## Многоуровневый подход

### Недостатки FD

Рассматривая классический алгоритм FD можно заметить, что с ростом числа вершин в графе время его итерации растёт с квадратом от количества вершин. Так же при большом размере графа и малой области разворачиванию графа могут препятствовать ограничения области.

### Подходы к улучшению

Ухудшению производительности связанному с ростом числа вершин либо с количество близлежащих вершин (находящихся в определённом радиусе) можно препятствовать алгоритмически. А именно при расчёте сил учитывать только близлежащие вершины.

Невозможность размещения в малой площади можно предотвратить, введя масштабирование и возможность размещения вершин между итерациями в неограниченной области.

Однако существенно отличающимся подходом является сжатие графа для расчётов, и его дальнейшее восстановление с целью получения размещения исходного графа.

### Многоуровневый подход

Суть алгоритма состоит в том, что в несколько этапов путём отождествления некоторой группы вершин одной вершине нового графа и дальнейших расчётах на меньшем графе получить результат для исходного графа.

Основными моментами в многоуровневом алгоритме стоит отметить:

* Редукцию графа
* Расчет позиций редуцированного графа
* Восстановление исходного графа
* Локальное уточнение восстановленных групп вершин

Немало важной особенностью является глубина сжатия и число итераций на каждом этапе. Так же можно производить несколько итераций многоуровневой схемы, что позволяет за счёт меньшего числа локальных итераций FD увеличить производительность алгоритма в разы.

#### Сжатие

Рассмотрим следующий алгоритм сжатия позволяющий уменьшить число вершин вдвое.

Шаг 1. Выбрать очередную не помеченную вершину - U

Шаг 2. Найти не помеченную вершину V смежную с текущей вершиной U. Если такой вершины не найдётся, тогда далее использовать только первую вершину U.

Шаг3. Пометить вершины U и V очередной меткой, повторить шаги до тех пор, пока в графе существуют не помеченные вершины.

Псевдокод 1

Таким образом, полученные метки являются номерами вершин сжатого графа.

#### Позиции вершин сжатого графа

В качестве позиции новой, объединенной вершины будем использовать среднее арифметическое позиций вершин исходного графа

Псевдокод 2

После размещения вершин необходимо выполнить несколько итераций FD а затем приступить к восстановлению графа.

#### Восстановление графа

Так как нам известны метки вершин исходного графа, мы можем разместить все вершины в соответствии с новыми координатами и группой вершин помеченных одной и той же меткой. Предлагается выполнить эту случайным образом в окрестности вершины редуцированного графа. Так же необходимо заметить, что на этапе редукции вершине отождествлённой с группой вершин необходимо увеличить радиус. А ребрам сжатого графа увеличить веса в соответствии со схождением в них ребёр исходного графа.

#### Локальное восстановление

Чтобы получить корректное размещение исходного графа после восстановления необходимо произвести несколько итераций FD чтобы случайно размещённые вершины заняли «свои » места.

### Реализация

Позже добавлю интерфейсы разработанных функций

### Эксперименты

Аналогично нужно будет добавить пару картинок в сравнение с FD