Методы разложения графов и их приложения

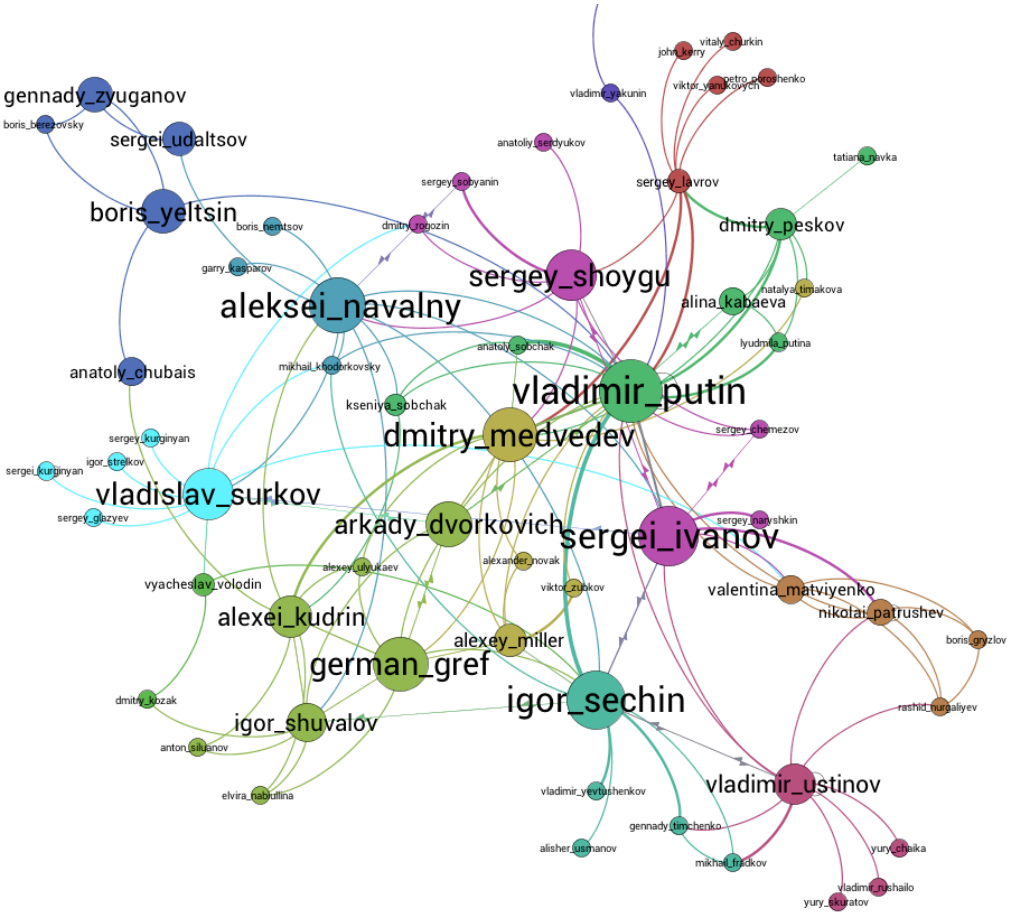
Аспирант 1ого года Уткина Ирина

Научный руководитель: Малышев Дмитрий Сергеевич

Цель и задачи

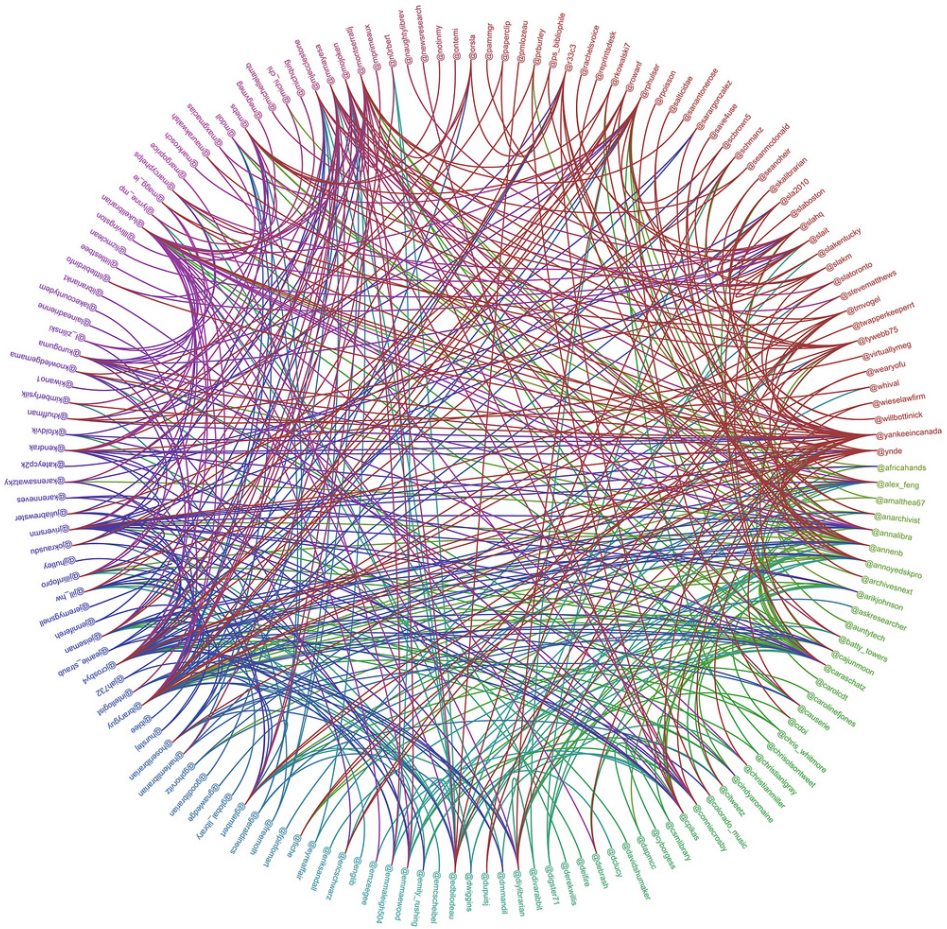
- Цель: Создать алгоритм для решения задачи нахождения максимальной клики на основе разложения графа
- Задачи:
  - Изучить методы разложения графов
  - Применить выбранный метод разложения для препроцессинга входного графа
  - Создать алгоритм для решения задачи нахождения максимальной клики на основе новых данных
  - Сравнить результаты

Причины


gennady_zyuganov
boris_berezovsky
sergei_udaltsov
boris_yeltsin
boris_nemtsov
garry_kasparov
aleksei_navalny
anatoly_chubais
mikhail_khodorkovsky
sergey_kurginyan
igor_strelkov
sergei_kurginyan
vladislav_surkov
sergey_glazyev
vyacheslav_volodin
alexei_kudrin
dmitry_kozak
igor_shuvalov
anton_siluanov
elvira_nabiullina
german_gref
arkady_dvorkovich
alexey_ulyukaev
alexander_novak
viktor_zubkov
alexey_miller
igor_sechin
vladimir_yevtushenkov
alisher_usmanov
gennady_timchenko
mikhail_fridkov
vladimir_ustinov
yury_chaika
vladimir_rushailo
yury_ekuratov
valentina_matviyenko
nikolai_patrushev
boris_gryzlov
rashid_nurgaliyev
sergei_ivanov
sergey_naryshkin
vladimir_putin
dmitry_medvedev
sergey_chemezov
kseniya_sobchak
anatoly_sobchak
lyudmila_putina
sergey_shoygu
alina_kabaeva
natalya_timakova
dmitry_peskov
tatiana_navka
sergey_lavrov
anatoliy_serdyukov
vladimir_yakunin
viktor_yanukovych
petro_poroshenko
john_kerry
vitaly_churkin
sergey_sobyanin
dmitry_rogozin

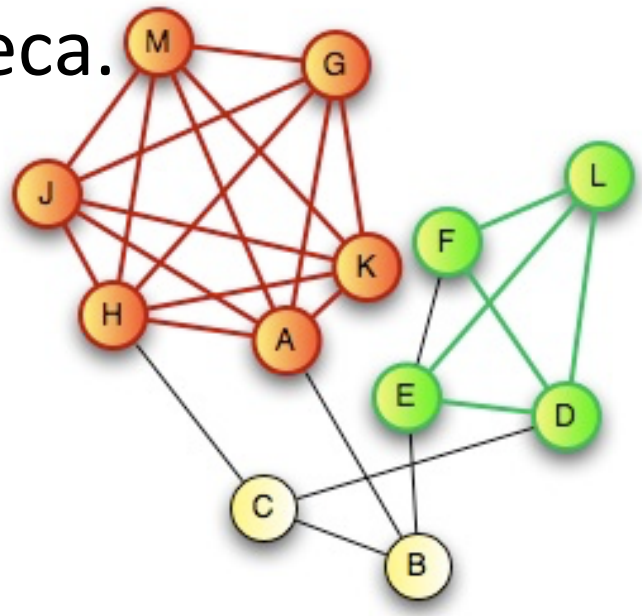
Причины

Причины



Поиск максимальной клики

- Клика графа G(V,E) – набор вершин связанных друг с другом, т.е. полный подграф.
- Максимальная клика – это клика наибольшего размера или веса.

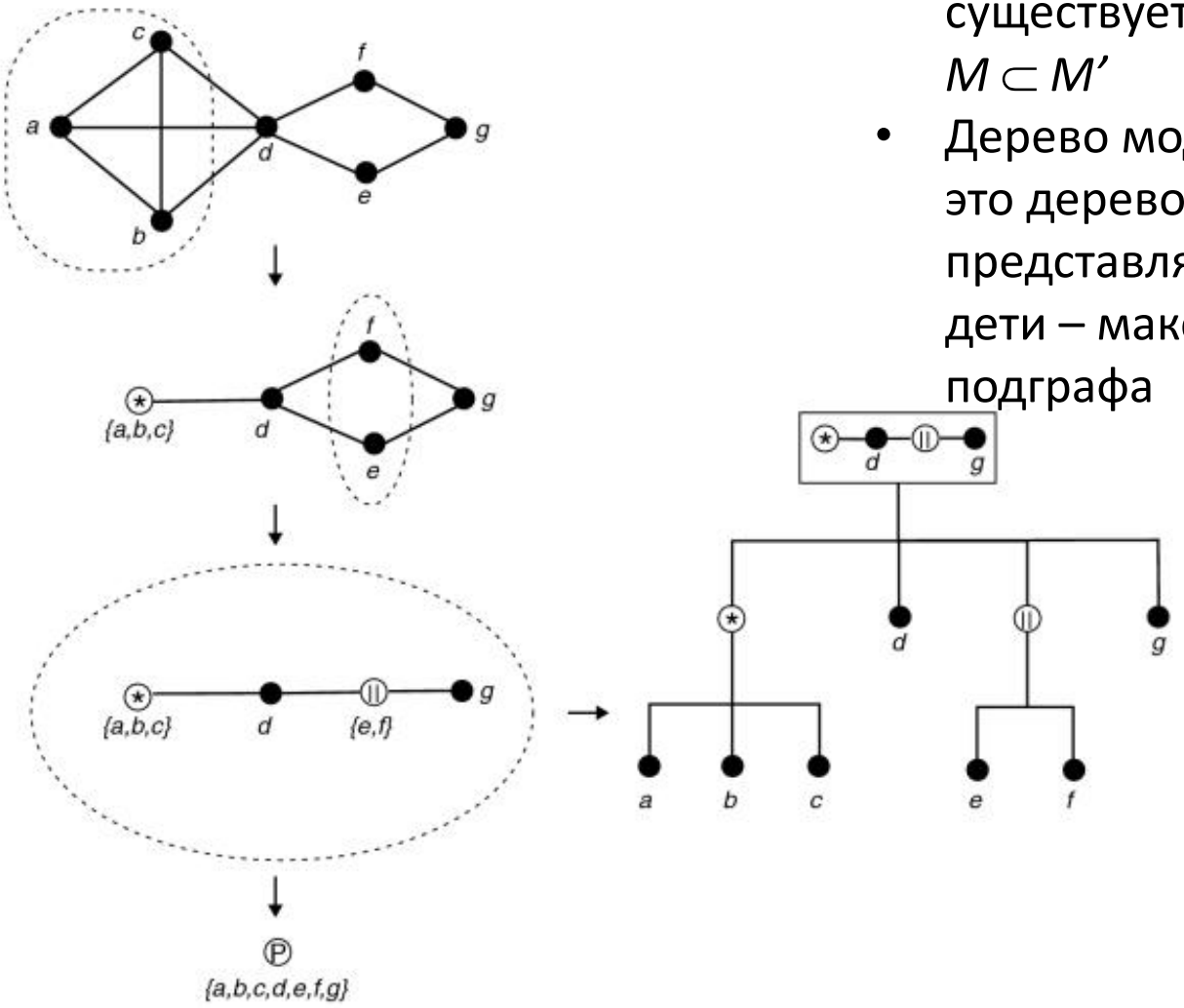


Методы разложения графов

- Разделяющая клика
- Модульное разложение

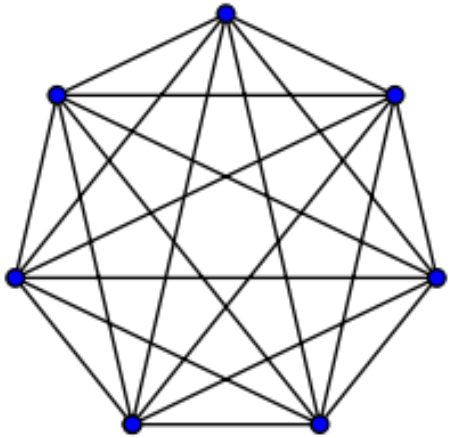
Модульное разложение

- Модуль – набор вершин, у которых одинаковый набор соседей вне этого набора.
- Модуль *M* максимален, если не существует модуля *M'* , такого что $M \subset M'$
- Дерево модульного разложения это дерево, где каждая вершина представляет собой подграф, а его дети – максимальные модули этого подграфа

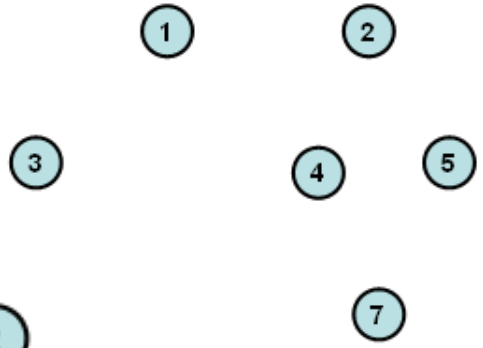


Типы вершин

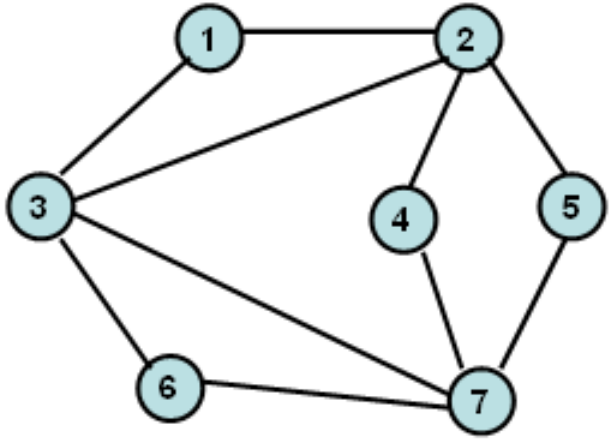
- Series



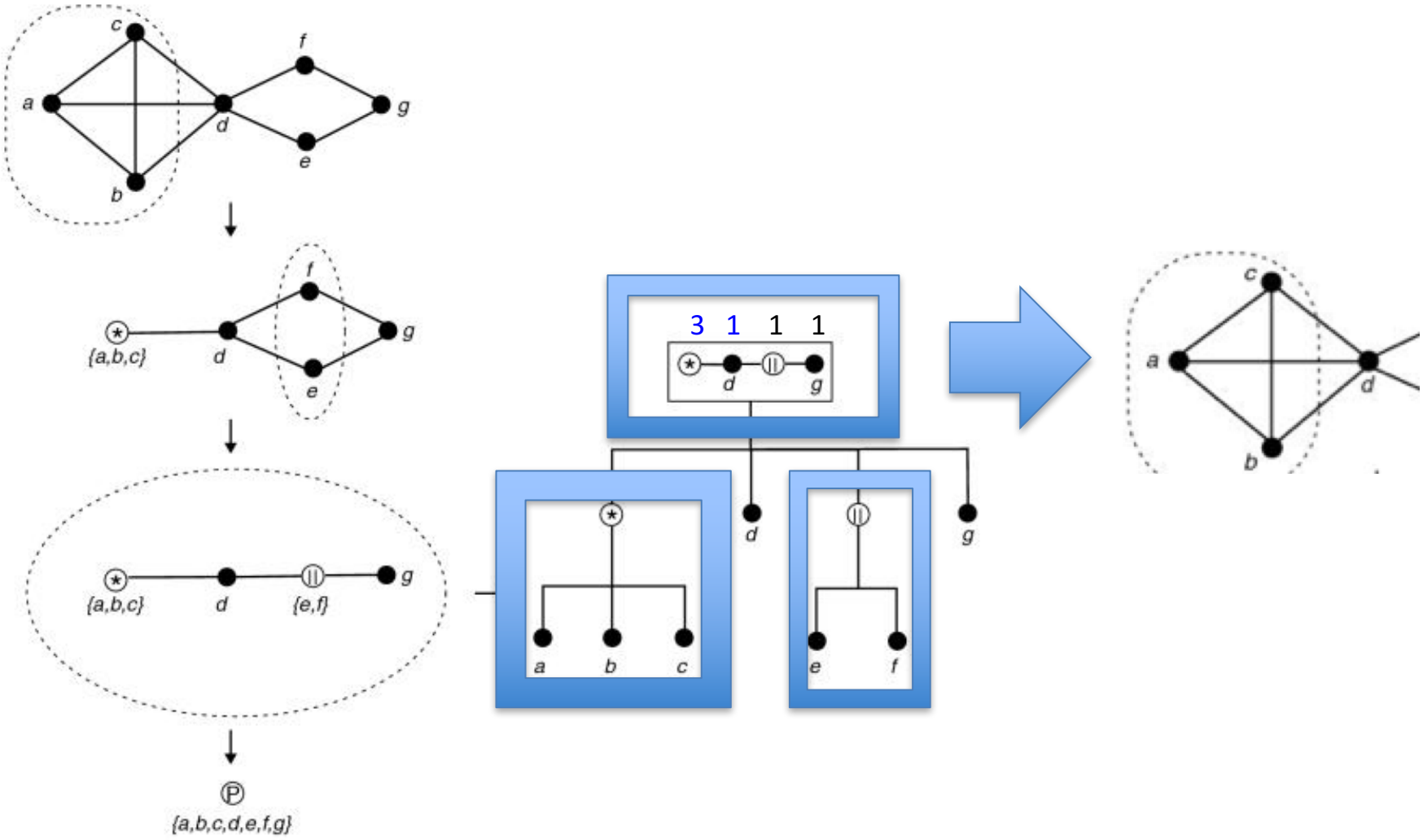
- Parallel



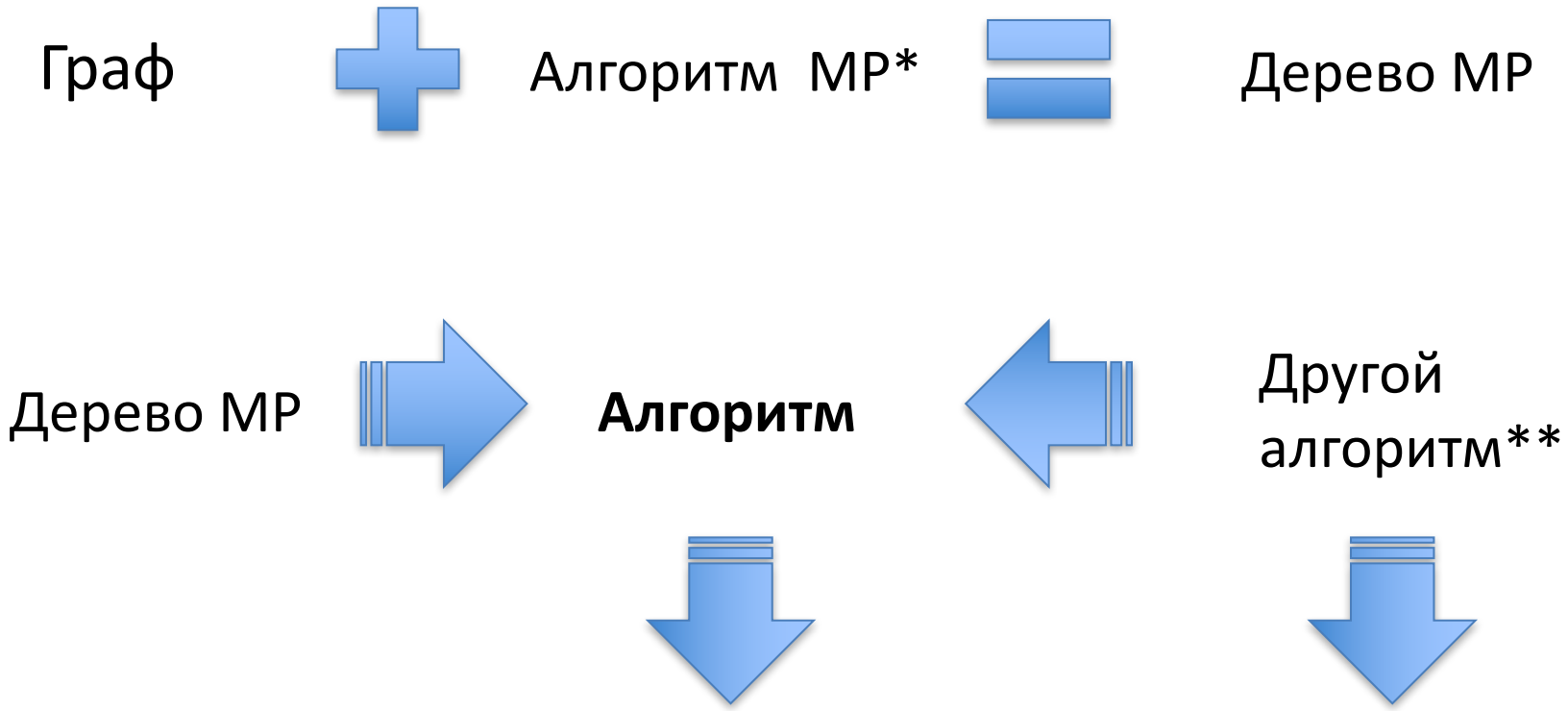
- Prime



Пример



Алгоритм



*Marc Tedder. Derek Corneil. Michel Habib. and Christophe Paul "Simpler Linear-Time Modular Decomposition via Recursive Factorizing Permutations"

** Patric R. J. Ostergard

DIMACS

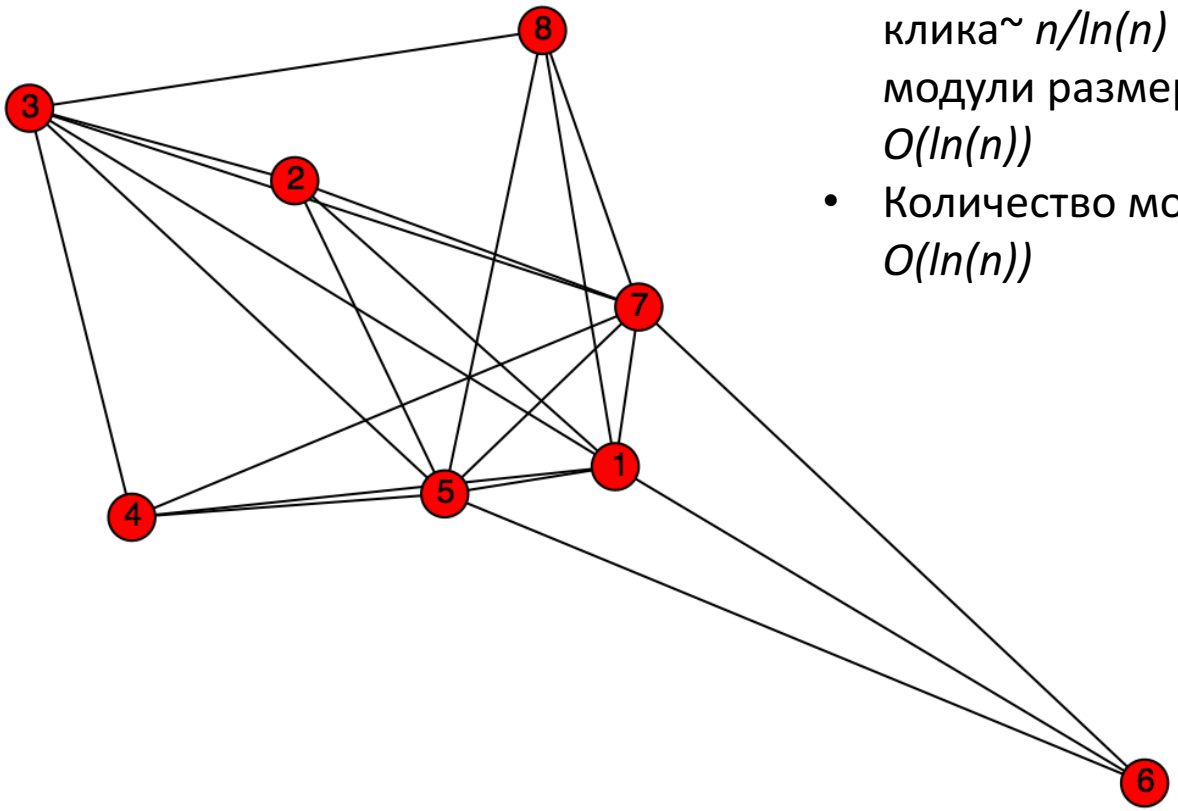
Size	My algorithm (s)	MD (s)	My + MD (s)	Ostergard (s)
c-fat200-1	0.000118	0.068662612	0.068780612	0.001059
c-fat200-2	0	0.104443933	0.104443933	0.001371
c-fat200-5	0	0.226667621	0.226667621	2.61894
c-fat500-1	0.015956	0.147924545	0.163880545	0.002184
c-fat500-10	0.006224	0.680824483	0.687048483	0.011369
c-fat500-2	0.011647	0.216331646	0.227978646	0.003734
c-fat500-5	0.009128	0.324011131	0.333139131	0.006868

Ко-графы

```
def partition(n):
    ns = []
    while n > 0:
        n_i = random.randint(1, n)
        ns.append(n_i)
        n -= n_i
    return ns

def generateCoGraph(n):
    if n > 1:
        ns = list(reversed(partition(n)))

        if random.randint(0,1) == 0:
            for n_i in ns:
                generateCoGraph(n_i)

        else:
            for n_i in ns:
                generateCoGraph(n_i)
            makeClique(ns)
```

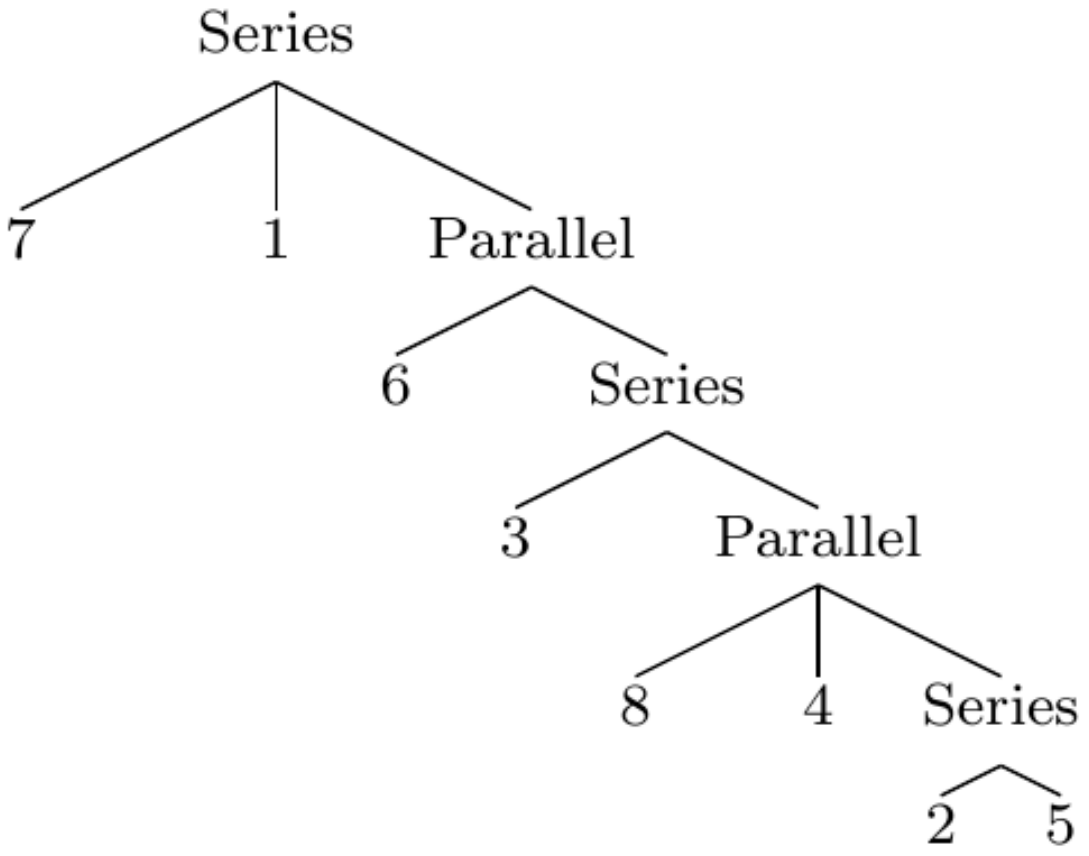
Size	My algorithm (s)	MD (s)	My + MD (s)	Ostergard (s)
500	107 0.000119	0.599508522	0.599627522	107 5.19397
550	219 0.000128	1.126792247	1.126920247	219 0.420204
600	206 0.000123	1.613567235	1.613690235	206 0.385463
650	143 0.000124	1.079984486	1.080108486	37 300
700	197 0.000151	1.49792208	1.49807308	197 0.10201
750	279 0.000184	2.111751452	2.111935452	279 0.096327
800	92 0.00017	0.592107428	0.592277428	37 300
850	155 0.000172	1.634152987	1.634324987	29 300
900	134 0.00016	1.25222059	1.25238059	36 300
1000	241 0.00018	3.105684402	3.105864402	241 4.2414
1050	338 0.000162	4.151108273	4.151270273	116 300
1100	329 0.000182	4.811745176	4.811927176	329 19.4748
1150	408 0.000139	5.756633889	5.756772889	408 0.302615
1200	362 0.000165	5.614880477	5.615045477	362 2.85711
1250	151 0.000193	2.125359674	2.125552674	54 300
1300	292 0.00019	5.384282824	5.384472824	292 12.8855
1350	178 0.000214	2.035435868	2.035649868	63 300
1400	216 0.000186	2.634297357	2.634483357	32 300
1450	279 0.00026	5.657872455	5.658132455	35 300
1500	377 0.000217	9.239033114	9.239250114	377 13.7176
1550	238 0.000268	6.540190934	6.540458934	238 19.8803
1600	409 0.000262	10.753900639	10.75416264	409 48.6752
1650	278 0.00028	10.456971305	10.45725131	41 300
1700	229 0.000327	10.474212124	10.47453912	30 300
1800	543 0.000345	21.152246751	21.15259175	47 300
1850	494 0.000296	18.978203233	18.97849923	43 300
1900	263 0.000303	5.338127994	5.338430994	103 300
1950	355 0.000308	17.914419038	17.91472704	44 300

Граф взаимной простоты



- Максимальная клика~ *n/ln(n)* Все модули размера *O(ln(n))*
- Количество модулей *O(ln(n))*

Графы взаимной простоты



Граф взаимной простоты

```
def createGraph(V):
    for i in V:
        for j in V:
            if gcd(i+1, j+1) == 1:
                addEdge(i, j)
```

Size	My algorithm (s)	MD (s)	My + MD (s)	Ostergard (s)
100	0	0.06906381	0.06906	0.001743
150	0	0.160086907	0.16009	0.004758
200	0	0.311571475	0.31157	0.006921
1000	0.00015	0.830395642	0.83055	300.001
1050	0.000132	2.77094437	2.77108	0.709009
1100	0.000149	2.962479483	2.96263	300
1150	0.00014	1.005088013	1.00523	300
1200	0.000178	1.102902136	1.10308	300
1250	0.000163	3.624508829	3.62467	300
1300	0.000237	6.191885617	6.19212	300
1350	0.000169	3.158776487	3.15895	300
1400	0.000192	0.448514811	0.44871	300
1450	0.000263	5.838179619	5.83844	300
1500	0.000273	8.211194675	8.21147	300
1550	0.000358	1.049262402	1.04962	300
1600	0.000262	1.861131225	1.86139	300
1650	0.000268	2.108732494	2.10900	300
1700	0.000283	13.137433843	13.13772	300
1750	0.000284	6.082156016	6.08244	300
1800	0.000295	2.649852882	2.65015	300
1850	0.00024	1.641608819	1.64185	300
1900	0.000331	22.834692949	22.83502	300
1950	0.000284	15.864648365	15.86493	300
2050	0.000315	19.971808447	19.97212	300
2100	0.000323	6.728151243	6.72847	300
2200	0.000331	3.152012792	3.15234	300
2250	0.000347	3.20890262	3.20925	300
2300	0.000519	26.991591521	26.99211	300
2350	0.000395	26.367774638	26.36817	300
2400	0.000368	17.514787453	17.51516	300
2450	0.00046	41.651295783	41.65176	300

Планы на будущее

- Использовать более новый алгоритм для решения задачи о поиске максимальной клики
- Попробовать ускорить алгоритм М.Теддера и др.
- Попробовать на других графах

Спасибо за внимание!