



# Курсовая работа

Выделение пересекающихся сообществ  
во взвешенных графах

---

Константин Славнов

16 июня 2016 г.

НИУ Высшая школа экономики,  
Институт проблем передач информации РАН

# Содержание

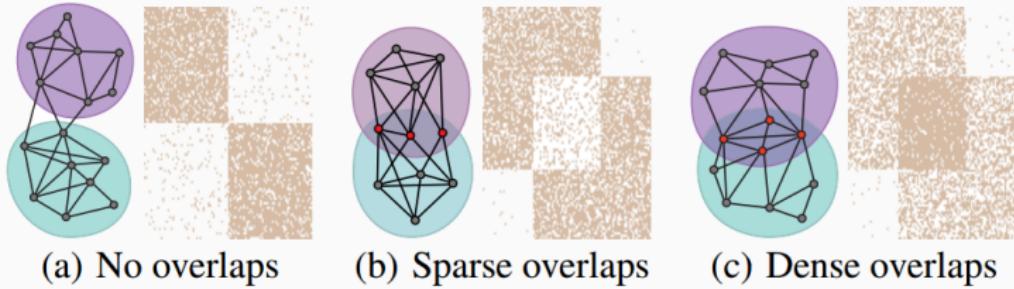
1. Введение, Постановка задачи
2. Метод BigClam
3. Новый метод для взвешенных графов
4. Эксперименты

## Введение, Постановка задачи

---

# Введение

**Задача:** выделение пересекающихся сообществ:

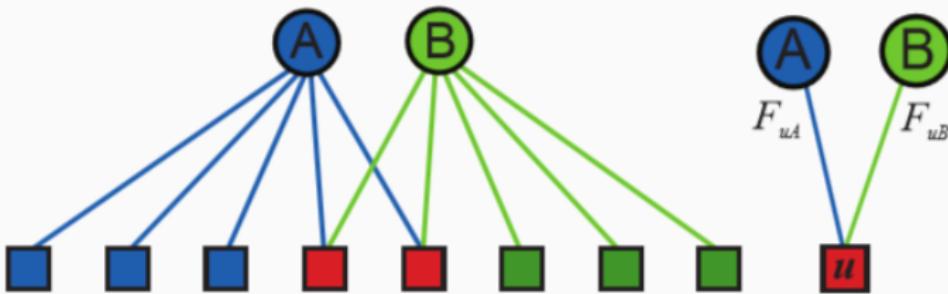


**Цель работы:** Создать новый метод выделения сообществ для взвешенных графов.

**Способ решения:** Обобщить метод BigClam на случай взвешенных графов.

# Постановка задачи

Рассмотрим двудольный граф



- Круглые вершины — сообщества ( $A, B$ ),
- Квадратные вершины — исходного графа ( $u$ ),
- $F_{uA} \geq 0$  — степень принадлежности вершины к сообществу.

**Задача:** Восстановить матрицу  $F$ .

## Метод BigClam

---

## Предположения

1. Введем  $X_{uv}^{(c)}$  — сила взаимодействия  $u, v$  в сообществе  $c$ . Пусть

$$X_{uv}^{(c)} \sim \text{Pois}(F_{uc} \cdot F_{vc})$$

2. Тогда  $X_{uv}$  — сила взаимодействия  $u, v$  в графе

$$X_{uv} \sim \text{Pois}\left(\sum_c F_{uc} \cdot F_{vc}\right) = \text{Pois}(F_u \cdot F_v^T).$$

3. Считаем, что ребро появляется если  $X_{uv} > 0$ . Тогда

$$p(u, v) = \mathbb{P}(X_{uv} > 0) = 1 - \exp(-F_u F_v^T),$$

# Cluster Affiliation Model for Big Networks (BigClam)

Выпишем правдоподобие

$$l(F) = \sum_{(u,v) \in E} \log(1 - \exp(-F_u F_v^T)) - \sum_{(u,v) \notin E} F_u F_v^T \rightarrow \max_{F \geq 0}.$$

Оптимизация методом блочного координатного спуска:

$$\nabla l(F_u) = \sum_{v \in \mathcal{N}(u)} F_u \frac{\exp(-F_u F_v^T)}{1 - \exp(-F_u F_v^T)} - \sum_{v \notin \mathcal{N}(u)} F_v^T.$$

Заметим, что

$$\sum_{v \notin \mathcal{N}(u)} F_v^T = \sum_v F_v - F_u - \sum_{v \in \mathcal{N}(u)} F_v.$$

Сложность одной итерации —  $O(\mathcal{N}(u))$ .

# Новый метод для взвешенных графов

---

# Наивный подход

Формально изменим функционал

$$I(F) = \sum_{(u,v) \in E} \log \left( 1 - \exp \left( -\frac{F_u F_v^T}{w_{uv}} \right) \right) - \sum_{(u,v) \notin E} F_u F_v^T \rightarrow \max_{F \geq 0}.$$

## Недостатки.

1. Наличие и отсутствие ребра вносят разный вклад в функционал.
2. Нельзя использовать для генерации графов.
3. Нет вероятностной интерпретации.

## Sparse Gamma Model. Предположения

Предположим, что данные генерируются по следующему алгоритму.

1. В графе создается ребро  $(u, v) \in E$  с вероятностью

$$1 - \exp(-\gamma F_u F_v^T)$$

2. Для созданных ребер генерируется их вес

$$w_{uv} \sim \Gamma \left( \sum_c F_{uc} F_{vc} + 1, 1 \right).$$

$\gamma \geq 0$  — дополнительный параметр модели.

$\Gamma(\theta, k)$  — Гамма распределение.

# Sparse Gamma Model. Анализ модели

Выпишем правдоподобие

$$\sum_{(u,v) \in E} \log P_\Gamma(w_{uv}) + \sum_{(u,v) \in E} \log (1 - \exp(-\gamma F_u F_v^T)) - \gamma \sum_{(u,v) \notin E} F_u F_v^T \rightarrow \max_{F \geq 0}.$$

$$\log P_\Gamma(w_{uv}) = [-\log \Gamma(F_u F_v^T + 1) + F_u F_v^T \cdot \log w_{uv} - w_{uv}].$$

Последние 2 слагаемых — это оригинальная BigClam модель для матрицы  $\sqrt{\gamma}F$ .

- Модель — комбинация BigClam и Gamma Модели весов.
- Скорость вычисления производной —  $O(\mathcal{N}(u))$ .
- Учитывает взвешенные ребра.
- Универсальность. Гамма заменяется на любое другое аддитивное распределение.

## Эксперименты

---

# Данные и методы

Модельные данные (бенчмарк от Lancichinetti).

Варьируется  $x$  — доля вершин в пересекающихся частях сообществ.

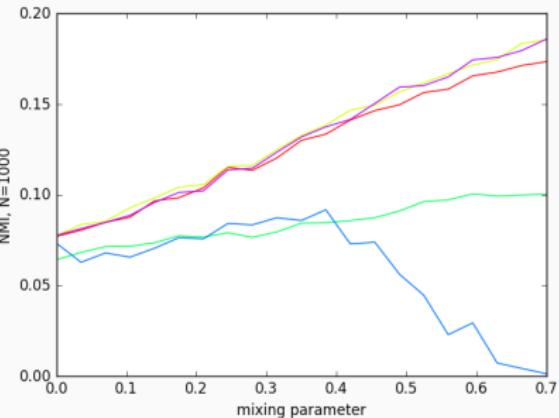
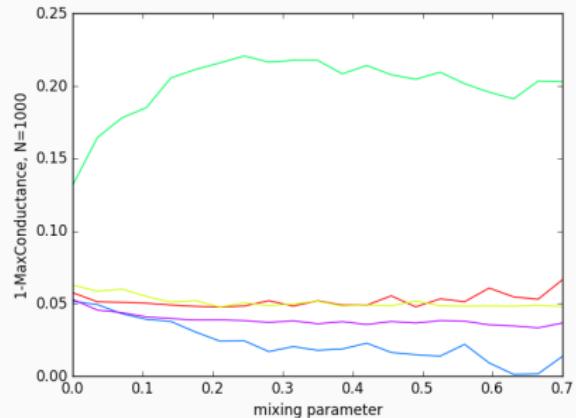
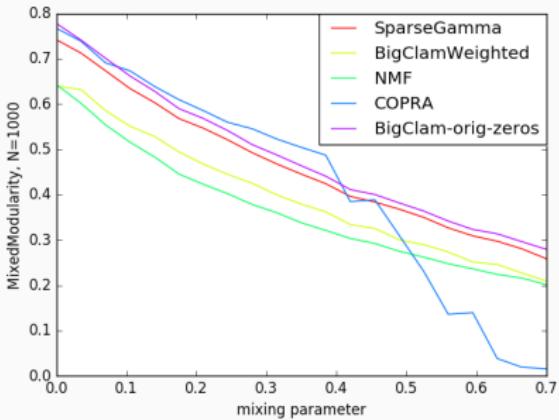
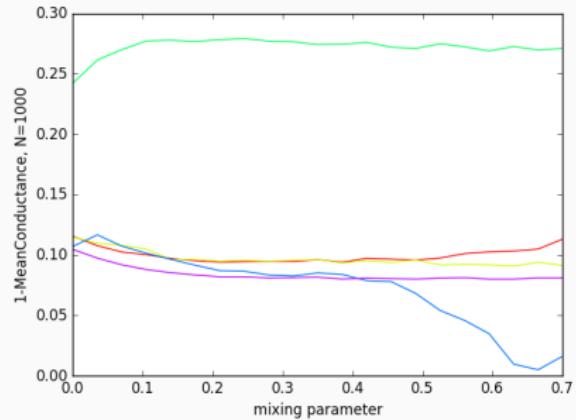
**Методы:**

1. **SparseGamma** — Разреженная гамма модель.
2. **BigClamWeighted** — наивный взвешенный BigClam.
3. **BigClam-orig-zeros** — оригинальный BigClam.
4. **COPRA** — label propagation для пересекающегося случая.
5. **NMF** — Неотрицательное матричное разложение.

**Метрики:**

Conductance(Среднее и максимум), Normalized Mutual Information,  
Модулярность.

# Результат

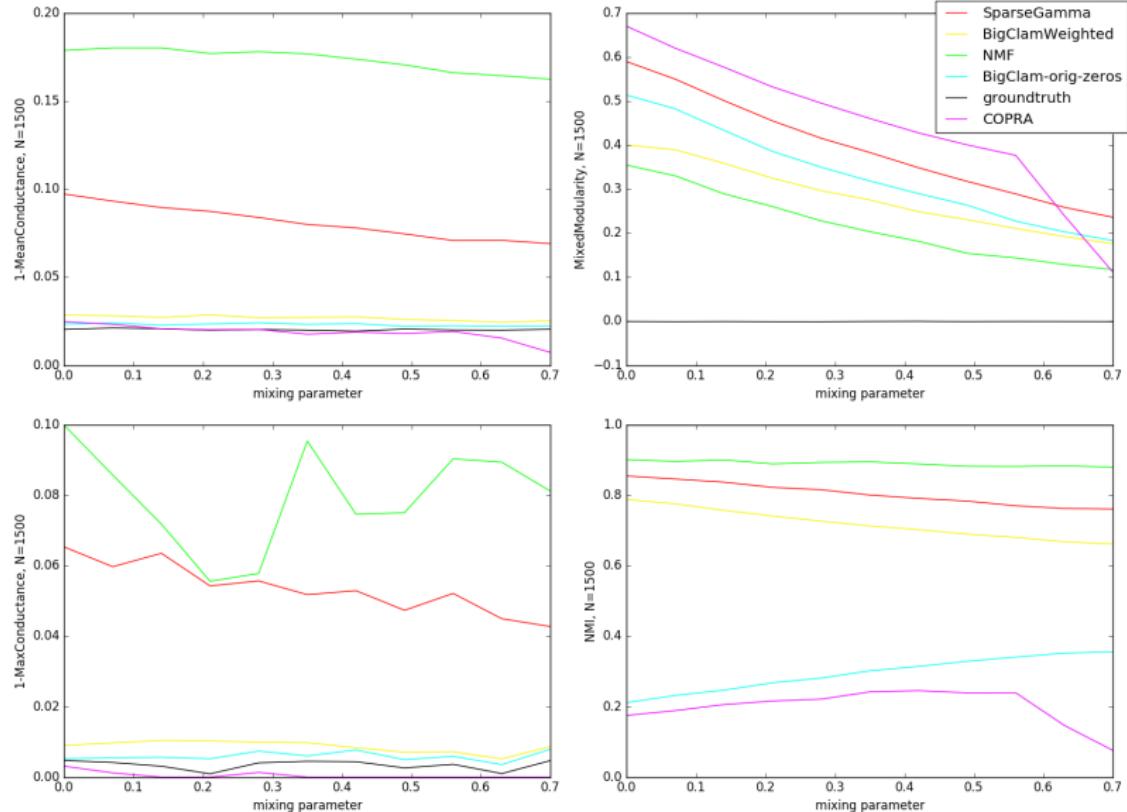


## Наблюдения.

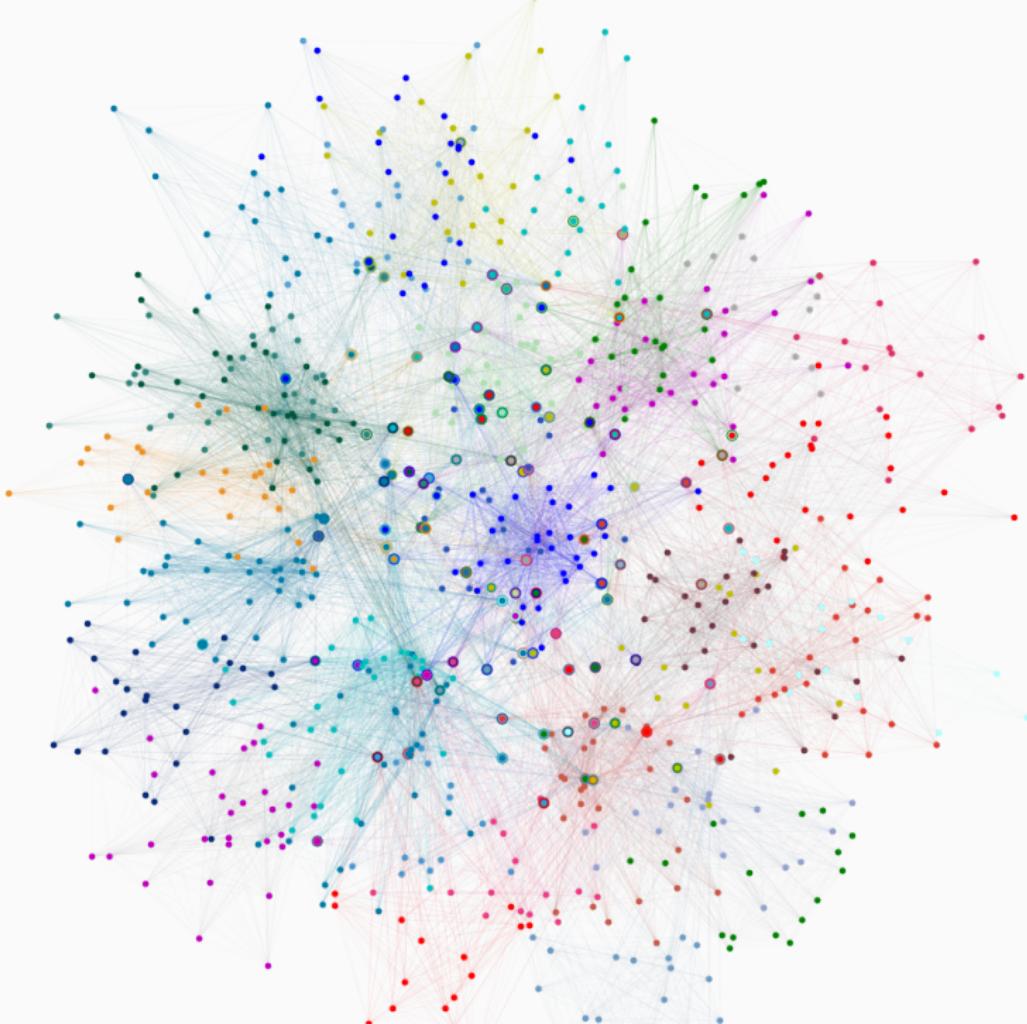
1. *NMF* имеет подавляющее преимущество по функционалам связанными с проводимостью, но не по NMI.
2. *COPRA* с некоторого момента начинает выдавать одно большое сообщество.
3. С ростом параметра  $\chi$  алгоритмы начинают лучше справляться с поставленной задачей.

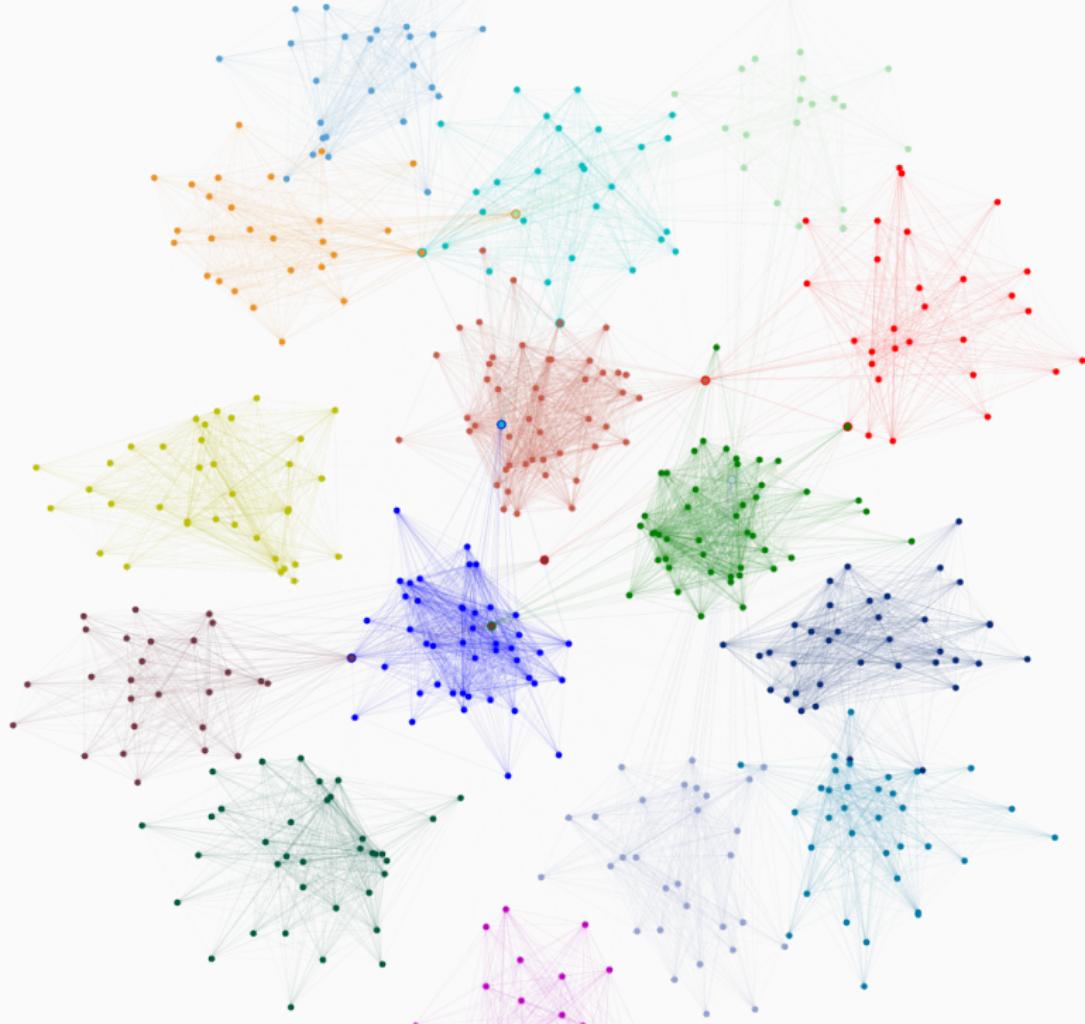
*SparseGamma* и *BigClamWeighted* ведут себя так же как *BigCLam*, принципиального улучшения не достигнуто.

## Результат 2











## Про инициализацию

---

## Оригинальный подход

Введем метрику  $\phi(S)$  — проводимость или Conductance  $S \subset V$ .

$$\phi(S) = \frac{\text{cut}(S)}{\min(\text{vol}(S), \text{vol}(\bar{S}))}$$

Выбираем эго-графы, которые достигают локального максимума  $\phi(S)$ , сортируем по возрастанию. Выбираем  $K$  первых для инициализации. Т.е. для  $S_1, \dots, S_K, i \in V$

$$F_{ij} = \begin{cases} 1, & \text{если } i \in S_j; \\ 0, & \text{иначе.} \end{cases}$$

### Недостатки.

1.  $F$  детерминирована.
2.  $F_{uv} \in \{0; 1\}$ , большая часть нули. А  $F = 0$  — точка локального минимума.
3. Часто 2 или больше  $S_i$  лежат в одном сообществе.

## Новый подход

Дополнительная пенализация.  $R$  — доля уже выбранных вершин  $F_{\text{selected}}$ , которые попали в рассматриваемый эго-граф  $F_{\text{ego}_i}$ .

$$R = \gamma \cdot \frac{{F_{\text{selected}}}^T F_{\text{ego}_i}}{\|F_{\text{selected}}\|}.$$

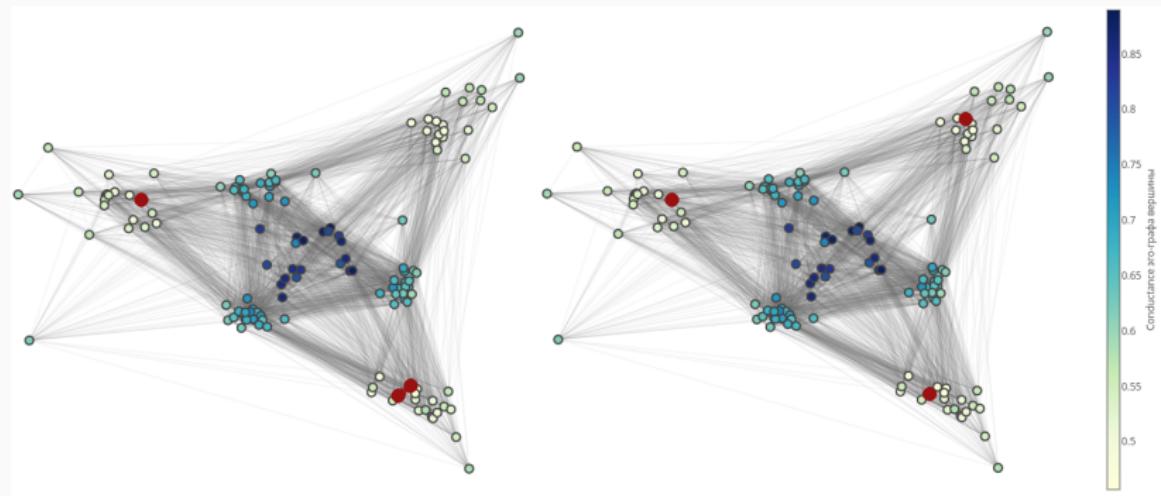
Вместо нулей — равномерный шум в диапазоне  $[0; 0, 1]$ .

## Новый подход

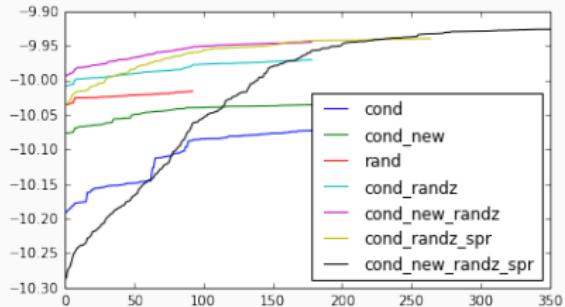
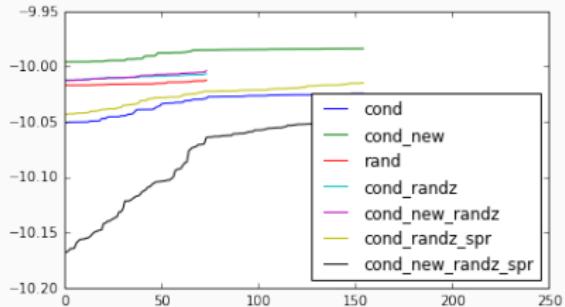
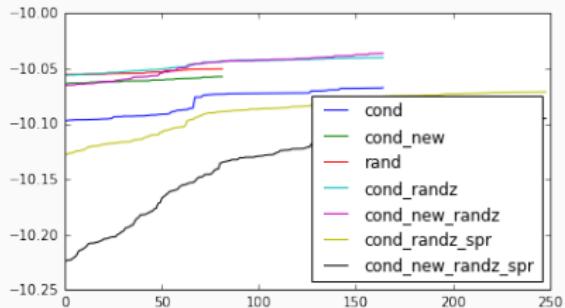
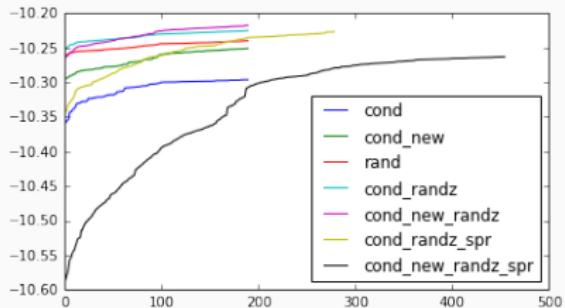
Дополнительная пенализация.  $R$  — доля уже выбранных вершин  $F_{\text{selected}}$ , которые попали в рассматриваемый эго-граф  $F_{\text{ego}_i}$ .

$$R = \gamma \cdot \frac{{F_{\text{selected}}}^T F_{\text{ego}_i}}{\|F_{\text{selected}}\|}.$$

Вместо нулей — равномерный шум в диапазоне  $[0; 0, 1]$ .



# Эксперименты



# Расшифровка названий методов

Обозначение	Описание
<i>rand</i>	Инициализация равномерным шумом от 0.75 до 1.25
<i>cond</i>	Инициализация в локальных максимумах проводимости (стандартный метод)
<i>cond_new</i>	Новый метод со штрафом за пересечение с уже выбранными вершинами
<i>cond_randz</i>	Дополнительно заменяем нули из метода *cond* на значения от 0 до 0.1
<i>cond_new_randz</i>	Дополнительно заменяем нули из метода *cond_new* на значения от 0 до 0.1
<i>cond_randz_spr</i>	Применяем метод <i>cond</i> . Соседние с найденными эгографами вершины получают половину его веса. Затем заменяем нули матрицы $F$ на значения от 0 до 0.1
<i>cond_new_randz_spr</i>	Применяем метод <i>cond_new</i> . Соседние с найденными сообществами вершины получают половину его веса. Затем заменяем нули матрицы $F$ на значения от 0 до 0.1