**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ**

**Кафедра дифференциальных уравнений и системного анализа**

**Обнаружение сообществ алгоритмом Лейдена**

Курсовая работа

Яблонской Анны Олеговны

студентки 3 курса, специальность 1-31 03 09 Компьютерная математика   
и системный анализ

Научный руководитель:  
кандидат физ.-мат. наук,  
доцент А.Э. Малевич

Минск, 2022

**Оглавление**

[введение 3](#_Toc103727143)

[Глава 1 Начальные сведения 4](#_Toc103727144)

[1.1 Кластеризация и поиск сообществ 4](#_Toc103727145)

[1.2 Основные понятия и обозначения 4](#_Toc103727146)

[1.3 Модульность 7](#_Toc103727147)

[1.4 CPM 7](#_Toc103727148)

[1.5 Параметр разрешения 8](#_Toc103727149)

[1.6 Алгоритмы обнаружения сообществ 9](#_Toc103727150)

[Глава 2 Алгоритм лейдена 11](#_Toc103727151)

[2.1 Общие сведения 11](#_Toc103727152)

[2.2 Гарантии алгоритма 12](#_Toc103727153)

[Глава 3 Реализация алгоритма лейдена 15](#_Toc103727154)

[Глава 4 Применение алгоритма лейдена 17](#_Toc103727155)

[2.1 Рекомендательная система фильмов 17](#_Toc103727156)

[заключение 23](#_Toc103727157)

[Список использованной литературы 24](#_Toc103727158)

введение

В данной работе будет разобран метод обнаружение сообществ в графах. Одной из важных особенностей графов, представляющих реальные системы, является структура сообщества, т.е. организация вершин в подгруппы, при этом множество ребер соединяют вершины одного подмножества, и сравнительно мало ребер соединяют вершины между подмножествами. Обнаружение сообществ имеет большое значение в социологии, биологии и компьютерных науках, дисциплинах, где системы часто представляются в виде графов.

В данной работе будет разобран один из популярных алгоритмов обнаружения сообществ: алгоритм Лейдена, который будет в дальнейшем применен для обнаружения сообществ в различных данных.

# Начальные сведения

## Кластеризация и поиск сообществ

Кластеризация — это метод машинного обучения, при котором объекты группируются таким образом, чтобы объекты из одного кластера были более похожи друг на друга, чем на объекты из других кластеров по какому-либо признаку.

Обнаружение сообществ — задача, которая старается найти лучшее разбиение графа на сообщества, исходя из того, как вершины связаны между собой. Здесь опять же связь вершин в сообществе должна быть лучше, чем связь между сообществами.

На первый взгляд не совсем понятно, в чем различия этих методов. На самом деле их принцип довольно схож, отличия лишь в понятии близости объектов внутри кластеров (сообществ). Для кластеризации понятием близости объектов является расстояние между ними (например, расстояние Уорда, расстояние дальнего/ ближнего соседа, среднее расстояние и т.д.). Для метода обнаружение сообществ понятие близости выражается в значении степени вершины. Чем выше степень вершины, тем больше у нее соседней, соответственно больше вероятности образовать сообщество.

## Основные понятия и обозначения

**Определение 1** граф с узлами и ребрами. Задавать графы можно различными способами: указанием множества вершин и ребер, графически, с помощью матрицы смежности или матрицы инцидентности. В данной работе графы будут задаваться при помощи матриц смежности A, для которой Ai,j=1, если между вершинами существует ребро, и 0, если ребра нет. Для взвешенных графов вес ребра определим как wi,j. Еслі граф невзвешенный, то wi,j=1.

**Определение 2** множество вершин называется сообществом, если связь вершин в данном множестве лучше, чем связь между вершинами множества и вершинами вне него. (На рисунках 4.3 и 4.4 вершины, выделенные одним цветом составляют сообщества)

**Определение 3** Разбиение графа состоит из сообществ, где каждое сообщество Ci Vсостоит из множества узлов , для любых . (Рисунки 4.3 и 4.4 являются примерами разбиений). Для двух множеств S и R будем использовать следующие обозначения:

**Определение 4** Обозначим—разбиение, которое получается при перемещении узла в сообщество разбиения .

**Определение 5** Обозначим как подграф, образованный сообществом, т.е. и . Сообщество называется связным, если связный граф. И наоборот, сообщество называется несвязным, если несвязный граф. (Рисунок)

**Определение 6** Пусть граф –– базовый (начальный). Обозначим агрегированный граф базового графа . Узлы агрегированного графа являются сообществами в разбиении базового графа , то есть . Ребра агрегированного графа являются многогранниками. Количество ребер между двумя узлами агрегированного графа равняется числу ребер между узлами в двух соответствующих сообществах базового гафа . Следовательно,где мультимножество. (Рисунок)

**Определение 7:** Мощность множества : , где если s не является множеством.

**Определение 8:** Функция качества или сообщества графа . Функция качества должна обладать свойством: , где обозначает одноэлементное разбиение агрегированного графа . Это гарантирует, что функция качества дает одинаковые результаты для базового и агрегированного графа.

**Определение 9:** Изменение функции качества после перемещения узла в сообщество обозначим, как . Другими словами: . Аналогично, обозначим изменение функции качества после перемещения множества узлов в сообщество: . Пустое сообщество обозначим . Следовательно, изменение функции качества после перемещения множества узлов в пустое, т.е. новое, сообщество обозначим .

**Определение 10:** Операция уплощения(сглаживания) для множества S определяется следующим образом:

Где , если не является множеством. Множество, которое было сглажено называется уплощенным(сглаженным).

Также нас понадобится операция уплощения разбиения :

Т.е. уплощение разбиения – это операция в которой каждое сообщество сглажено. Разбиение, к которому была применена операция уплощения, называется уплощенным (сглаженным).

## Модульность

Одним из способов нахождения сообществ является модульность. Этот метод максимизирует разницу между фактическим и ожидаем количеством ребер в сообществе C. Связи внутри сообществ должны быть относительно частыми, в то время как связи между сообществами – редкими. Исходя из этого мы будем “поощрять” связи в сообществах и штрафовать за отсутствие связей в сообществе (1):

(1)

где , если и 0 в противном случае, веса . Минимальное значение данной функции соответствует наилучшему разделению. Выбор весов очень важен, т.к. они оказывают основное влияние на то, какие сообщества будут выбраны.

Существуют различные виды модульности, однако нам интересна лишь Константная модель Поттса (CMP), поскольку именно она применяется как функция качества в алгоритмах Лувенского и Лейдена.

## CPM

Для начала определим веса модели : , где это параметр разрешения, получаем (2):

(2)

Если мы определим количесво ребер в сообществе как (3),

(3)

a количество вершин /

Тогда уравнение (2) можно переписать в следующем виде (4):

(4)

Получаем что модель максимизирует число внутренних ребер, сохраняя небольшие сообщества. Параметр разрешения выполняет роль порога. Например, если сообщество состоит из ребер и вершин, тогда его лучше разделить на два сообщества R и S, если , где количество ребер между сообществами R и S. Такое соотношение определяет плотность связей между сообществами R и S. Получаем, что плотность связи между сообществами должна быть ниже , а внутри сообществ выше . Более высокой значение приводит к большему количеству сообществ и наоборот.

## Параметр разрешения

Разберем параметр подробнее, поскольку от выбора данного параметра зависит качество разбиения графа на сообщества.

В общем случае соответствует решению, когда все вершины находятся в одном сообществе. В тоже время соответствует решению только тогда, когда каждая вершина сама по себе является сообществом. Таким образом параметр разрешения необходимо выбирать из отрезка . Однако какой именно выбрать параметр?

Для разных моделей модульности следует выбирать различные параметры разрешения. Данный параметр зависит от количества кликов. Кликой называется подмножество вершин графа, любые две из которых соединены ребром. В общем случае для CMP стоит выбирать такой разрешающий параметр, что .

## Алгоритмы обнаружения сообществ

Существуют различные алгоритмы обнаружения сообществ. Наиболее популярными являются алгоритм Лувена и алгоритм Лейдена. Оба этих алгоритма основаны на модульности и рассматриваются как методы быстрого обнаружения сообществ в сети. Более подробно данные алгоритмы будут рассмотрена в следующих главах.

Алгоритм Лувена был предложен в 2008 году как метод быстрого развертывания сообщества для больших сетей. Он основан на оптимизации функции модульности. Он очень популяр из-за простоты реализации и скорости. Однако у него есть несколько недостатков.

Во-первых, данный алгоритм может находить “плохо” связанные сообщества, т.е. сообщества, которые не связаны внутренне. В частности, он находит несвязные сообщества. Получаем, что одна часть сообщества, может достигнуть другой, только через внешний путь, т.е. вершину, принадлежащую другому сообществу. Такая вершина, являющаяся мостом между разъединенными частями сообщества, сама может быть перемещена в другое сообщество. Удаление такого узла приводит к появлению несвязного сообщества. И части данного сообщества необязательно будут образовывать новые сообщества, наоборот зачастую они так и остаются в одном сообществе. Это объясняется тем, что вершины внутри такого сообщества могут быть связаны сильнее между собой, чем с другими вершинами за пределами сообщества. В этом заключается недостаток алгоритма Лувена: он может находить сколько угодно слабо связанных сообществ.

Во-вторых, данный алгоритм дает лишь одну гарантию: раздел на сообщества, вершины которых невозможно объединения в другие сообщества. Другими словами, он гарантирует, что сообщества хорошо разделены. Более сильные гарантии можно получить, повторяя алгоритм, т.е. используя предыдущие разбиения в качестве отправной точки следующей итерации. Таким образом качество разделения будет увеличиваться до тех пор, пока алгоритм не сможет вносить дальнейшие изменения. На этом этапе гарантируется, что каждый узел оптимально распределен. Такой способ итерации алгоритма гарантирует две вещи: 1) никакие сообщества не могут быть объединены; 2) никакие узлы не могут быть перемещены. Однако повторение алгоритма лишь усугубляет проблему слабо связанных сообществ, поскольку теперь алгоритм использует свое же предыдущее разбиение. Т.е. после первой итерации алгоритма было получено некоторое разбиение. Далее на первом шаге следующей итерации алгоритм снова переместит отдельные узлы. Некоторые из узлов могут быть мостами между узлами сообщества и перемещая их появляются слабо связанные сообщества. Более того алгоритм не обладает механизмом выявления таких сообществ. Таким образом повторение алгоритма с одной стороны улучшает разделение сообществ, а с другой ухудшает его. Поскольку решить проблему слабо связанных сообществ фактически невозможно, был придуман еще один алгоритм, алгоритм Лейдена, который решает данную проблему.

Все данные недостатки были устранены в алгоритме Лейдена, поэтому он и был выбран в данной работе.

# Алгоритм лейдена

## Общие сведения

После того как было обнаружено, что алгоритм Лувена находит плохо связанные сообщества был предложен новый алгоритм: алгоритм Лейдена, который частично основан на алгоритме Лувена и является его усовершенствованной версией. Алгоритм Лейдена решает проблему плохо связанных сообществ и в отличие от предыдущего алгоритма, гарантирует, что сообщества хорошо связаны.

Данный алгоритм состоит из трех фаз:

1. Быстрое локальное перемещение узлов. В отличие от Алгоритма Лувена, данный алгоритм не посещает все узлы до тех пор, пока они не перестанут перемещаться, тем самым посещая и те узлы, которые нельзя перемещать. В процедуре быстрого локального перемещения посещаются только узлы, окрестности которых изменились.
2. Уточнение разделения. На данном этапе алгоритм находит разбиение , которое является уточнением разбиения , полученного на этапе 1 алгоритма. Здесь сообщества из могут быть разделены на под сообщества
3. Агрегация сети. На этом этапе происходит агрегация графа с учетом уточненного разбиения.

Более подробно данный алгоритм рассмотрен в следующей главе.

## Гарантии алгоритма

Гарантии алгоритма Лейдена частично полагаются на случайность выбора в алгоритме. За случайность отвечает параметр , который используется на шаге уточненного разделения. Для начала введем некоторые определения.

γ-разделимость: два сообщества называются γ-разделимыми, если . Соообщество γ-разделимо, если γ-разделимо относительно всех . Разбиение γ-разделимо, если все сообщества γ-разделимы.

γ-связность: Множество вершин сообщества γ-связно, если или S может быть разделен на два подмножества R и T таких, что  
 и R и T γ-связаны. Сообщество γ-связно, если γ-связно. Разбиение γ-связно, если все сообщества γ-связны.

γ-плотность по подразбиениям: Множество вершин γ-плотно по подразбиениям, если выполняются два условия: 1) ; 2). или S может быть разделен на два подмножества R и T таких, что  
 и R и T γ-плотно по подразбиениям. Сообщество γ-плотно по подразбиениям, если также γ-плотно по подразбиениям. Разбиение γ-плотно по подразбиениям, если все γ-плотны по подразбиениям.

Оптимальность по узлам: Сообщество называется оптимальным по узлам, если для любых и все (или ). Разбиение оптимально по узлам. Если все оптимальны по узлам.

Равномерная γ-плотность: Сообщество называется равномерно γ-плотным, если для всех . Разбиение равномерно γ-плотно, если все равномерно γ-плотны.

Оптимальность по подмножеству: Сообщество называется оптимальным по подмножеству, если для всех и все (или ). Разбиение оптимально по подмножеству, если все оптимальны по подмножеству.

Итерация алгоритма Лейдена, при котором разбиение не меняется, называется стабильной итерацией. После стабильной итерации алгоритм гарантирует, что:

1. все узлы локально оптимально назначены, т.е. нет отдельных узлов, которые можно было бы переместить
2. все сообщества являются γ-плотными по подразбиениям, т.е. сообщество можно разделить на две части так, что: эти две части хорошо связаны и ни одна из частей не может быть отделена от своего сообщества; каждая часть сама также γ-плотна по подразбиениям. γ-плотность подразбиения не означает, что отдельные узлы локально оптимально назначены. Это означает только то, что отдельные узлы хорошо связаны со своим сообществом.

В случае алгоритма Лувена после стабильной итерации все последующие итерации также будут стабильными, значит дальнейшие улучшения невозможны. В отличие от алгоритма Лейдена, в котором после стабильной итерации могут быть дальнейшие улучшения в более поздних итерациях. Фактически, когда мы продолжаем повторять алгоритм Лейдена, он сойдется к разбиению, для которого гарантируется, что:

1. все сообщества равномерно γ-плотны, т.е. нет подмножеств сообщества, которые можно отделить от него. Равномерная γ-плотность означает, что независимо от того, как сообщество разделено на две части, эти две части всегда будут хорошо связаны друг с другом. Кроме того, если все сообщества в разделении равномерно γ-плотны, качество раздела не слишком далеко от оптимального.
2. все сообщества оптимальны для подмножества.

Данный алгоритм также гарантирует, что:

1. все сообщества γ-разделимы;
2. все сообщества γ-связаны.

В этих свойствах γ относится к параметру разрешения в функции качества CMP. Свойство γ-разделения также гарантируется алгоритмом Лувена, т.к. в нем говорится, что нет сообществ, которые можно объединить. Такое свойство является несколько более сильным вариантом обычной связности.

1. Самой надежной гарантией алгоритма Лейдена является оптимальность подмножеств. Сообщество является оптимальным по подмножеству, если все подмножества сообщества локально оптимально назначены, т.е. никакое подмножество не может быть перемещено в другое сообщество. Из данной гарантии вытекают γ-плотность и все остальные свойства.

Глава 3  
Реализация алгоритма лейдена

Первая стадия алгоритма — это быстрое локальное перемещение узлов. Алгоритм обходит все вершины в случайном порядке. Для каждого узла определяется, можно ли повысить функцию качества, переместив его из текущего сообщества в другое. Сообщество выбирается исходя из изменения значения функции качества . Перемещение, при котором изменение функции качество максимально определяет сообщество, к которому принадлежит вершина. Переместив вершину в сообщество, алгоритм запоминает всех ее соседей и добавляет их в список для посещения. Так продолжается до тех пор, пока все вершины не будут посещены.

Следующим этапом является уточнение разделения. Изначально на этом этапе все вершины разделены так, что сами по себе образуют сообщества. Т.е. они образуют новое разбиение . Затем алгоритм локально объединяет узлы в . На этом этапе рассматриваются сообщества разбиения . Для каждого рассматриваются все узлы, хорошо с ним связанные. Узлы хорошо связаны с подмножеством, если вес ребер, соединяющих вершину с подмножеством, больше, чем произведение γ, степени вершины и разности между порядком подмножества и степенью вершины. Далее алгоритм рассматривает все подмножества сообщества, хорошо связанные с ним, которые выбираются по такому же принципу, как и хорошо связанные вершины. Из этих подмножеств выбирается случайное, в которое перемещается вершина. Степень случайности выбора сообщества определяется параметром . Как можно заметить, в отличие от фазы локального перемещения узлов, здесь узлы не обязательно сливаются с подмножеством, которое дает наибольший прирост функции качества. Вместо этого узел может быть объединен с любым подмножеством, для которого функция качества увеличивается. Чем больше увеличение функции качества, тем больше вероятность того, что подмножество будет выбрано. Случайность выбора сообщества позволяет более широко исследовать пространство разбиения. Слияния узлов, приводящие к уменьшению функции качества, не учитываются. Это делает этап уточнения более эффективным.

Важно отметить, что слияния выполняются только внутри каждого сообщества разделения . Кроме того, узел объединяется с сообществом в только в том случае, если оба они достаточно хорошо связаны со своим сообществом в . После завершения фазы уточнения сообщества в часто будут разделены на несколько сообществ в ,но не всегда.

Последним этапом алгоритма является агрегация графа. Она основана на уточненном разделении . Узлы агрегированного графа являются сообществами в разбиении . Ребра агрегированного графа являются мультимножествами:

Создавая совокупную сеть на основе , алгоритм имеет больше возможностей для идентификации высококачественных связей.

Все эти этапы продолжаются до тех пор, пока количество вершин в не будет равно количеству сообществ в нем, т.е. каждая вершина должна являться сообществом.

Последним шагом алгоритма является уплощение графа.

Реализованный алгоритм возвращает значение функции качества и полученный после разбиения граф. (Рисунок и код)

Глава 4  
Применение алгоритма лейдена

## 4.1 Рекомендательная система фильмов

Первые данные, на которых был опробован алгоритм, представляют собой рекомендательную систему фильмов. Здесь поиск сообществ играет огромную роль. Поскольку рекомендательные системы сейчас пользуются большим спросом, очень важно подбирать правильные рекомендации. Поиск сообществ в графе фильмов позволит определить интересы пользователей, а также объединить фильмы по каким-либо признакам. Данные состоят из 36472 строк и 8 колонок:

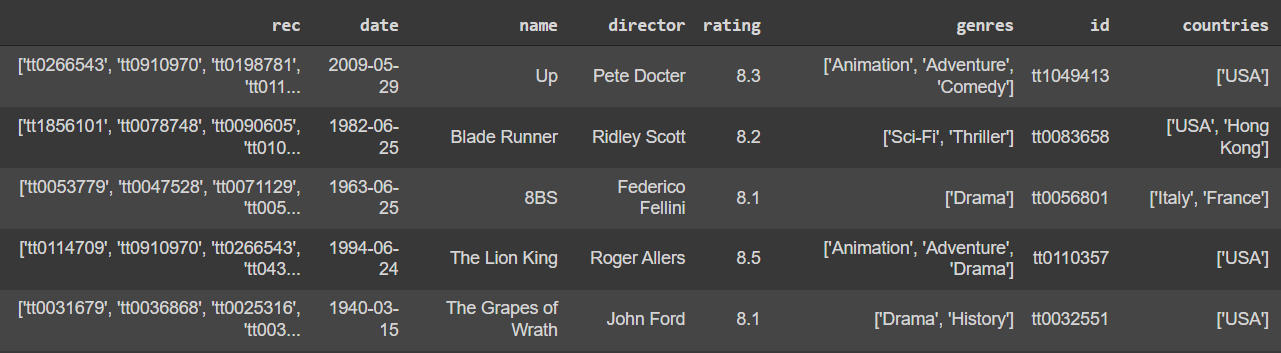
1. rec: содержит уникальные id фильмов, попадающих в рекомендации вместе с данным фильмом
2. date: дата примьеры фильма
3. name: название фильма
4. director: имя режиссера фильма
5. rating: рэйтинг (шкала от 1 до 10, где 1 самы низки1, а 10 самый высокий)
6. genres: жанр фильма
7. id: уникальный id фильма
8. countries: страна выпуска

Рисунок 4.1 Внешний вид данных

В таких данных не сложно определить граф. Здесь вершинами будут являться уникальные id фильмов, а ребра будут соединять фильмы, которые рекомендуются при просмотре данного. Таким образом, чтобы составить граф достаточно использовать две колонки: id и rec.

Поскольку все id представляли из себя объекты типа string, была создана отдельная колонка, в которой хранятся id фильмов в виде объектов типа int. Далее тоже самое было проделано с колонкой rec. Затем были образованы ребра в графе, т.е. пары id фильмов, которые рекомендуются при просмотре. Эти пары будут указывать координаты нахождения 1 в матрицы смежности, а все остальные элементы матрицы равны 0.

|  |  |
| --- | --- |
| Параметр разрешения | Значение функции качества |
| 1 | 0,418 |
| 0,5 | 0,598 |
| 0,05 | 0,831 |
| 0,01 | 0,891 |
| 0,001 | 0,929 |
| 0,0005 | 0,936 |
| 0,0001 | 0,940 |
| 0,00001 | 0,799 |

Таблица 4.1 Зависимость функции качества от параметра разрешения для рекомендательной системы фильмов

Можно заметить, что, чем меньше значение параметра разрешения, тем больше значение функции качества. Однако при 0,00001 значение модульности снова уменьшается.

## 4.2 «Отверженные»

Еще одни данные, на которых был использован алгоритм, содержат в себе взвешенную сеть взаимосвязей героев романа Виктора Гюго «Отверженные». Этот граф является примером простого социального взаимодействия людей, что является одной из самых популярных задач, где применяется алгоритм. Здесь вершинами являются id героев, а ребра соединяют любую пару персонажей, появляющихся в одной главе. Вес ребра определяет количество таких взаимосвязей.

|  |  |  |
| --- | --- | --- |
| Параметр разрешения | Степень случайности | Значение функции качества |
| 0,8 | 6 | 0,378 |
| 0,8 | 5 | 0,396 |
| 0,2 | 3 | 0,481 |
| 0,2 | 7 | 0,489 |
| 0,15 | 7 | 0,493 |
| 0,11 | 1 | 0,526 |
| 0,108 | 1 | 0.514 |

Таблица 4.2 Зависимость функции качества от параметра разрешения и степени случайности для графа «Отверженные»

Данный граф оказался более чувствителен к параметру степени случайности. Здесь как и в первом случае функция качества выше при меньшем значении параметра разрешения, однако при значении 0,108 качество сново ухудшается.

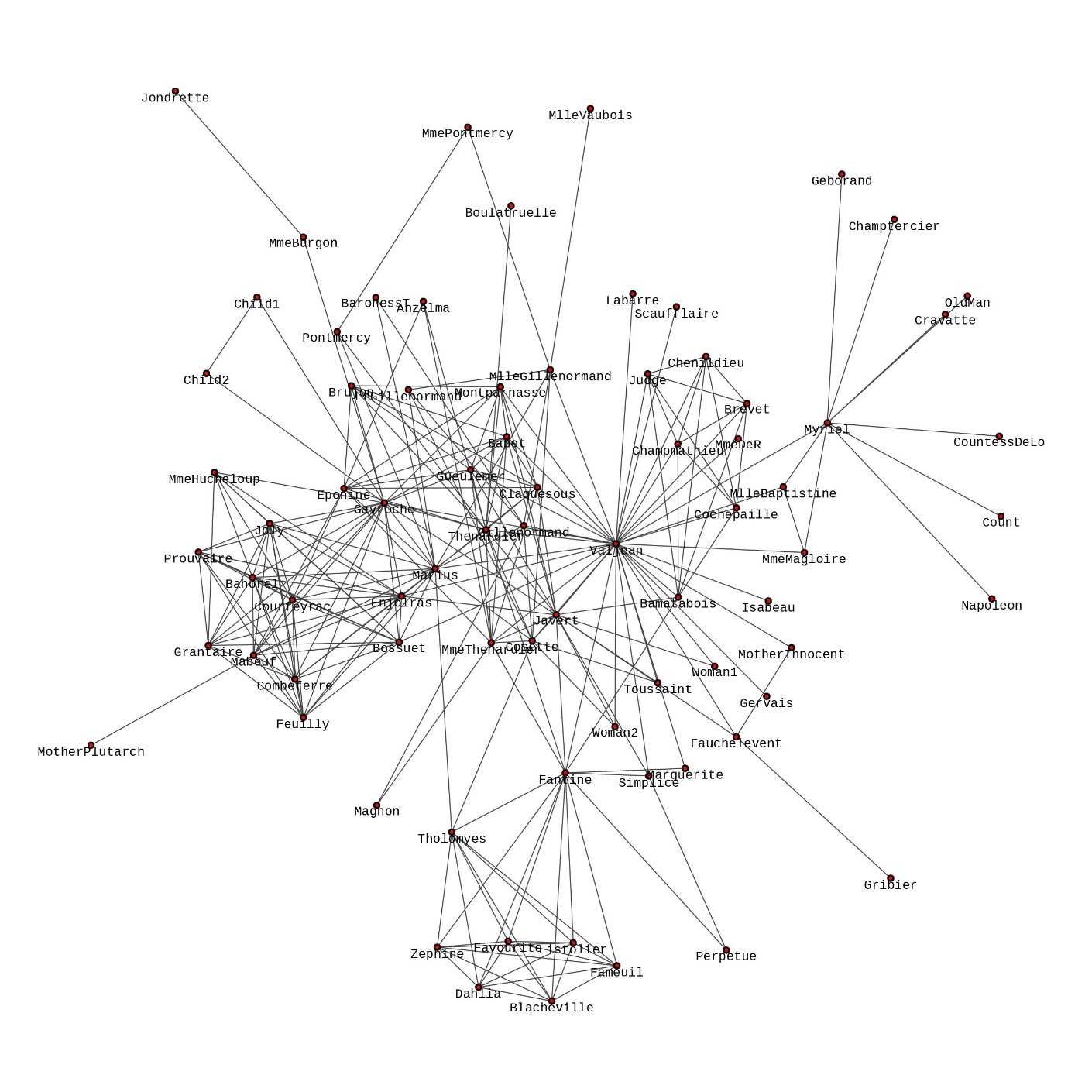


Рисунок 4.2 Граф «Отверженные»

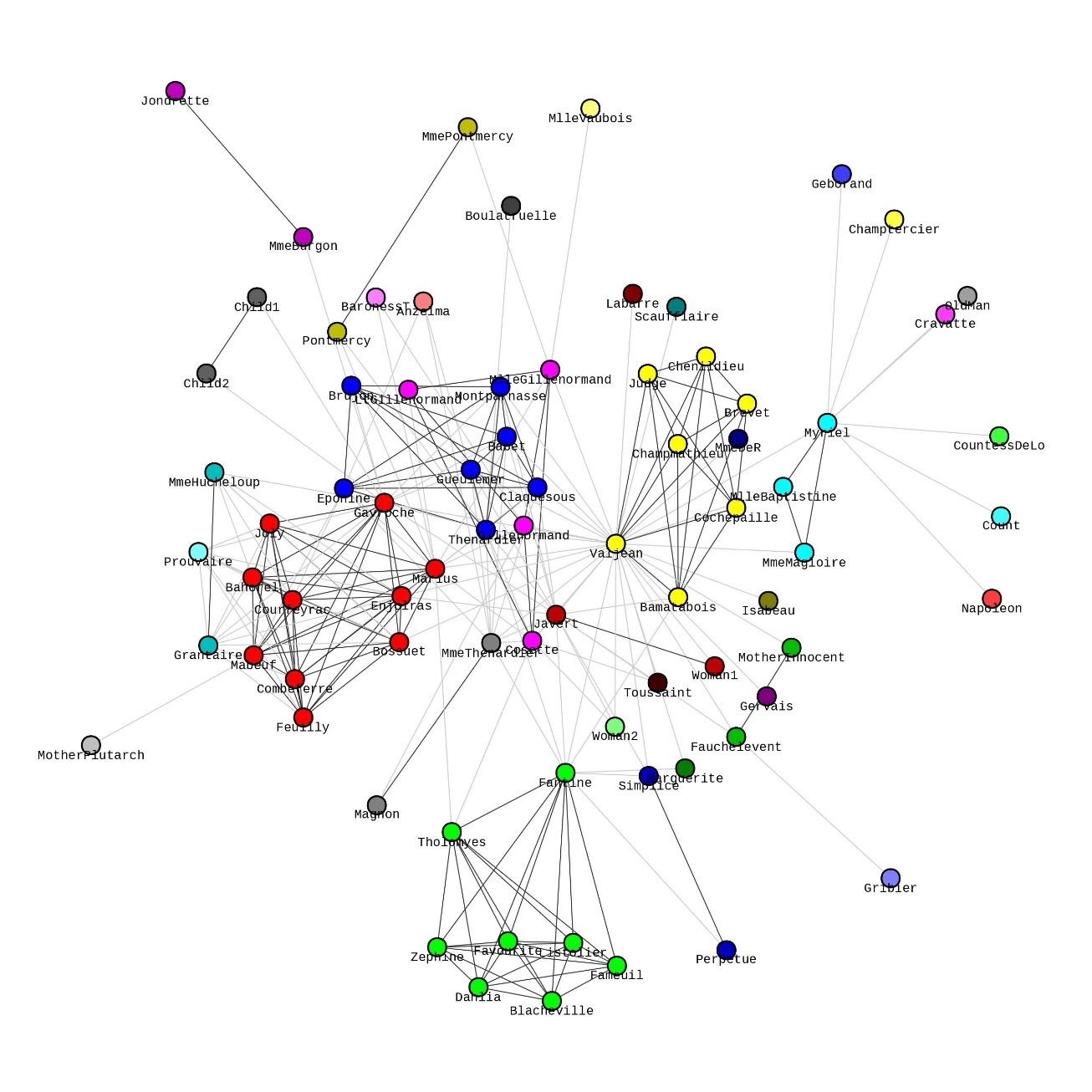


Рисунок 4.3 Граф «Отверженные» после разбиения с параметром разрешения 0,11; степью случайности 1.

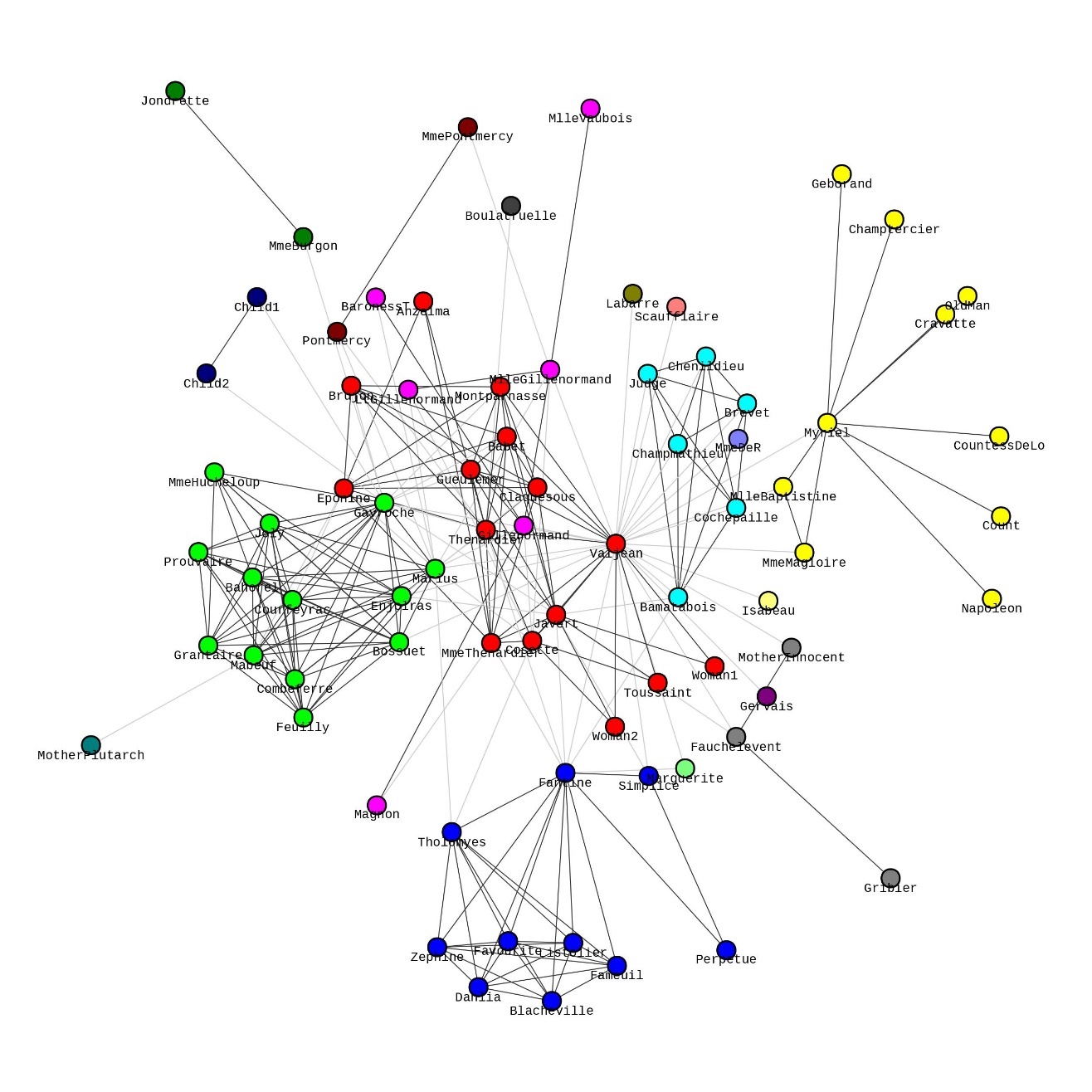


Рисунок 4.4 Граф «Отверженные» после разбиения с параметром разрешения 0,8; степью случайности 6.

## 4.3 Футбол

Следующий пример содержит сеть игр по американскому футболу между различными колледжами. Вершинами являются названия команд, а ребра соединяют команды, которые играли между собой. Таблица и описание

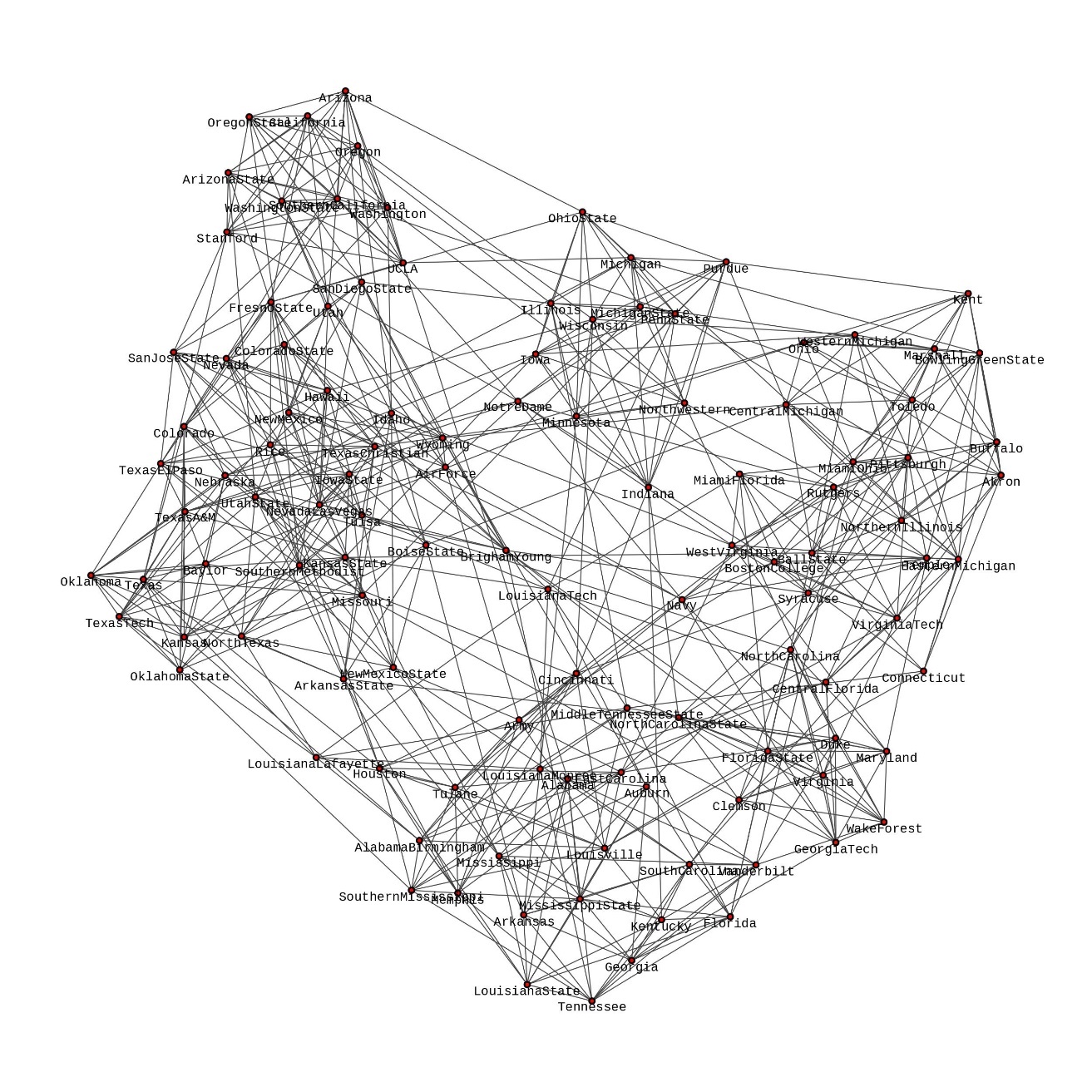


Рисунок 4.5 Граф «Футбол»

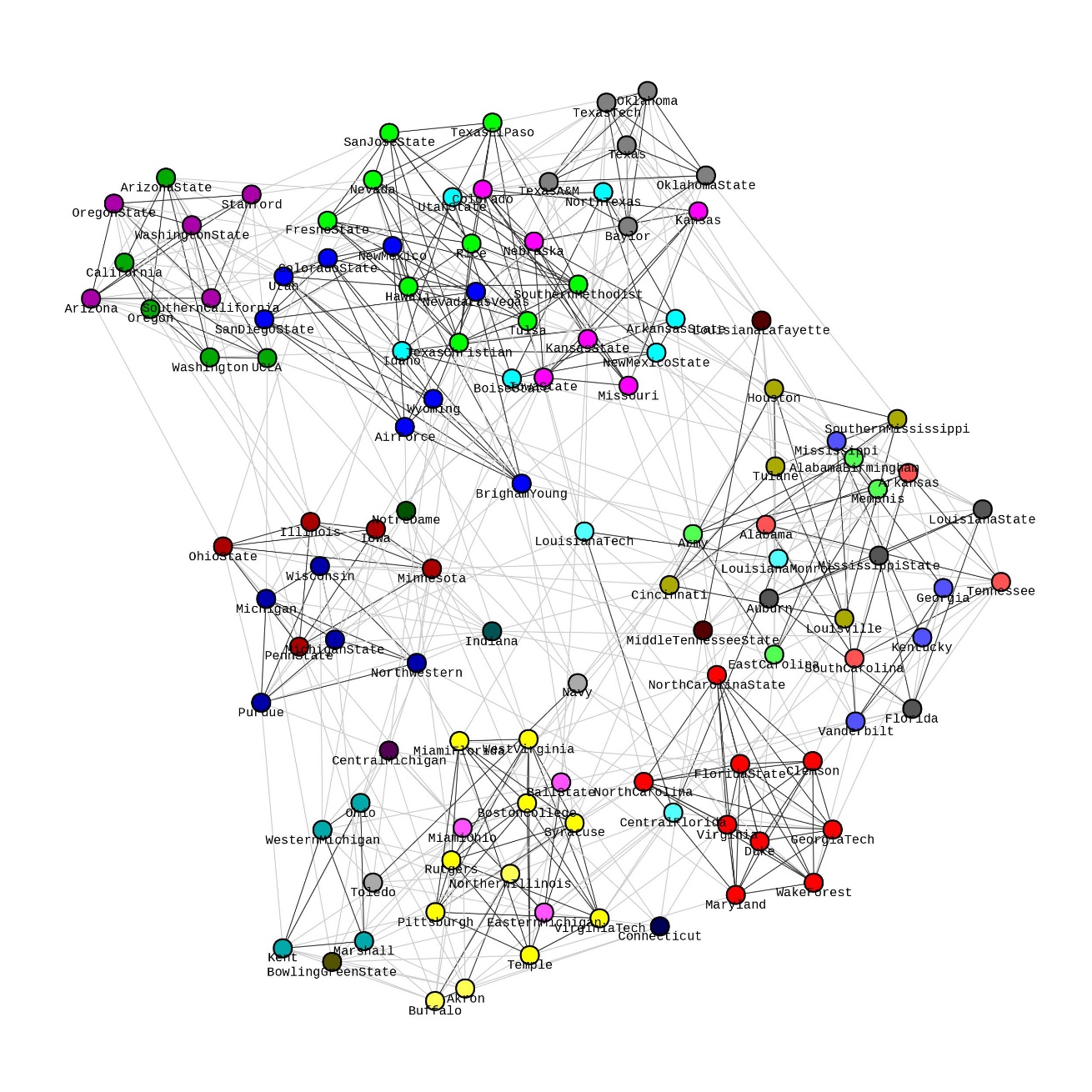


Рисунок 4.6 Граф «Футбол» после разбиения с параметром разрешения 0,8, степенью случайности 5

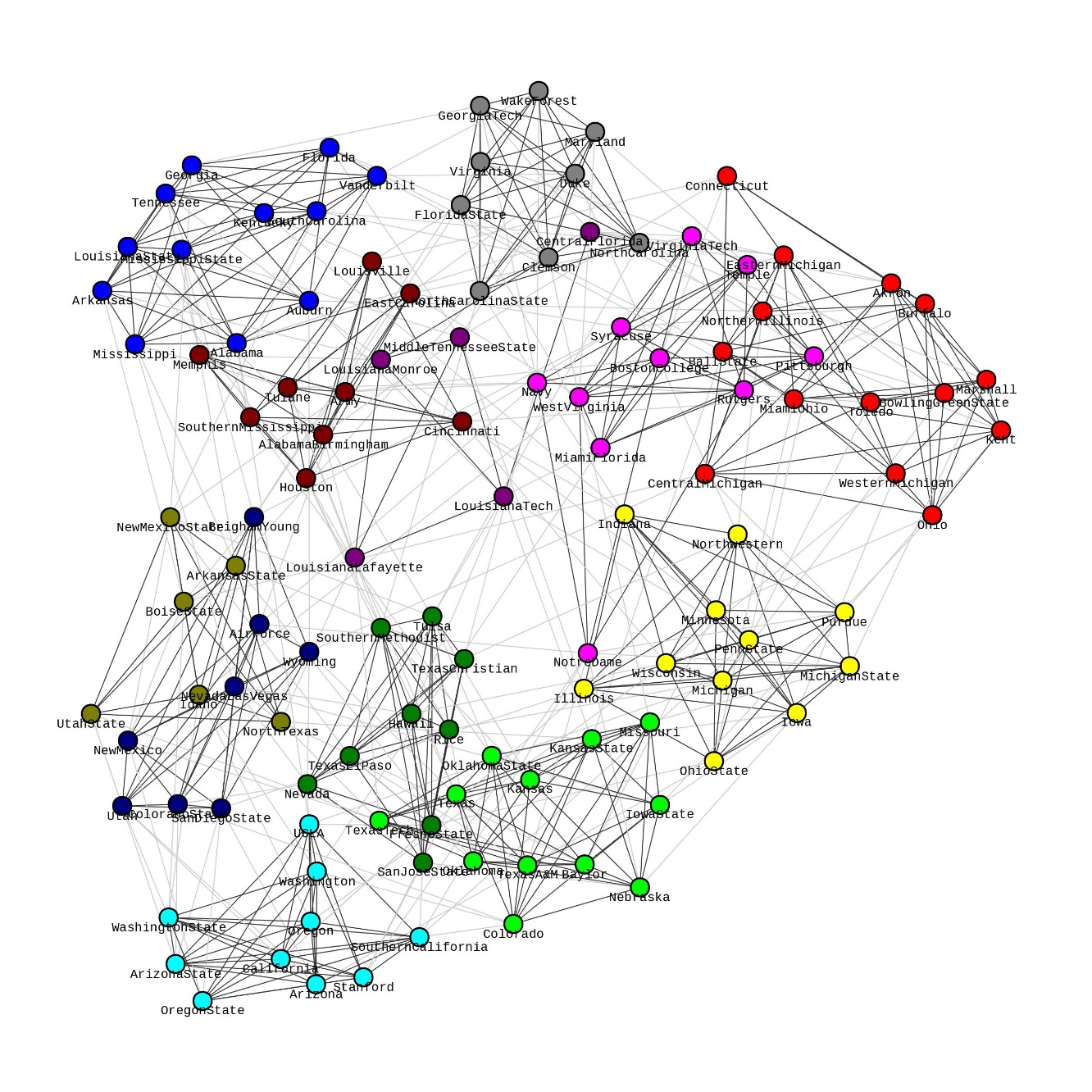


Рисунок 4.7 Граф «Футбол» после разбиения с параметром разрешения 0,1; степью случайности 41.

заключение

В данной работе был исследован и реализован алгоритм Лейдена. Далее он был протестирован на нескольких графах.

Список использованной литературы

From Louvain to Leiden: guaranteeing well-connected communities [Электронный ресурс] – Режим доступа: <https://www.nature.com/articles/s41598-019-41695-z> – Дата доступа: 26.03.2019.

Narrow scope for resolution-limit-free community detection [Электронный ресурс] – Режим доступа: <https://arxiv.org/pdf/1104.3083.pdf> – Дата доступа: 02.08.2011.