НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

НАУЧНЫЙ ДОКЛАД

по результатам подготовленной

научно-квалификационной работы (диссертации)

ФИО: Евстропов Глеб Олегович

Направление подготовки: 02.06.01 Компьютерные и информационные науки

Профиль (направленность) программы: 05.13.17 Теоретические основы информатики

Аспирантская школа по компьютерным наукам

Аспирант _______________ /Евстропов Г.О./

Научный руководитель ______________/Бабенко М.А./

Директор аспирантской школы __________________ /Объедков С.А./

Москва, 2018

Содержание

1. Аннотация

Тема диссертационного исследования: комбинаторные алгоритмы для мультипотоковых задач. Перед автором данной работы была поставлена задача исследования возможности адаптации алгоритма Гольдберга-Рао поиска максимального потока в сети к задаче поиска максимального целочисленного кососимметрического потока. С помощью техники использования «начального приближения» для кососимметрического потока, алгоритма Гольдберга-Рао поиска максимального потока, и поиска регулярного пути в кососимметрическом графе с битовыми оптимизациями операций над множествами данная работа предлагает алгоритм поиска максимального целочисленного кососимметрического потока за время $O(min\left(n^{\frac{2}{3}}, m^2\right) m \log\frac{n^2}{m} \log C + \frac{n^3}{\log n})$, что улучшает оценку $O(nm)$ для плотных графов, то есть графов у которых $\mathrm{m} = \Theta(n^2)$. Здесь и далее мы будем предполагать все ограничения на пропускные способности целочисленными, и будем обозначать через n, m и C, количество вершин в сети, количество рёбер в сети и максимальную пропускную способность одной дуги.

2. Введение

Тема диссертационного исследования: комбинаторные алгоритмы для мультипотоковых задач. Перед автором данной работы была поставлена задача исследования возможности адаптации алгоритма Гольдберга-Рао[1] поиска максимального потока в сети к задаче поиска максимального целочисленного кососимметрического потока.

Задача о целочисленном кососимметрическом потоке является обобщением классической задачи о максимальном потоке, позволяет решать различные потоковые задачи с некоторыми дополнительными ограничениями. Например, с помощью алгоритмов поиска максимального потока можно искать максимальное паросочетание в двудольных графах, в то время как с помощью поиска целочисленного кососимметрического потока можно искать паросочетания уже в графах произвольного вида.

Одним из результатов теории кососимметрических потоков является алгоритм, который позволяет любой алгоритм поиска обычного потока применить к кососимметрическим сетям с использованием дополнительного слагаемого порядка $O(nm)$. Такой алгоритм позволяет без потери асимптотики применить к задаче о целочисленном кососимметрическом потоке в общем случае любой алгоритм поиска потока, кроме алгоритма Гольдберга-Рао — наилучшего с теоретической точки зрения слабо-полиномиального алгоритма, который, вообще говоря, имеет оценку лучше $O(nm)$. Попытки же адаптировать этот алгоритм под кососимметрические сети аналогично другим алгоритмам сталкиваются с довольно интересным препятствием, которое совершенно не выглядит неразрешимым.

Хотя задача преодоления барьера $O(nm)$ в общем случае нами не решена, в данной работе приведено описание алгоритма поиска максимального целочисленного кососимметричского потока за время $O(min\left(n^{\frac{2}{3}}, m^2\right) m \log\frac{n^2}{m} \log C + \frac{n^3}{\log n})$, что для плотных графов может быть записано как $O(n^{\frac{8}{3}} \log C + \frac{n^3}{\log n})$. Такое ускорение, записанной нами как деление на $\log n$ в некоторых источниках записывается как деление на $wordlen$, то есть длину машинного слова. Первая запись используется всё же чаще, так как позволяет не вводить в перегруженную формулу ещё один параметр, оставляя оценку справедливой, так как в RAM-модели предполагается возможность прямой индексации памяти, то есть $\log n \leq wordlen$[3].

В заключительной части работы мы упоминаем возможные направления развития, которые в случае успеха позволят полиномиально улучшить время поиска максимального целочисленного кососимметрического потока в плотных сетях.

3. Объект исследования, основные определения

Сформулируем **задачу о максимальном потоке**[2] (Max Flow problem): пусть дана сеть (G, c, s, t), где G – орграф с множеством вершин V = V(G) и множеством рёбер E = E(G). На множестве рёбер определена неотрицательная вещественнозначная функция пропускных способностей (capacity) u: E(G) → R+. Из множества вершин выбраны источник s (source) и сток t (sink). Потоком (flow) в G является действительная функция f: V × V → R, удовлетворяющая следующим трём условиям.

Ограничение пропускной способности (capacity constraint): f(u, v) ≤ c(u, v) для всех u, v принадлежащих V.

Антисимметричность (skew symmetry): f(u, v) = −f(v, u) для всех u, v ∈ V.

Сохранение потока (flow conservation): для всех u ∈ V – {s, t}

$$\sum_{\mathrm{v} \in \mathrm{V}} f(u, v) = 0.$$

Величиной (value) потока называется суммарный поток, выходящий из источника, то есть

$$|f| = \sum_{\mathrm{v} \in \mathrm{V}} f(s, v).$$

В классической формулировке задачи о максимальном потоке ставится задача отыскания потока f(u, v) максимальной величины в сети (G, c, s, t).

Задача о максимальном потоке легко может быть сформулирована в виде линейной задачи:

максимизировать $\sum_{e \in \delta^{+}(s)} x_e - \sum_{e \in \delta^{-}(s)} x_e$

при условиях $\sum_{e \in \delta^{+}(v)} x_e = \sum_{e \in \delta^{-}(v)} x_e \ \ (v \in \mathrm{V(G)} \ \{s, t\})$,

$$x_e \le u(e) \ \ \big(e \in \mathrm{E(G)}\big),$$

$$x_e \geq 0 \quad \big(e \in \mathrm{E(G)}\big).$$

Данная программа, очевидно, ограничена и нулевой поток $\mathrm{f} \equiv 0$ всегда допустим, из чего следует, что задача о максимальном потоке всегда имеет оптимальное решение. Если нас дополнительно интересуют только те решения, в которых все искомые переменные целочисленные, то говорят, что ставится **задача о максимальном целочисленном потоке**.

Теперь сформулируем более общую задачу о максимальном кососимметрическом потоке. Пусть некоторые две функции $\delta_V(v)$ и $\delta_E(e)$ определяют разбиение на пары вершин и рёбер графа, причём отображение по рёбрам сохраняет симметрию по вершинам. Формально,

1. $\delta_V(v) \neq v$ и $\delta_V(\delta_V(v)) = v$, будем обозначать $\delta_V(v)$ как v'.
2. $\delta_E(e) \neq e$ и $\delta_E(\delta_E(e)) = e$, будем обозначать $\delta_E(e)$ как e' .
3. $\delta_V(head(e)) = tail\big(\delta_E(e)\big)$ и $\delta_V(tail(e)) = head\big(\delta_E(e)\big)$, то есть ориентированному ребру uv соответствует ребро $\delta_E(uv) = v'u'$.

Для удобства объединим δ_V и δ_E в одну биекцию δ. Кососимметрической сетью будем называть четвёрку (G, c, δ, s), состоящую из описания графа G, функции пропускной способности c, биекции δ и стартовой вершины s. Конечной вершиной для всех путей, доставляющих поток, будем являться обратная к s вершина s'. Теперь мы поставим задачу найти функцию потока $f(u,v)$ удовлетворяющую всем тем же самым свойствам, что и в задаче о максимальном потоке, плюс свойству $f(u,v) = f(v',u')$. С точки зрения рассмотрения задачи о потоке как задачи об упаковке в граф ориентированных простых путей, теперь перед нами стоит задача упаковки в граф пар ориентированных кососимметрических путей.

Как и в случае обычного потока, если дополнительно наложить ограничение целочисленности на функцию $f(u,v)$, то можно говорить о задаче о максимальном целочисленного кососимметрическом потоке, которая и является основным предметом рассмотрения данной работы.

4. Обзор результатов теории потоков

Первый комбинаторный алгоритм нахождения максимального потока был опубликован в 1956 году Л.Р. Фордом младшим и Д.Р. Фалкерсоном[4]. Алгоритм этот хорошо известен, поэтому приводить его мы здесь не будем. Алгоритм устанавливает строгую комбинаторную двойственность между максимальным потоком и минимальным разрезом, и демонстрирует, что решение задачи о максимальном потоке обладает той же дробностью, что и исходные данные. В частности, из этого следует, что задача о максимальном целочисленном потоке всегда имеет решение. К существенным недостаткам данного алгоритма можно отнести что, во-первых, он не является полиномиальным; во-вторых, при ряде допущений (веса иррациональные числа, работа с которыми выполняется бесконечно точно) может и вовсе не завершать свою работу, даже не приближаясь к правильному ответу. Таким образом, алгоритм Форда-Фалкерсона, вообще говоря, не доказывает существование максимального потока при нецелочисленных пропускных способностях (что, впрочем, всё равно следует из свойств линейной программы).

Более сильные результаты были опубликованы в 1970 году советским математиком Ефимом Диницем и, независимо, в 1972 году Джеком Эдмондсом и Ричардом Карпом. Оба результата предъявляли комбинаторный сильно полиномиальный алгоритм поиска максимального потока, основанный на насыщении кратчайшего пути. Алгоритм Эдмондса-Карпа имеет сложность $O(m^2n)$, а алгоритм Диница совершенствуя эту технику достигает сложности $O(n^2m)$. Наилучший теоретический сильно-полиномиальный результат для общего случая задачи о максимальном потоке принадлежит Джеймсу Орлину[5], работа была опубликована недавно – в 2012 году и в ней описан алгоритм нахождения максимального потока имеющий сложность O(nm). Отметим отдельно, что этот результат важен ещё тем, что показывает возможность найти функцию потока за то же (асимптотически) время, что и декомпозировать. Более того, могут быть построены примеры, которые показывает, что, вообще говоря, декомпозиция потока имеет размер $\Omega(nm)$.

А. Карзанов показал[7] ряд оценок на количество фаз алгоритма Диница (одна фаза выполняет один поиск блокирующего потока) в стандартных графах, которые позволяют в частности оценить время поиска максимального паросочетания в двудольном графе алгоритмом Диница как $O(m\sqrt{n})$. Комбинирую идеи из оценок Карзанова на число фаз, линейное время поиска блокирующего потока в графе с единичными пропускными способностями и идею масштабирования, Гольдберг и Рао смогли получить первый алгоритм, время работы которого составляет $o(nm)$, а точнее $O(min\left(n^{\frac{2}{3}}, m^2\right) m \log\frac{n^2}{m} \log C)$, что означает, что для некоторых графов поток может быть найден даже быстрее, чем потом займёт его декомпозиция. Заметим, что время работы этого алгоритма зависит от ограничения на пропускные способности, а вопрос о существовании сильно-полиномиального алгоритма, преодолевающего барьер $\mathrm{O}(nm)$ остаётся открытым.

Сразу отметим, что в случае кососимметрических сетей интерес для исследования представляет только задача о целочисленном потоке. Действительно, если бы не ограничение целочисленности значений функции $f(u,v)$, то задача могла бы быть решена следующим алгоритмом.

1. Установим симметричные значения для пропускным способностей, то есть для каждой упорядоченной пары (u,v) выберем минимум из $\mathrm{c}(u,v)$ и $\mathrm{c}(v',u')$, после чего установим оба значения пропускной способности равными этому минимуму.

2. Теперь опустим ограничение $f(u,v) = f(v',u')$ и найдём в графе максимальный целочисленный поток.

3. Для каждой пары (u,v) заменим величину потока на среднее арифметическое $\frac{(f(u,v)+f(v',u'))}{2}$. Легко проверить прямой подстановкой, что все

ограничения накладываемые аксиомами потока будут выполнены, плюс станет выполняться и условие симметрии. Единственное, что будет нарушено – это дробность такого решения. При целых ограничениях на $c(u, v)$ итоговые значения потока могли получиться полуцелыми.

Задача поиска максимального целочисленного потока в кососимметрических сетях подробно изучалась в работах Гольдберга и Карзанова[8], и в работах Татта[9]. Интересно, что почти все основные утверждения и объекты теории потоков находят свои кососимметрические аналоги. Так, наиболее простой алгоритм поиска ненасыщенного пути в остаточной сети соответствует алгоритму поиска регулярного пути в остаточной сети для кососимметрического случая. Вместо понятия минимального разреза теория кососимметрических потоков оперирует понятием s-барьера (рис. 1) и так далее[10]. Были получены аналоги алгоритмов Эдмондса-Карпа и алгоритма Диница, а также оценка $O(\min\left(n^{\frac{2}{3}}m, m^{\frac{3}{2}}\right))$ на время работы для сетей с единичными пропускными способностями, которая повторяет аналогичную оценку для случая обычных сетей. Всё это позволяет надеяться, что и результаты Гольдберга-Рао могут быть перенесены на кососимметрические сети без потерь (или с небольшими потерями) асимптотики. Также стоит упомянуть, что как и для обычных сетей[12], для кососимметрических сетей был получен результат[11], показывающий всякий ациклический целочисленный кососимметрический поток в единичной кососимметрической сети без кратных рёбер имеет носитель (то есть множество рёбер, на которых функциях $f(u, v)$ будет не равна нулю) размера $O(n\sqrt{n})$.

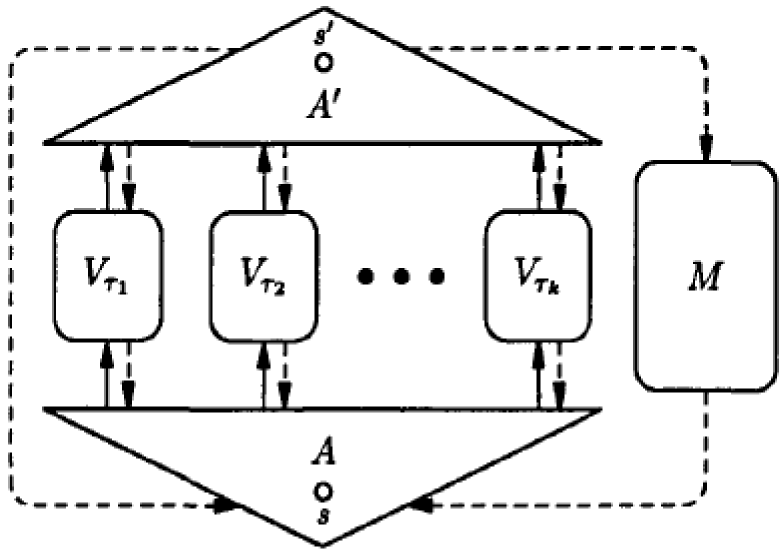

Рис. 1. S-барьер. [11]

Отдельным результатом теории потоков является тот факт, что любой алгоритм решения задачи о целочисленном максимальном потоке может быть применён к задаче о целочисленном кососимметрическом потоке с уплатой штрафного слагаемого $O(nm)$, то есть целочисленный кососимметрический поток в сети с целочисленными пропускными способностями не превосходящими C может быть найден за время $O(t(n,m,C) + nm)$, где $t(n,m,C)$ означает время, необходимое для поиска обычного потока. Поскольку данный алгоритм играет ключевую роль в дальнейшем изложении, то опишем здесь его краткую схему.

1. Опустим условие целочисленности и найдём точное решение задачи о максимальном кососимметрическом потоке с помощью решения задачи о максимальном потоке аналогично тому, как это описывалось выше.

2. Теперь мы хотим округлить наше решение, чтобы построить корректное решение задачи и целочисленном потоке, не очень сильно отличающееся от максимального. Для этого мы рассмотрим множество всех рёбер, имеющих полуцелые величины потока, выделим компоненты слабой связности по таким рёбрам и, рассмотрев их в неориентированном смысле в каждой найдём Эйлеров цикл. Каждая компонента будет либо симметрична

сама себе, либо симметрична какой-то ещё компоненте. Теперь изменим величину потока в каждом цикле при каком-нибудь порядке обхода.

3. Мы получили решение, отличающееся от оптимального не более, чем на количество компонент связности, то есть на n. В полученном графе мы сделаем не более чем n итераций увеличения потока на 1 вдоль какого-нибудь регулярного пути. Время поиска регулярного пути составляет $O(nm)$, что и даёт нам заявленную асимптотику.

5. Операции с битовыми масками, обход орграфов

Обсудим применение битовых операций для эффективной реализации различных операций работы с множествами. Пусть нам доступны числа, состоящие из w бит и стандартный набор битовых операций над этими числами: побитовое И, побитовое ИЛИ, исключающее ИЛИ, побитовое отрицание, сдвиги на заданное количество бит в старшую или младшую сторону.

Также пусть имеется некоторое множество размера U (для простоты считаем, что U делится на w) и требуется реализовать стандартные операции теории множеств над его подмножествами: объединение, пересечение, добавление и удаление элементов, вычитание одного подмножества из другого, извлечение любого элемента множества. Заведём $\frac{U}{w}$ чисел по w бит в каждом, получив таким образом U битов, с которыми соотнесём элементы нашего множества. Теперь каждое подмножество мы будем описывать с помощью $\frac{U}{w}$ чисел, явно представляющих используемые элементы. Заметим, что мы можем реализовать все стандартные операции теории множеств.

1. Вставка элемента i производится за время $O(1)$ выставлением $i \ mod \ w$ бита $(i \ div \ w)$-го числа.

2. Удаление элемента i из какого-либо множество также происходит за время $O(1)$ аналогичными действиями.

3. Для объединения двух множеств, задаваемых числовыми последовательностями $a_0, a_1, \dots, a_k$ и $b_0, b_1, \dots, b_k$ мы рассматриваем все пары

соответствующих элементов a_i и b_i, берём для них побитовое ИЛИ. Такой способ объединения двух множеств работает за время $O(\frac{U}{w})$.

4. Для поиска пересечения двух множеств за $O\left(\frac{U}{w}\right)$ мы сделаем всё аналогично предыдущему пункту, лишь заменим побитовое ИЛИ на побитовое И.

5. Для выполнения вычитания множества В из множества А мы сперва с помощью битового отрицания найдём дополнение множества В, а затем пересечём это дополнение с А. Время работы - $O\left(\frac{U}{w}\right)$.

6. Наконец, как будет устроена операция получения любого представителя подмножества. Сперва просмотрим последовательность $a_0, a_1, \dots, a_k$ и выберем из неё любой элемент, отличный от нуля. Если такого элемента не нашлось, то последовательность описывает пустое множество. В противном случае нужно решить задачу поиска в ненулевом числе единичного бита. Это может быть сделано за время $O(\log w)$, или с помощью предподсчёта за время $O(1)$, если положить $w = \log n$. В дальнейшем мы будем игнорировать эту компоненту в оценках и полагать, что сложность поиска представителя составляет $O\left(\frac{U}{w}\right)$.

Рассмотрим теперь, как можно ускорить поиск пути между парой вершин в плотном орграфе используя вышеописанные способы хранения множества. Пусть граф хранится в виде n множеств N_i, для каждой вершины хранятся все вершины, в которые можно попасть из данной одним переходом по ребру. Будем итеративно строить ориентированное остовное дерево вершин, достижимых из s. Пусть сейчас это дерево включает в себя некоторое множество вершин T, причём $s \in T$, а t – нет. Тогда нам требуется запросить любой элемент множества, являющегося объединением N_{v_0}, N_{v_1}, ..., N_{v_k} по всем элементам $v_i \in T$, из которого вычли собственно T. Это будет соответствовать ребру из T в остальной граф и позволит расширить наше множество. Такую операцию нужно проделать не более, чем n раз, при этом после каждого добавления требуется выполнить до-объединения с новым множеством N_x.

Таким образом мы получаем алгоритм, который даёт нам время работы $O\left(\frac{n^2}{\log n}\right)$ поиска одного пути в орграфе и позволяет найти в сети поток величины k за время $O\left(\frac{kn^2}{\log n}\right)$. Комбинация этой техники с алгоритмом Диница позволяет найти максимальное паросочетание в двудольном графе за время $O\left(n^2\sqrt{\frac{n}{\log n}}\right)$.

6. Поиск регулярного пути с помощью битовых операций

Теперь нам хотелось бы перенести эти операции с битовыми множествами на задачу поиска регулярного пути в кососимметрической сети. Как и в задаче поиска обычного пути нам требуется оракул, который позволяет получить какое-нибудь ребро выходящее из компоненты T. Операции, требующиеся для реализации этого шага ничем не отличаются от операций из предыдущего раздела, поэтому мы пропустим их описание и перейдём сразу к вырезанию ростков.

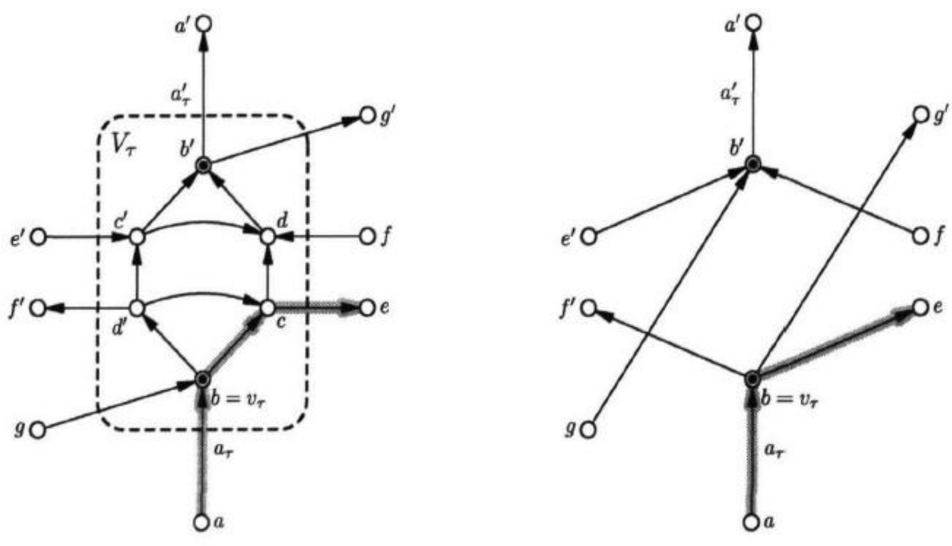

Рис. 2. Вырезание ростков. [11]

Какие операции с множествами здесь требуется применить? Для всех рёбер исходящих из ростка требуется сделать их исходящими из базы ростка. Такая операция эквивалентна объединению множеств и может быть выполнена за $O(\frac{n}{\log n})$. Хуже обстоят дела с переносом всех входящих в росток рёбер в вершину b'. Чтобы матрица смежности поддерживала применение групповых битовых операций как по строкам, так и по столбцам, потребуется использовать другой способ её представления.

Вместо разбиения каждой строки на кусочки по $\log n$, мы покроем таблицу квадратами размером $\sqrt{\log n}x\sqrt{\log n}$. Теперь одно число будет хранить биты данного квадрата, допуская как групповые операции по строке так и групповые операции по столбцу. Однако, за удобство делать операции с множествами в любом направлении мы платим штраф в $\sqrt{\log n}$, теперь время выполнения одной операции с множествами составляет $O(\frac{n}{\sqrt{\log n}})$.

Чтобы не платить штраф в множитель $\sqrt{\log n}$ вернёмся к предыдущему способу хранения матрицы смежности. Заметим, что в используемом нами алгоритме поиска регулярного пути используется только поиск какого-нибудь исходящего ребра. При этом необходимости сканировать список входящих рёбер у нас нет, а значит, мы можем просто отложить операцию их переноса как пока не требующуюся. Вместо этого, мы используем реализацию системы непересекающихся множеств с амортизированным общим временем выполнения операций слияния $O(n\log n)$ и временем ответа на запрос $O(1)$, чтобы эффективно находить реальные концы рёбер, для которых мы не выполняли перенос.

Таким образом, общее время поиска регулярного пути в кососимметрической сети составляет $O(\frac{n^2}{\log n})$, после чего в матрицу смежности требуется внести $O(n)$ точечных изменений.

7. Заключение

Нами был описан алгоритм поиска максимального целочисленного кососимметрического потока со временем работы $O(min\left(n^{\frac{2}{3}}, m^2\right) m \log\frac{n^2}{m} \log C + \frac{n^3}{\log n})$. Хотя этот алгоритм уже проходит барьер декомпозиции в случае плотных графов, хотелось бы во-первых пройти отметку $O(nm)$ для произвольных графов, а во-вторых получить улучшение на полиномиальный множитель.

Автор планирует продолжить исследование по данному направлению и проверить применимость к задаче следующих соображений и фактов.

1. Поиск путей за время меньшее количества дуг в графе с помощью линейных набросков из теории поточных алгоритмов.

2. Использовать алгоритмы быстрого матричного умножения. Так, например, был получен не комбинаторный алгоритм поиска максимального паросочетания в произвольном графе за время $O(n^{2.376})$[14].

3. Благодаря алгоритму поиска приближённого решения, которое находит целочисленный кососимметричный поток отличающийся от оптимального не более чем на n, мы можем решать задачу адаптации алгоритма Гольдберга-Рао для более слабой задачи, когда известно, что требуется пустить не более чем n единиц. Возможно это открывает нам некоторый зазор в выборе параметров и даёт степень свободы в применении алгоритма, что позволит избежать тех проблем, которые сейчас мешают прямо перенести алгоритм на кососимметрические сети.

8. Список использованной литературы

[1] Goldberg, A. V.; Rao, S. (1998). "Beyond the flow decomposition barrier". *Journal of the ACM*. 45 (5): 783. *doi*:*10.1145/290179.290181*

[2] Кормен, Томас Х., Лейзерсон, Чальз И., Ривест, Рональд Л., Штайн, Клиффорд. Алгоритмы: построение и анализ, 2-е издание. : пер. с англ. – М. : Издательский дом «Вильямс», 2009. – 1296 с. : ил. – Парал. тит. Англ.

[3] Ахо А., Хопкрофт Д., Ульман Д. Построение и анализ вычислительных алго-
ритмов. М.: Мир, 1999.

[4] Ford L. R., Fulkerson D. R. Maximal flow through a network // Canadian Journal of Mathematics. 1956. № 8. doi:10.4153/CJM-1956-045-5.

[5] James B. Orlin, Max Flow in O(nm) Time, or Better. 2012.

[6] Кормен, Томас Х., Лейзерсон, Чальз И., Ривест, Рональд Л., Штайн, Клиффорд. Алгоритмы: построение и анализ, 2-е издание. : пер. с англ. – М. : Издательский дом «Вильямс», 2009. – 1296 с. : ил. – Парал. тит. Англ.

[7] On the efficiency of Maximum-Flow Algorithms on Networks with Small Integer Capacities. David Fernandez-Baca and Charles U.Martel

[8] Goldberg A. V., Karzanov A. V. Path Problems in Skew-Symmetric Graphs // Combinatorica.
1996. V. 16, № 3. P. 353-382.

[9] Tutte W. T. Antisymmetrical Digraphs // Canadian Journal of Mathematics. 1967.
V. 19. P. 1101-1117.

[10] Goldberg A. V., Karzanov A. V., Maximum skew-symmetric flows and matchings Math. Program., Ser. A 100: 537–568 (2004)

[11] Бабенко М.А. Сложность некоторых алгоритмических проблем для кососимметрических графов. Диссертация на соискание ученой степени кандидата физико-математических наук (2007)

[12] Karger D. R., Levine M. S. Finding Maximum Flows in Undirected Graphs Seems
Easier than Bipartite Matching // Proceedings of the Thirtieth Annual ACM Symposium
on Theory of Computing. NY: ACM Press, 1998. P. 69-78.

[13] Alt, H.; Blum, N.; Mehlhorn, K.; Paul, M. (1991), "Computing a maximum cardinality matching in a bipartite graph in time ", Information Processing Letters, 37 (4):

[14] Mucha, M.; Sankowski, P. (2004), *"Maximum Matchings via Gaussian Elimination" (PDF), Proc. 45th IEEE Symp. Foundations of Computer Science*, pp. 248–255