**Московский Авиационный Институт**

**(Национальный Исследовательский Университет)**

**Факультет: «Информационные технологии и прикладная математика»**

**Кафедра 805 «Математическая кибернетика»**

**Курсовая работа**

Нахождение максимально внутренне устойчивых подмножеств графа

Студент: Коростелев Д.В.

Группа: М80-108Б-18

Преподаватель: Смерчинская С.О.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: 04.06.2019

**Содержание**

1. Теоретические сведения. Описание алгоритма..............................................2
2. Алгоритм ...........................................................................................................3
3. Логическая блок-схема.....................................................................................4
4. Оценка сложности алгоритма..........................................................................5
5. Тестовые примеры.............................................................................................5
6. Список использованной литературы...............................................................6

**Теоретические сведения. Описание алгоритма**

Внутренне устойчивое подмножество. Пусть задан граф G = (E,Г); подмножество S ⊂ E называется *внутренне устойчивым,* если S ⋂ ГS = ø. Другими словами, никакие две вершины S не смежны. Если S’ ⊂ S, то S’ – также внутренне устойчивое подмножество.

Максимальное внутренне устойчивое подмножество – внутренне устойчивое подмножество, не являющееся собственным подмножеством никакого другого внутренне устойчивого подмножества.

Число внутренней устойчивости. – число вершин наибольшего из внутренне устойчивых подмножеств.

Отыскание семейства максимально внутренне устойчивых подмножеств (*метод Магу*). Этот метод использует свойства булевых уравнений.

Будем рассматривать графы без петель, так как вершина, имеющая петлю, не может принадлежать внутренне устойчивому подмножеству.

Пусть S – некоторое внутренне устойчивое подмножество. Свяжем с любой вершиной Xi графа булеву переменную xi и положим:

если Xi ∈ S, то ¬xi = 1 или xi = 0,

если Xi ∈ ГXi, j ≠ i, то αij = 1;

если Xi ∉ ГXi, j ≠ i, то αij = 0.

Для любой пары вершин Xi и Xi (с учетом S ⋂ ГS = ø) справедливо утверждение: ( i ≠ j; Xj ∈ Г Xi или Xi ∈ Г Xj ) ⇒ Xi ∉ S или Xj ∉ S). Это можно записать так: i ≠ j, ¬αij ¬αij ∔ ¬xi∔ ¬xj = 1

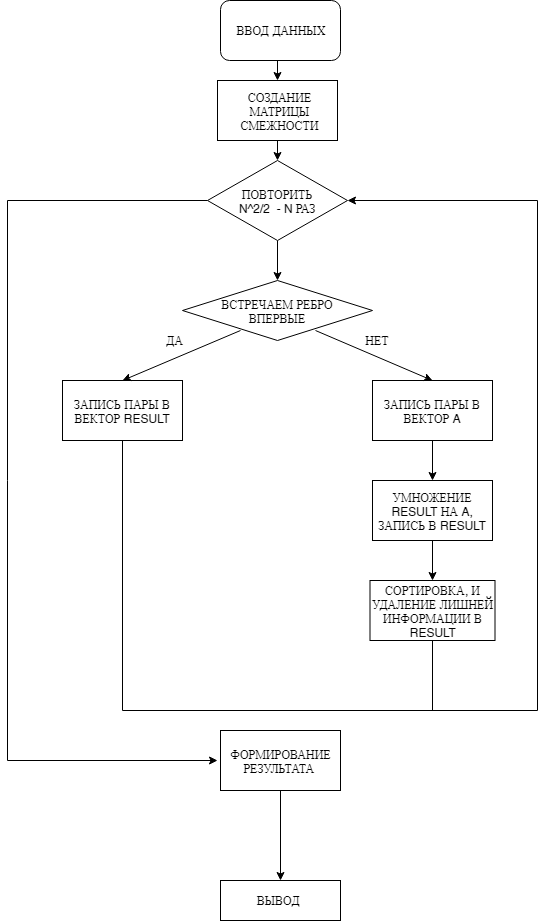
Беря произведение по всем вершинам графа, получаем уравнение:

ɸS ¬αij ¬αij ∔ ¬xi∔ ¬xj)= 1.

Далее раскроем скобки и приведем подобные члены, учитывая, что a + ab = a. Тогда для каждого члена совокупность всех вершин, соответствующих переменным, отсутствующим в нем, дает максимальное внутренне устойчивое подмножество графа. Действительно, такой член содержит лишь переменные с отрицанием и поэтому к множеству вершин Xi, соответствующих переменным xi, не встречающимся в этом члене, нельзя добавить никакой другой.

**Алгоритм**

Алгоритм основан на выше расписанном методе Магу, написан на языке С++, программа на вход получает число вершин в графе, далее формируется двумерный вектор n на n, который в последствии требуется заполнить пользователю, далее следуя по алгоритму Магу, формируются пары, по следующему принципу, если в графе на месте G[i][j] стоит единица, то формируется пара I и J, которая из целочисленного типа переводится в символьный и если ребро встречаем впервые , то есть вектор result пустой, записываем эту пару в result, иначе, записываем эту пару в буферную переменную A. Далее умножаем result и a, путем записи в result всех возможных комбинации строк векторов result и a, далее следует применить правило a + ab = a, для этого сортируем полученный result, по возрастанию длин строк, сортируем сами строки (элементы result) по возрастанию, удаляем повторяющиеся буквы в строках. После этого запускаем функцию, которая выполняет упрощение a + ab = a, посредством поиска подстрок. Повторяем данные действия, пока не закончиться матрица, количество шагов в итоге равно n^2/2 – n. После выполнения всех шагов в result получаем набор строк с переменными, из этих строк получаем уравнение ɸS, далее делаем инверсию элементов result (если в строке было записано AB и число вершин в графе было равно 5, то получаем строку CDE), формируем и выводим результат.

**Логическая блок-схема**

**Оценка сложности алгоритма**

Сложность алгоритма составляет О(( – 1)), так как программа запускает функцию умножению ( – 1)n раз, а число вложенных циклов у функции умножения не превышает 2, значит ее сложность O().

**Тестовые примеры**

|  |  |
| --- | --- |
| 8  0 1 0 0 1 1 0 0  1 0 1 1 0 0 0 0  0 0 0 0 0 0 0 0  0 0 0 0 1 0 0 0  0 0 0 1 0 1 0 0  0 0 0 0 1 0 1 0  0 0 0 0 0 0 0 1  0 0 0 1 0 1 0 0 | [A,C,D,G]  [C,E,H]  [C,E,G]  [B,E,H]  [B,E,G]  [A,C,H]  [C,D,F]  [B,F] |
| 4  0 1 0 0  0 0 1 0  0 0 0 0  0 0 1 0 | [A,D]  [A,C]  [B,D] |
| 3  0 1 0  0 0 1  0 0 0 | [A,C]  [B] |
| 5  0 1 0 1 0  0 0 0 0 1  0 0 0 1 0  0 0 0 0 0  0 0 0 0 0 | [A,C,E]  [D,E]  [B,D]  [B,C] |
| 5  0 1 1 1 1  0 0 1 1 1  1 1 0 1 1  1 1 1 0 1  1 1 1 1 0 | [E]  [D]  [C]  [B]  [A] |

**Список использованной литературы**

1. *Методические указания к выполнению курсовой работы по теории графов циклу дисциплин 2018. 46стр. Смерчинская С.О., Яшина Н.П.*
2. *Draw.io [Эдектронный ресурс]//*  [*https://www.draw.io/*](https://prog-cpp.ru/)*URL:* [*https://www.draw.io*](https://www.draw.io)
3. *Введение в прикладную комбинаторику. А. Кофман Издательство «Наука». Главная редакция физико-математической литературы. Москва 1975. 480стр, перевод с французского В.П. Мякишева и В.Е. Тараканова, под редакцией Б.А. Севастьянова*