Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего профессионального образования

Кубанский государственный технологический университет

(ФГБОУ ВПО «КубГТУ»)

Факультет компьютерных технологий и автоматизированных систем

Кафедра информационных систем и программирования

**КУРСОВАЯ РАБОТА**

По дисциплине: «Дискретная математика»

На тему: «Задачи и алгоритмы дискретной математики»

Выполнила студентка группы 12-КБ-ПИ1 Дорошенко Алена Игоревна

(Ф.И.О.)

Допущен к защите:

Руководитель работы Е.А.Симоненко

(подпись, дата, расшифровка подписи)

Защищён \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка

(дата)

Члены комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.А.Симоненко

А.Г.Волик

Краснодар

2013

Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего профессионального образования

Кубанский государственный технологический университет

(ФГБОУ ВПО «КубГТУ»)

Факультет компьютерных технологий и автоматизированных систем

Кафедра информационных систем и программирования

УТВЕРЖДАЮ

Зав. кафедрой ИСиП

профессор Л.А. Видовский

«\_\_\_» 2013 г.

**ЗАДАНИЕ**

на курсовую работу

Студентке Дорошенко А.И. группы 12-КБ-ПИ-1 2 курса

факультета Компьютерных технологий и автоматизированных систем

направления 230700.62 – Прикладная информатика \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тема работы: «Задачи и алгоритмы дискретной математики»

Содержание задания: Изучить темы «Покрытие множества» и «**Паросочетания», провести исследование алгоритмов и**  **реализовать (написать программы)** на одном из языков программирования.

Объём курсовой работы:

а) пояснительная записка \_\_\_\_\_13\_\_\_\_\_ стр.;

б) программы.

Рекомендуемая литература: Скиена С. «Алгоритмы. Руководство по разработке », Кристофидес Н. «Теория графов. Алгоритмический подход»,

Окулов С.М. «Дискретная математика. Теория и практика решения задач по информатике».

Срок выполнения работы: с «02» сентября 2013г. по «21» декабря 2013г.

Срок защиты: «27 » декабря 2013г.

Дата выдачи задания: « 02 » сентября 2013г.

Дата сдачи работы на кафедру: с «02» декабря 2013г. по «28» декабря 2013г.

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Е.А.Симоненко

(подпись)

Задание приняла студентка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.И.Дорошенко

(подпись)

**РЕФЕРАТ**

**РЕФЕРАТ**

Пояснительная записка к курсовой работе 13 страниц.

ПОКРЫТИЕ МНОЖЕСТВА, ПОИСК ПОКРЫТИЯ МНОЖЕСТВА, ПАРОСОЧЕТАНИЯ, ДВУДОЛЬНЫЕ ГРАФЫ, АЛГОРИТМ НАХОЖДЕНИЯ НАИБОЛЬШЕГО И МАКСИМАЛЬНОГО ПАРОСОЧЕТАНИЯ, АЛГОРИТМ КУНА.

В данной курсовой работе рассматриваются две темы из курса «Дискретная математика»- «Покрытия множества» и «Паросочетания». Цель курсовой работы – реализация алгоритмов решения данных задач на одном из языков программирования.

**Содержание**

[1 Нормативные ссылки 5](#_Toc375898571)

[Введение 6](#_Toc375898572)

[Основная часть 7](#_Toc375898573)

[Глава 1. Покрытие множества 7](#_Toc375898574)

[1.1 Формулировка задачи 7](#_Toc375898575)

[1.2 Сведение к двудольному графу 7](#_Toc375898576)

[Глава 2. Паросочетания 9](#_Toc375898577)

[2.1Формулировка задачи 9](#_Toc375898578)

[2.2 Алгоритм Куна: 9](#_Toc375898579)

[Список используемой литературы 10](#_Toc375898580)

[Приложение A – Листинг программы 11](#_Toc375898581)

[Приложение В – Листинг программы 13](#_Toc375898582)

# 1 Нормативные ссылки

В данной пояснительной записке использованы ссылки на нижеперечисленные стандарты.

ГОСТ 2.105-95. Общие требования к текстовым документам

ГОСТ Р 50739-95. Государственный стандарт РФ. Средства вычислительной техники. Защита от несанкционированного доступа к информации. Общие технические требования.

ГОСТ 7.32-2001. СИБИД. Отчет о НИР. Структура и правила оформления.

ГОСТ 7.1-84 СИБИД. Библиографическое описание документа. Общие требования и правила составления.

ГОСТ 7.80-2000 Библиографическая запись. Заголовок. Общие требования и правила составления.

ГОСТ 19.701-90 (ИСО 5807-85) ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения.

ГОСТ 34.601-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.

# Введение

Матрица Татта — это изящный подход к решению задачи о паросочетании в произвольном (не обязательно двудольном) графе. Правда, в простом виде алгоритм не выдаёт сами рёбра, входящие в паросочетание, а только размер максимального паросочетания в графе.

Паросочетанием называется набор попарно несмежных рёбер графа (иными словами, любой вершине графа должно быть инцидентно не более одного ребра из множества ). Мощностью паросочетания назовём количество рёбер в нём. Наибольшим (или максимальным) паросочетанием назовём паросочетание, мощность которого максимальна среди всех возможных паросочетаний в данном графе. Все те вершины, у которых есть смежное ребро из паросочетания (т.е. которые имеют степень ровно один в подграфе, образованном ),назовём насыщенными этим паросочетанием.

# Основная часть

# Глава 1. Покрытие множества

# Формулировка задачи

Дан ориентированный ациклический граф. Требуется покрыть его наименьшим числом путей, т.е. найти наименьшее по мощности множество непересекающихся по вершинам простых путей, таких, что каждая вершина принадлежит какому-либо пути.

# Сведение к двудольному графу

Пусть дан граф G с n вершинами. Построим соответствующий ему двудольный граф H стандартным образом, т.е.: в каждой доле графа H будет по n вершин, обозначим их через a и b соответственно. Тогда для каждого ребра (i, j)исходного графа G проведём соответствующее ребро .

Каждому ребру G соответствует одно ребро H, и наоборот. Если мы рассмотрим в любой путь , то ему ставится в соответствие набор рёбер . Более просто для понимания будет, если мы добавим "обратные" рёбра, т.е. образуем граф из графа добавлением рёбер вида . Тогда пути  в графе будет соответствовать путь .

Обратно, рассмотрим любой путь  в графе , начинающийся в первой доле и заканчивающийся во второй доле. Очевидно, снова будет иметь вид , и ему можно поставить в соответствие в графе G путь . Однако здесь есть одна тонкость: v1 могло совпадать с, поэтому путь получился бы циклом. Однако по условию граф G ациклический, поэтому это вообще невозможно (это единственное место, где используется ацикличность графа G; тем не менее, на циклические графы описываемый здесь метод вообще нельзя обобщить).

Итак, всякому простому пути в графе , начинающемуся в первой доле и заканчивающемуся во второй, можно поставить в соответствие простой путь в графе G, и наоборот. Но заметим, что такой путь в графе — это паросочетание в графе H. Таким образом, любому пути из можно поставить в соответствие паросочетание в графе H, и наоборот. Более того, непересекающимся путям в G соответствуют непересекающиеся паросочетания в H.

Последний шаг. Заметим, что чем больше путей есть в нашем наборе, тем меньше все эти пути содержат рёбер. А именно, если есть p непересекающихся путей, покрывающих все n вершин графа, то они вместе содержат  рёбер. Итак, чтобы минимизировать число путей, мы должны максимизировать число

рёбер в них.

Итак, мы свели задачу к нахождению максимального паросочетания в двудольном графе . После нахождения этого паросочетания (см. Алгоритм Куна) мы должны преобразовать его в набор путей в (это делается

тривиальным алгоритмом, неоднозначностей здесь не возникает). Некоторые вершины могут остаться ненасыщенными паросочетанием, в таком случае в ответ надо добавить пути нулевой длины из каждой из этих вершин.

Реализация алгоритма в Приложении А

# Глава 2. Паросочетания

# 2.1Формулировка задачи

Алгоритм Куна. Нахождения наибольшего паросочетания в двудольном графе.

Дан двудольный граф G , содержащий n-вершин и m-рёбер. Требуется найти наибольшее паросочетание, т.е. выбрать как можно больше рёбер, чтобы ни одно выбранное ребро не имело общей вершины ни с каким другим выбранным ребром.

# 2.2 Алгоритм Куна:

Его можно кратко описать так: сначала возьмём пустое паросочетание, а потом — пока в графе удаётся найти увеличивающую цепь, — будем выполнять чередование паросочетания вдоль этой цепи, и повторять процесс поиска увеличивающей цепи. Как только такую цепь найти не удалось — процесс останавливаем, — текущее паросочетание и есть максимальное. Осталось детализировать способ нахождения увеличивающих цепей. Алгоритм Куна — просто ищет любую из таких цепей с помощью обхода в глубину или в ширину. Алгоритм Куна просматривает все вершины графа по очереди, запуская из каждой обход, пытающийся найти увеличивающую цепь, начинающуюся в этой вершине.

Реализация алгоритма в Приложения В

# 

# Список используемой литературы

1. Кристофидес Н. Теория графов. Алгоритмический подход. – Пер. с англ. – М.: Мир, 1978. – 432 с.;-глава10.

2. Седжвик Р. Алгоритмы на C++. – Пер. с англ. – М.: Вильямс, 2011. – 1056 с.;-глава 17.7.

3. Скиена С. Алгоритмы. Руководство по разработке. – 2-е изд.: пер. с англ. – СПб.: БХВ-Петербург, 2011. – 720 с.;-глава 16.5 и глава 14.6.

4. Окулов С.М. Дискретная математика. Теория и практика решения задач по информатике: учебное пособие. – М.: БИНОМ. Лаборатория знаний, 2008. – 422 с.;- глава 3.9.

# Приложение A – Листинг программы

int N=7,n=9,j,i=0;  
// Togda polnoe mnojestvo delaetsya tak:  
long U = 0;  
for(i=0; i<7;i++ )  
{  
U |= (1L«i);  
}  
// Dlya polnogo perebora toje ispol'zuem shkalu  
long P = 0;  
for(i=0; i<n; i++)  
{  
P |= (1L«i);  
}  
double a[9]; //vesa  
double s; // cena  
/// Pust' mnojestva zadany chislami long A[n]  
long A[7]={ {10100010}, {101001010}, {110011},  
{100100100},{1010001},{1101},{10110101},};  
long X = 0; // X soderjit mnojestva, uchastvuyuschie v probnom pokrytii  
long Y; // Chastichnoe pokrytie  
while(X!=P) {  
Y = 0;  
for(i=0; i<n; i++) {  
if (Y==U) {   
s = 0;  
for (j=0; j<i; j++) { // Pechat' naydennogo pokrytiya i ego ceny  
if ( (X & (1L«j))!=0 ) {  
printf("%c ", 'A'+j);  
s += a[j];  
}  
}  
printf("s=%f\n", s);  
break;  
}  
if ( (X & (1L«i))!=0 )  
{  
Y = (Y | A[i]);  
}  
}  
X++;  
}  
}

# Приложение В – Листинг программы

На языке С++:

int n, k;

vector < vector<int> > g;

vector<int> mt;

vector<char> used;

bool try\_kuhn (int v) {

if (used[v]) return **false**;

used[v] = **true**;

for (size\_t i=0; i<g[v].size(); ++i) {

int to = g[v][i];

if (mt[to] == -1 || try\_kuhn (mt[to])) {

mt[to] = v;

return **true**;

}

}

return **false**;

}

int main() {

... чтение графа ...

mt.assign (k, -1);

for (int v=0; v<n; ++v) {

used.assign (n, **false**);

try\_kuhn (v);

}

for (int i=0; i<k; ++i)

if (mt[i] != -1)

printf ("%d %d\n", mt[i]+1, i+1);

}