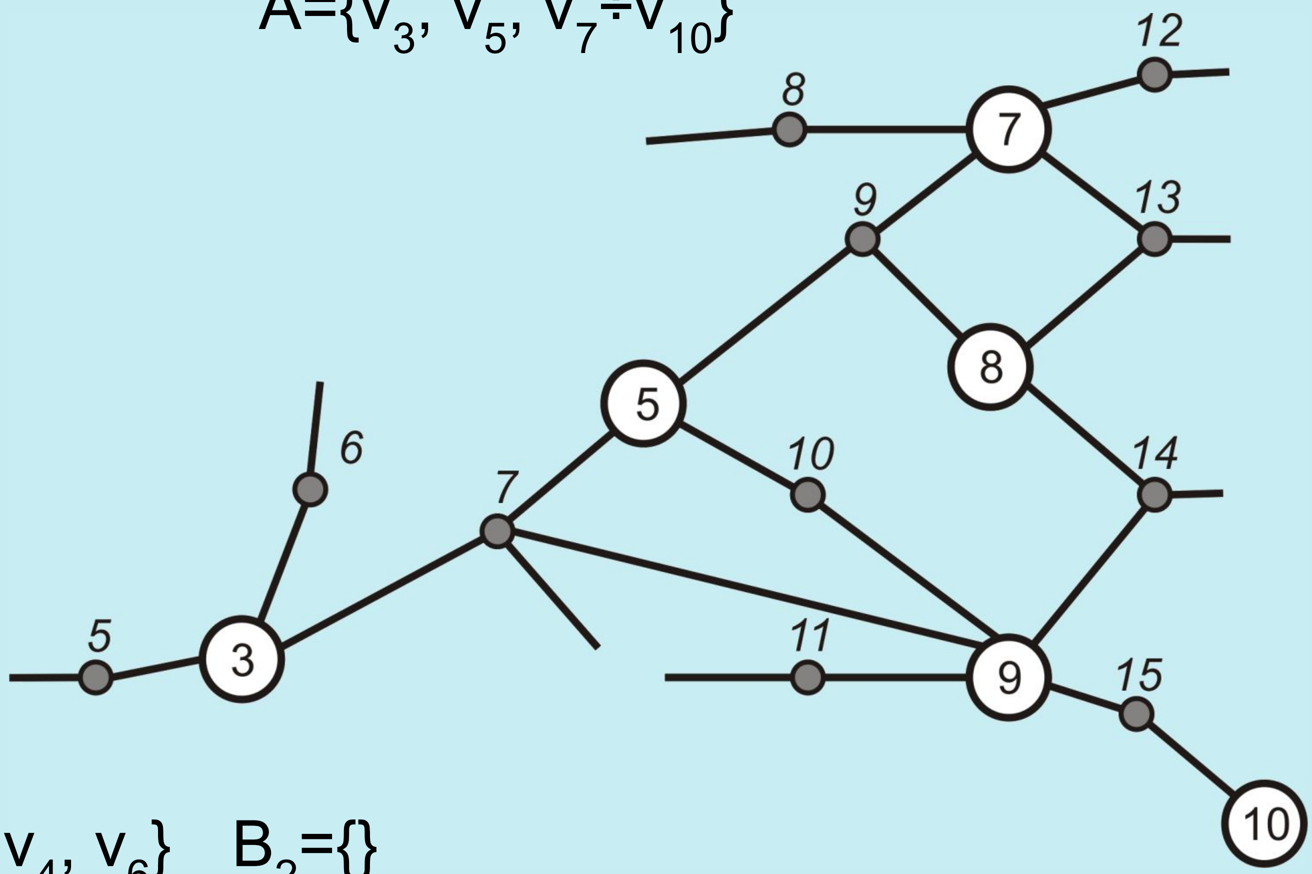
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_3, v_5, v_7 \div v_{10}\}$

Выполнение п. 2

$$con(a, A \setminus a) \rightarrow \min$$

$a=v_3$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{\}$

#a	3	5	7	8	9	10
con(a,A\a)	1	x	2	x	4	x

Алгоритм Кодреса

Алгоритм

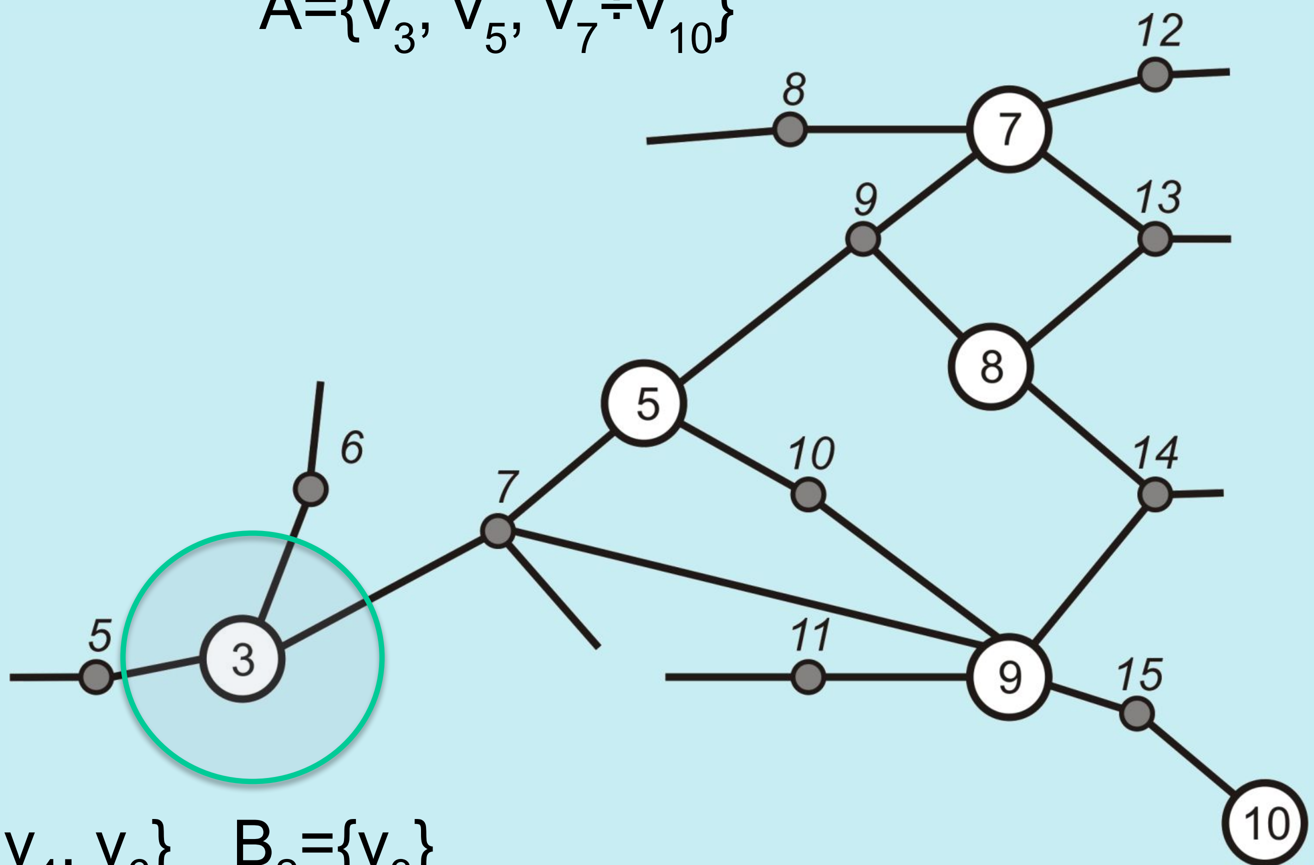
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_3, v_5, v_7 \div v_{10}\}$

Выполнение п. 3

$con(a, B_2) \to \max$

$a=\{v_5, v_9\}$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3\}$

#a	**3**	**5**	**7**	**8**	**9**	**10**
con(a,B$_2$)	x	1	0	0	1	0

Алгоритм Кодреса

$S_{max} = 10$
$T_{max} = 8$

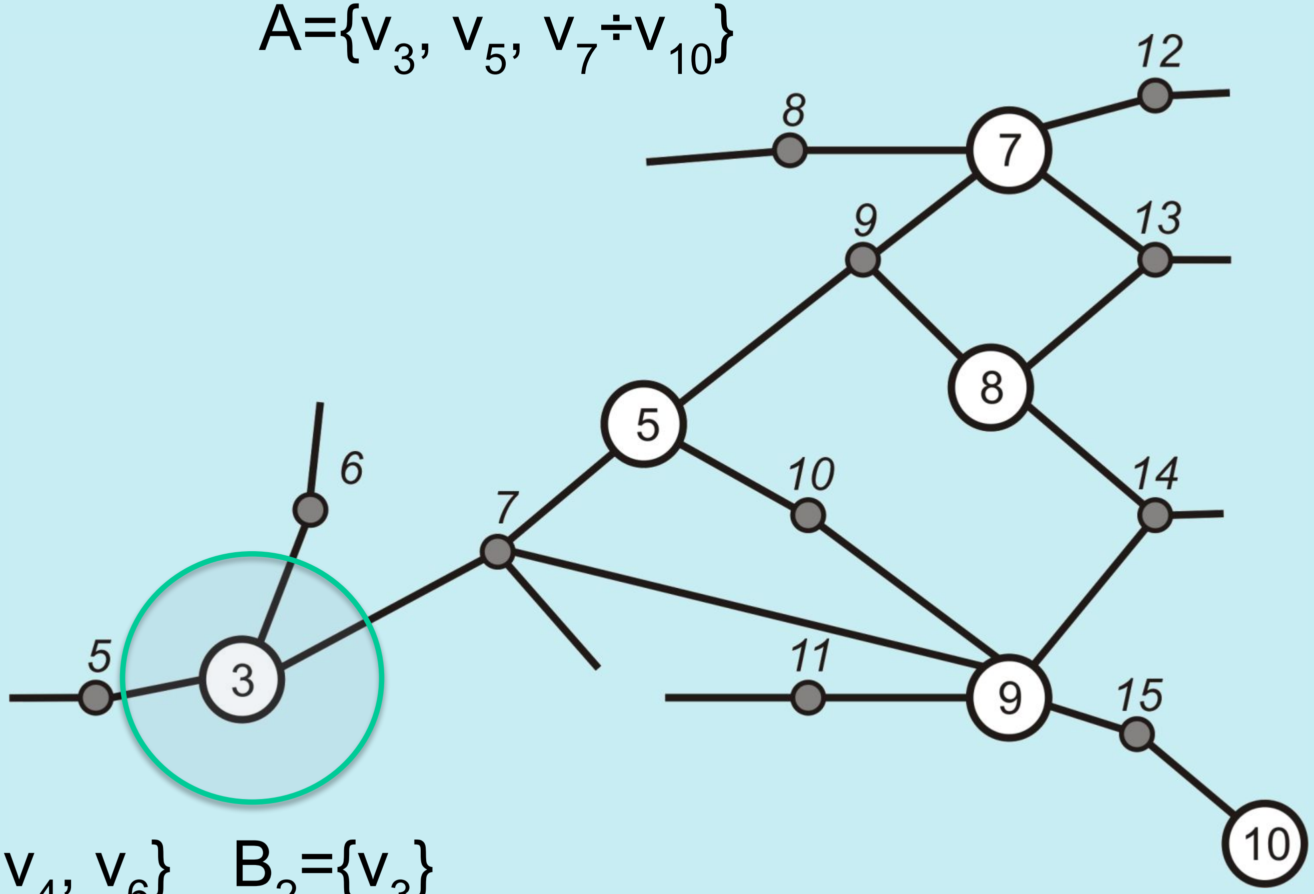
Выполнение п. 3

$dis(a, B_2) \to \min$

$a = v_5$

Алгоритм

$A = \{v_3, v_5, v_7 \div v_{10}\}$



$B_1 = \{v_1, v_2, v_4, v_6\}$ $B_2 = \{v_3\}$

#a	3	5	7	8	9	10
dis(a,B_2)	x	4	x	x	6	x

Алгоритм Кодреса

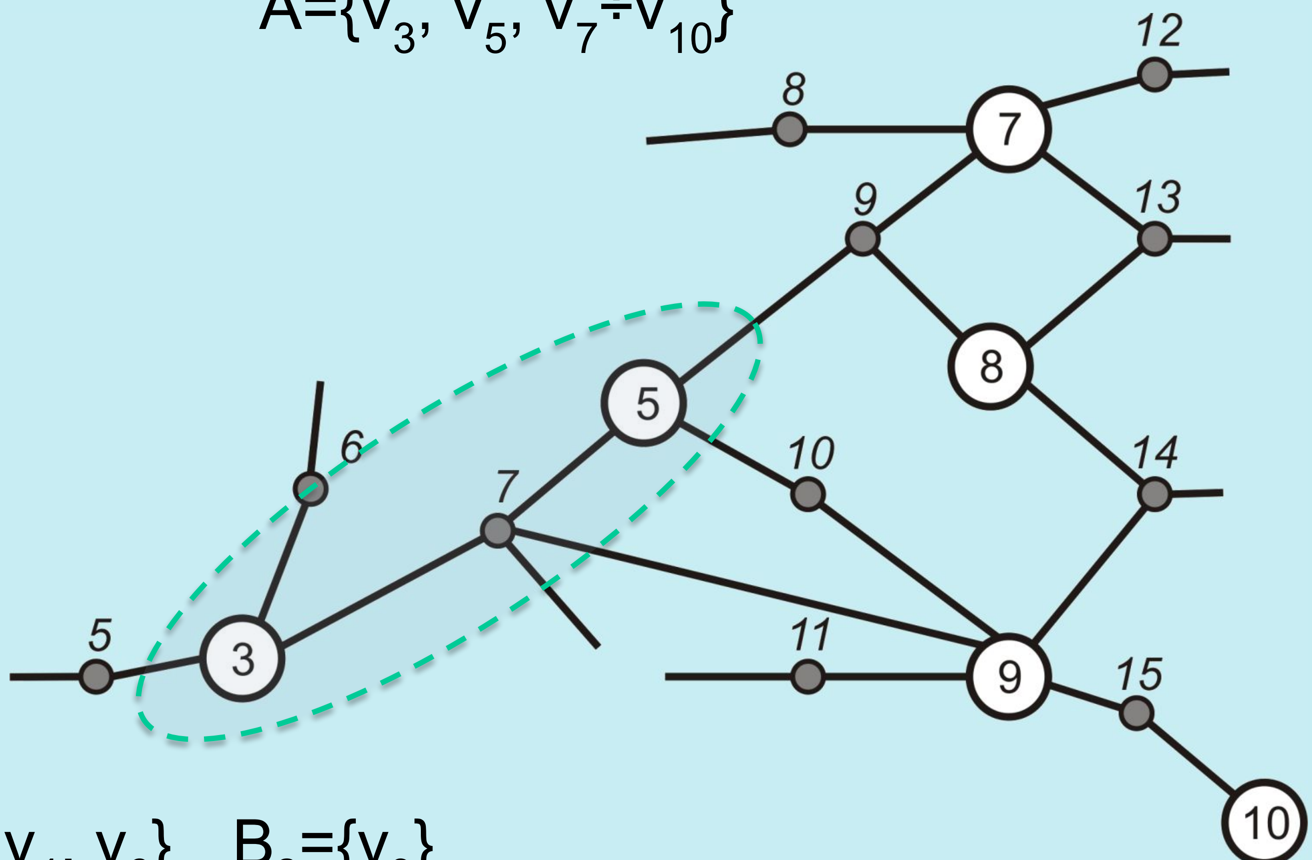
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_3, v_5, v_7 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_2 \cup v_5) = 6 \\ T(B_2 \cup v_5) = 5 \end{cases}$$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3\}$

Алгоритм Кодреса

Алгоритм

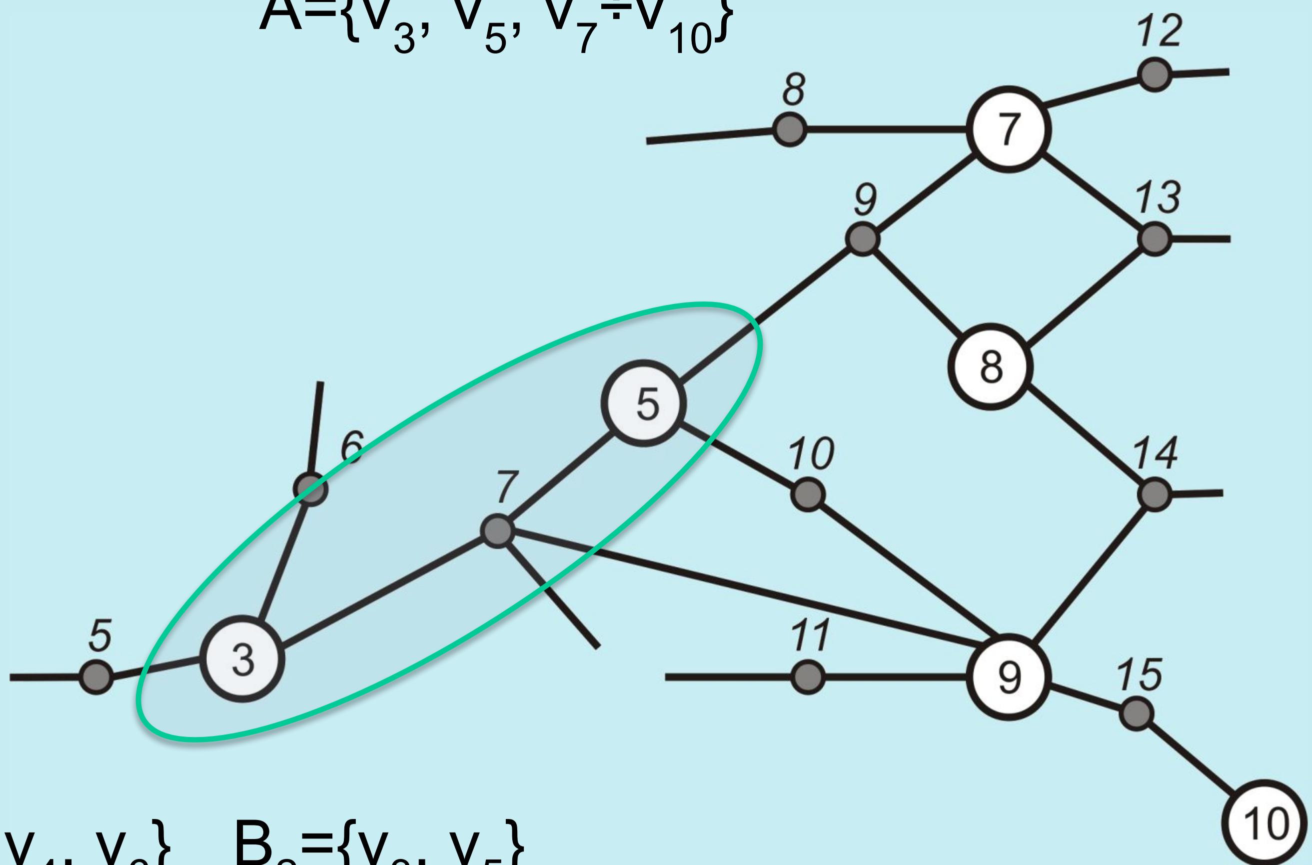
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_3, v_5, v_7 \div v_{10}\}$

Выполнение п. 3

$con(a, B_2) \to \max$

$a=v_9$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5\}$

#a	3	5	7	8	9	10
con(a,B_2)	x	x	1	1	2	0

Алгоритм Кодреса

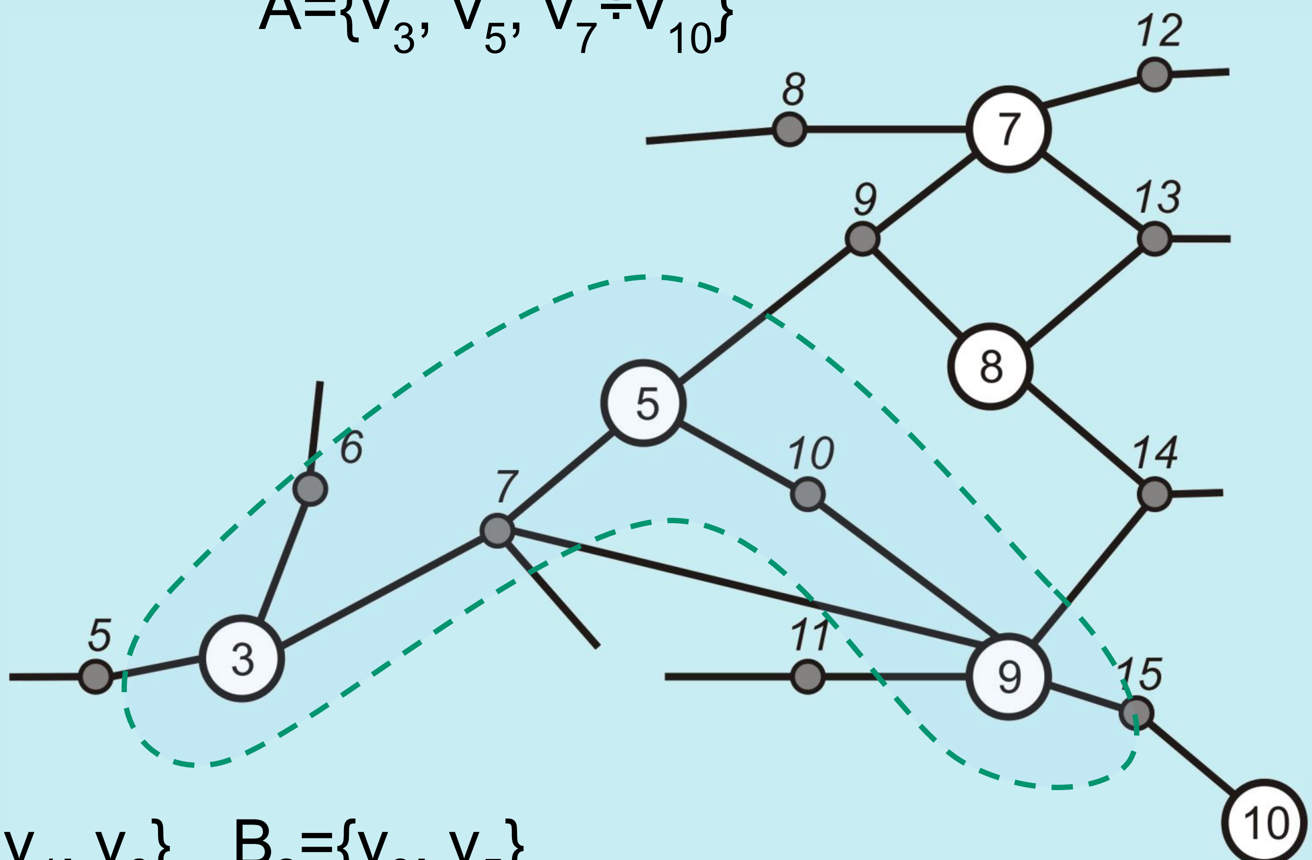
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A = \{v_3, v_5, v_7 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_2 \cup v_9) = 11 \\ \end{cases}$$



$B_1 = \{v_1, v_2, v_4, v_6\}$ $B_2 = \{v_3, v_5\}$

Алгоритм Кодреса

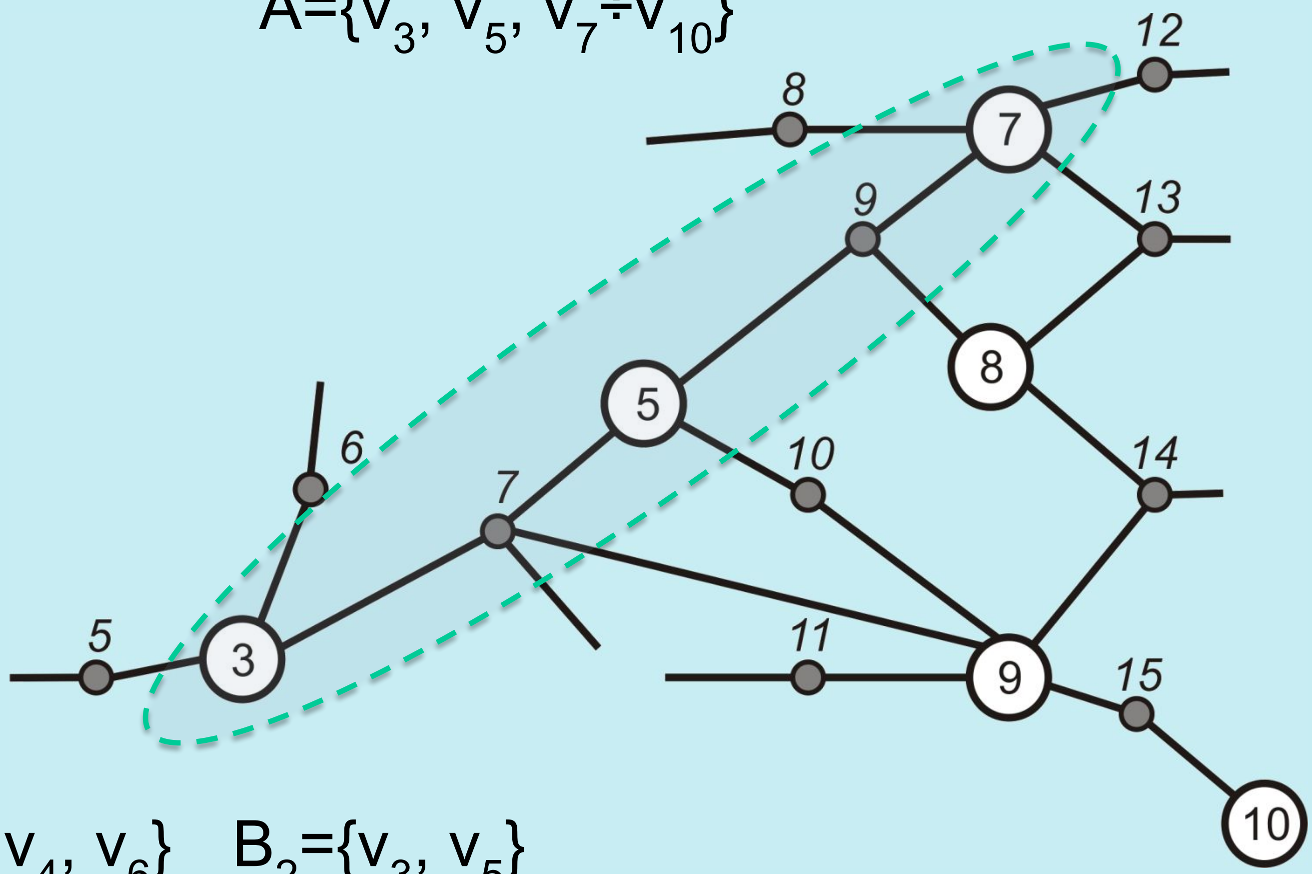
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_3, v_5, v_7 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_2 \cup v_7) = 10 \\ T(B_2 \cup v_7) = 8 \end{cases}$$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5\}$

Алгоритм Кодреса

Алгоритм

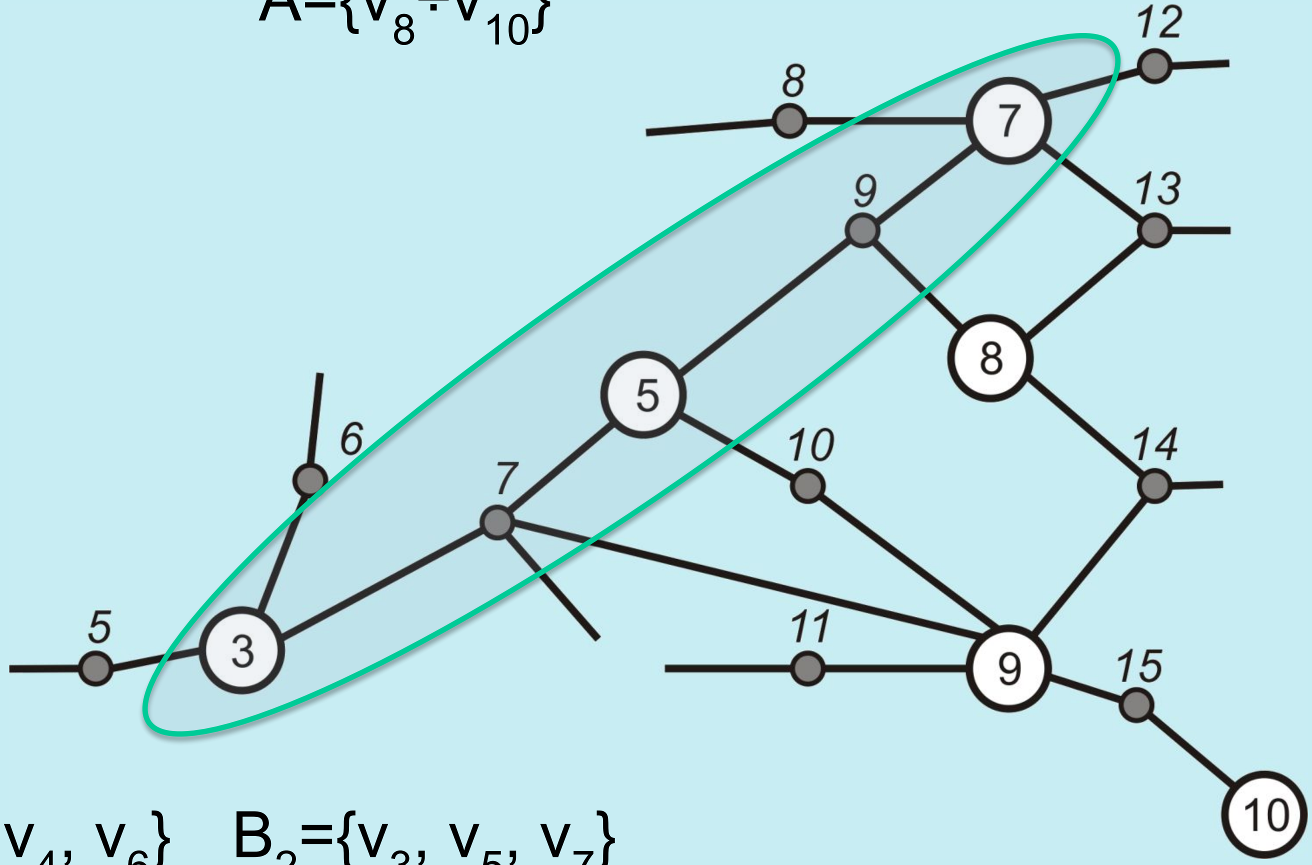
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_8 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_2 \cup v_7) = 10 & + \\ T(B_2 \cup v_7) = 8 & + \end{cases}$$

i=3



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5, v_7\}$

Алгоритм Кодреса

Алгоритм

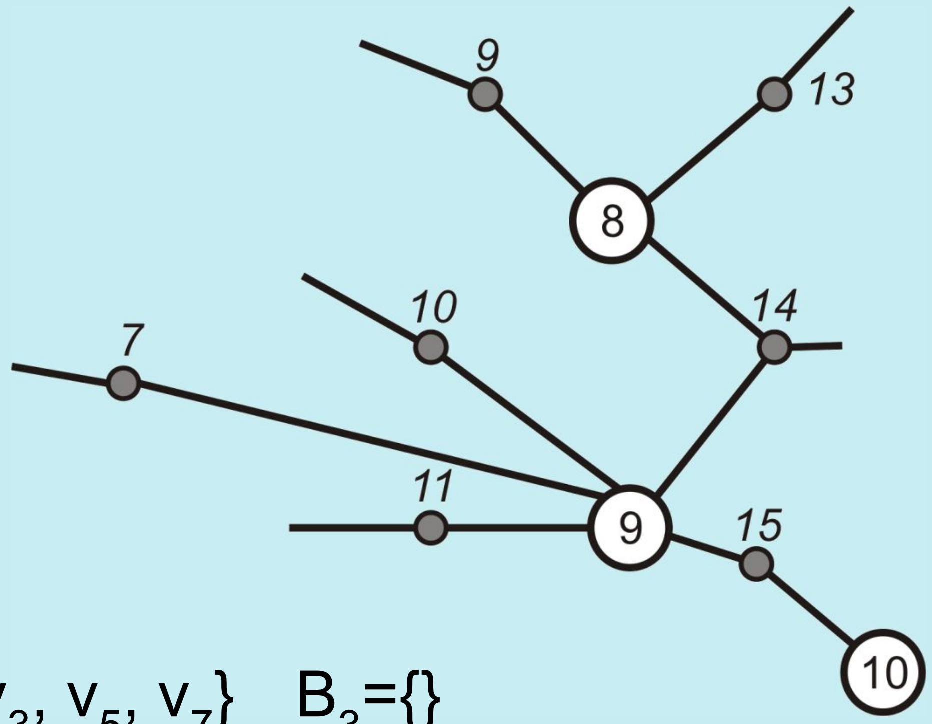
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_8 \div v_{10}\}$

Выполнение п. 2

$con(a, A) \rightarrow \max$

$a=v_9$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5, v_7\}$ $B_3=\{\}$

#a	**8**	**9**	**10**
con(a,A)	3	4	0

Алгоритм Кодреса

$S_{max} = 10$
$T_{max} = 8$

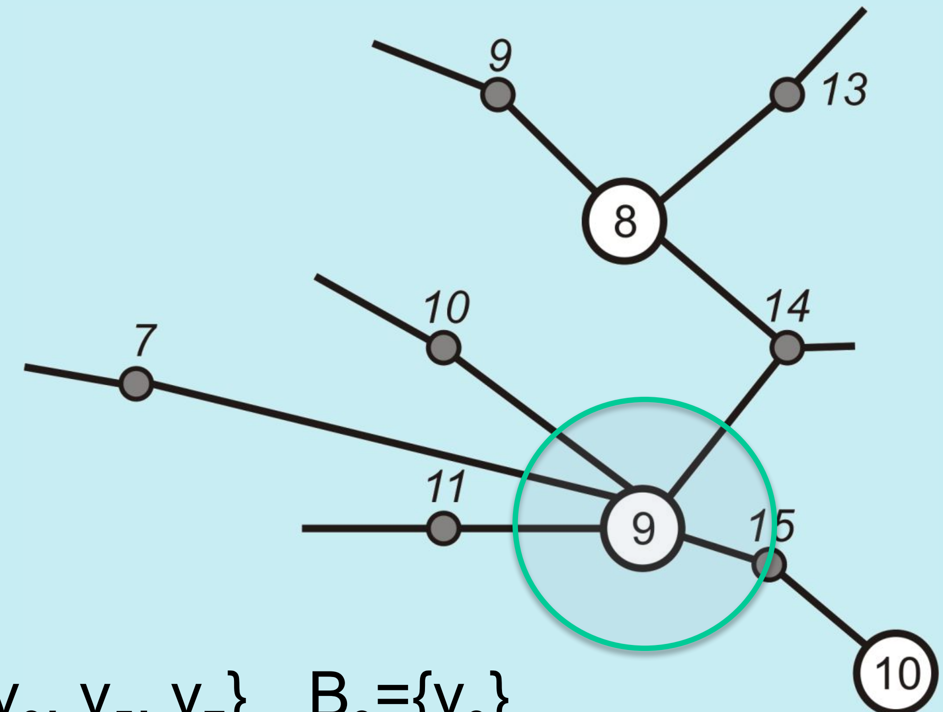
Алгоритм

$A=\{v_8 \div v_{10}\}$

Выполнение п. 3

$con(a, B_3) \rightarrow \max$

$a=\{v_8, v_{10}\}$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5, v_7\}$ $B_3=\{v_9\}$

#a	**8**	**9**	**10**
con(a,B_3)	1	x	1

Алгоритм Кодреса

Алгоритм

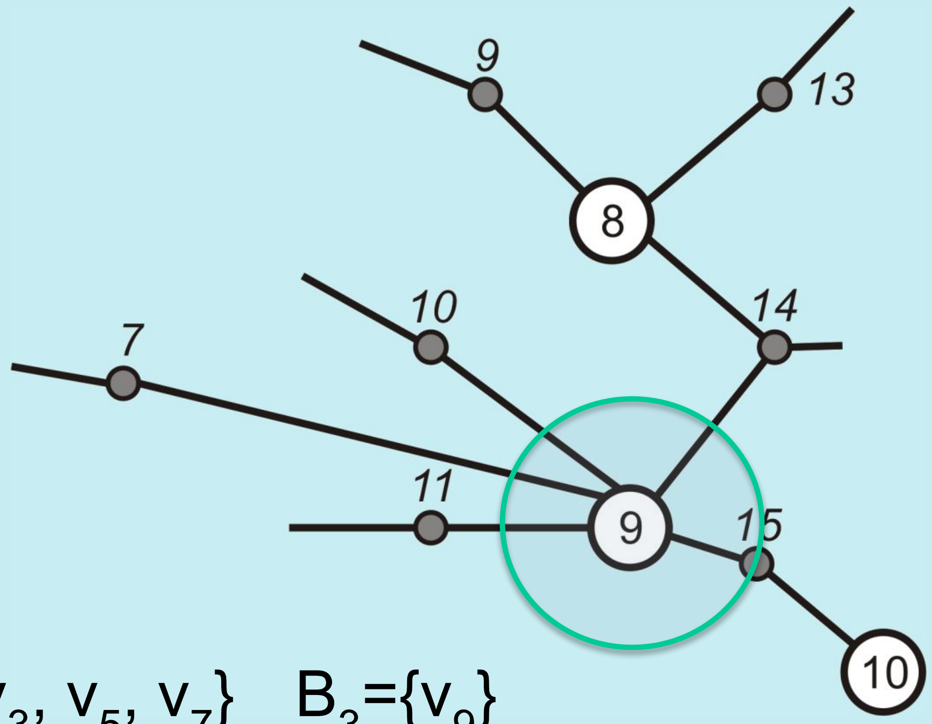
$S_{max} = 10$
$T_{max} = 8$

$A=\{v_8 \div v_{10}\}$

Выполнение п. 3

$$dis(a, B_3) \to \min$$

$a=v_{10}$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5, v_7\}$ $B_3=\{v_9\}$

#a	**8**	**9**	**10**
dis(a,B$_3$)	6	x	4

Алгоритм Кодреса

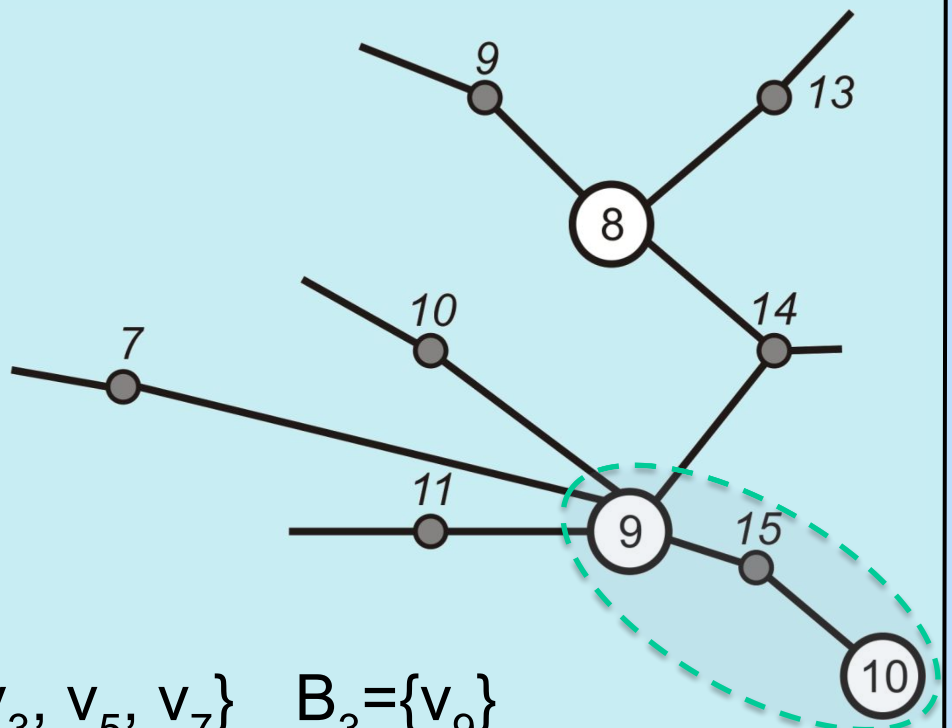
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_8 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_3 \cup v_{10}) = 6 \\ T(B_3 \cup v_{10}) = 4 \end{cases}$$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5, v_7\}$ $B_3=\{v_9\}$

Алгоритм Кодреса

Алгоритм

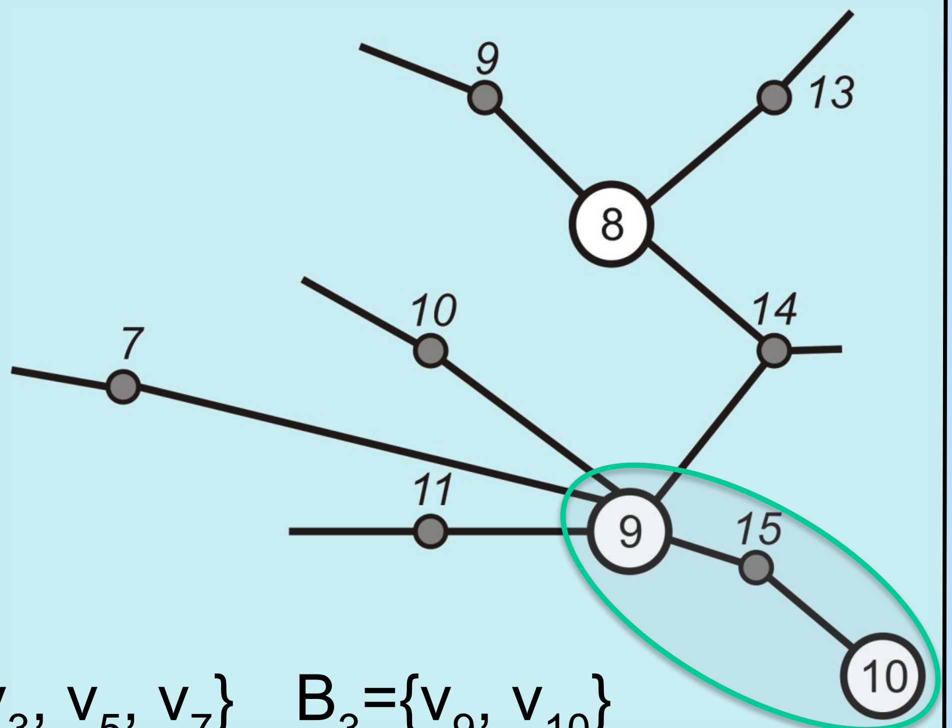
$A=\{v_8 \div v_{10}\}$

$S_{max} = 10$
$T_{max} = 8$

Выполнение п. 3

$con(a, B_3) \to \max$

$a=v_8$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5, v_7\}$ $B_3=\{v_9, v_{10}\}$

#a	8	9	10
con(a,B_3)	1	x	x

Алгоритм Кодреса

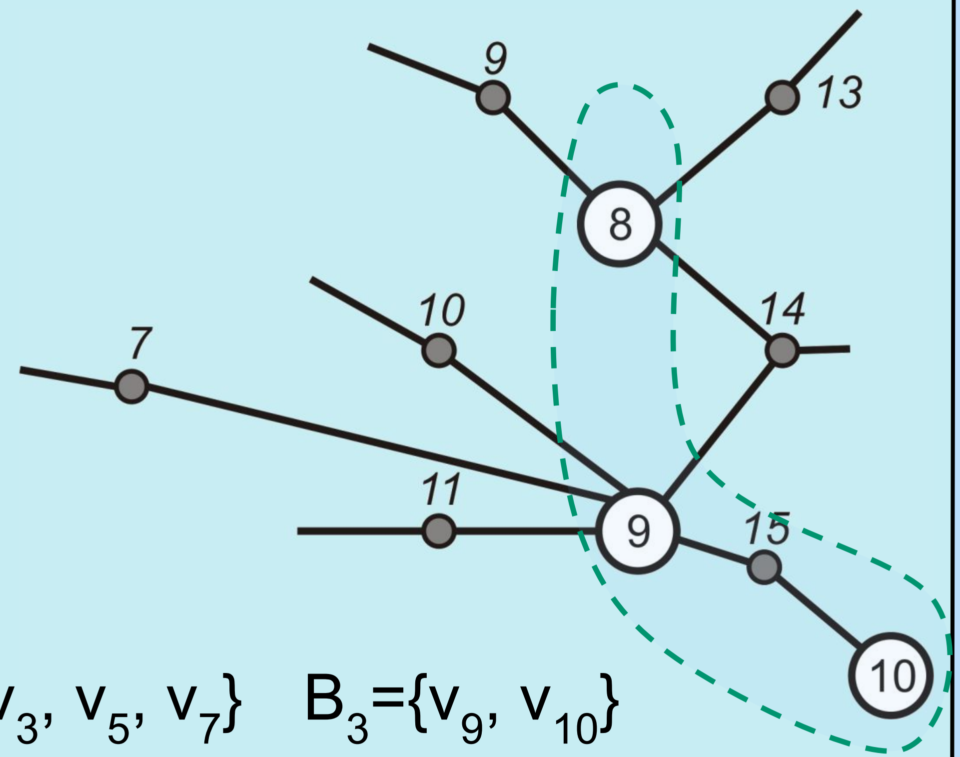
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_8 \div v_{10}\}$

Выполнение п. 4

$$\begin{cases} S(B_3 \cup v_8) = 6 \\ T(B_3 \cup v_8) = 9 \end{cases}$$



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5, v_7\}$ $B_3=\{v_9, v_{10}\}$

Алгоритм Кодреса

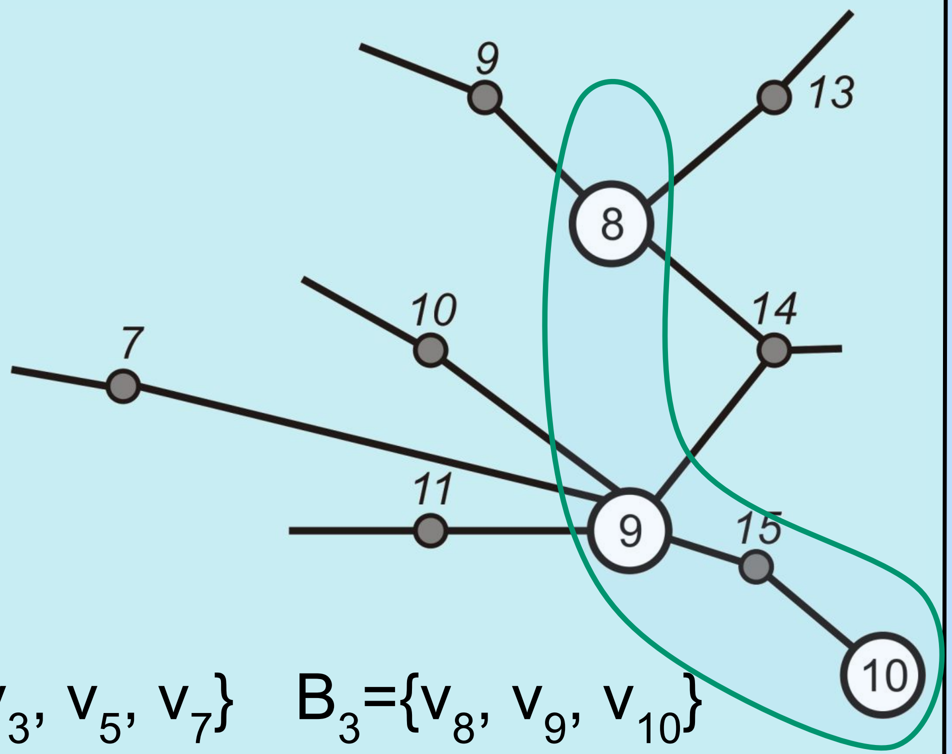
Алгоритм

$S_{max} = 10$
$T_{max} = 8$

$A=\{v_8 \div v_{10}\}$

Выполнение п. 3

Больше элементов нет, КОНЕЦ



$B_1=\{v_1, v_2, v_4, v_6\}$ $B_2=\{v_3, v_5, v_7\}$ $B_3=\{v_8, v_9, v_{10}\}$

Алгоритмы перемещения групп

- Принадлежат к итерационным алгоритмам
- Эти алгоритмы начинают работы с некоторого начального разбиения, обычно случайного
- В разбиении производятся локальные изменения (перестановки элементов) для уменьшения стоимости разреза
- Процесс повторяется, пока больше не будет улучшений

Алгоритм Кернигана-Лина

Входные данные:

Граф G=(V,E) с 2n вершинами, каждая вершина имеет одинаковый вес.

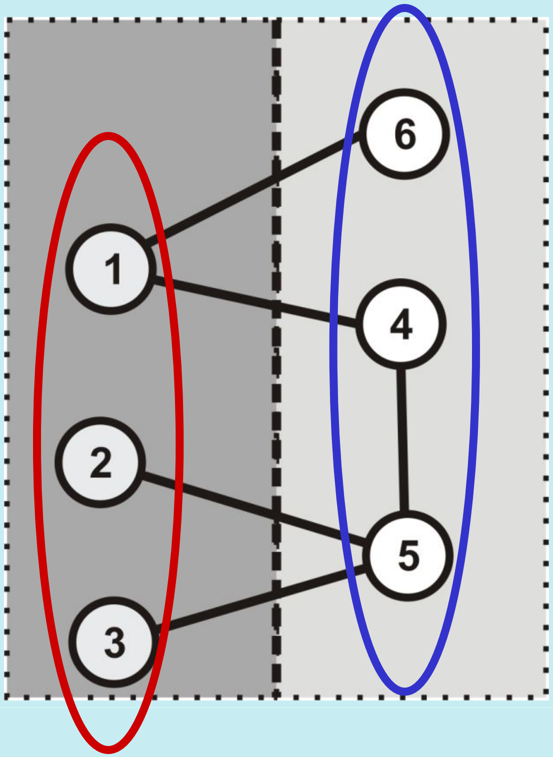
Задача:

Разделить граф на два непересекающихся подмножества A и B с минимальной стоимостью разреза,

$|A| = |B| = n$.

Пример: $n = 3$

Блок *A*



Блок *B*

Алгоритм Кернигана-Лина

D(*v*) – цена перемещения *v*

$D(v) = E(v) – I(v)$

E(*v*) – число ребер v, пересекающих разрез

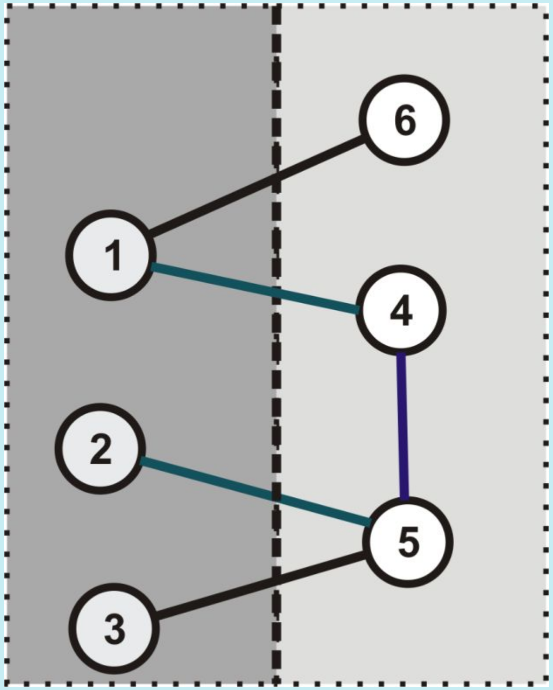
I(*v*) - число ребер с вершинами внутри блока

***D* > 0**

Перемещать вершину выгодно

***D* < 0**

Цена разреза увеличится



$D(4) = E(4) – I(4) = 1\text{-}1 = 0$

Алгоритм Кернигана-Лина

D(*v*) – цена перемещения *v*

D(*v*) = *E*(*v*) – *I*(*v*)

E(*v*) – число ребер v, пересекающих разрез

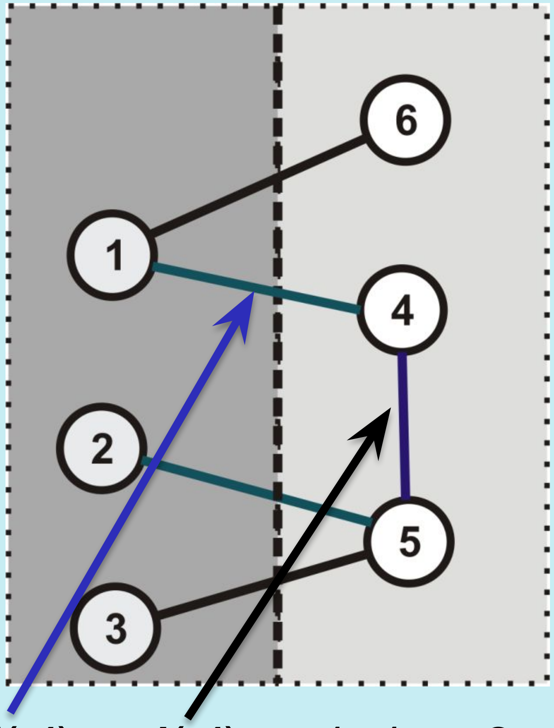
I(*v*) - число ребер с вершинами внутри блока

***D* > 0**

Перемещать вершину выгодно

***D* < 0**

Цена разреза увеличится



D(4) = *E*(4) – *I*(4) = 1-1 = 0

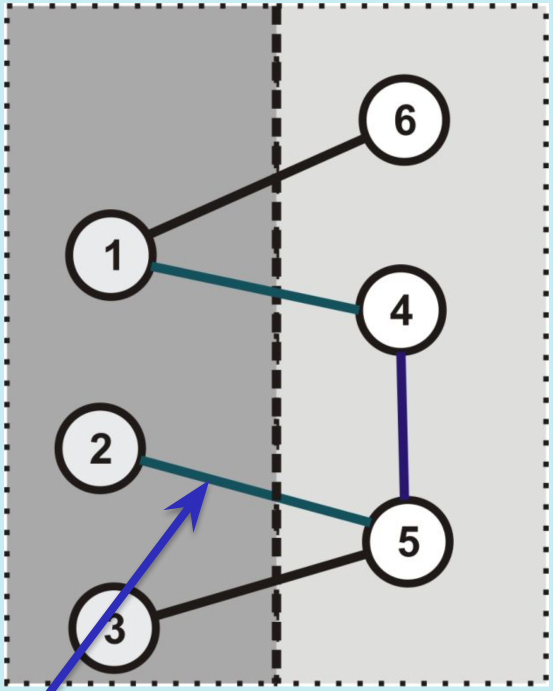
Алгоритм Кернигана-Лина

$D(v)$ – цена перемещения v

$D(v) = E(v) – I(v)$

$E(v)$ – число ребер v, пересекающих разрез

$I(v)$ - число ребер с вершинами внутри блока



$D > 0$

Перемещать вершину выгодно

$D < 0$

Цена разреза увеличится

$D(2) = E(2) – I(2) = 1\text{-}0 = 1$

Алгоритм Кернигана-Лина

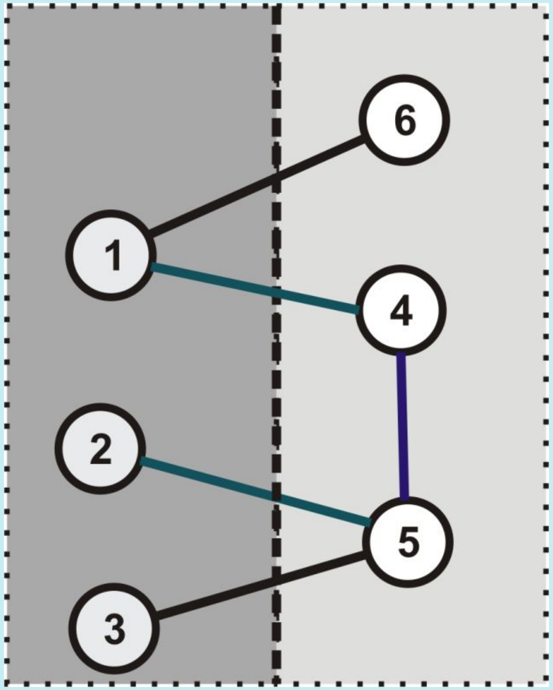
Перестановка пары вершин

Стоимость перестановки *a* и *b* местами:

$\Delta g = D(a) + D(b) - 2^* c(a,b)$,

D(*a*), *D*(*b*) – стоимость перестановки a, b

c(*a*,*b*) – связанность a и b



- Если ребра (a,b) не существует, то *c*(*a*,*b*) = 0
- Δg показывает пользу от перестановки вершин между собой
- Чем больше Δg, тем меньше будет стоимость разреза

$\Delta g(2,4) = 0 + 1 - 2^* 0 = 1$

Алгоритм Кернигана-Лина

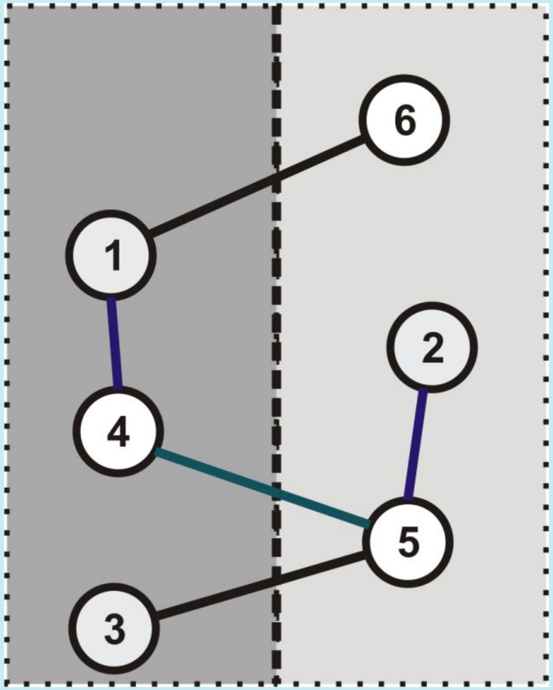
Перестановка пары вершин

Стоимость перестановки *a* и *b* местами:

$\Delta g = D(a) + D(b) - 2^* c(a,b)$,

$D(a)$, $D(b)$ – стоимость перестановки a, b

$c(a,b)$ – связанность a и b



- Если ребра (a,b) не существует, то $c(a,b) = 0$
- Δg показывает пользу от перестановки вершин между собой
- Чем больше Δg, тем меньше будет стоимость разреза

$\Delta g(2,4) = 0 + 1 - 2^* 0 = 1$

Алгоритм Кернигана-Лина

Перестановка пары вершин

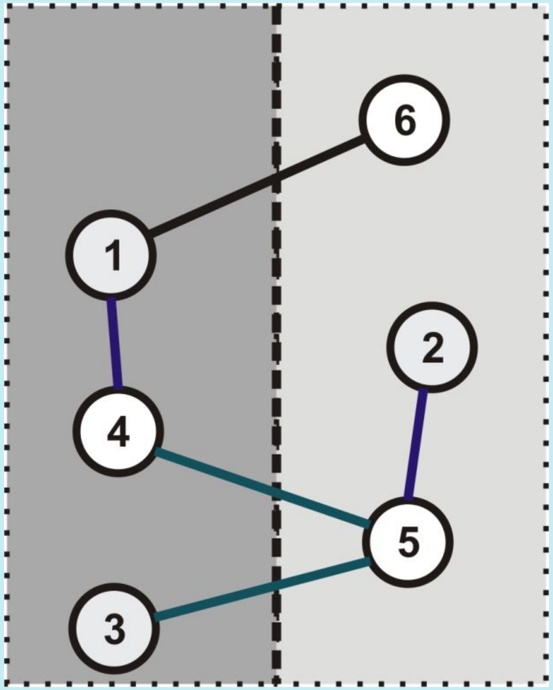
Стоимость перестановки *a* и *b* местами:

$\Delta g = D(a) + D(b) - 2 * c(a,b)$,

$D(a)$, $D(b)$ – стоимость перестановки a, b

$c(a,b)$ – связанность a и b

- Если ребра (a,b) не существует, то $c(a,b) = 0$
- Δg показывает пользу от перестановки вершин между собой
- Чем больше Δg, тем меньше будет стоимость разреза



$\Delta g(4,5) = 0 + 1 - 2 * 1 = -1$

Алгоритм Кернигана-Лина

Перестановка пары вершин

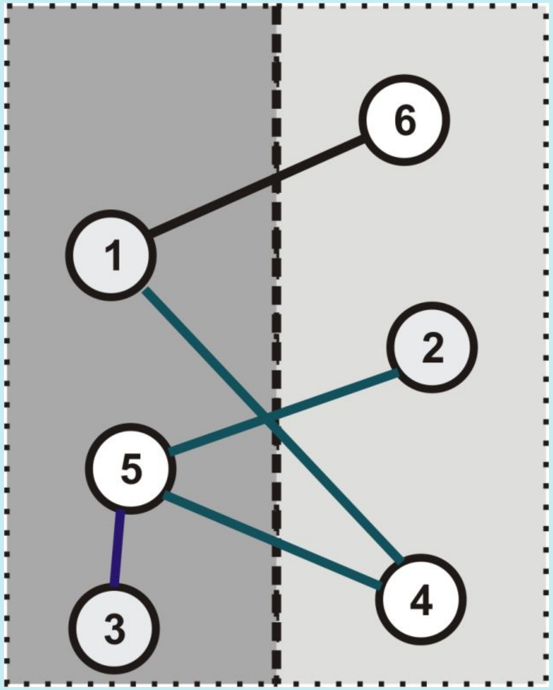
Стоимость перестановки *a* и *b* местами:

$\Delta g = D(a) + D(b) - 2 * c(a,b)$,

$D(a)$, $D(b)$ – стоимость перестановки a, b

$c(a,b)$ – связанность a и b

- Если ребра (a,b) не существует, то $c(a,b) = 0$
- Δg показывает пользу от перестановки вершин между собой
- Чем больше Δg, тем меньше будет стоимость разреза



$\Delta g(4,5) = 0 + 1 - 2 * 1 = -1$

Алгоритм Кернигана-Лина

Стоимость последовательности перестановок G_m

- За один проход алгоритма необходимо максимизировать G_m
- G_m обо В конце прохода из свей последовательности шагов выбирается такая подпоследовательность длиной m, что $G_m \rightarrow max$
- За выбранные m перестановок стоимость разреза уменьшится лучше всего

$$G_m = \sum_{i=1}^{m} \Delta g_i$$

Алгоритм Кернигана-Лина

Алгоритм

1. Разбить V на A и B, |A|=|B|, A∩B=V
2. i=1

   Вычислить D(v) для всех v∈V
3. Пока есть незафиксированные вершины

   Выбрать (a,b) с $\Delta g \rightarrow max$

   Поменять a и b местами

   Зафиксировать a и b

   Пересчитать D() для всех незафиксированных вершин, связанных с a и b

   i=i+1
4. Найти подпоследовательность 1,…,m; $1 \le m \le i$ с $G_m \rightarrow max$

   Если $G_m > 0$

   Воспроизвести перестановки 1,…,m

   Перейти на п. 2

   Иначе КОНЕЦ

Алгоритм Кернигана-Лина

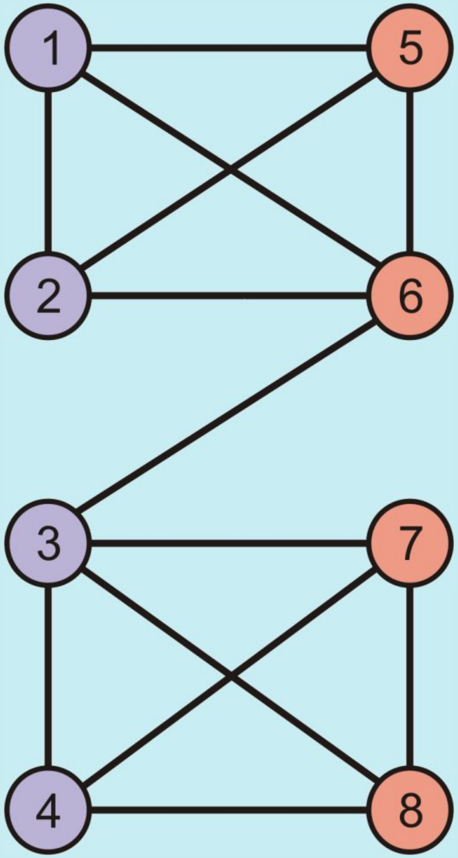
Пример

Случайно разобьем V на A и B

A={1,2,3,4}

B={5,6,7,8}

Вычислить D(v) для всех вершин



Алгоритм Кернигана-Лина

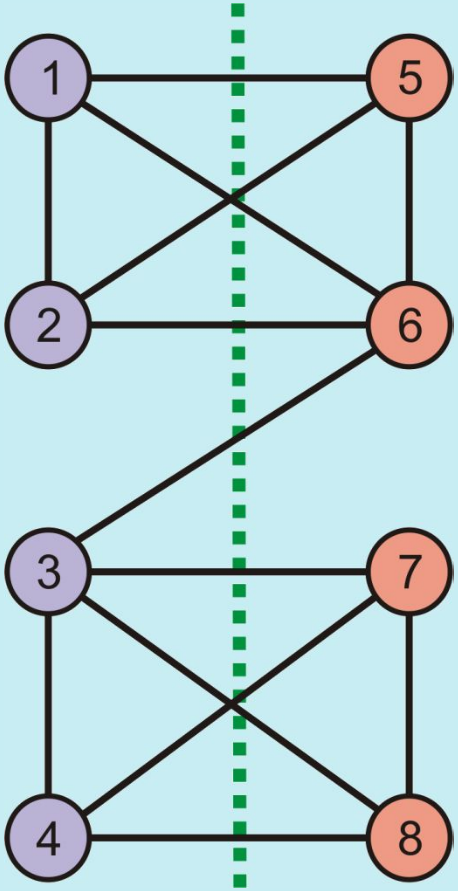
Пример

Случайно разобьем V на A и B

A={1,2,3,4}

B={5,6,7,8}

Вычислить D(v) для всех вершин



Цена разреза: 6

#v	1	2	3	4	5	6	7	8
D(v)	1	1	2	1	1	2	1	1

Алгоритм Кернигана-Лина

Пример

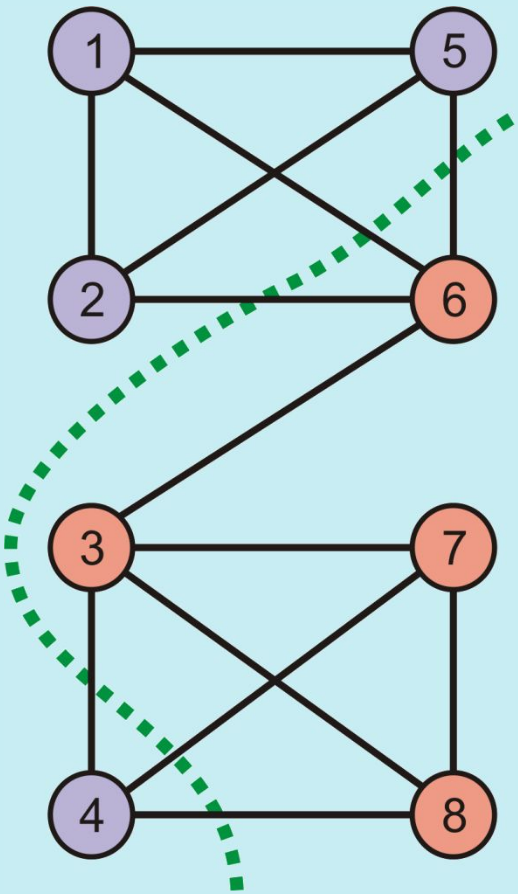
Для пары **(3,5):**

$\Delta g_1 = 2+1-0 = 3$

Переместить (3,5)

$G_1 = \Delta g_1 = 3$

Зафиксировать (3,5)



i	(x,y)	Δg_i	G_m
1	**(3,5)**	3	3

Цена разреза: 6

#v	1	2	3	4	5	6	7	8
D(v)	1	1	**2**	1	**1**	2	1	1

Алгоритм Кернигана-Лина

Пример

Обновление D(v)

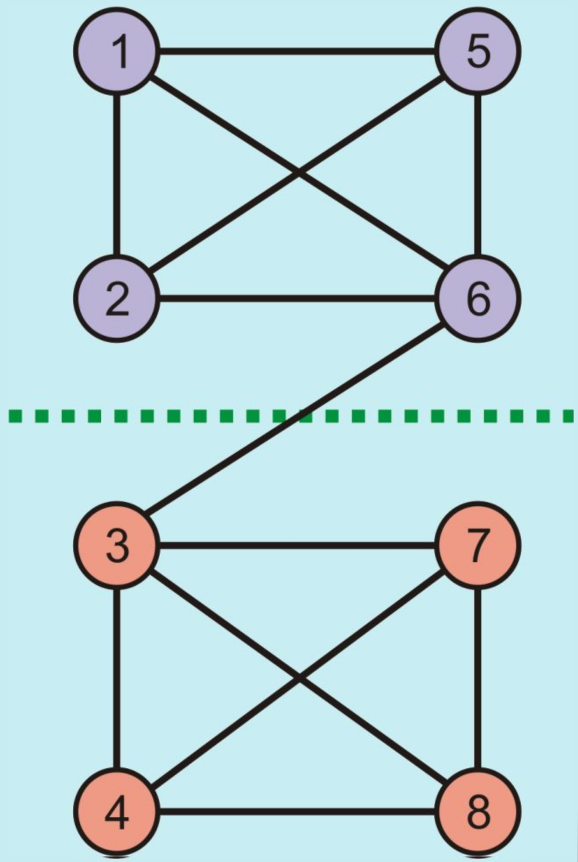
Для пары **(4,6):**

$\Delta g_2 = 3+2-0 = 5$

Переместить (4,6)

$G_2 = G_1 + \Delta g_2 = 8$

Зафиксировать (4,6)



i	(x,y)	Δg_i	G_m
1	**(3,5)**	3	3
2	**(4,6)**	5	8

Цена разреза: 1

#v	1	2	3	4	5	6	7	8
D(v)	-1	-1	x	3	x	2	-1	-1

Алгоритм Кернигана-Лина

Пример

Обновление D(v)

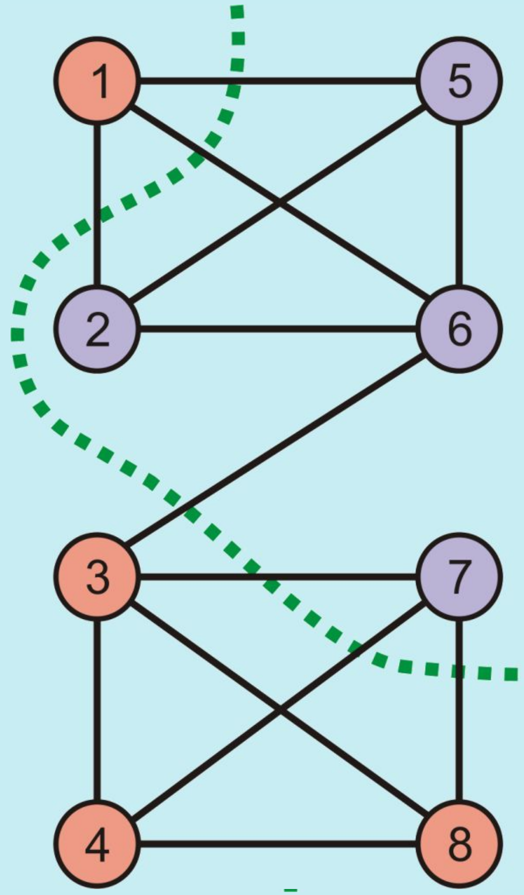
Для пары **(1,7):**

Δg_3 = -3-3-0 = -6

Переместить (1,7)

$G_3 = G_2 + \Delta g_3 = 2$

Зафиксировать (1,7)



i	(x,y)	Δg_i	G_m
1	**(3,5)**	3	3
2	**(4,6)**	5	8
3	**(1,7)**	-6	2

Цена разреза: 7

#v	1	2	3	4	5	6	7	8
D(v)	-3	-3	x	x	x	x	-3	-3

Алгоритм Кернигана-Лина

Пример

Обновление D(v)

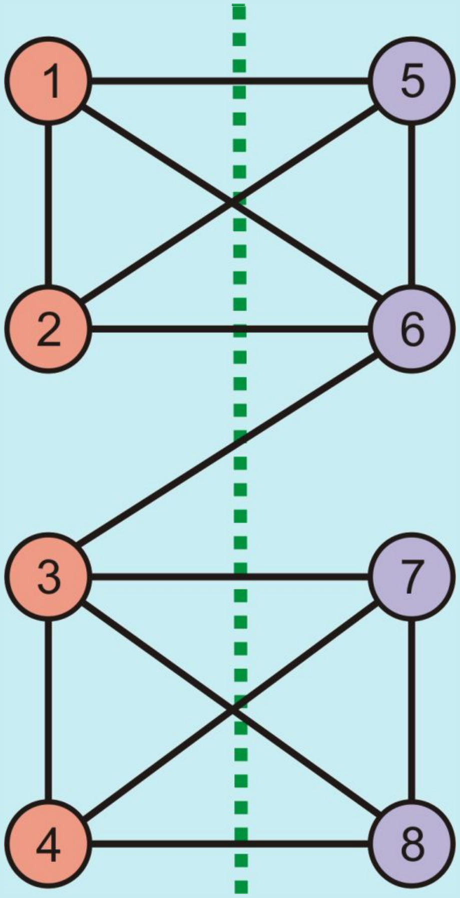
Для пары **(2,8):**

$\Delta g_4 = -1-1-0 = -2$

Переместить (2,8)

$G_4 = G_3 + \Delta g_4 = 0$

Зафиксировать (2,8)



i	(x,y)	Δg_i	G_m
1	**(3,5)**	3	3
2	**(4,6)**	5	8
3	**(1,7)**	-6	2
4	**(2,8)**	-2	0

Цена разреза: 9

#v	1	2	3	4	5	6	7	8
D(v)	x	-1	x	x	x	x	x	-1

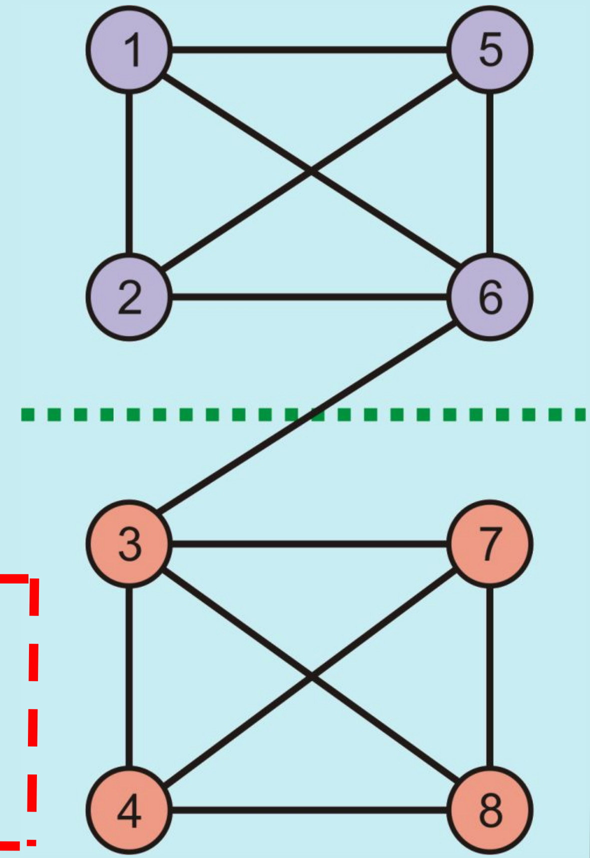
Алгоритм Кернигана-Лина

Пример

Выбрать М с $G_m \rightarrow max$

m=**2**

$G_2 = 8$

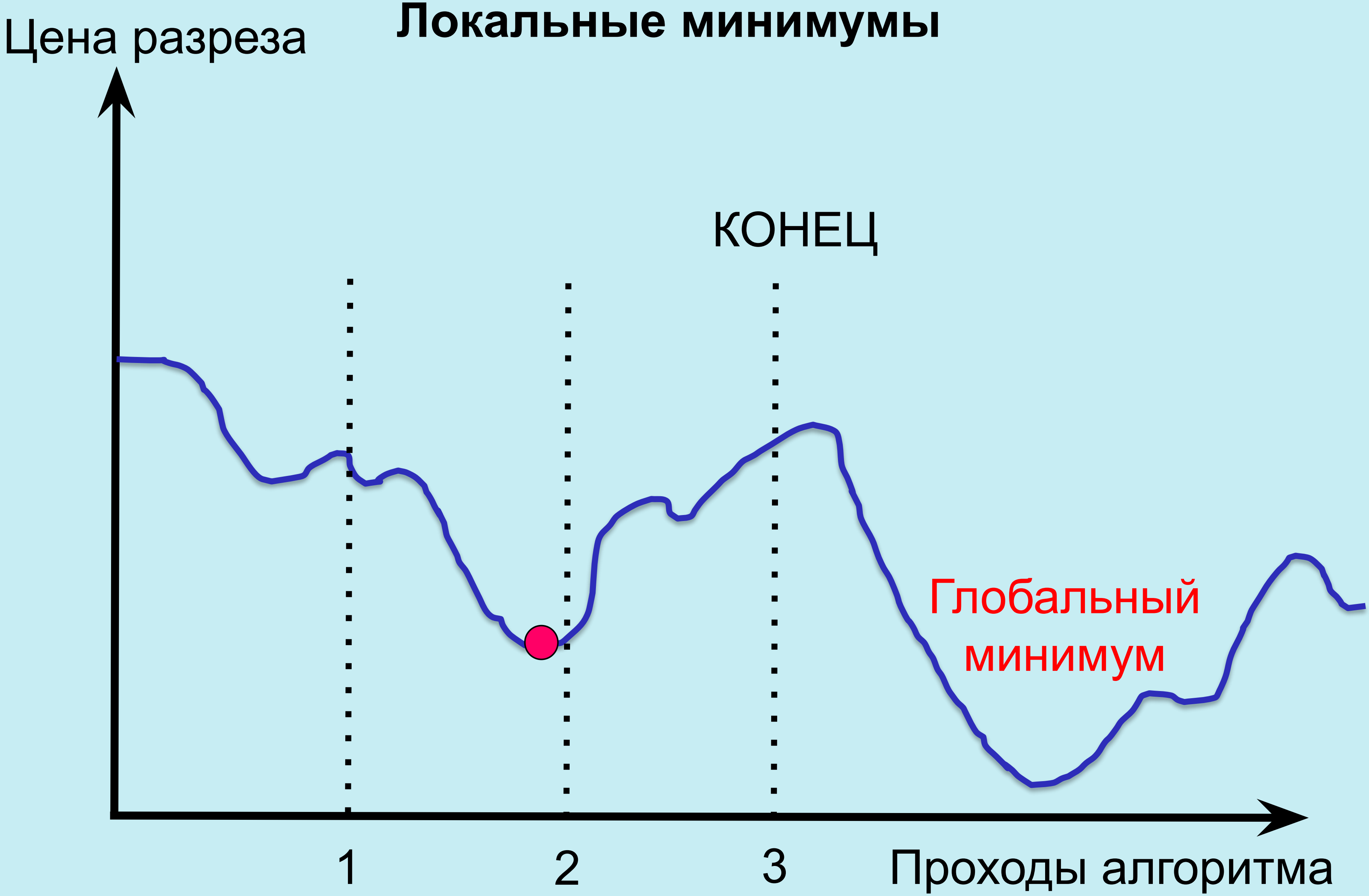


i	(x,y)	Δg_i	G_m
1	**(3,5)**	3	3
2	**(4,6)**	5	8
3	**(1,7)**	-6	2
4	**(2,8)**	-2	0

Цена разреза: 1

#v	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
D(v)	x	-1	x	x	x	x	x	-1

Алгоритм Кернигана-Лина



Анализ алгоритма

Временная сложность:

На каждый проход $O(n^2)$ при выборе лучшей пары

Всего n/2 итераций на проходе

Всего: $O(n^3)$

Недостатки:

- «Сваливается» в локальный минимум
- Только равные части
- Не учитывает вес элементов
- Низкое быстродействие
- Нет поддержки гиперребер

Расширения алгоритма

- Перемещаются только одиночные вершины вместо пары
- Переделывается подсчет стоимости разреза для поддержки гиперребер
- Учитывается вес каждой вершины
- Вводится структура данных для ускорения выбора перемещаемых вершин

Алгоритм Фидуччи-Маттеуса

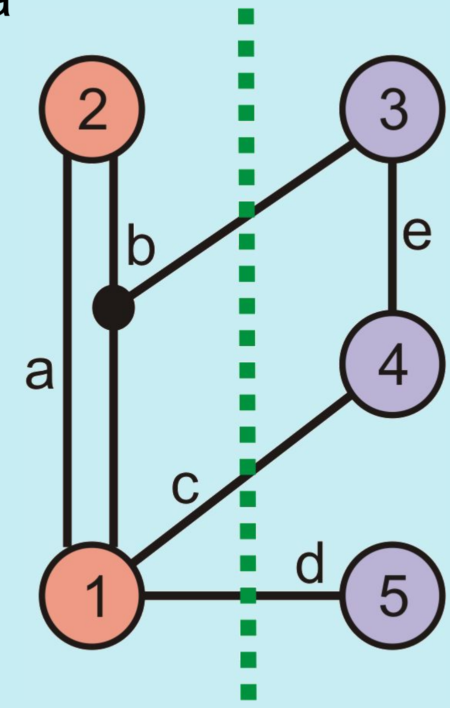
Входные данные:

Граф G=(V,E) со *взвешенными вершинами и ребрами*

Задача:

Разделить все вершины на части A и B, чтобы была минимальной цена разреза

$A \cap B = \varnothing$



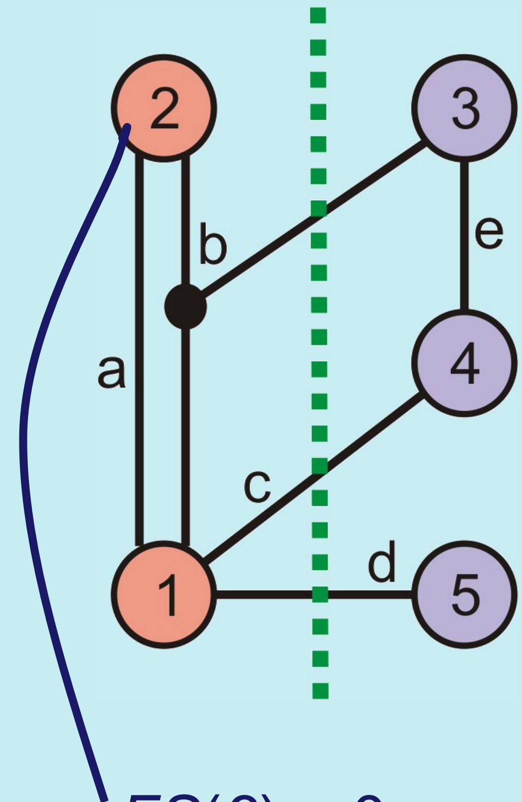
Алгоритм Фидуччи-Маттеуса

Цена перестановки:

$\Delta g(c) = FS(c) - TE(c)$

FS(*c*) – количество гиперребер, для которых c – единственная вершина в блоке

TE(*c*) - количество гиперребер связанных с c, у которых нет вершин в другом блоке



$FS(2) = 0$
$TE(2) = 1$
$\Delta g(2) = -1$

- *FS*(*c*) – «сила притягивания» (From Single)
- TE(*c*) – «сила отталкивания» (To Empty)
- Чем выше $\Delta g(c)$, тем больше выгода от перестановки c

Алгоритм Фидуччи-Маттеуса

Стоимость последовательности перестановок G_m

Как и в алгоритме Кернигана-Лина

- За один проход алгоритма необходимо максимизировать G_m
- G_m обо В конце прохода из свей последовательности шагов выбирается такая подпоследовательность длиной m, что $G_m \to max$
- За выбранные m перестановок стоимость разреза уменьшится лучше всего

$$G_m = \sum_{i=1}^{m} \Delta g_i$$

Алгоритм Фидуччи-Маттеуса

Балансирование блоков

Балансное соотношение:

- Задает относительный размер блоков A и B: area(A) и area(B)
- Предотвращает перенос всех элементов в один блок

$$r = \frac{area(A)}{area(A) + area(B)}$$

Критерий сбалансированности:

- Помогает сбалансировать размер блоков A и B
- Усовершенствует балансное соотношение
- Учитывает максимальный размер элемента $area_{max}(V)$ для перемещения через разрез

$$[\, r \cdot area(V) - area_{max}(V) \,] \le area(A) \le [\, r \cdot area(V) + area_{max}(V) \,]$$

Алгоритм Фидуччи-Маттеуса

Алгоритм

1. Рассчитать критерий сбалансированности

2. i=1

   Вычислить Δg_i для всех ячеек

3. Пока есть незафиксированные вершины

   Найти, переместить и заблокировать вершину c_i , что $\Delta g_i \rightarrow max$

   Обновить Δg для всех вершин, связанных с вершиной c_i

   i=i+1

3. Найти подпоследовательность 1,…,m; $1 \leq m \leq i$ с $G_m \rightarrow max$

   Если $G_m > 0$

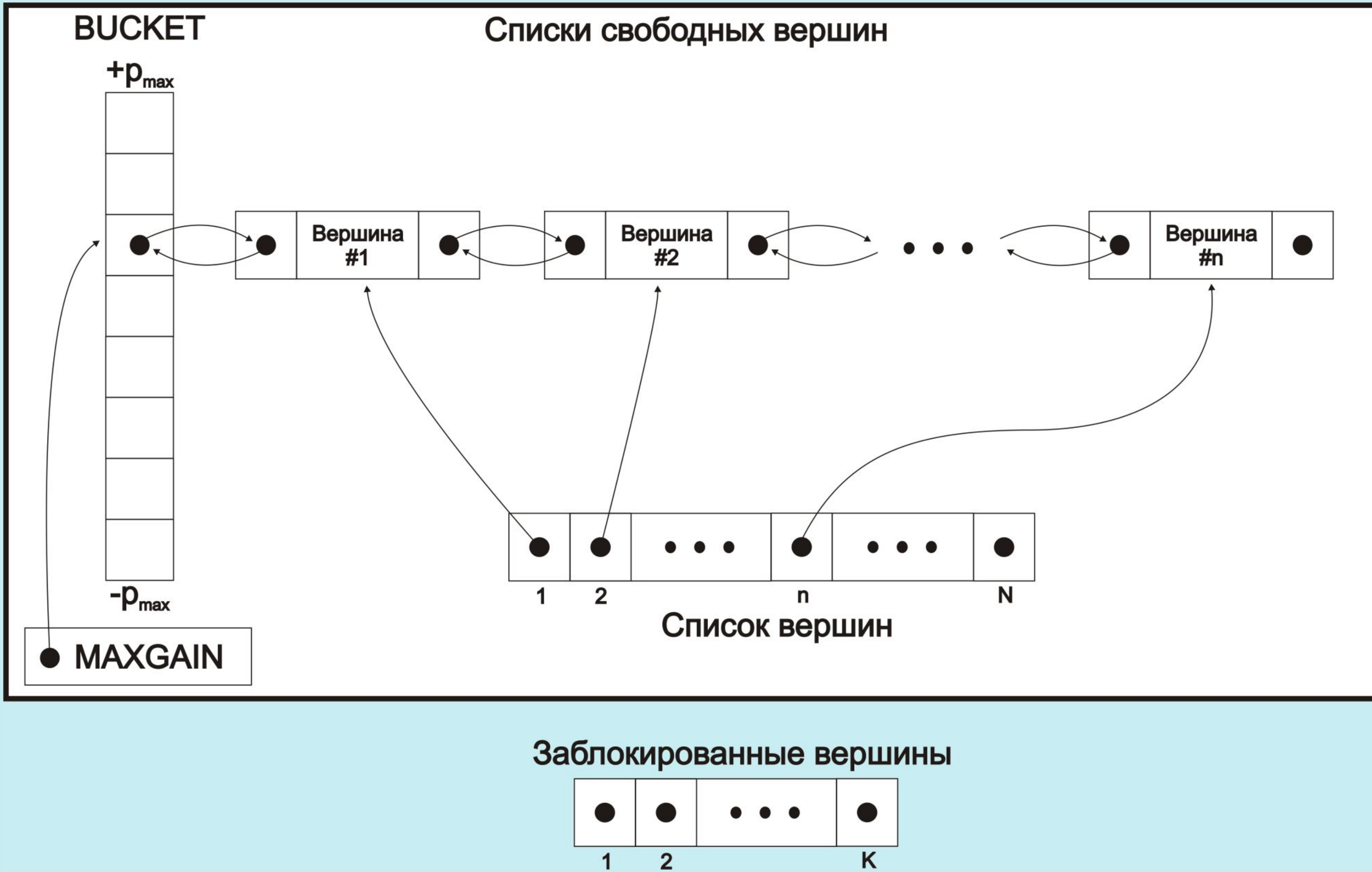
   Воспроизвести перестановки 1,…,m

   Перейти на п. 2

   Иначе КОНЕЦ

Алгоритм Фидуччи-Маттеуса

Структура данных



Алгоритм Фидуччи-Маттеуса

Пример

Исходные данные:

Балансное соотношение $r = 0{,}375$

$area(1) = 2$

$area(2) = 4$

$area(3) = 1$

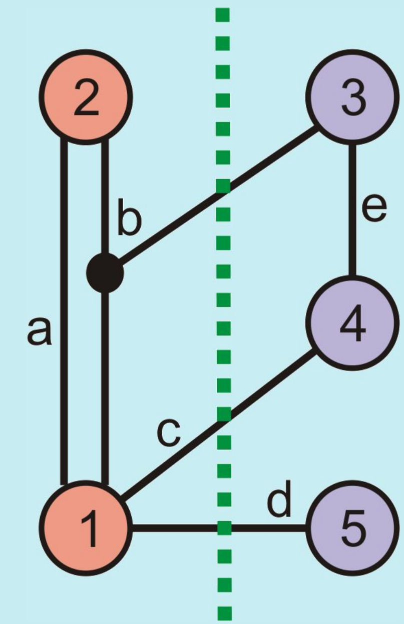
$area(4) = 4$

$area(5) = 5.$

$1 \le area(A) \le 11$

$1 = 0{,}375 * 16 – 5$

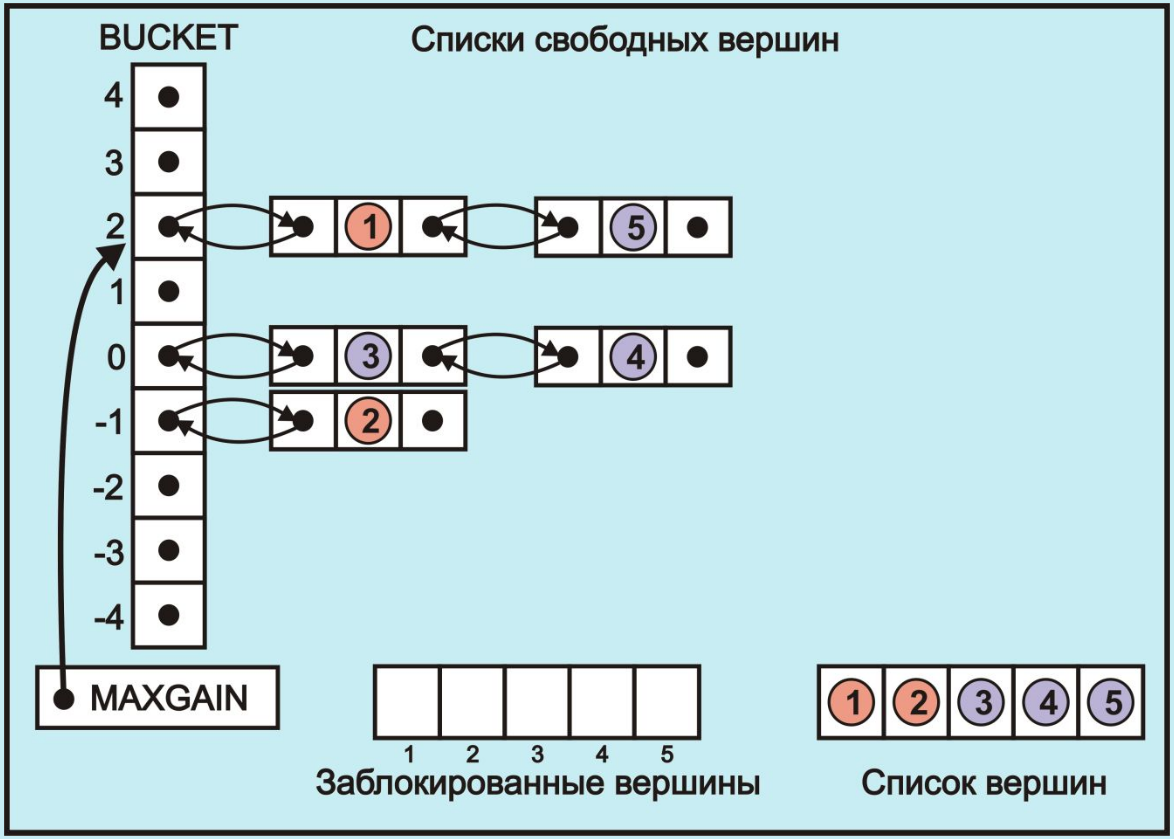
$11 = 0{,}375 * 16 + 5$

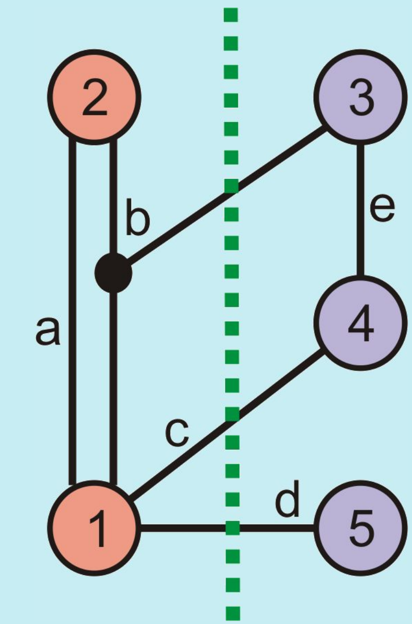


Цена разреза: 9

Алгоритм Фидуччи-Маттеуса

Пример





Цена разреза: 9

После c_1 *: area(A)=4*

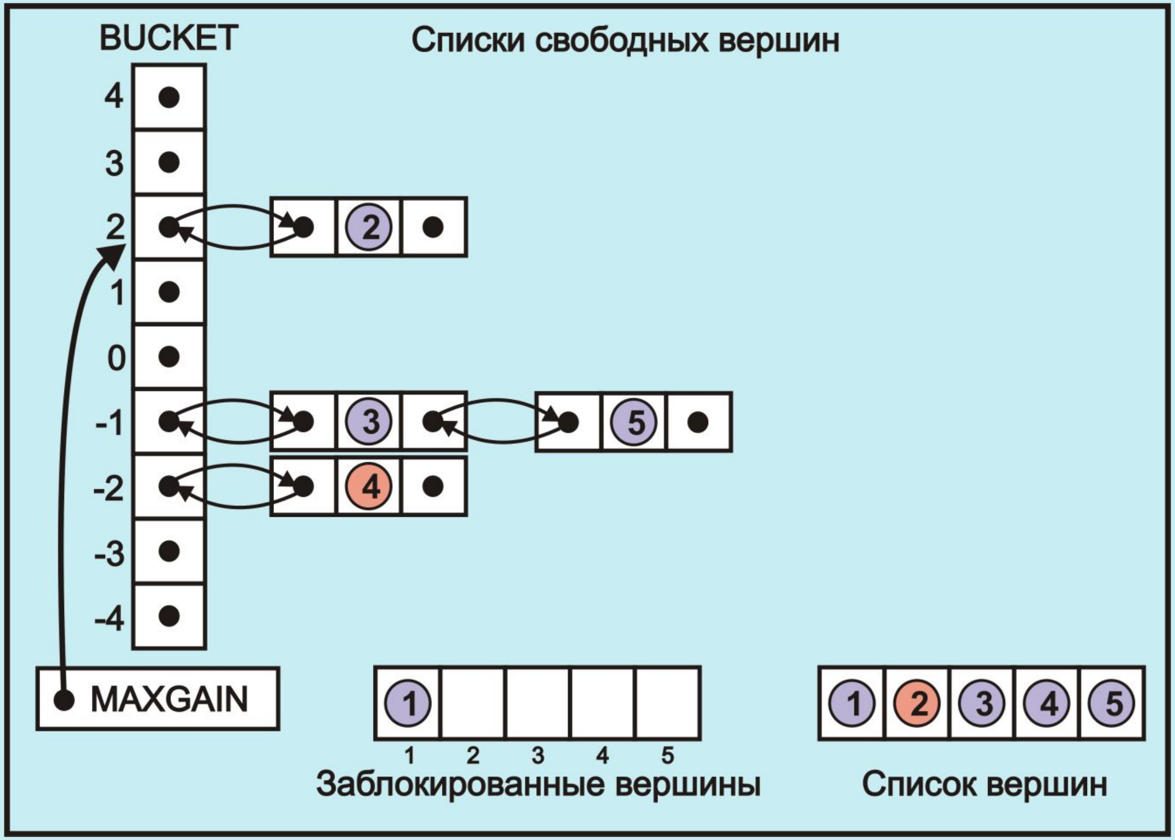
После c_5 *: area(A)=11*

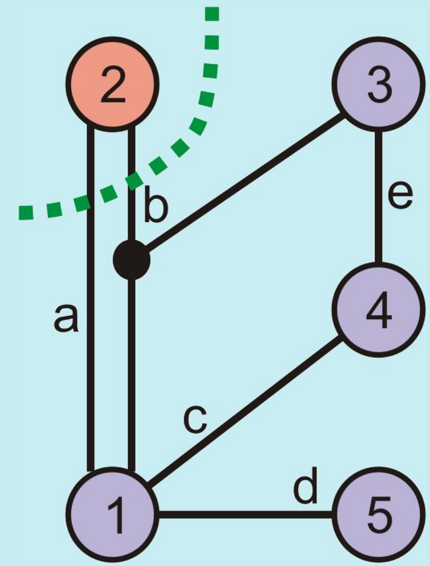
c_1 *меньше нарушает баланс*

#c	1	2	3	4	5
FS(c)	2	0	1	1	1
TE(c)	1	1	1	1	0
Δg_c	1	-1	0	0	1

Алгоритм Фидуччи-Маттеуса

Пример





Цена разреза: 2

c_2 нарушает баланс

После c_3 *: area(A)=5*

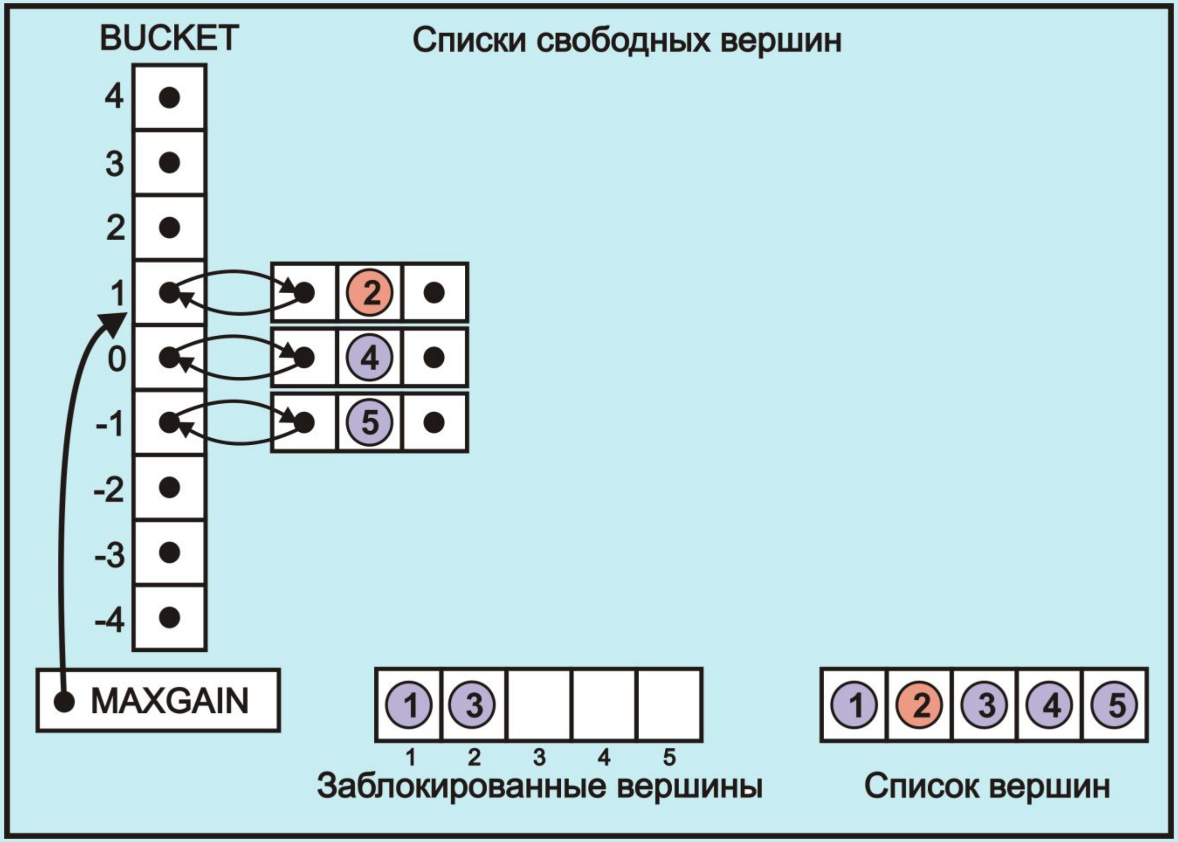
После c_5 *: area(A)=9*

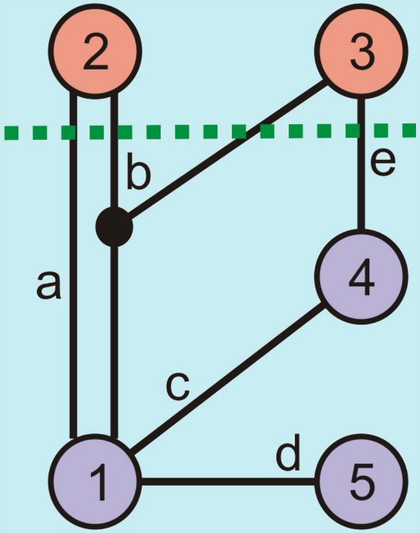
c_3 *меньше нарушает баланс*

#c	1	2	3	4	5
FS(c)	x	2	0	0	0
TE(c)	x	0	1	2	1
Δg_c	x	2	-1	-2	-1

Алгоритм Фидуччи-Маттеуса

Пример





Цена разреза: 3

Почему?

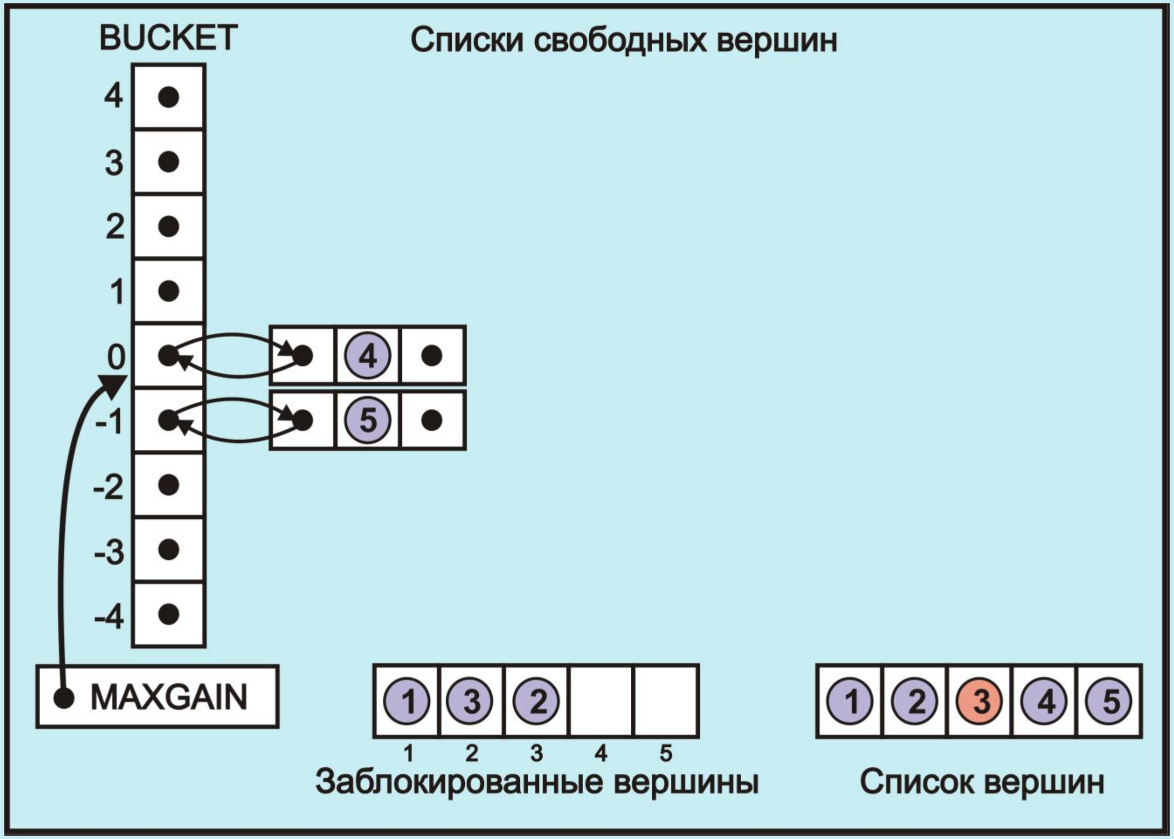
После $\mathbf{c_2}$ *: area(A)=1*

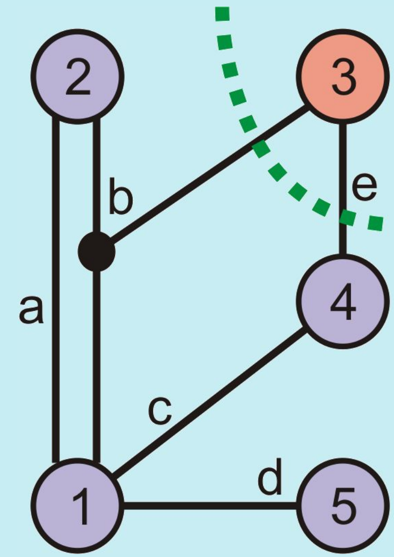
$\mathbf{c_2}$ *не нарушает баланс*

#c	1	2	3	4	5
FS(c)	x	1	x	1	0
TE(c)	x	0	x	1	1
Δg_c	x	1	x	0	-1

Алгоритм Фидуччи-Маттеуса

Пример





Цена разреза: 2

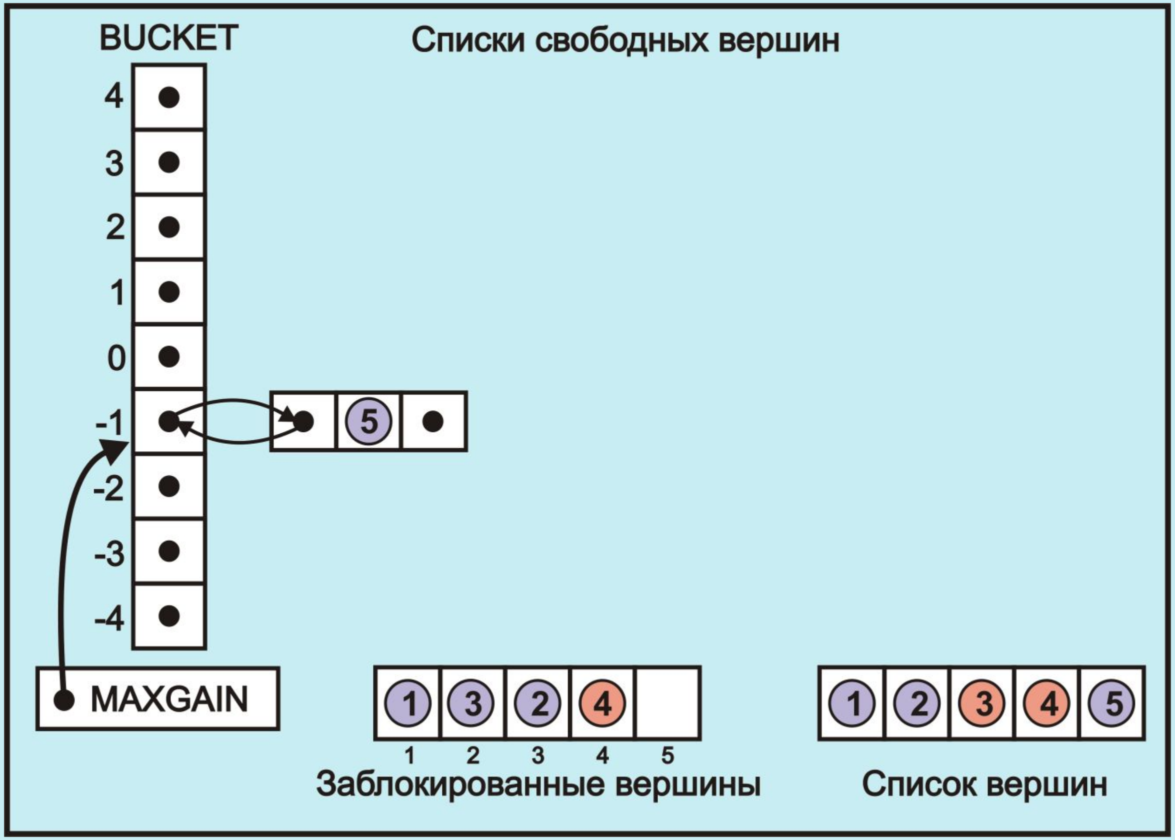
После c_4 *: area(A)=5*

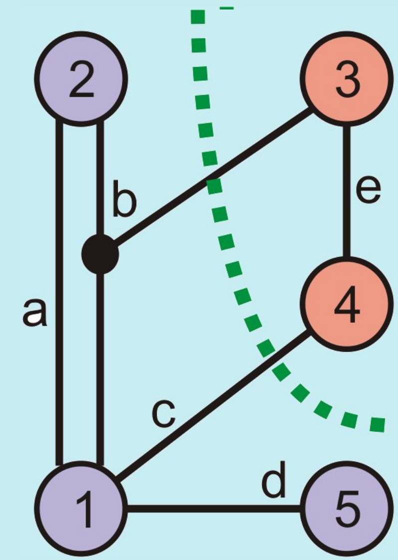
c_4 *не нарушает баланс*

#c	1	2	3	4	5
FS(c)	x	x	x	1	0
TE(c)	x	x	x	1	1
Δg_c	x	x	x	0	-1

Алгоритм Фидуччи-Маттеуса

Пример





Цена разреза: 2

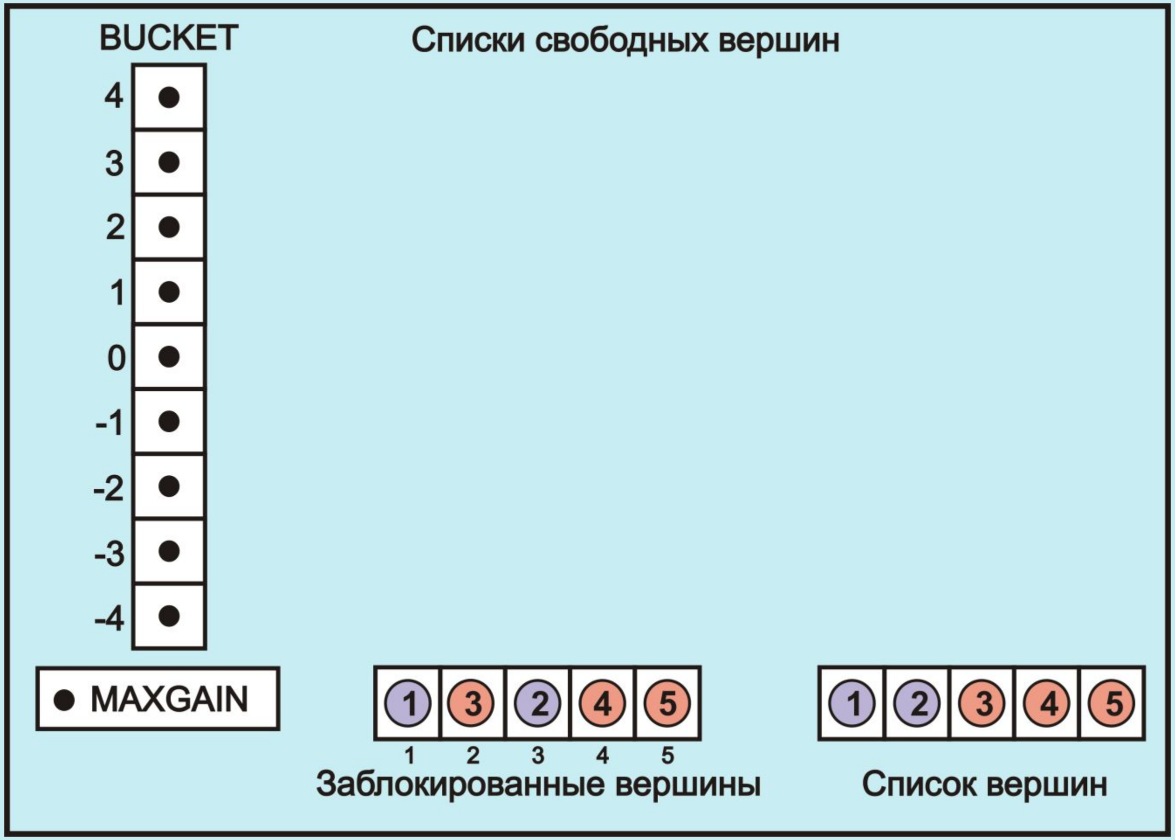
После c_5 *: area(A)=10*

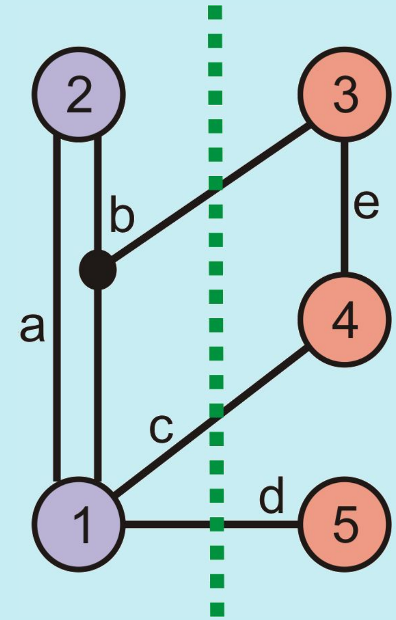
c_5 *не нарушает баланс*

#c	1	2	3	4	5
FS(c)	x	x	x	x	0
TE(c)	x	x	x	x	1
Δg_c	x	x	x	x	-1

Алгоритм Фидуччи-Маттеуса

Пример





Цена разреза: 3

#c	1	2	3	4	5
FS(c)	x	x	x	x	x
TE(c)	x	x	x	x	x
Δg_c	x	x	x	x	x

Алгоритм Фидуччи-Маттеуса

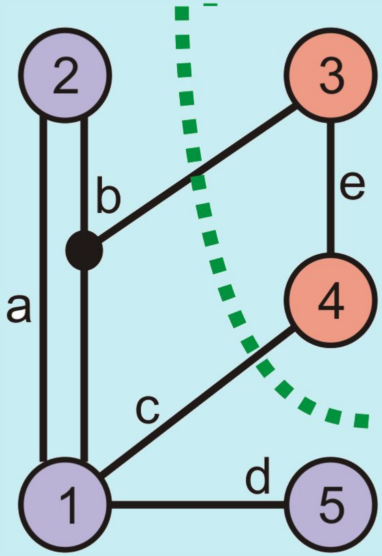
Пример

Выбор наилучшей последовательности ходов $c_1 \dots c_m$

$$G_m = \sum_{i=1}^{m} \Delta g_i = 1$$

i	c	Δg_i	G_m
1	**1**	1	1
2	**3**	-1	0
3	**2**	1	1
4	**4**	0	1
5	**5**	-1	0

Результат:



Кандидаты: m=1,3,4

Для m=4 $area(A)$ = 5
Лучше сбалансировано

Цена разреза: 2

Лекция окончена