Белорусский государственный университет информации и радиоэлектроники

Расчётная работа

«Решение теоретико-графовой задачи 1.19 для неориентированного графа»

Подготовила:

Кравцова Вероника Кирилловна

Гр. 421701

Проверил:

Зотов Н.В.

Минск 2024

Цель: составить алгоритм определения двудольного графа для неориентированного графа, заданного списком инцидентности (1.19).

Алгоритм:

1. Задаём количество вершин и рёбер.
2. Нумеруем вершины, начиная с 0.
3. Нумеруем рёбра, начиная с 0.
4. Переходим к ребру 0.
   4.1. Задаём для выбранного ребра две вершины.
   4.2. Если все рёбра имеют вершины, переходим к пункту 5.
   4.3. Иначе переходим к следующему ребру.
   4.4. Возвращаемся к пункту 4.1.
5. Инициализируем цвета.
   5.1. Если вершина не имеет цвета, то она не раскрашена и имеет число -1.
   5.2. Если вершина не раскрашена, то:
      5.2.1. Если у вершины нет соседей со значением 0, то можно её раскрасить цветом 0.
      5.2.2. Иначе – цветом 1.
   5.3. Первоначально все вершины не раскрашены.
6. Переходим к вершине 0.
7. Если вершина 0 является единственной, то граф двудольный. Завершение алгоритма.
8. Иначе:
   8.1. Называем выбранную вершину основной.
   8.2. Переходим к соседней вершине в порядке возрастания рёбер.
   8.3. Если соседняя вершина не раскрашена, то:
      8.3.1. Если соседняя вершина является последней, то:
         8.3.1.1. Раскрашиваем основную вершину цветом 0.
         8.3.1.2. Переходим к пункту 8.6.
      8.3.2. Иначе:
         8.3.2.1. Переходим к следующей соседней вершине.
         8.3.2.2. Возвращаемся к пункту 8.3.
   8.4. Если соседняя вершина раскрашена цветом 0, то:
      8.4.1. Если соседняя вершина является последней, то:
         8.4.1.1. Раскрашиваем основную вершину цветом 1.
         8.4.1.2. Переходим к пункту 8.6.
      8.4.2. Иначе:
         8.4.2.1. Переходим к следующей соседней вершине.
         8.4.2.2. Если соседняя вершина раскрашена в цвет 1, то граф не является двудольным. Завершение алгоритма.
         8.4.2.3. Иначе возвращаемся к пункту 8.4.1.
   8.5. Если соседняя вершина раскрашена цветом 1, то:

8.5.1. Если вершина является последней, то:

8.5.1.1. Раскрашиваем основную вершину цветом 0.

8.5.1.2. Переходим к пункту 8.6.

8.5.2. Иначе:

8.5.2.1. Переходим к следующей соседней вершине.

8.5.2.2. Если соседняя вершина раскрашена в цвет 0, то граф не является двудольным. Завершение алгоритма.

8.5.2.3. Иначе возвращаемся к пункту 8.5.1.

8.6.Если вершина является последней, то переходим к пункту 8.8.

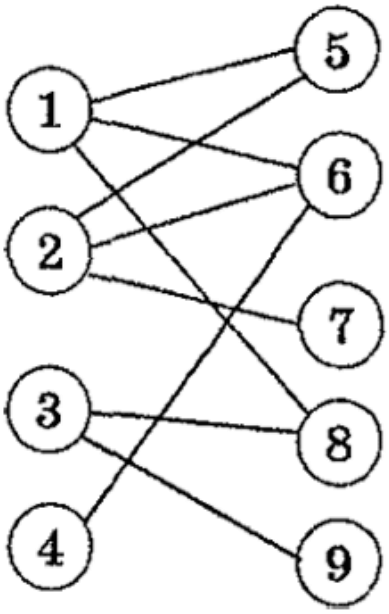
8.7.Иначе:

8.7.1. Переходим к следующей вершине.

8.7.2. Возвращаемся к пункту 8.1.

8.8.Граф является двудольным. Завершение алгоритма.

Пример двудольного графа:



Описание кода:

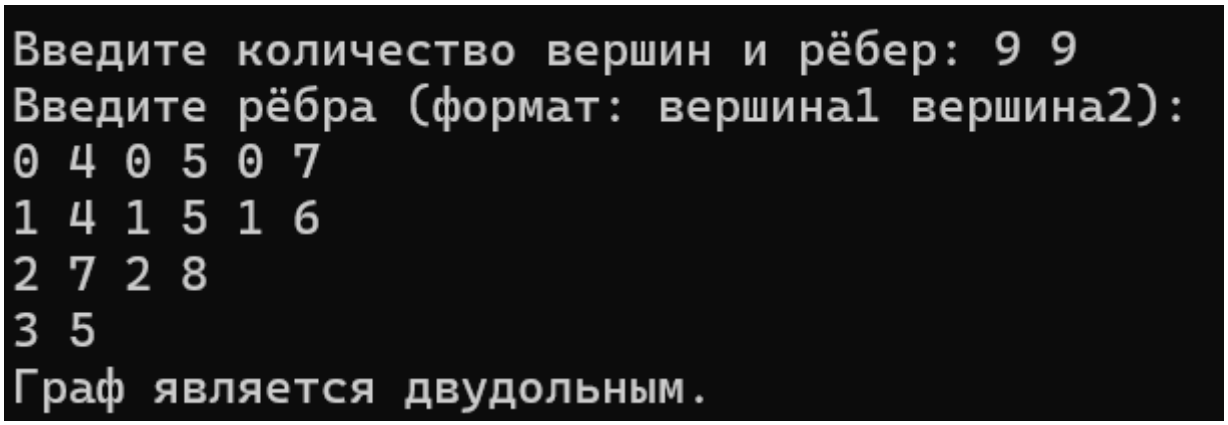
Функция isBipartite(int, vector<vector<int>>) возвращает истинное значение, если переданный ей граф двудольный, либо ложное значение, если переданный граф не является двудольным. В данной функции реализован алгоритм решения графовой задачи.

Функция main() обеспечивает ввод с клавиатуры и вывод на консоль информации, связанной с графовой задачей.

```cpp
bool isBipartite(int v, vector<vector<int>>& adj) {
    vector<int> color(v, -1);
    for (int start = 0; start < v; ++start) {
        if (color[start] == -1) {
            queue<int> q;
            q.push(start);
            color[start] = 0;

            while (!q.empty()) {
                int node = q.front();
                q.pop();


                for (int neighbor : adj[node]) {
                    if (color[neighbor] == -1) {
                        color[neighbor] = 1 - color[node];
                        q.push(neighbor);
                    }
                    else if (color[neighbor] == color[node]) {
                        return false;
                    }
                }
            }
        }
    }
    return true;
}
```

```cpp
int main() {
    setlocale(LC_ALL, "ru");
    int vertices, edges;
    cout << "Введите количество вершин и рёбер: ";
    cin >> vertices >> edges;

    vector<vector<int>> adj(vertices);

    cout << "Введите рёбра (формат: вершина1 вершина2):" << endl;
    for (int i = 0; i < edges; ++i) {
        int u, v;
        cin >> u >> v;

        if (u < 0 || u >= vertices || v < 0 || v >= vertices) {
            cout << "Ошибка: вершины должны быть в диапазоне от 0 до " << vertices - 1 << "." << endl;
            return 1;
        }

        adj[u].push_back(v);
        adj[v].push_back(u);
    }

    if (isBipartite(vertices, adj)) {
        cout << "Граф является двудольным." << endl;
    }
    else {
        cout << "Граф не является двудольным." << endl;
    }

    return 0;
}
```

Пример работы программы:



Вывод: в ходе расчётной работы было определено, что для определения двудольного графа необходимо и достаточно, чтобы граф можно было раскрасить в два цвета так, чтобы соседние вершины были разных цветов; также в ходе расчётной работы был выведен алгоритм определения двудольного графа и перенесён на язык программирования C++.