МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное

учреждение высшего образования

«Казанский национальный исследовательский технический

университет им. А.Н. Туполева-КАИ»

(КНИТУ-КАИ)

Колледж информационных технологий «КИТ»

Дисциплина:

«Теория алгоритмов»

ЛАБОРАТОРНАЯ РАБОТА № 3

на тему:

РАЗРАБОТКА АЛГОРИТМА ПОИСКА МАКСИМАЛЬНОГОПОТОКАВТРАНСПОРТНОЙ СЕТИ

|  |  |  |
| --- | --- | --- |
| Выполнил: | Обучающийся группы 4335 |  |
|  | Желваков А. С. |  |
| Проверил: | Валова П.А. |  |

Казань 2024г.

**Вариант 9**

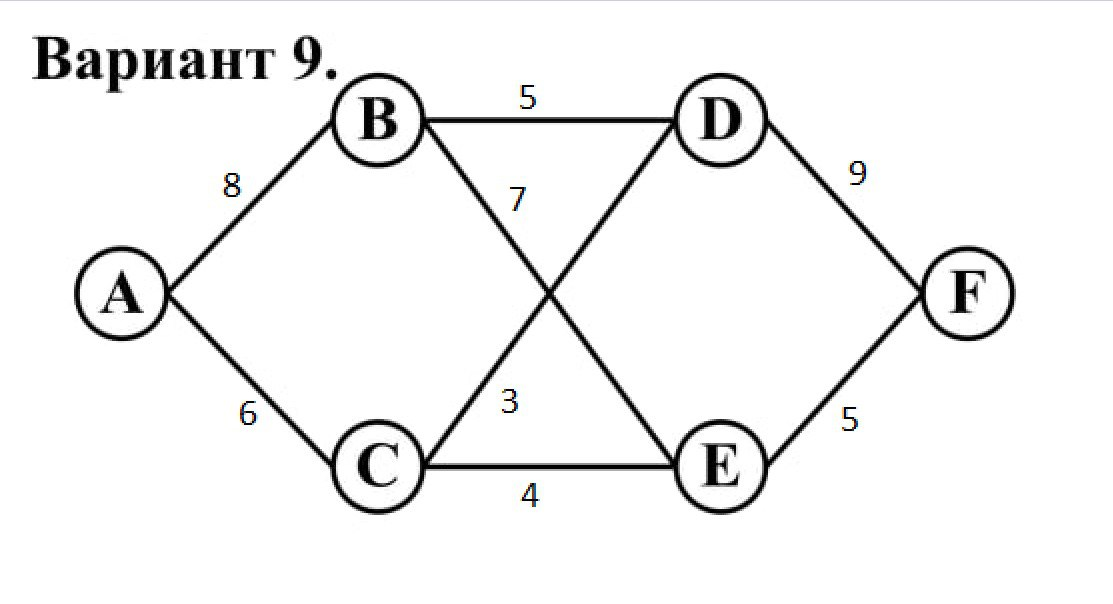
Цель: Разработать алгоритм поиска максимального потока в транспортной сети.

Задание:

1. Самостоятельно задать пропускные способности дуг и построить максимальный поток в транспортной сети.

2. Найти минимальный разрез сети и проверить справедливость теоремы Форда – Фалкерсона.

Выполнение работы:

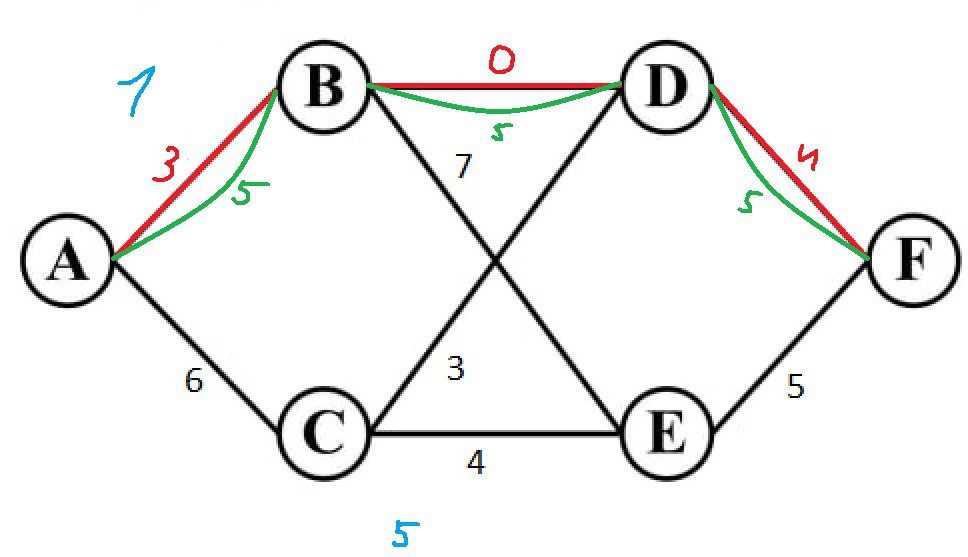
Рис. 1 - Построенный граф

Поиск максимального потока:

* Красным цветом выделены пути от A до F
* Зеленым цветом выделены пути от F до A
* Синим цветом обозначен шаг и сумма максимального потока

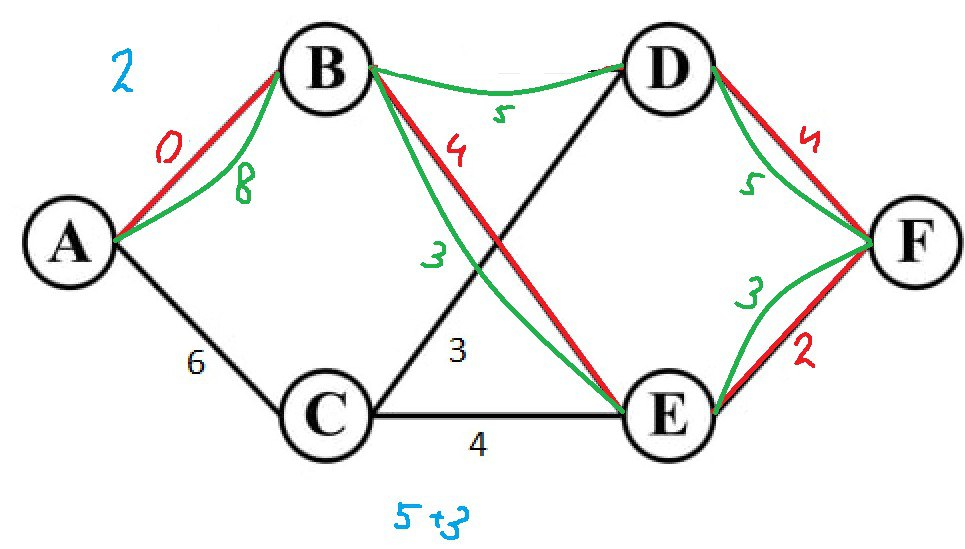
1. Шаг 1

* Прокладываем путь A – B – D – F
* Больше 5 единиц потока через этот путь пропустить нельзя
* Пропускаем поток из истока в сток
* Удаляем ребро с нулевым весом
* Добавляем к максимальному потоку 5 единиц

Рис. 2 - Шаг 1

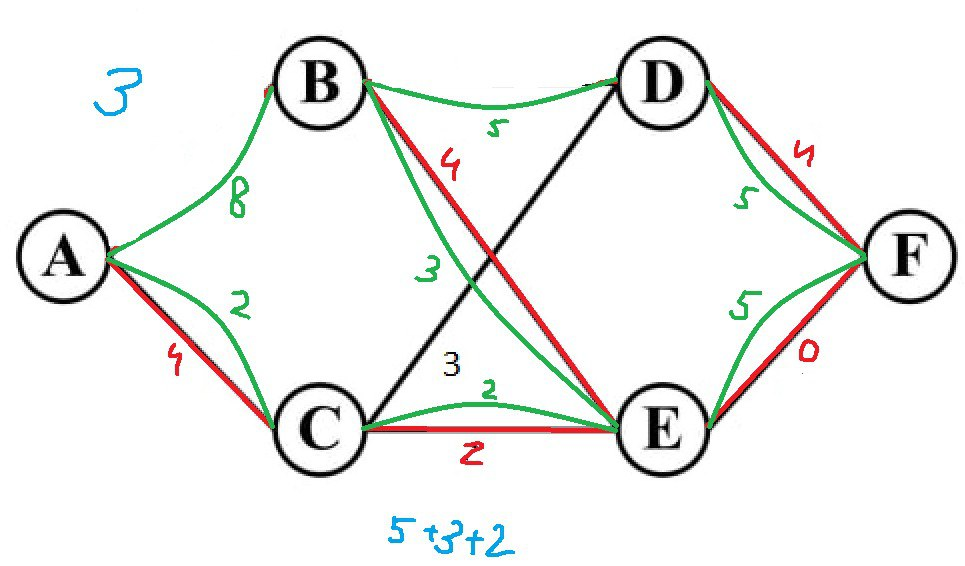
1. Шаг 2

* Прокладываем путь A – B – E – F
* Больше 3 единиц потока через этот путь пропустить нельзя
* Пропускаем поток из истока в сток
* Удаляем ребро с нулевым весом
* Добавляем к максимальному потоку 3 единицы

Рис. 3 - Шаг 2

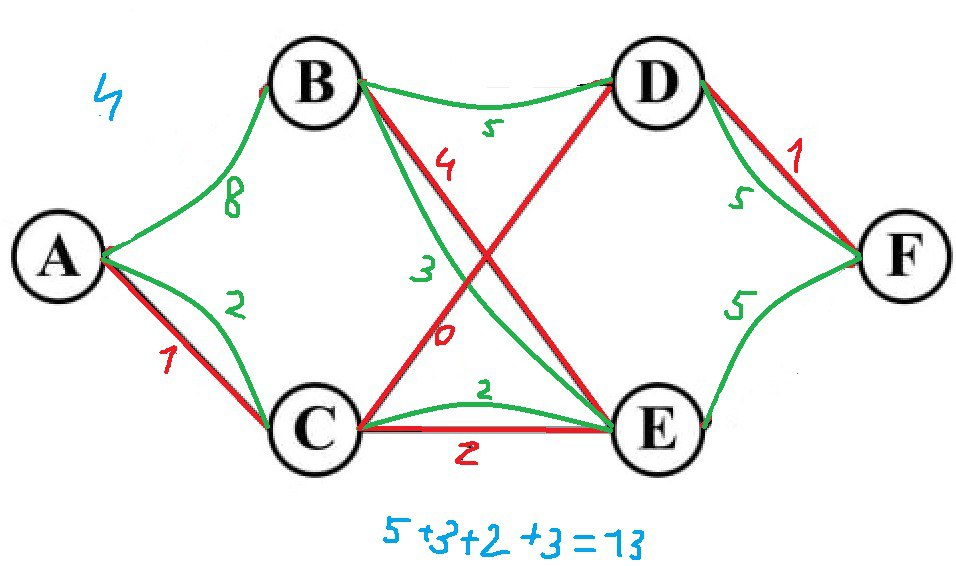
1. Шаг 3

* Прокладываем путь A – С – E – F
* Больше 2 единиц через этот путь пропустить нельзя
* Пропускаем поток из истока в сток
* Удаляем ребро с нулевым весом
* Добавляем к максимальному потоку 2 единицы

Рис. 4 - Шаг 3

1. Шаг 4

* Прокладываем путь A – C – D – F
* Больше 3 единиц через этот путь пропустить нельзя
* Пропускаем поток из истока в сток
* Удаляем ребро с нулевым весом
* Добавляем к максимальному потоку 3 единицы
* Видим, что от A к F не осталось прямых путей
* Складываем сумму максимального потока и получаем 13 единиц

Рис. 5 - Шаг 4

Поиск минимального разреза:

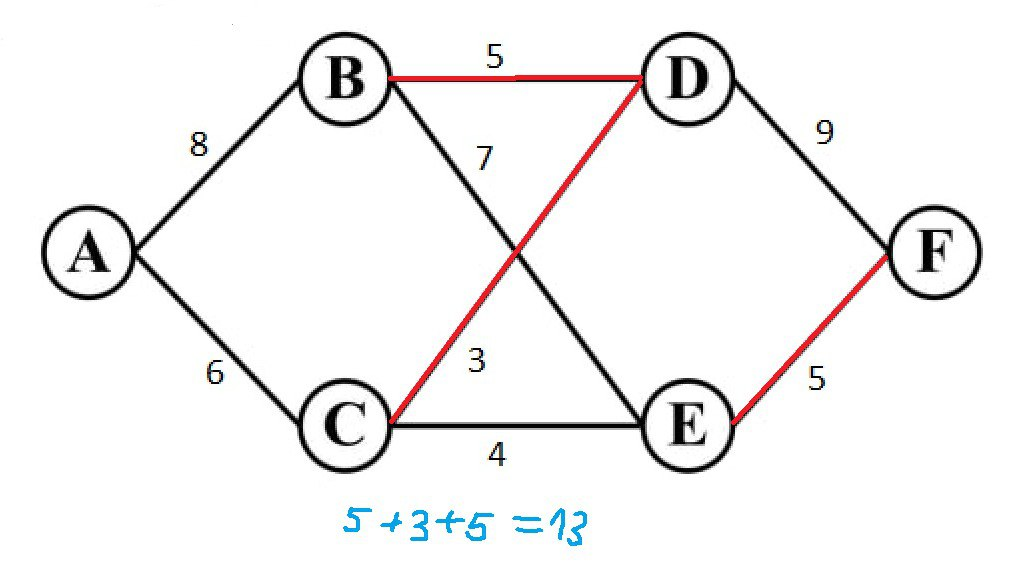
* Убираем ребра:

B – D

C – D

E – F

И получаем сумму 13.

Рис. 6 - Минимальный разрез

Вывод:

Максимальный поток равен с минимальным разрезом - теорема Форда - Фалкерсона справедлива.

ЗАДАНИЕ 1.

1. Paзpaбoтaть cxeмy aлгopитмa и пpoгpaммy, peaлизyющиe следующие мeтoды пoиcкa пoдcтpoки: алгopитм Kнyтa-Moppиca-Пpaтта, алгоритм Бойера-Мура-Хорспула

2. Cохранить фaйл “text.txt” в кaтaлoг, гдe лeжит пpoгpaммa.

3. Дoпoлнить пpoгpaммy тaк, чтoбы oнa иcкaлa вce вxoждeния cлoвa,ввeдeннoгo c клaвиaтypы, и oбщee кoличecтвo вxoждeний, в фaйлe“text.txt”.

5. Пepeпиcaть пpoгpaммy тaким oбpaзoм, чтoбы мexaнизм пoиcкa был peaлизoвaн oтдeльнoй фyнкциeй.

6. Cpaвнить вpeмя paбoты пpoгpaммы co вpeмeнeм paбoты пpoгpaммы пoиcкa дpyгим cпocoбoм.

7. Cдeлaть вывoды.

Выполнение работы:

1. Схемы алгоритмов

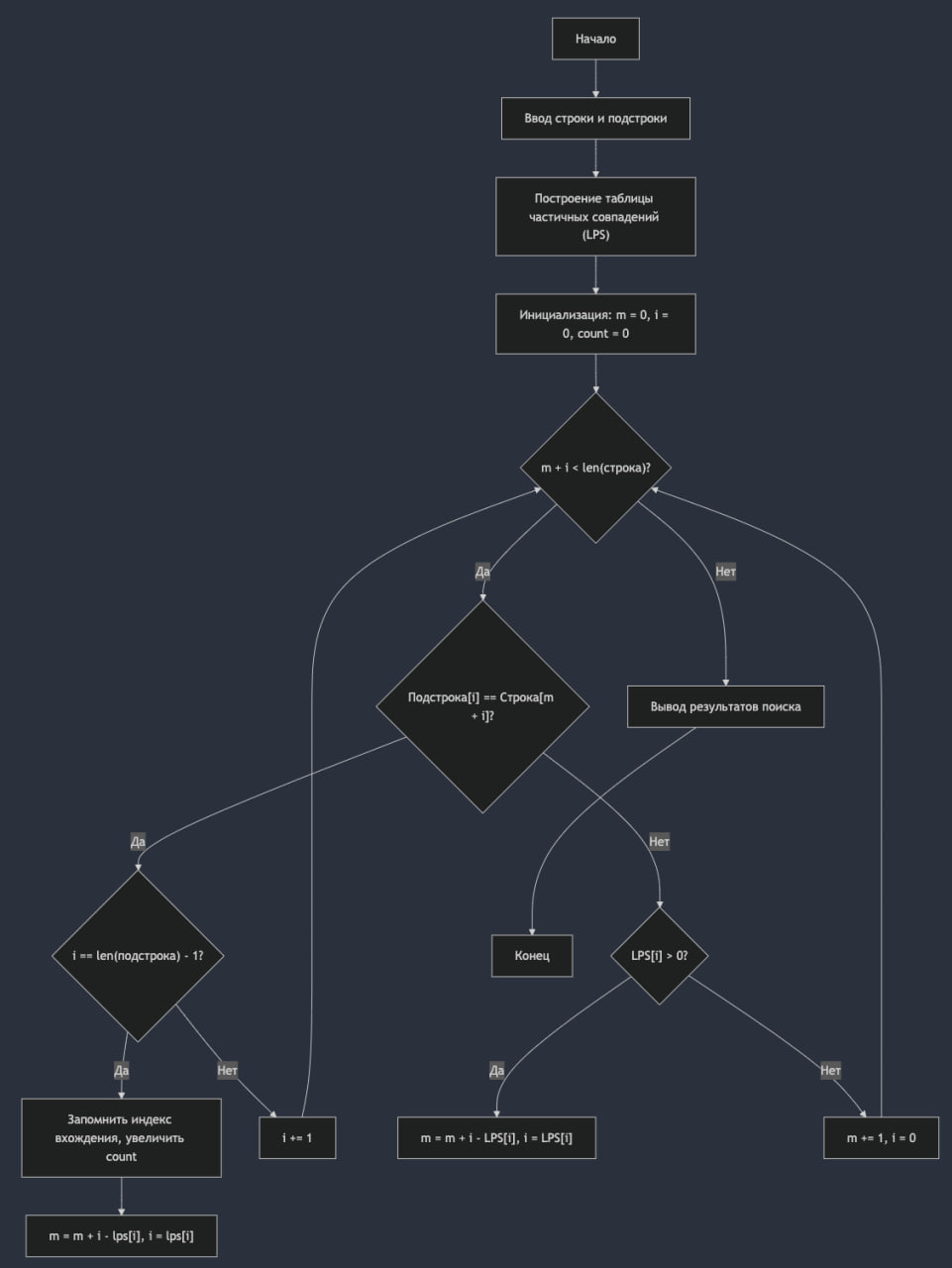


Рис. 7 - Блок-схема для алгоритма Кнута-Морриса-Пратта (КМП)

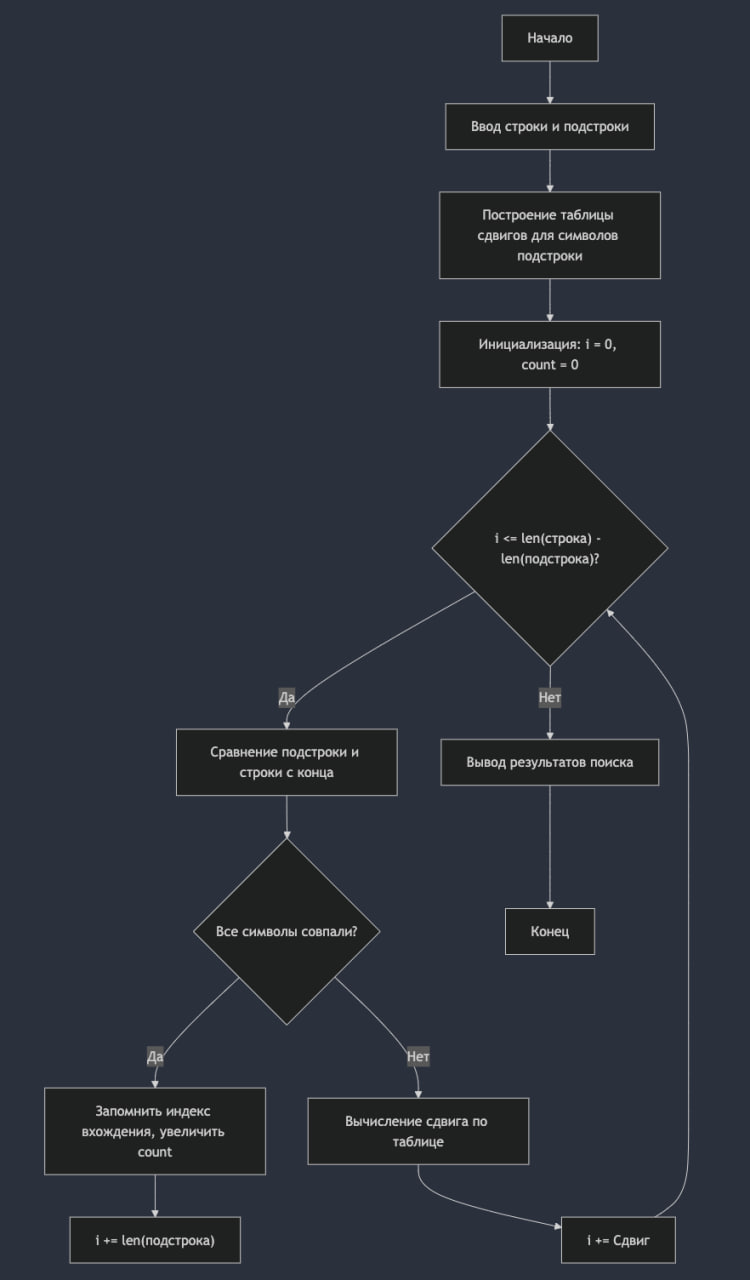


Рис. 8 - Блок-схема для алгоритма Бойера-Мура-Хорспула

1. Результат работы программы

|  |
| --- |
| Введите слово для поиска: ИВАНОВ    Поиск с помощью алгоритма Кнута-Морриса-Пратта (КМП):  Найдено вхождений: 3  Индексы вхождений: [149, 170, 183]  Время выполнения: 0.000116 секунд    Поиск с помощью алгоритма Бойера-Мура-Хорспула:  Найдено вхождений: 3  Индексы вхождений: [149, 170, 183]  Время выполнения: 0.000090 секунд    Алгоритм Бойера-Мура-Хорспула быстрее. |

1. Вывод

|  |
| --- |
| В данном эксперименте были протестированы два алгоритма поиска подстроки в тексте: алгоритм Кнута-Морриса-Пратта (КМП) и алгоритм Бойера-Мура-Хорспула. Оба алгоритма нашли три вхождения подстроки "ИВАНОВ" в тех же индексах ([149, 170, 183]), что указывает на их корректную работу.    Однако время выполнения этих алгоритмов различалось:  - Кнута-Морриса-Пратта (КМП): 0.000116 секунд  - Бойера-Мура-Хорспула: 0.000090 секунд    Алгоритм Бойера-Мура-Хорспула в данном тесте показал себя быстрее, чем алгоритм Кнута-Морриса-Пратта, сэкономив примерно 22% времени на поиске. Это объясняется особенностями алгоритма Бойера-Мура-Хорспула, который эффективно использует сдвиги при неудачных совпадениях, позволяя пропускать части текста. Алгоритм Кнута-Морриса-Пратта также эффективен, но его подход с предварительным построением таблицы частичных совпадений не всегда дает преимущество в скорости для всех типов подстрок и текстов. |

Листинг:

<https://github.com/ArseniyZh/CIT/tree/main/3rd_year/%D0%A2%D0%90/%D0%BB%D0%B0%D0%B1%D1%8B/3>