О методе ускорения поиска кратчайших путей на основе удаления вершин малых степеней

Зизов Вадим Сергеевич, м.н.с
каф. МК ВМК МГУ имени М.В. Ломоносова

Постановка задачи

Рассматривается задача поиска
кратчайших взвешенных путей
в ориентированном разреженном графе
большого размера
(V ~ 70000, log_2(V) ~ 16)
с вещественными весами рёбер
от одного источника ко всем вершинам.

SSSP

Существующие решения

	Предобработка	Запрос (общий граф)	Запрос (планарный граф, V = n, E = O(n))	Особенности
Дейкстра (бинарная куча)	–	O((V + E) log V)	O(n log n)	Просто и эффективно
Дейкстра (куча Фибоначчи)	–	O(E + V log V)	O(n log n)	Сложнее, бо’льшая константа
SPFA (модификация Беллмана-Форда)	–	В среднем O(E log V), В худшем O(VE)	O(n log n) - O(n^2)	Быстр на редких графах. Не гарантирует среднюю оценку.
Торуп (Thorup, 1999)	O(E)	O(E)		Целые положительные числа, большая константа
Henzinger et al. (1997)	O(n)	–	O(n)	Теоретически оптимальный для SSSP на планарных графах
Klein (2005)	O(n log n)	–	O(log^2 n)	Множественные запросы от источников на границе области
Klein, Mozes, Sommer (2010)	O(n log n)	–	O(log^2 n)	Приближённые расстояния
BFS (на дереве)	–	-	O(n) на дереве	Быстрый алгоритм для деревьев.
Contraction Hierarchies	O(n^2)	?	O(log n)	Только точка-точка

C24 Алгоритм

Параметр: k, Вход: граф G = (V, E), w: E -> Float.

1. Рекурсивно удаляем вершины с минимальной степенью не более чем k.

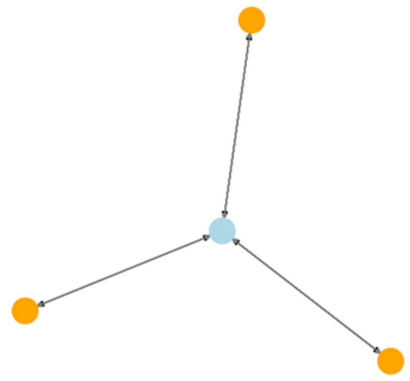
2. Запоминаем границы удалённых кластеров и строим «быстрый» способ расчёта от границы кластера до всех точек

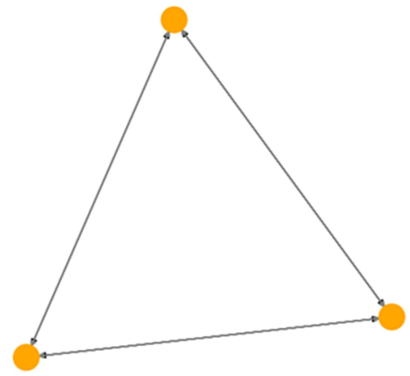
3. Используем двухступенчатый SSSP (например, Dijkstra):

a) сначала локально в кластере,
b) затем глобально по сжатому графу,
c) затем рассчитываем пути до вершин, не попавших в эти расчёты.

C24 Компрессия

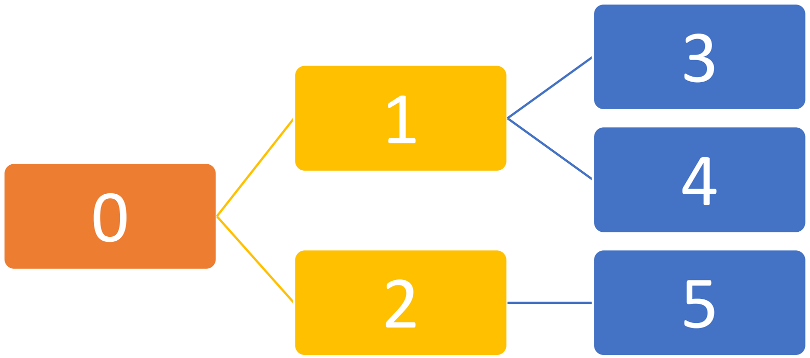
Удаляем вершины со степенью ≤ limit (например, 4).
1. Для каждой удалённой вершины добавляем ребра между её предками и потомками.
2. Сохраняем:
какие вершины были удалены,
новые рёбра и их веса,
при конфликтах выбираем минимальный вес
3. Плюсы:
уменьшение числа рёбер и вершин
сокращение кратчайших расстойний



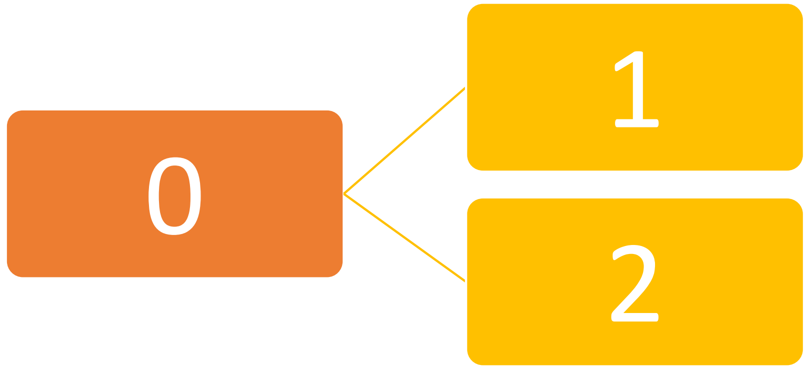


k = 1, рассмотрим подграф с корнем в 0

Удалим все вершины степени 1



Повторим рекурсивно

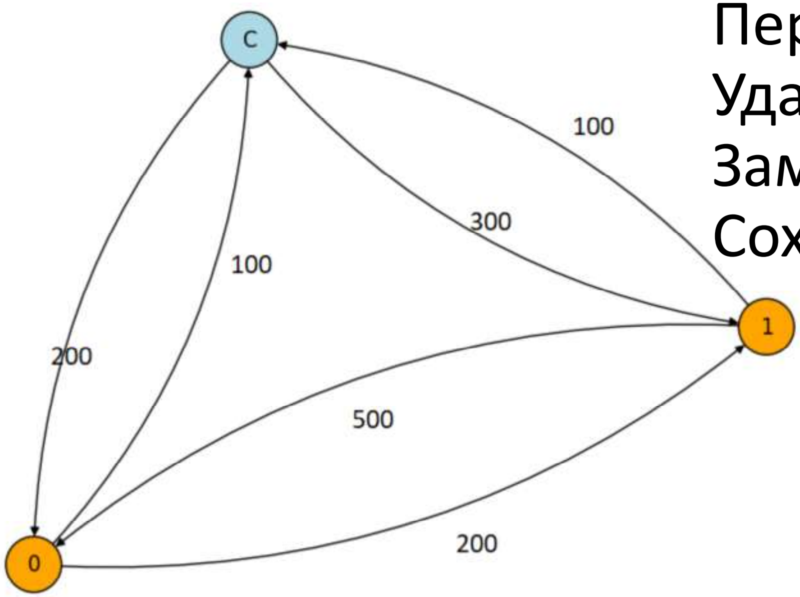


1. Если найдены все пути в графе, и до вершины 0 расстояние P, то O(E) просмотр рёбер даст расстояние до всех вершин дерева
2. Если source находится внутри поддерева, то а O(E) можно найти расстояние до всех вершин в дереве.

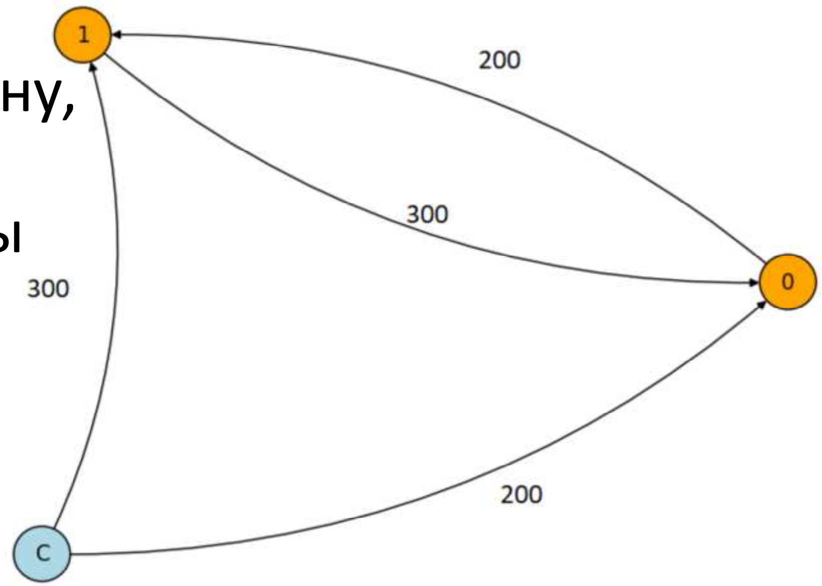
k = 2

Проведём аналогичную процедуру для k = 2, всего N вершин требуется O(kN) операций удаления и добавления рёбер

1. Всё ещё линейный просмотр вершин в очереди



Переход:
Удаляем вершину,
Заменяем веса,
Сохраняем хопы



2. Число рёбер всегда уменьшается либо на 1, либо на 2.
Степени (оставшихся) вершин не увеличиваются.

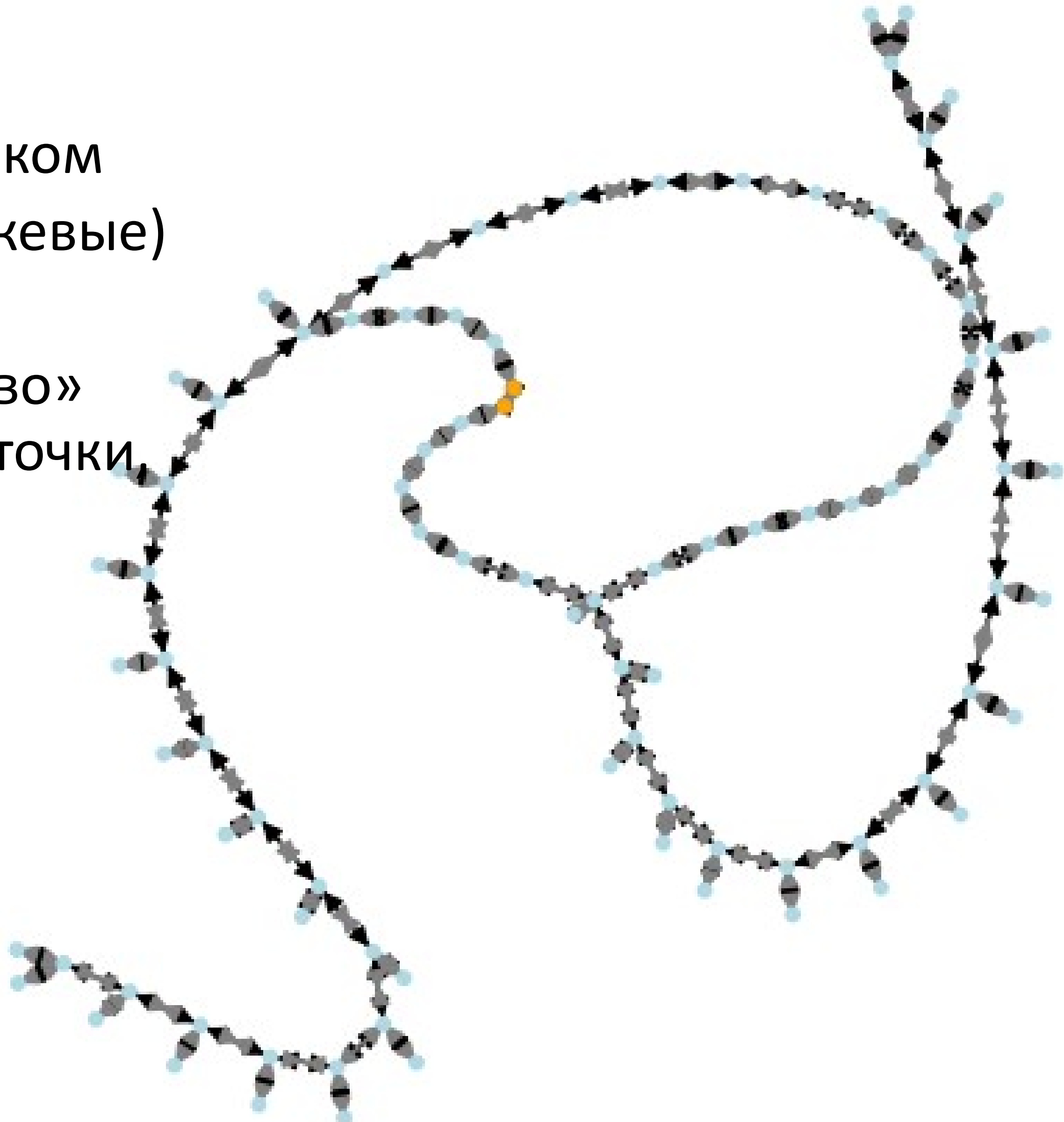
k = 2, Lace-графы

на рисунке подграф при k<=2 целиком

1. Существует k точек входа (оранжевые) (*граница*)

2. Можно представить как «кружево» из дуг, либо имеющих две общие точки, либо не имеющих общих точек, и деревьев (от k = 1)

3. Формально это граф, не содержащий подграфа, гомеоморфного К4



Релаксация

4. O(kN) поиск всех путей от k входов

От каждого входа до каждой точки существует кратчайший путь

(0, b, c, e)

(1, d, c, e)

Тогда вычислив

(0, b), (1, d), в точке с получим альт.

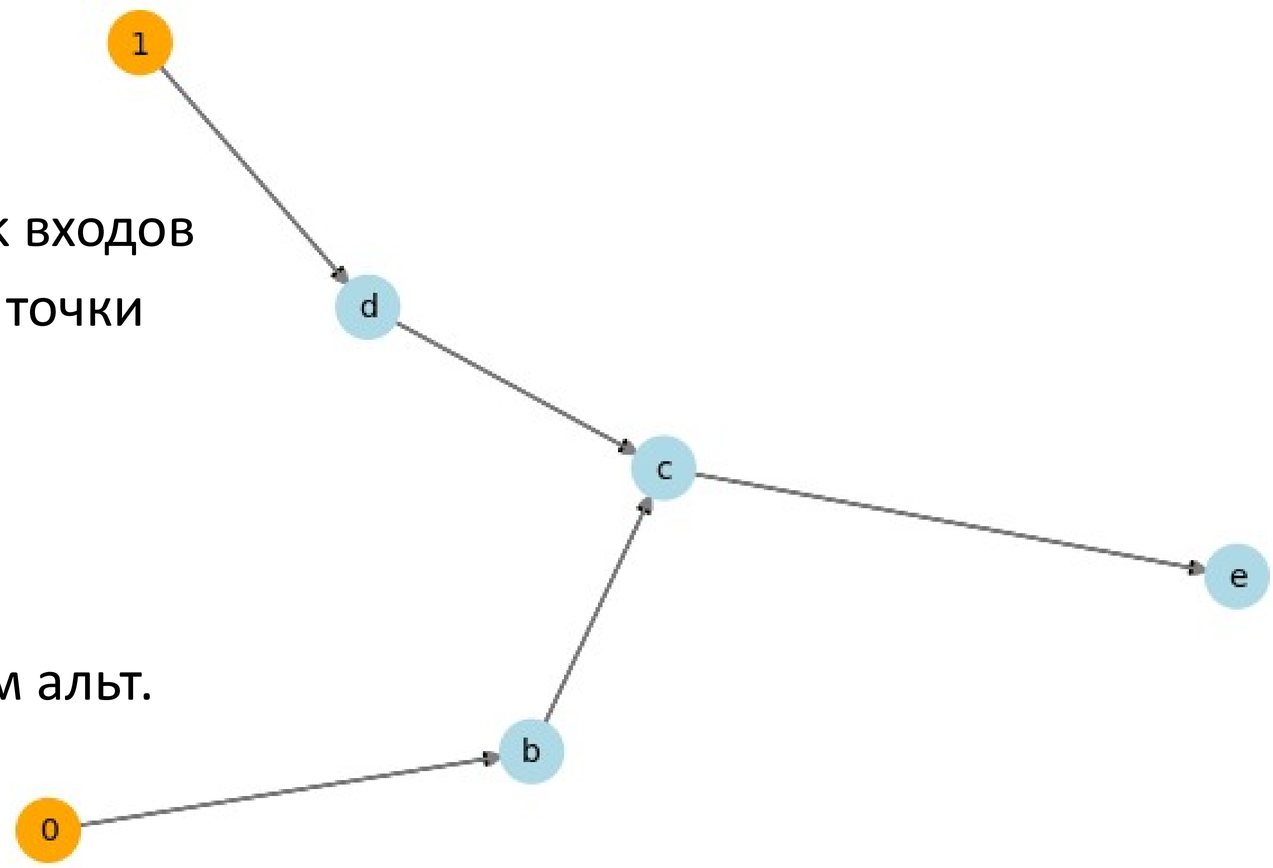
min((b, c), (d, c))

(c, e)

Эту последовательность можно записать как программу вычислений

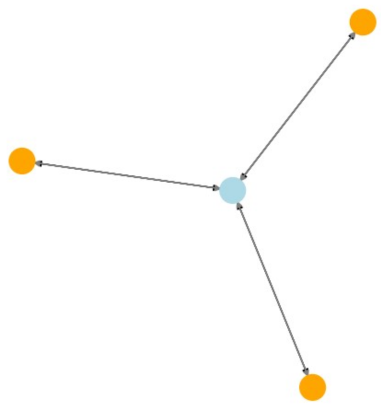
5. поиск пути

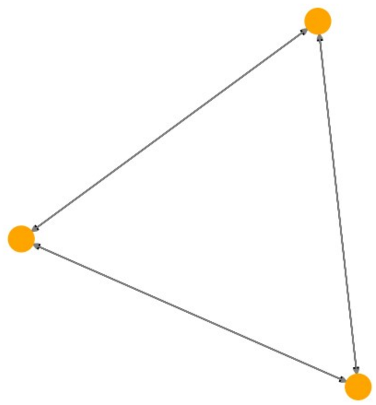
от *истока* до *границы*



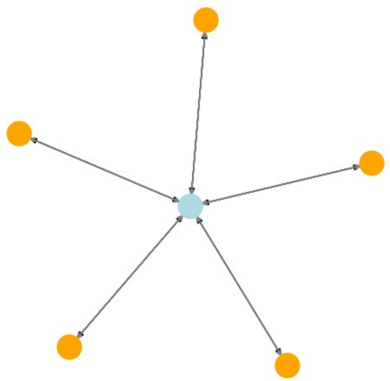
k = 3, и общий случай

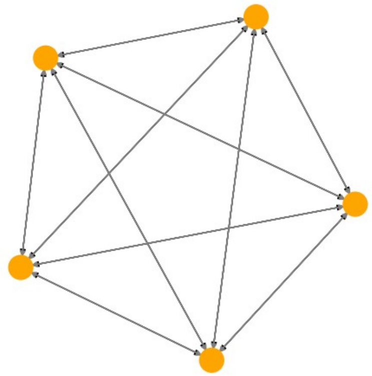
1. При k = 3
Число рёбер **не меняется**
Степени оставшихся вершин
увеличиваются на 1

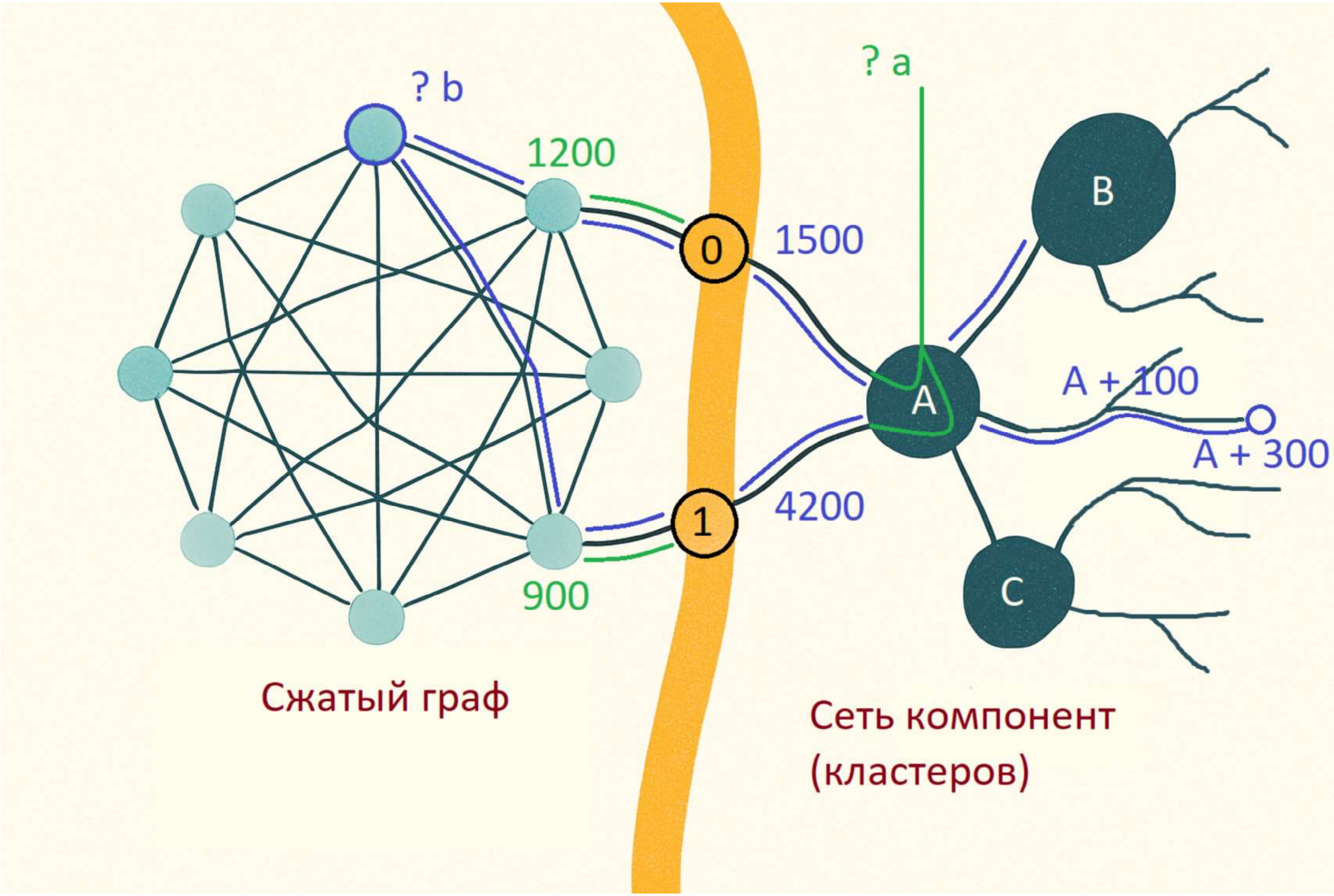




2. При k >= 4 Число рёбер возрастает как k^2/2 - 3k/2,
Степени вершин возрастают на k-2 каждая.







? b
1200
0
1500
? a
B
A
A + 100
A + 300
1
4200
900
C
Сжатый граф
Сеть компонент
(кластеров)

Предобработка:
Линейный проход по вершинам графа,
N операций по удалению вершин, #удалённых вершин
kN операций по добавлению / изменению рёбер

Амортизированная сложность $O(kV)$

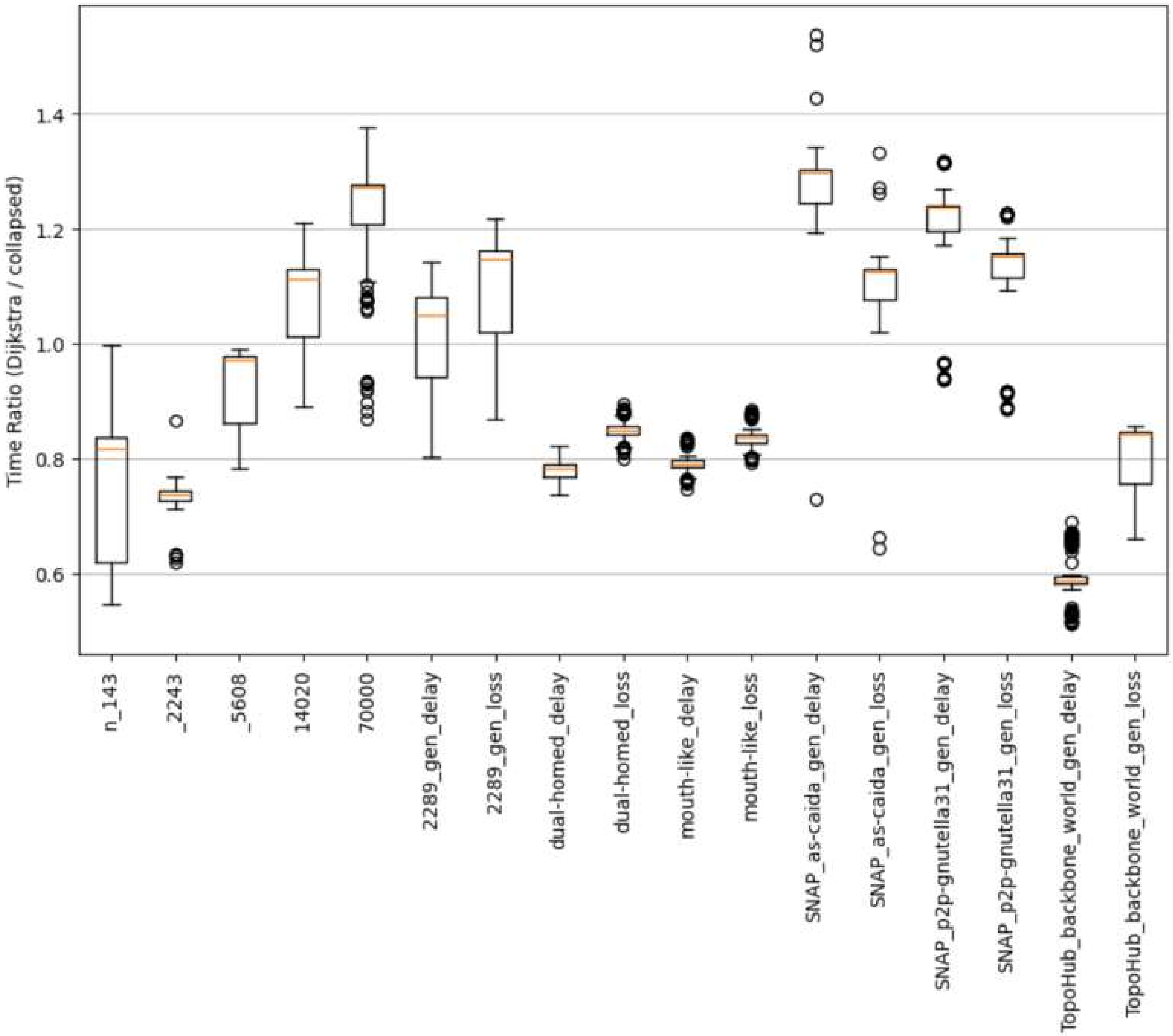
$V = N + M$, $E = E'' + E'$,
$p <= 1$ - если наибольшая компонента размера p
Запрос:
$O((M + E') \log M) + O(Np \log (Np)) + O(kN)$

	k = 1	k = 2	k=3	k = ?
Поиск пути до границы (практика)	O(N log N)	O(N log N)	O(N log N)	O(N log N)
Поиск пути до границы (теория)	O(1)	O(1)	O(1)	O(k)
Обработка (практика)	Общая	Общая	Общая	Общая
Обработка (теория)	Отдельная	Отдельная	Общая	Общая
Поиск пути в кластере (практика)	O(N log N)	O(N log N)	O(N log N)	O(N log N)
Поиск пути в кластере (теория)	N	O(N)	O(N log N)	O(N log N)
Ограничения	Нет K3	Нет K4	Нет K5	Нет K(k+2)
Поиск пути в графе (практика)	O((M + E') log M)			
Поиск пути в графе (теория)	min(O(E'), O(M log M + E'))			
Релаксация	N	2N	3N	kN
Поиск T кратчайших путей	1	(T-1)N	(T-1) N log N	?

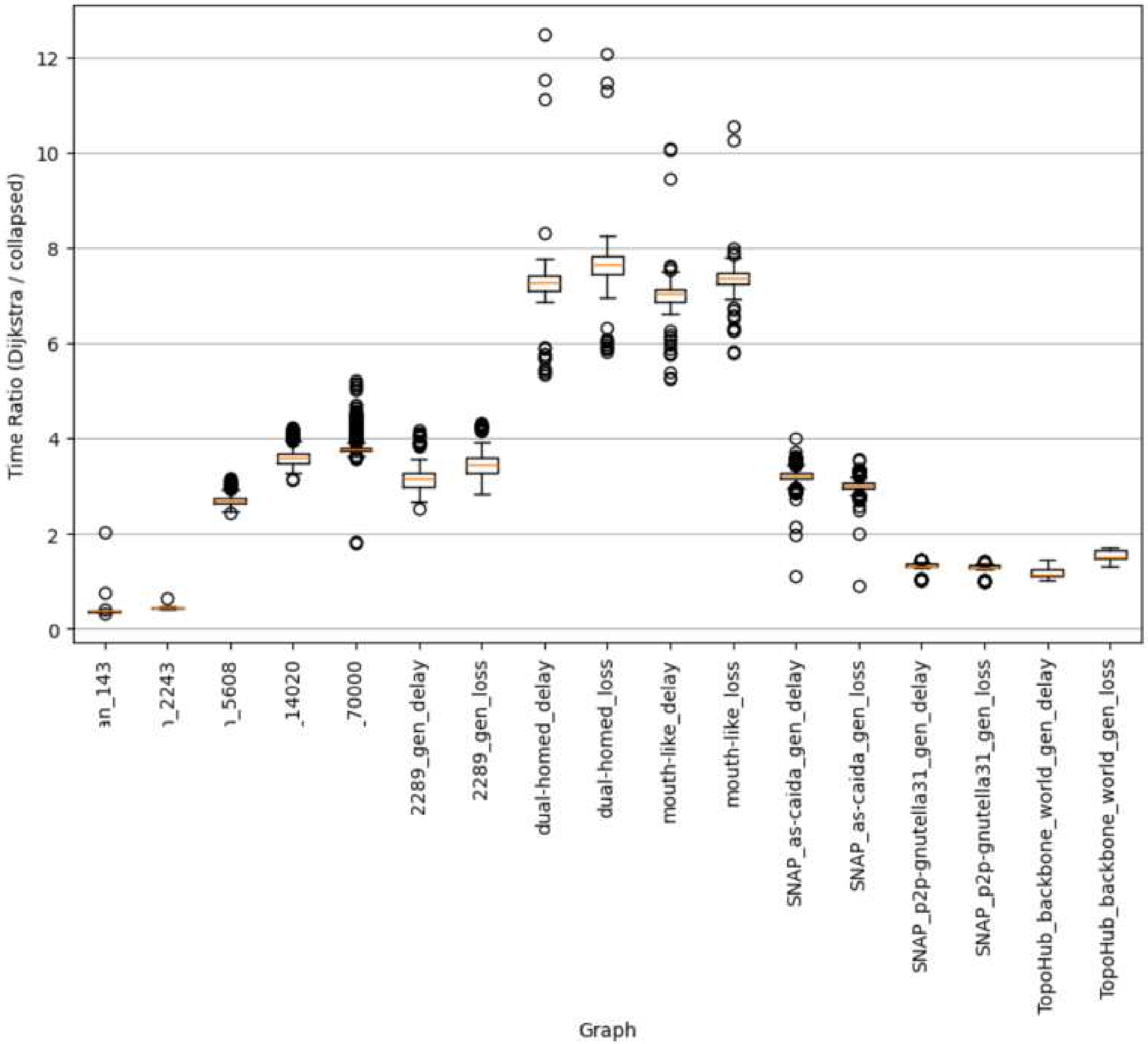
Тестирование

Name	Nodes	Edges
*	70100	223106
dual-homed	1000	2060
mouth-like	1000	1570
TopoHub_backbone_world	3815	5189
*	12812	61440
*	49568	454656
910D	16400	200704
SNAP_p2p-gnutella31	62586	147892
SNAP_as-caida	26475	106762
SNAP_cisco-secure-workload	278739	2158346

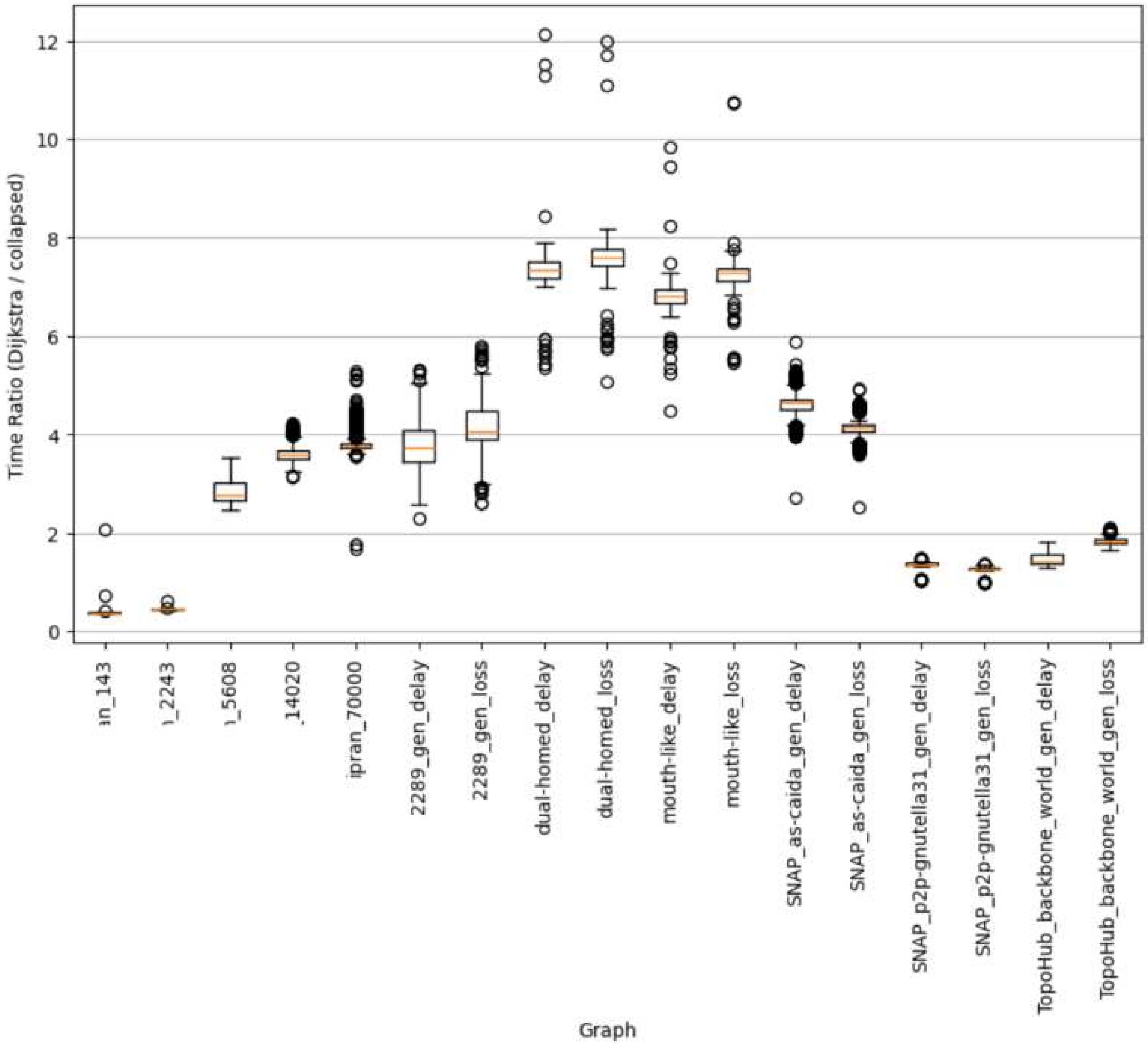
Тестирование k=1



Тестирование k=2



Тестирование k=3



Спасибо за внимание!