ЖУРНАЛ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И МАТЕМАТИЧЕСКОЙ ФИЗИКИ, 2023, том 63, № 3, с. 517–530

ИНФОРМАТИКА

УДК 519.86

О СТАБИЛЬНЫХ ПОТОКАХ И ПРЕДПОТОКАХ

© 2023 г. А. В. Карзанов[1,*]

[1] 117418 Москва, Нахимовский пр-т, 47, ЦЭМИ РАН, Россия

*e-mail: akarzanov7@gmail.com

Поступила в редакцию 11.08.2022 г.
Переработанный вариант 26.08.2022 г.
Принята к публикации 17.11.2022 г.

Предлагается новый алгоритм построения стабильного потока в сети с несколькими источниками и стоками. Он основан на идее *предпотоков* (примененной в 1970-х годах для более быстрого решения классической задачи о максимальном потоке) и имеет временну́ю сложность $O(nm)$ для сети с n вершинами и m ребрами. Полученные результаты затем распространяются на более широкий класс объектов – т.н. стабильные квазипотоки с ограниченными отклонениями от балансовых соотношений в нетерминальных вершинах. Библ. 12.

Ключевые слова: стабильный поток в сети, стабильное распределение, предпоток, квазипоток.

DOI: 10.31857/S0044466923030079, **EDN:** EBUSXU

1. ВВЕДЕНИЕ

Область теоретических и прикладных задач о стабильных договорах явилась предметом интенсивного изучения в математической экономике, теории игр и комбинаторной оптимизации, и ей посвящены многочисленные работы нескольких последних десятилетий. Отправной точкой многих исследований послужила классическая работа Гейла и Шепли [1] о *стабильных марьяжах* (SM).

Согласно одной из распространенных формулировок этой задачи имеется двудольный граф $G = (V, E)$, и для каждой вершины v задан (строгий) линейный порядок $<_v$ на инцидентных ребрах. (В [1] рассматриваются полные двудольные графы, но это не существенно. В популярной интерпретации ребра в G представляют возможные союзы персон разного пола, а порядок $<_v$ – предпочтения персоны v: если для ребер $e = vu$ и $e' = vw$ выполняется $e <_v e'$, то v предпочитает союз с u союзу с w.) В задаче требуется найти паросочетание (matching) $M \subseteq E$, являющееся стабильным относительно всех этих порядков. Это означает, что для любого ребра e из $E - M$ есть ребро $e' \in M$ такое, что e и e' имеют общую вершину v, и при этом выполняется $e' <_v e$. Было показано, что стабильное паросочетание в двудольном графе всегда существует, и оно может быть построено комбинаторным алгоритмом с линейной верхней оценкой числа действий (*временно́й сложностью*) $O(n + m)$, где n и m – число вершин и ребер в G соответственно.

В последующих работах многих авторов были исследованы различные обобщения задачи SM. Укажем два направления обобщений, связанные с графами (оставляя в стороне постановки, в которых в договорах могут участвовать более двух агентов, или предпочтения агентов задаются другими способами). Одно из них – переход от двудольного к произвольному графу G. Соответствующий аналог задачи SM, известный под названием задачи о *стабильном размещении пар соседей по комнатам* (stable roommates problem), был исследован Ирвингом в работе [2], где был дан алгоритм линейной сложности для построения стабильного паросочетания в G либо доказательства его несуществования. Важные дополнительные структурные и алгоритмические результаты были представлены в [3].

Другой тип обобщений, который более важен для нас, оставляет граф $G = (V, E)$ двудольным, но добавляет числовые параметры. Среди задач этого типа весьма общей выглядит задача о *стабильном распределении* (stable allocation problem – SA), введенная Бейу и Балински [4]. Здесь распределением считается назначение каждому ребру $e \in E$ величины $x(e) \geq 0$, не превышающей предписанной пропускной способности $c(e)$, и при этом сумма назначений по ребрам, инцидентным вершине $v \in V$, не должна превышать предписанной “квоты” $q(v)$. (В случае задачи

SM, все $c(e)$ и $q(v)$ равны единице, а $x(e)$ принимает значение 0 или 1. В общем случае задачи SA, число $x(e)$ на ребре $e = uv$ может пониматься, например, как размер участия "работника" u в "работе" v.) В [4] доказывается разрешимость SA при любых неотрицательных вещественных c, q (давая целочисленное x при целочисленных c, q) и предложен сильно полиномиальный алгоритм решения, т.е. имеющий временну́ю сложность, зависящую только от размеров графа и выражающуюся полиномом от n и m. Дин и Мунши [5] описали улучшенную версию алгоритма из [4], которая строит решение за время $O(nm)$; более того, они показали, что можно добиться даже временно́й оценки $O(m\log n)$, если применить для ряда процедур мощные структуры данных, такие как динамические и самонастраивающиеся деревья Слейтора и Таржана [6], [7]. (Это дает теоретическое ускорение, но алгоритм такого рода громоздкий и едва ли может быть применен для практических целей.)

В свою очередь задача SA может быть представлена как частный случай задачи о *стабильном потоке* (SF). Последняя была сформулирована Флейнером в 2010-е годы в работе [8] (как обобщение задачи Островского [9] для ациклических сетей с единичными пропускными способностями ребер). В ней задана сеть, состоящая из ориентированного графа $G = (V, E)$ с пропускными способностями $c(e) \geq 0$ ребер $e \in E$ и двумя выделенными вершинами ("терминалами") s и t. Для каждой *внутренней* вершины $v \in V - \{s,t\}$ заданы линейный порядок на входящих ребрах и линейный порядок на выходящих ребрах. (Допустимый) *поток* – это неотрицательная вещественная функция f на ребрах, удовлетворяющая верхним ограничениям по пропускным способностям и имеющая нулевые эксцессы для всех внутренних вершин, где под эксцессом в вершине v понимается разность $\mathrm{ex}_f(v)$ между общим потоком по входящим ребрам и общим потоком по выходящим ребрам в v. (Внутреннюю вершину можно интерпретировать как "игрока", "торговца" или "агента", который, получая некоторый объем однородного продукта по входящим ребрам, посылает его далее по выходящим ребрам, руководствуясь своей функцией полезности, зависящей от указанных порядков на ребрах.) Поток считается стабильным, если он не допускает локальных улучшений, использующих "ненасыщенные" пути; точное определение будет дано в разд. 2.

Между задачами SA и SF есть тесная связь. Задача SA с двудольным графом $G = (V_1 \sqcup V_2, E)$ сводится к SF с графом, получаемым из G (с ребрами, ориентированными от V_1 к V_2) путем добавления терминалов s и t, ребер (s,u) с пропускными способностями $q(u)$ для вершин u из доли V_1, и ребер (v,t) с пропускными способностями $q(v)$ для вершин v из доли V_2.

С другой стороны, Флейнер [8] показал, что задача SF с графом $G = (V, E)$ может быть сведена к задаче SA с графом, получаемым расщеплением вершин в G и добавлением $O(|V|)$ новых ребер. В результате было установлено существование стабильного потока для любой сети (и целочисленного стабильного потока при целочисленном c), и возможность построения такого потока при помощи алгоритмов для SA, получая временну́ю сложность того же порядка.

Впоследствии появились прямые алгоритмы нахождения стабильного потока. Недавно Чех и Матушке [10] предложили прямой алгоритм для сети с одним источником и одним стоком, который имеет сложность $O(nm)$ и основан на комбинации идей метода увеличивающих путей Форда и Фалкерсона для задачи о максимальном потоке и метода "отсроченных принятий" (deferred aceptance), восходящего к Гейлу и Шепли [1].

В настоящей работе предлагается альтернативный алгоритм построения стабильного потока в сети $N = (G, S, T, c)$ с произвольными множествами источников S и стоков T при условии отсутствия ребер, входящих в источники или выходящих из стоков. Алгоритм прямой и чисто комбинаторный (не апеллирует к задаче SA и не использует сложных структур данных), он основан на методе предпотоков. Напомним, что *предпотоком* в сети называется неотрицательная функция на ребрах, ограниченная пропускными способностями и имеющая неотрицательные эксцессы во всех внутренних вершинах. (Это понятие было введено в [11] и использовалось в алгоритме нахождения максимального потока в сети.) Заметим, что предпотоки уже применялись ранее в работе [12] для построения стабильного потока в сети с целочисленными пропускными способностями за псевдополиномиальное время (линейно зависящее от суммы пропускных способностей ребер).

Алгоритм имеет базовую и модифицированную (ускоренную) версии. Обе начинаются с построения некоторого стабильного предпотока, и на каждой последующей итерации текущий стабильный предпоток перестраивается с целью избавления от положительных эксцессов во внутренних вершинах. Как только эксцессы всех внутренних вершин обнулятся, будет построен ис-

комый стабильный поток (причем целочисленный при целочисленном c). Базовый алгоритм конечен при любых неотрицательных вещественных пропускных способностях c. Модифицированный алгоритм сильно полиномиальный; он использует дополнительные преобразования предпотоков и строит стабильный поток с временно́й сложностью $O(nm)$ (подобно алгоритму в [10]).

Мы затем рассматриваем более общую задачу для сети $N = (G, S, T, c)$, в которой для каждой внутренней вершины v заданы два параметра $\beta(v) \geq 0$ и $\gamma(v) \geq 0$ и требуется найти стабильный "квазипоток" f с ограничениями вида $-\beta(v) \leq \mathrm{ex}_f(v) \leq \gamma(v)$. (Это превращается в стабильный поток при $\beta, \gamma = 0$.) Показывается, что такой "квазипоток" существует и также может быть найден за время $O(nm)$. (В приложениях число $\gamma(v)$ можно понимать как разрешение "агенту" v распоряжаться по своему усмотрению частью полученного продукта в размере, не превышающем $\gamma(v)$, а число $\beta(v)$ – привлекать "со стороны" дополнительный продукт в размере не более $\beta(v)$.)

Данная статья организована следующим образом. В разд. 2 приводятся основные определения и точные формулировки задач SF и SA. В разд. 3 излагается базовый алгоритм нахождения стабильного потока в сети методом предпотоков и показывается его конечная сходимость (предложение 1). Модифицированная версия алгоритма, имеющая временну́ю сложность $O(nm)$, описывается в разд. 4. Раздел 5 содержит обобщения полученных результатов на предпотоки и квазипотоки с ограниченными эксцессами (предложения 3 и 4). В заключительном разд. 6 обсуждаются три дополнительных свойства: 1) максимум величин (т.е. суммарных эксцессов в стоках) для стабильных предпотоков достигается на стабильном потоке; 2) стабильные потоки в фиксированной сети имеют одинаковые величины; и 3) стабильные потоки образуют решетку. (Свойства в п. 2 и 3 показываются в [8] через сведение к соответствующим свойствам в задаче SA; мы описываем прямые конструкции.)

2. ОПРЕДЕЛЕНИЯ И ПОСТАНОВКИ

Мы рассматриваем сеть $N = (G, S, T, c)$, состоящую из *ориентированного* графа $G = (V, E)$ (без петель и кратных ребер), выделенных непересекающихся подмножеств вершин S (*источники*) и T (*стоки*), называемых также *терминалами*, и функции $c : E \to \mathbb{R}_+$ *пропускных способностей* ребер. (Здесь и далее $\mathbb{R}_+$ и $\mathbb{Z}_+$ – множества неотрицательных вещественных и неотрицательных целых чисел соответственно.) Для вершины $v \in V$ обозначим через $\delta^{\mathrm{in}}(v)$ и $\delta^{\mathrm{out}}(v)$ множества ребер *входящих* в v и *выходящих* из v соответственно. Для большей простоты нашего изложения мы предполагаем, что выполняется

$$\delta^{\mathrm{in}}(s) = \varnothing \quad \forall s \in S \quad \text{и} \quad \delta^{\mathrm{out}}(t) = \varnothing \quad \forall t \in T. \tag{2.1}$$

Определение 1. Функция $f : E \to \mathbb{R}_+$ называется *допустимой* (относительно c), если она удовлетворяет ограничениям $f(e) \leq c(e)$ для всех ребер $e \in E$. Определим *эксцесс* для f в вершине $v \in V$ как

$$\mathrm{ex}_f(v) := \sum_{e \in \delta^{\mathrm{in}}(v)} f(e) - \sum_{e \in \delta^{\mathrm{out}}(v)} f(e).$$

Допустимая функция f называется *предпотоком* в N (следуя терминологии в [11]), если каждая вершина $v \in V - S$ имеет неотрицательный эксцесс: $\mathrm{ex}_f(v) \geq 0$. *Поток* (из S в T) – это предпоток f с нулевыми эксцессами всех *внутренних* (нетерминальных) вершин $v \in V - (S \cup T)$, а его *величиной* $\mathrm{val}(f)$ считается $\mathrm{ex}_f(T) := \sum_{t \in T} \mathrm{ex}_f(t)$.

Путь в G – это последовательность $P = (v_0, e_1, v_1, \ldots, e_k, v_k)$, где e_i – ребро, соединяющее вершины v_{i-1} и v_i. Ребро e_i в P называется *прямым* (forward), если $e_i = v_{i-1}v_i$, и *обратным* (backward), если $e_i = v_i v_{i-1}$. (Мы обозначаем ребро, выходящее из u и входящее в v как uv, вместо обычного (u, v)). Путь называется *ориентированным* (directed), если все его ребра прямые, и называется *простым* (simple), если все его вершины различны. Противоположный (обратный) путь $(v_k, e_k, v_{k-1}, \ldots, e_1, v_0)$ обозначается через P^{-1}. Путь из вершины u в вершину v может быть назван *u–v путем*. Если не оговорено противное, то, говоря о пути, мы считаем его нетривиальным, т.е. имеющим по крайней мере одно ребро.

Для допустимой функции f ребро e с $f(e) = c(e)$ ($f(e) < c(e)$; $f(e) = 0$) называется *насыщенным* (соответственно, *ненасыщенным*; *свободным* (от f)). Путь считается ненасыщенным, если таковы все его ребра.

В рассматриваемой нами задаче каждая внутренняя вершина v сети N снабжена линейным (полным строгим) порядком $<_v^-$ на множестве $\delta^{\text{in}}(v)$ и линейным порядком $<_v^+$ на множестве $\delta^{\text{out}}(v)$. Они интерпретируются как "отношения предпочтения", а именно, $e <_v^- e'$ означает, что вершина ("агент") v предпочитает ребро e ребру e'; в этом случае мы также будем говорить, что e расположена в $\delta^{\text{in}}(v)$ раньше или левее e', а e' – позже или правее e. В соответствии с порядком $<_v^-$ множество $\delta^{\text{in}}(v)$ организуется в виде списка (double-linked list); таким образом, первый (последний) элемент в списке наиболее (соответственно, наименее) предпочтительный. И аналогично для порядка $<_v^+$.

Определение 2. Поток f в сети N с указанными порядковыми оснащениями для внутренних вершин называется *стабильным*, если каждый ненасыщенный ориентированный путь $P = (v_0, e_1, v_1, \ldots, e_k, v_k)$ удовлетворяет по крайней мере одному из следующих двух условий (где допускается случай $v_0 = v_k$):

$$\begin{gathered}\text{начальная вершина } v_0 \textit{ внутренняя, и } P \text{ доминируется в } v_0;\\ \text{это означает, что каждое ребро } e \in \delta^{\text{out}}(v_0) \text{ позднее } e_1 \text{ свободно от } f;\end{gathered} \tag{2.2}$$

$$\begin{gathered}\text{конечная вершина } v_k \text{ внутренняя, и } P \text{ доминируется в } v_k,\\ \text{т.е. каждое ребро } e \in \delta^{\text{in}}(v_k) \text{ позднее } e_k \text{ свободно от } f.\end{gathered} \tag{2.3}$$

В частности, нет ненасыщенного ориентированного пути из S в T. Предпоток f называется стабильным в аналогичном случае.

Нас интересует *задача о стабильном потоке* (SF): найти стабильный поток для сети N. Флейнер [8] установил, что эта задача всегда разрешима, а также имеет место свойство целочисленности: если сеть $N = (G = (V, E), S, T, c)$ целочисленная (т.е. $c \in \mathbb{Z}_+^E$), то существует целочисленный стабильный поток.

Это доказывалось путем редукции задачи SF для N к *задаче о стабильном распределении* (SA) на двудольном графе $G' = (V', E')$ с весами $c'(e) \geq 0$ ребер $e \in E'$ и "квотами" $q(v) \geq 0$ вершин $v \in V'$. (Строго говоря, в [8] рассматривался случай, когда $|S \cup T| = 2$, и условие (2.1) не накладывается, но указанные свойства верны и для нашего случая.) При этом редукция линейна по размерам, а именно: $|V'| = 2|V|$ и $|E'| < |E| + 2|V|$, и сохраняет целочисленность: c' и q целочисленные при целочисленном c. Задача SA была введена и исследована в [4], где были доказаны ее разрешимость для любой сети и свойство целочисленности и предложен сильно полиномиальный алгоритм решения.

Как было отмечено во введении, в [5] показано, что при использовании продвинутых структур данных, т.н. динамических и самонастраивающихся деревьев, задачу о стабильном распределении в графе с n вершинами и m ребрами можно решить с временно́й сложностью $O(m \log n)$ (а без использования таких структур можно добиться оценки $O(nm)$). Это, в силу указанной редукции, приводит к аналогичной оценке алгоритмической сложности и для задачи SF. Прямой алгоритм для SF в случае $|S| = |T| = 1$, предложенный в [10], имеет сложность $O(nm)$.

В настоящей работе предлагается альтернативный прямой алгоритм нахождения стабильного потока в произвольной сети $N = (G, S, T, c)$ (при условии (2.1)), основанный на методе предпотоков. В следующем разделе мы описываем базовую версию алгоритма (конечную при любом c), а в разд. 4 – модифицированную версию (с временно́й сложностью $O(nm)$).

3. БАЗОВЫЙ АЛГОРИТМ

Алгоритм нахождения стабильного потока в сети $N = (G = (V, E), S, T, c)$ состоит из последовательности итераций; как правило (но не всегда) итерация имеет две *фазы*: *балансирование* (balancing) и *достройка* (pushing) (подобно структуре этапа (большой итерации) в алгоритме построения максимального потока в [11]). Каждая итерация преобразует один блокирующий предпоток

в другой, и процесс завершается, когда текущий предпоток становится потоком. Подчеркнем, что термин “блокирующий” заимствован из языка в [11] и имеет иной смысл, чем тот, что обычно понимают в задачах о стабильности.

Определение 3. Для предпотока f в сети N скажем, что вершина v *эксцессная*, если $\mathrm{ex}_f(v) > 0$. Предпоток f называется *блокирующим*, если каждый ориентированный путь из S в T имеет насыщенное ребро. Если к тому же всякий ориентированный путь, идущий из эксцессной внутренней вершины в T имеет насыщенное ребро, то мы называем f *полностью блокирующим*.

Замечание 1. Полностью блокирующий предпоток не обязательно стабильный, и наоборот. В то же время всякий стабильный поток является блокирующим (и автоматически полностью блокирующим). В целом уже в ранжированной ациклической сети построение стабильного потока выглядит более сложной задачей, чем построение блокирующего потока. Последняя решается на этапе алгоритма в [11] за время $O(n^2)$, что быстрее, чем $O(nm)$ в алгоритме разд. 4 для SF.

3.1. Начальная итерация

На этой и последующих итерациях поддерживаются два списка Old и New, элементами которых являются эксцессные внутренние вершины текущуго предпотока. Также для каждой внутренней вершины v указан элемент $\tilde{e}(v)$, который либо принимает пустое значение: $\tilde{e}(v) = \{\varnothing\}$, либо является выделенным ненасыщенным ребром в $\delta^{\mathrm{out}}(v)$, называемым *активным* ребром. В начале полагается Old := New := $\varnothing$, и для каждого $v \in V - (S \cup T)$ активным полагается первое (самое предпочтительное) ребро в $\delta^{\mathrm{out}}(v)$.

Начальная итерация состоит из одной фазы – достройки, которая начинается с тривиального блокирующего предпотока f, определяемого как $f(e) := c(e)$ для всех ребер $e \in \delta^{\mathrm{out}}(s)$, $s \in S$, и $f(e) := 0$ для остальных ребер e. Каждая вершина v, соединенная ребром $e = sv$ с источником $s \in S$, заносится в список New (так как эксцесс в v становится положительным). Затем мы сканируем вершины текущего списка New.

Для очередной вершины $v \in$ New мы изменяем f на $\delta^{\mathrm{out}}(v)$ так, чтобы либо свести эксцесс в v к нулю, либо насытить все ребра в $\delta^{\mathrm{out}}(v)$ (либо и то и другое). Более точно, ребра $e \in \delta^{\mathrm{out}}(v)$ обрабатываются в порядке $<_v^+$ (“слева-направо”), начиная с активного ребра $e = \tilde{e}(v)$. Если текущий эксцесс $\Delta := \mathrm{ex}_f(v)$ еще ненулевой, назначаем $f(e) := \min\{c(e), f(e) + \Delta\}$. Если новый эксцесс обнулился (в случае $c(e) \geq \Delta$), обработка v заканчивается; иначе (когда $c(e) < \Delta$), переходим к следующему ребру в списке $\delta^{\mathrm{out}}(v)$, и т.д. Также при изменении f на ребре $e = vw$, если вершина w (в которой происходит увеличение эксцесса) не фигурирует ни в Old, ни в New, то мы добавляем w в текущий список New.

При окончании работы с v эта вершина удаляется (быть может временно) из списка New, и если эксцесс в v все еще ненулевой, v заносится в список Old. Новым активным ребром $\tilde{e}(v)$ становится самое левое ненасыщенное ребро, а если такого нет (в частности, когда $\mathrm{ex}_f(v) > 0$), то полагается $\tilde{e}(v) := \{\varnothing\}$. Затем переходим к обработке другой вершины в New, и т.д. Итерация заканчивается, когда New становится пустым.

Заметим, что в процессе итерации одна и та же внутренняя вершина v может несколько раз появляться и исчезать в New. Однако мы покажем позднее, что начальная (и каждая последующая) итерация всегда заканчивается. Можно видеть следующее.

> При завершении начальной итерации полученная функция f является полностью блокирующим предпотоком f с тем свойством, что для каждой внутренней вершины v : если $\tilde{e}(v) = \{\varnothing\}$, то все ребра в $\delta^{\mathrm{out}}(v)$ насыщены, а если $\tilde{e}(v) \neq \{\varnothing\}$, то все ребра в $\delta^{\mathrm{out}}(v)$ раньше (левее) e насыщены, а все ребра после e свободны от f. При этом все эксцессные внутренние вершины v (и только они) включены в список Old, и для таких v выполняется $\tilde{e}(v) = \{\varnothing\}$. (3.1)

Из (3.1) легко заключить выполнение (2.2) для всех ненасыщенных ориентированных путей; следовательно, начальный предпоток f стабильный.

В оставшейся части этого раздела мы сначала описываем фазу балансирования для общей итерации. Затем уточняем условия, которым должен удовлетворять текущий предпоток перед началом фазы достройки. Наконец, мы описываем эту фазу для общей итерации.

3.2. Балансирование

Эта процедура производится, когда множество Old эксцессных внутренних вершин для текущего предпотока f непусто (в то время как $\mathrm{New} = \varnothing$). Она применяется к одной выбранной вершине $v \in \mathrm{Old}$. Положим $\Delta := \mathrm{ex}_f(v)$ (> 0), и пусть e – последнее (самое правое, наименее предпочтительное) ребро в $\delta^{\mathrm{in}}(v)$ с $f(e) > 0$. Уменьшим f в e на $\min\{\Delta, f(e)\}$. Если эксцесс в v стал нулевым (в случае $\Delta \le f(e)$), заканчиваем. Иначе (когда $\Delta > f(e)$), берем последнее ребро e' с $f(e') > 0$ для обновленного f и уменьшаем f в e' аналогичным образом. И так далее, пока эксцесс в v не станет нулевым.

Для каждого ребра $e = uv$, в котором при данном балансировании было уменьшение f, проверяем вершину u, и если она внутренняя и не содержится в Old, то добавляем ее в список New. (Таким образом, New – это множество новых эксцессных вершин u, образовавшихся в результате уменьшения f на ребрах uv.) Самое левое (хронологически последнее) ребро, где было уменьшение f, назовем *критическим* и обозначим через $\hat{e} = \hat{e}(v)$. Это ребро и все ребра $e = uv$ в $\delta^{\mathrm{in}}(v)$ после (правее) него помечаются как *закрытые* (только для последующего увеличения!). При этом если такое e было активным ребром в $\delta^{\mathrm{out}}(u)$, то новым активным ребром назначается первое после e незакрытое (при более ранних балансированиях) ребро в $\delta^{\mathrm{out}}(u)$, а если такого нет, то полагается $\tilde{e}(u) := \{\varnothing\}$.

Замечание 2. Множества Old и New задаются в виде обычных двухсторонних списков. В процессе алгоритма надо поддерживать величины эксцессов для внутренних вершин (корректируя их за $O(1)$ действий при каждом изменении f).

Мы предполагаем (по индукции), что после балансирования в вершине v текущее f является предпотоком, который не обязан быть полностью блокирующим, но для которого выполнены следующие три свойства.

(C1) Каждое ненасыщенное ребро в $\delta^{\mathrm{out}}(s)$, $s \in S$, закрытое.

(C2) Если для эксцессной внутренней вершины u множество $\delta^{\mathrm{out}}(u)$ содержит незакрытое ненасыщенное ребро, то $u \in \mathrm{New}$.

(C3) Для каждой внутренней вершины u справедливо: а) если в $\delta^{\mathrm{out}}(u)$ есть активное ребро $\tilde{e}(u) \ne \{\varnothing\}$, то это ребро ненасыщенное и незакрытое, все ребра после него свободны от f, а каждое ребро до $\tilde{e}(u)$ – насыщенное или закрытое (ставшее таковым при последнем или более раннем балансировании); б) если $\tilde{e}(u) = \{\varnothing\}$, то все ребра в $\delta^{\mathrm{out}}(u)$ насыщенные или закрытые; и в) если u когда-либо подвергалась балансированию, то $\tilde{e}(u) = \{\varnothing\}$, в свою очередь в $\delta^{\mathrm{in}}(u)$ есть критическое ребро $\hat{e}(u)$, это ребро ненасыщенное, а все ребра после него свободны от f.

3.3. Достройка (после балансирования в вершине v)

Она состоит в увеличении текущего предпотока f на некоторых ребрах, чтобы сделать его полностью блокирующим; это схоже с построением начального предпотока, но имеет ряд особенностей. Достройка немедленно заканчивается, когда начальный список New пустой. Иначе она начинается с некоторой вершины $u \in \mathrm{New}$, и мы стремимся максимально уменьшить эксцесс в u. Для этого сканируем ребра в $\delta^{\mathrm{out}}(u)$ в порядке $<^+_u$, пропуская закрытые ребра. Сканирование начинается с активного ребра $\tilde{e}(u) \ne \{\varnothing\}$ (если $\tilde{e}(u) = \{\varnothing\}$, то вершина u просто переносится из New в Old, и работа с u заканчивается). Как и при начальной итерации, для очередного обрабатываемого ребра $e = uw$ полагаем $f(e) := \min\{c(e), f(e) + \Delta\}$, где Δ – текущий эксцесс в u, и одновременно добавляем вершину w в текущее множество New, если его еще не было в $\mathrm{Old} \cup \mathrm{New}$

(так как эксцесс в w увеличивается). Если новый эксцесс в u все еще ненулевой, переходим к следующему незакрытому ребру в $\delta^{\text{out}}(u)$, и т.д. Если эксцесс обнулился, то работу с u заканчиваем и удаляем u из New. Если все ребра отсканированы, но эксцесс в u все еще ненулевой, то переносим u из New в Old. Окончив работу с u и соответственно скорректировав активное ребро $\tilde{e}(u)$ в $\delta^{\text{out}}(u)$ (получая $\tilde{e}(u)$, удовлетворяющее (C3)а,б), переходим к другой вершине в текущем New, и т.д. Заканчиваем процесс достройки, когда множество New становится пустым.

При завершении достройки (которая конечна согласно предложению 1 ниже) построенное f является предпотоком, и можно видеть, что f имеет свойства (C1) и (C3), а (C2) заменяется на следующее свойство (ср. (C3)б).

(C2') Каждая эксцессная внутренняя вершина u содержится в Old, и $\tilde{e}(u) = \{\varnothing\}$.

Лемма 1. *Предпоток f, полученный при достройке, является стабильным и полностью блокирующим.*

Доказательство. Рассмотрим ненасыщенный ориентированный путь $P = (u_0, e_1, u_1, \ldots, e_k, u_k)$. Предположим, что либо $u_0 \in S$, либо P не доминируется в u_0 (когда u_0 внутренняя). Тогда ребро e_1 закрытое (ввиду (C1) и (C3)). Значит, вершина u_1 подвергалась балансированию и перед этим была эксцессной. Применяя (C2') и (C3)б к вершине u_1 и предпотоку f' в момент перед тем балансированием, заключаем, что ребро e_2 было закрытым в этот момент. Следовательно, e_2 закрытое и для f. Рассуждая так и далее, заключаем, что ребра $e_3, \ldots, e_k$ – тоже закрытые. (Случай $u_k \in T$ невозможен, так как в момент балансирования в вершине u_{k-1} эта вершина была эксцессной, и случай ненасыщенности e_k невозможен.) Так как e_k закрытое, оно должно встречаться в $\delta^{\text{in}}(u_k)$ после критического ребра $\hat{e}(u_k)$ либо совпадать с $\hat{e}(u_k)$. Тогда из (C3)в следует, что все ребра в $\delta^{\text{in}}(u_k)$ после e_k свободны от f. Следовательно, P доминируется в u_k, и выполняется (2.3). Это означает, что f стабильный.

Тот факт, что f полностью блокирующий, выводится из (C1),(C3),(C2') при аналогичных рассуждениях. Лемма доказана.

Используя указанные выше свойства предпотока, полученного при достройке, можно заключить, что при балансировании на следующей итерации возникает предпоток со свойствами (C1)–(C3). Это обосновывает корректность алгоритма.

3.4. Сходимость алгоритма

Из леммы 1 получаем

Следствие 1. Если на очередной итерации для текущего предпотока f эксцессы всех внутренних вершин стали нулевыми, то f – стабильный поток.

В этом случае f – требуемое решение задачи, и работа алгоритма заканчивается. Поскольку при каждом балансировании или достройке число эксцессных внутренних вершин может как уменьшиться, так и увеличиться, то для установления конечности алгоритма нужен дополнительный анализ.

Для текущего f назовем ребро e *средним*, если оно несвободное и ненасыщенное: $0 < f(e) < c(e)$. Пусть $\Gamma = (V_\Gamma, E_\Gamma)$ обозначает подграф в G, индуцированный средними ребрами. В частности, Γ содержит все несвободные критические ребра. Мы также добавим к Γ каждое *свободное* активное ребро (напомним, что активное ребро всегда незакрытое и ненасыщенное). Сделаем следующие наблюдения.

1. Для ребра $e = uv$ момент его насыщения назовем *событием S*, а момент освобождения (перехода от положительного к нулевому $f(e)$) – *событием F*. Изменения в e имеют "однопиковый" характер: вначале $f(e)$ монотонно увеличивается при достройках в u, а при первом уменьшении $f(e)$ ребро e становится закрытым и в дальнейшем $f(e)$ может только уменьшаться (при балансированиях в v).

2. Ребро e может добавиться в граф Γ не более двух раз: первый раз – когда e становится активным, и второй раз – когда e становится критическим.

Пусть α_S, α_F, α_M обозначают, соответственно, число событий S, число событий F и число событий M, состоящих в изменении графа Г. Из наблюдений выше следует, что

$$\text{каждое из чисел } \alpha_S, \alpha_F, \alpha_M \text{ оценивается как } O(m). \tag{3.2}$$

Таким образом, для анализа сходимости алгоритма надо оценить число подряд идущих итераций, на которых не происходит событий S, F и M. Для этого заметим следующее.

3. В процессе алгоритма для каждой внутренней вершины v активное ребро в $\delta^{\text{out}}(v)$ может сдвигаться только вправо (при достройках в v), а критическое ребро в $\delta^{\text{in}}(v)$ – только влево (при балансированиях в v); при этом статус таких ребер меняется: для старого активного ребра происходит событие S или ребро становится закрытым, а для старого критического ребра – событие F. В силу этого, обозначая $E^+ = E^+(f)$ и $E^- = E^-(f)$ множества активных и критических ребер в Г, соответственно, получаем, что

$$\begin{gathered}\text{в результате операции (балансирования или достройки) граф } \Gamma \\ \text{сохраняется тогда и только тогда, когда сохраняются оба } E^+ \text{ и } E^-.\end{gathered} \tag{3.3}$$

Можно также видеть, что эти множества дают разбиение E_Γ:

$$E^+ \cup E^- = E_\Gamma \quad \text{и} \quad E^+ \cap E^- = \varnothing$$

(учитывая (C3) и то, что критическое ребро $e = uv$ делается закрытым и не может оставаться активным в $\delta^{\text{out}}(u)$). Таким образом, при сохранении Г вместе с сохранением фиксированного разбиения (E^+, E^-) на подряд идущих итерациях каждое изменение предпотока f состоит в его уменьшении в некотором критическом ребре или его увеличении в некотором активном ребре. Заметим также, что активное ребро может стать критическим, но не наоборот. Эти обстоятельства вместе со свойствами (3.2) и (3.3) позволяют установить следующее

Предложение 1. *Предложенный алгоритм нахождения стабильного потока в сети $N = (G, S, T, c)$ конечный и при целочисленных пропускных способностях с находит целочисленный стабильный поток.*

Доказательство. Надо показать конечность последовательности итераций, на которых сохраняются E^+ и E^-. Пусть ε – минимальный положительный эксцесс среди внутренних вершин в начале этой последовательности. Предположим по индукции, что перед началом очередного изменения предпотока f на итерации этой последовательности множество $\text{Old} \cup \text{New}$ эксцессных вершин непусто, и (*): каждая из них имеет эксцесс не менее ε. Если в этот момент производится балансирование в вершине v, то, поскольку E^- сохраняется, балансирование сводится только к уменьшению f на величину $\Delta := \text{ex}_f(v) \geq \varepsilon$ в критическом ребре $\hat{e}(v) = uv$. Это обнуляет эксцесс в v и приводит к увеличению эксцесса в вершине u на ту же величину Δ; следовательно, свойство (*) верно для нового f (в случае $u \in S$ множество $\text{Old} \cup \text{New}$ просто уменьшается на один элемент v). Если же производится достройка во внутренней вершине u с активным ребром $\tilde{e}(u) \neq \varnothing$, то ввиду сохранения E^+, достройка сводится к увеличению f в этом ребре $\tilde{e}(u) = uw$ на величину $\Delta := \text{ex}(u) \geq \varepsilon$. Это обнуляет эксцесс в u и увеличивает эксцесс в вершине w на величину Δ; следовательно, свойство (*) верно для нового f (в случае $w \in T$ множество $\text{Old} \cup \text{New}$ уменьшается на один элемент u).

Таким образом, каждое изменение f состоит в его уменьшении на $\geq\varepsilon$ в ребре в E^- либо увеличении на $\geq\varepsilon$ в ребре в E^+. Значит, число итераций в данной последовательности не превосходит $c(E^- \cup E^+)/\varepsilon$, что дает конечность алгоритма.

При целочисленном c каждая операция изменяет предпоток в ребре на целое число, и, следовательно, результирующий стабильный поток целочисленный. Предложение доказано.

Несмотря на конечность алгоритма число итераций в нем может быть очень большим, как показывает следующий пример.

Пример. Пусть в G имеются вершины u, v, w и ребра uv, vw и uw, для которых $uv <_u^+ uw$ и $uw <_w^- vw$. Предположим, ребро vw критическое в $\delta^{\text{in}}(w)$, ребро uv критическое в $\delta^{\text{in}}(v)$, а ребро uw активное в $\delta^{\text{out}}(u)$. Будем считать, что величины $f(uv)$, $f(vw)$ и $c(uw) - f(uw)$ достаточно большие, эксцесс Δ в w – положительный и достаточно малый, а эксцессы в u и v равны нулю. Работа алгоритма может проходить так: сначала $f(vw)$ уменьшается на Δ, затем $f(uv)$ уменьшается на Δ, затем $f(uw)$ увеличивается на Δ. И мы возвращаемся к начальной вершине w с прежним эксцессом Δ (и при $\text{ex}(u) = \text{ex}(v) = 0$) и снова движемся по тому же циклу $w \to v \to u \to w$, и так много раз.

В следующем разделе мы расскажем, как модифицировать алгоритм, чтобы он приводил к более быстрому решению.

4. МОДИФИЦИРОВАННЫЙ АЛГОРИТМ

В изложенном выше базовом алгоритме число итераций, на которых граф Γ не меняется, может быть очень большим (как показывает пример в разд. 3); для удобства здесь и далее мы включаем разбиение (E^+, E^-) в описание Γ. В этом разделе мы описываем модифицированную версию, для которой число изменений предпотока f при постоянном $\Gamma = (V_\Gamma; E^+, E^-)$ имеет порядок $O(n)$. Это эквивалентно тому, что $O(n)$ изменений f приводят к событию S, F или M (включая тот случай, когда некоторое активное ребро закрывается и становится критическим).

Путь в Γ назовем *правильным*, если все активные ребра в нем прямые, а все критические ребра обратные. Последовательность итераций с постоянным Γ назовем *большой итерацией*. Она начинается с выбора эксцессной вершины v_0 в (G, f), и порядок обработки вершин уточняется следующим образом (полагая, что v_0 принадлежит Γ).

(A) Если при балансировании очередной эксцессной вершины v, состоящем в уменьшении f в критическом ребре uv (которое сохраняется в Γ), выясняется, что вершина u внутренняя и имеет пустое активное ребро (и тогда все ребра в $\delta^{\text{out}}(u)$ насыщенные или закрытые (ср. (C3)б), ввиду чего достройка в u невозможна), то переходим к балансированию в вершине u.

(B) Если при достройке в вершине u, состоящей в увеличении f в активном ребре $e = uw$ выясняется, что вершина w внутренняя и имеет непустое активное ребро wz, то далее производится достройка в w, а если $\tilde{e}(w) = \{\varnothing\}$, то переходим к балансированию в w. При этом балансировании, если критическим ребром становится то же самое ребро $e = uw$, то оно закрывается и перестает быть активным в u, в результате чего граф Γ изменяется, и большая итерация заканчивается.

Из этих уточнений следует, что последовательность обрабатываемых вершин и ребер образует правильный путь $P = (v_0, e_1, v_1, \ldots, e_i, v_i, \ldots)$. Для очередной вершины v_k в P возможны следующие особые ситуации:

(Q1) v_k совпадает с ранее пройденной вершиной v_i;

(Q2) v_k является терминалом.

Рассмотрим эти ситуации более подробно. В результате изменения f на $e_1, \ldots, e_k$ эксцессы всех вершин в P, кроме v_k, обнуляются.

1. В случае (Q1) выделяем правильный простой цикл $C = (v_i, e_{i+1}, \ldots, v_k)$. Подобно тому, как это делается для ротаций в алгоритмах для стабильных b-матчингах, распределениях, и др., мы изменяем f вдоль C на максимальную возможную величину Δ, равную минимуму величин $f(e)$ для $e \in E_C \cap E^-$ и величин $c(e') - f(e')$ для $e' \in E_C \cap E^+$, увеличивая f на Δ в прямых ребрах и уменьшая на Δ в обратных ребрах цикла C. В результате некоторое ребро в C насыщается или освобождается, и происходит событие M (вместе с S или F), завершая большую итерацию. При этом эксцессы всех вершин сохраняются, и можно видеть, что предпоток f продолжает быть стабильным и полностью блокирующим.

2. В случае (Q2) мы запоминаем P (где теперь эксцессы всех внутренних вершин нулевые) и, в предположении сохранения Γ, продолжаем большую итерацию, выбирая новую эксцессную

вершину v_0' в (G, f) и строя новый правильный путь $P' = (v_0', e_1', v_1, \ldots)$. Рассмотрим возможные варианты.

а) Если путь P' зацикливается, то действуем, как указано в 1, и заканчиваем большую итерацию.

б) Если P' встречает предыдущий путь P (ведущий в терминал) в вершине $v_r' = v_i$, то объединяем P' с P, получая дерево $\mathcal{T}$ (с корнем в некотором источнике $s \in S$ или "антикорнем" в стоке $t \in T$); при этом все внутренние вершины в $\mathcal{T}$, кроме v_r', имеют нулевой эксцесс. Продолжаем большую итерацию с новой эксцессной вершиной v_0'' в (G, f).

в) Если же P' попадает в терминал, отличный от терминала в P, то запоминаем P' (помимо P) и продолжаем с новой эксцессной вершиной.

г) В общем случае каждый новый построенный путь, если он не зацикливается и не попадает в новый терминал, встречается либо с деревом $\mathcal{T}$ с корнем в S, либо с деревом $\mathcal{T}'$ с антикорнем в T, и мы добавляем этот путь к данному дереву.

В результате при сохранении Γ (в частности, при отсутствии зацикливания) получаем следующее: в Γ построены несколько непересекающихся правильных деревьев $\mathcal{T}_1, \ldots, \mathcal{T}_k$ с корнями в S и деревьев $\mathcal{T}_1', \ldots, \mathcal{T}_\ell'$ с антикорнями в T, и все внутренние вершины в G, не лежащие в этих деревьях, имеют нулевые эксцессы.

При этом число изменений предпотока f равно $\left|E_{\mathcal{T}_1}\right| + \cdots + \left|E_{\mathcal{T}_k}\right| + \left|E_{\mathcal{T}_1'}\right| + \cdots + \left|E_{\mathcal{T}_\ell'}\right| \simeq O(n)$. Осталось объяснить, как избавляться от этих деревьев.

Предположим $\ell \neq 0$, и обозначим Z множество эксцессных вершин в $\mathcal{T}_1'$ (это в точности множество точек ветвления в $\mathcal{T}_1'$). Обойдя $\mathcal{T}_1'$, выделим в нем минимальное поддерево W с антикорнем в $t \in T$, содержащее Z, и перенумеруем ребра в W в топологическом порядке $w_1, \ldots, w_p$ (так, что если w_j лежит на пути, соединяющем w_i с t, то $i < j$). Перебирая ребра в этом порядке, изменяем f в них естественным образом, получая одно из двух: либо (i) эксцессы всех вершин в $\mathcal{T}_1' - \{t\}$ становятся нулевыми, либо (ii) некоторое промежуточное ребро становится насыщенным или свободным, что заканчивает большую итерацию.

Аналогично действуем и с другими деревьями $\mathcal{T}_i'$ и $\mathcal{T}_j$.

Таким образом, общая работа с деревьями осуществляется за время $O(n)$ и заканчивается либо событием M (вместе с S или F), либо избавлением от всех эксцессных внутренних вершин в G, и, следовательно, получением стабильного потока f. В течение большой итерации каждое ребро в Γ рассматривается не более $O(1)$ раз. Поэтому временна́я сложность большой итерации – $O(n)$. Этот факт вместе с (3.2) приводят к следующему результату.

Предложение 2. *Модифицированный алгоритм находит стабильный поток f в сети $N = (G = (V, E), S, T, c)$ за время $O(nm)$ (причем f целочисленный при целочисленном c).*

Замечание 3. С достаточной уверенностью можно сказать, что данный алгоритм может быть ускорен до формальной временно́й оценки $O(m \log n)$ путем применения структур данных как в [6], [7] для операций на графе Γ (таких как выделение и перестройку компонент и циклов в Γ, агрегированное изменение предпотока на циклах и деревьях, и др.), в том же духе, как подобного рода процедуры для подграфа "слабых ребер" ускоряются в алгоритме [5]. Мы оставляем проработку этих технических деталей за рамками данной статьи, сохраняя достаточную простоту и практическую применимость изложенного выше метода.

5. ОБОБЩЕНИЯ

Задача о стабильном потоке в сети $N = (G = (V, E), S, T, c)$ может быть обобщена путем введения для каждой внутренней вершины v верхней границы $\gamma(v) \in \mathbb{R}_+$ разрешенного эксцесса в v. (Как и прежде мы предполагаем условие (2.1).)

Определение 4. Предпоток $f : E \to \mathbb{R}_+$ в N назовем *γ-предпотоком*, если он допустимый (т.е. $f \leq c$) и удовлетворяет ограничениям

$$0 \leq \mathrm{ex}_f(v) \leq \gamma(v) \quad \text{для всех} \quad v \in V - (S \cup T). \tag{5.1}$$

В прикладных интерпретациях можно понимать ограничения в (5.1) как разрешение “агенту” v не передавать далее весь продукт, доставленный по входящим ребрам-каналам $e \in \delta^{\text{in}}(v)$, а откладывать в запас (или отправлять “на сторону”) часть полученного продукта в размере, не превышающем $\gamma(v)$.

Мы называем γ-предпоток *стабильным*, если для каждого ненасыщенного ориентированного u–v пути P верно по крайней мере одно из двух: а) вершина v внутренняя, и P удовлетворяет (2.3) (при $v_k = v$), т.е. P доминируется в v; или б) вершина u внутренняя, и P *сильно доминируется* в u, что означает выполнение (2.2) (при $v_0 = u$) и отсутствие эксцесса в u: $\text{ex}_f(u) = 0$. При $\gamma = 0$ это превращается в определение стабильного потока.

Обобщение полученных выше результатов на γ-предпотоки выглядит так.

Предложение 3. *Для сети $N = (G = (V, E), S, T, c)$ и вектора $\gamma \in \mathbb{R}_+^{V-(S\cup T)}$ стабильный γ-предпоток существует и может быть найден за время $O(nm)$.*

Доказательство. Данная задача сводится к задаче о стабильном потоке в расширенной сети N' с графом $G' = (V, E')$, полученным из G добавлением для каждой внутренней вершины v ребра vt пропускной способности $c(vt) := \gamma(v)$, которое ставится в конец порядка на $\delta^{\text{out}}(v)$, а именно, $e <_v^+ vt$ для всех $e \in \delta_G^{\text{out}}(v)$. Здесь t – выделенный сток в T. Пусть f' – стабильный поток для N', и f – его ограничение на G. Тогда f – это γ-предпоток, и его стабильность следует из стабильности f'. (Действительно, если P – ненасыщенный ориентированный u–v путь в G, который, будучи рассмотрен как путь в G', доминируется в u, то выполняется $f'(ut) = 0$, и мы получаем $\text{ex}_f(u) = 0$.) Предложение доказано.

Мы можем обобщить задачу далее, вводя для каждой внутренней вершины v, помимо $\gamma(v)$ как выше, границу $\beta(v) \in \mathbb{R}_+$ для величины $-\text{ex}(v)$. Иначе говоря, мы рассматриваем допустимую (по c) функцию $f : E \to \mathbb{R}_+$, удовлетворяющую

$$-\beta(v) \le \text{ex}_f(v) \le \gamma(v) \quad \text{для всех} \quad v \in V - (S \cup T). \tag{5.2}$$

Такое f назовем (β,γ)-*квазипотоком*. Можно понимать ограничение $-\beta(v) \le \text{ex}_f(v)$ в (5.2) как разрешение “агенту” v брать из запаса или привлекать со стороны продукт в размере не более $\beta(v)$ для передачи далее вместе с продуктом, доставленный по входящим ребрам-каналам (при этом, как и прежде, “агенту” разрешается отложить в запас или отправить на сторону часть, не превышающую $\gamma(v)$). Проще говоря, для (β,γ)-квазипотока f, если величина $\Delta(v) := \text{ex}_f(v)$ положительная, то “агент” v запасает $\Delta(v)$ единиц продукта, а если отрицательная – берет из запаса $-\Delta(v)$ единиц.

Назовем (β,γ)-квазипоток *стабильным*, если для каждого ненасыщенного ориентированного u–v пути P верно по крайней мере одно из двух: а) вершина u внутренняя, и P сильно доминируется в u (относительно γ) в том смысле, что выполняется (2.2) (при $v_0 = u$), и эксцесс в u неположительный (и не менее $-\beta(u)$); или б) вершина v внутренняя, и P сильно доминируется в v (относительно β), что означает выполнение (2.3) (при $v_k = v$) и неотрицательный эксцесс в v (не более $\gamma(v)$). При $\beta = \gamma = 0$ получаем определение стабильного потока.

Для данного обобщения справедливо

Предложение 4. *Для сети $N = (G = (V, E), S, T, c)$ и векторов $\beta,\gamma \in \mathbb{R}_+^{V-(S\cup T)}$ стабильный (β,γ)-квазипоток существует и может быть найден за время $O(nm)$.*

Доказательство. Рассмотрим задачу о стабильном потоке в расширенной сети N' с графом $G' = (V', E')$, полученным из G: а) расщеплением каждой внутренней вершины v на две копии v' и v'', где v' наследует входящие ребра из $\delta^{\text{in}}(v)$, а v'' – выходящие ребра из $\delta^{\text{out}}(v)$ (с теми же пропускными способностями и порядками на них); б) добавлением ребра $v'v''$ большой пропускной способности; и в) добавлением ребра $v't$ пропускной способности $c(v't) := \gamma(v)$ и ребра sv'' пропускной способности $c(sv'') := \beta(v)$, где $v't$ и sv'' полагаются менее приоритетными, чем ребро $v'v''$, и где s и t – выделенные источник и сток соответственно. Пусть f' – стабильный поток для N', и f – его “образ” в G. Можно проверить, что f – это (β,γ)-квазипоток в N, и его стабиль-

ность следует из стабильности f'. Здесь существенен тот факт, что для каждой внутренней вершины $v \in V$ по крайней мере одно из значений $f'(sv'')$ и $f'(v't)$ должно быть нулевым (что следует из рассмотрения ненасыщенного пути $(v', v'v'', v'')$). Предложение доказано.

6. ДОПОЛНИТЕЛЬНЫЕ ЗАМЕЧАНИЯ

В этом заключительном разделе указываются три дополнительных интересных свойства стабильных потоков и предпотоков. Как и выше, мы рассматриваем ориентированную сеть $N = (G = (V, E), S, T, c)$ с линейными порядками на $\delta^{\text{in}}(v)$ и $\delta^{\text{out}}(v)$ для внутренних вершин $v \in V - (S \cup T)$ (и при условии (2.1)).

I. Для предпотока f в сети $N = (G, S, T, c)$ будем считать величиной f суммарный эксцесс в стоках: $\text{val}(f) := \text{ex}_f(T)$ $(= \sum_{t \in T} \text{ex}_f(t))$. Справедливо следующее свойство:

> для стабильного предпотока f в N найдется стабильный поток f' в N такой, что $f(e) \le f'(e)$ для всех ребер, входящих в T, и, следовательно, выполняется $\text{val}(f) \le \text{val}(f')$. (6.1)

Действительно, если f не является потоком (т.е. имеет эксцессную внутреннюю вершину), то f перестраивается в стабильный поток f' применением последовательности итераций базового алгоритма. Каждое балансирование не изменяет значений в ребрах, входящих в T, а достройка может только увеличивать эти значения, что дает требуемое свойство (6.1).

II. Что касается величин стабильных потоков, то, обобщая классические результаты для стабильных марьяжей, стабильных двудольных b-матчингов и др., Флейнер [8, Sec. 4] установил, что:

> для любых стабильных потоков f и g в сети N значения $f(e)$ и $g(e)$ совпадают для каждого ребра e, инцидентного терминалу. (6.2)

Как следствие, $\text{val}(f) = \text{val}(g)$. Свойство (6.2) доказывалось в [8] (где рассматривались двухполюсные сети и допускались произвольные ребра, инцидентные терминалам) путем сведения к соответствующему свойству для стабильных распределений. Для наших сетей можно дать прямое и наглядное доказательство.

А именно, свяжем с функцией $f - g$ соответствующее разложение по путям и циклам. Более точно, поскольку f и g не имеют эксцессных внутренних вершин, можно образовать семейство $\mathcal{C}$, состоящее из простых путей и циклов C с весами $\Delta(C) > 0$ так, чтобы выполнялось следующее:

(i) для каждого $C \in \mathcal{C}$ его прямые ребра e удовлетворяют $f(e) > g(e)$, а обратные ребра e' удовлетворяют $f(e') < g(e')$;

(ii) для каждого ребра $e \in E$ сумма весов $\Delta(C)$ по путям/циклам C, содержащим e, равна $|f(e) - g(e)|$;

(iii) каждый путь в $\mathcal{C}$ соединяет два разных терминала, а каждый цикл содержит не более одного терминала.

Предположим, f и g не совпадают на некотором ребре e, инцидентном терминалу, и рассмотрим случай, когда e выходит из источника $s \in S$ (т.е. $e \in \delta^{\text{out}}(s)$). Для определенности положим $f(e) > g(e)$. Тогда имеется $C \in \mathcal{C}$, содержащее e (как прямое ребро); при этом либо а) C – путь из s в $t \in T$, либо б) C – путь из s в $s' \in S$ (возможно, $s' = s$).

В случае а) имеется последовательность $s = v_0, v_1, \ldots, v_k = t$ вершин в C (где k нечетное), для которой часть C от v_{i-1} до v_i имеет только прямые ребра при нечетном i, и только обратные ребра при четном i. Тогда

при нечетном i имеется ориентированный путь P_i из v_{i-1} в v_i, на ребрах которого f превосходит g, и, следовательно, P_i ненасыщенный для g и не содержит ребер,свободных от f; в свою очередь, при четном i имеется ориентированный путь Q_i из v_i в v_{i-1}, на ребрах которого g превосходит f, и, следовательно, Q_i ненасыщенный для f и не содержит ребер, свободных от g. (6.3)

(Такие пути фигурируют в конкатенации $P_1 \cdot Q_2^{-1} \cdot P_3 \cdot Q_4^{-1} \cdot \ldots \cdot Q_{k-1}^{-1} \cdot P_k$ для C, где Q^{-1} обозначает путь, обратный пути Q.) Обозначим p_i (p_i') первое (соответственно, последнее) ребро в P_i, и обозначим q_j (q_j') первое (соответственно, последнее) ребро в Q_j. Можно видеть, что

$$p_i', q_{i+1}' \in \delta^{\text{in}}(v_i) \text{ при нечетном } i < k, \text{ и } q_i, p_{i+1} \in \delta^{\text{out}}(v_i) \text{ при четном } i. \tag{6.4}$$

Теперь применим (6.3) и (6.4), двигаясь шаг за шагом по последовательности путей $P_1, Q_2, P_3 \ldots$. Так как P_1 начинается в источнике $v_0 = s$ и является ненасыщенным для g, и так как выполняется $g(q_2') > 0$, то из стабильности g следует, что $q_2' <^-_{v_1} p_1'$. Тогда из стабильности f, ненасыщенности Q_2 для f, неравенства $f(p_3) > 0$, и свойства, что Q_2 не доминируется в v_1 (в силу $f(p_1') > 0$), получаем $p_3 <^+_{v_2} q_2$. И так далее. Дойдя до пути Q_{k-1}, получаем $p_k <^+_{v_{k-1}} q_{k-1}$. Но тогда последний путь P_k (который заканчивается в стоке $v_k = t$ и является ненасыщенным для g) не доминируется в начальной вершине v_{k-1}, вопреки стабильности g.

В случае б) рассуждения схожи. Здесь мы имеем дело с конкатенацией путей вида $P_1, Q_2^{-1}, \ldots, P_{k-1}, Q_k^{-1}$ (при четном k). При этом последний путь Q_k начинается в источнике $v_k = s'$, является ненасыщенным для f, и не доминируется в концевой вершине v_{k-1} (в силу $f(p_{k-1}') > 0$ и $q_k' <^-_{v_{k-1}} p_{k-1}'$), что противоречит стабильности f. По соображениям симметрии, случаи несовпадений f и g в ребре, входящем в T, также невозможны. Этим заканчивается доказательство (6.2).

III. Предыдущая конструкция может быть продолжена. А именно, рассмотрим два стабильных потока f и g в сети $N = (G, S, T, c)$. Пусть H – подграф в G, порожденный ребрами в $A := \{e : f(e) > g(e)\}$ и $B := \{e : g(e) > f(e)\}$, и назначим ребрам e в H веса $\omega(e) := |f(e) - g(e)|$. Согласно (6.2), H не содержит терминальных вершин. Более того, ω раскладывается в неотрицательную линейную комбинацию характеристических функций простых правильных циклов (и тем самым ω может рассматриваться как “циркуляция”).

Здесь мы говорим, что (не обязательно простой) цикл C в H является *правильным*, если все прямые ребра в нем принадлежат A, а обратные принадлежат B. Для вершины v в таком цикле обозначим $\Pi_C(v)$ множество пар подряд идущих ребер, инцидентных v и следующих в направлении цикла. Назовем пару $\pi \in \Pi_C(v)$ *особой*, если π содержит ребро как из A, так и из B (эквивалентно, оба ребра в π либо входят в v, либо выходят из v). В этом случае скажем, что пара $\pi = (e, e')$ *левая*, если e предпочтительнее для v, чем e'; иначе пара называется *правой*. Рассуждая как при доказательстве (6.2), из стабильности f и g можно заключить, что

для любого правильного цикла C особые пары ребер для всех вершин в C имеют одинаковую ориентацию: либо все левые, либо все правые. (6.5)

Это свойство распространяется на компоненты связности K в H: все особые пары ребер, встречающиеся в правильных циклах в K, одновременно либо левые, либо правые (это следует из того, что две такие пары π, π' можно включить в один правильный (не обязательно простой) цикл в K). В соответствии с этим выделим четыре типа компонент K. А именно, скажем, что K имеет: *тип A* (*тип B*), если все ребра в K принадлежат множеству A (соответственно, B); и имеет

тип L (*тип R*), если K имеет ребра в обоих множествах A и B (в этом случае назовем компоненту K *богатой*), и ориентации всех особых пар в K левые (соответственно, правые). Эквивалентно,

богатая компонента K имееи тип L, если для любой особой вершины v в K
в множестве $\delta^{\text{in}}(v)$ ребра из A предшествую требрам из B, а в множестве $\delta^{\text{out}}(v)$ (6.6)
ебра из B предшествуют ребрам из A; в случае типа R ситуация противоположная.

Пользуясь этим, можно явно описать множество стабильных потоков в N в виде решетки (что делается в [8] путем сведения к задаче о стабильном распределении и апелляции к соответствующему результату в [4]). А именно, определим следующие функции h, ℓ на E:

h совпадает с f на компонентах типа A и R и совпадаеи с g
на компонентах типа B и L, а ℓ определяется противоположным образом; (6.7)
на остальных ребрах e полагается $h(e) := \ell(e) := f(e) = g(e)$.

Можно видеть, что h и ℓ являются потоками. А также для каждой внутренней вершины v поток h доминирует потоки f, g на множестве $\delta^{\text{out}}(v)$, и доминируется этими потоками на множестве $\delta^{\text{in}}(v)$, в то время как ℓ ведет себя противоположным образом. (Здесь для числовых функций a, b на упорядоченном множестве $(S, \prec)$ считаем, что a *доминирует* b, если либо $a = b$, либо имеется $e \in S$ такое, что $a(e) > b(e)$, $a(e') \geq b(e')$ при $e' \prec e$, и $a(e'') \leq b(e'')$ при $e \prec e''$.)

Из вышесказанного можно получить (мы опускаем подробности), что h совпадает с указанным в [8, Sec. 4] потоком $f \vee g$, а ℓ совпадает с потокам $f \wedge g$; в частности, оба h, ℓ стабильные.

Автор выражает благодарность рецензенту за анализ начальной версии статьи и указание работ [10], [12], не известных автору при ее написании.

СПИСОК ЛИТЕРАТУРЫ

1. *Gale D., Shapley L.S.* College admissions and the stability of marriage // Amer. Math. Monthly. 1962. V. 69. № 1. P. 9–15.
2. *Irving R.W.* An efficient algorithm for the “stable roommates” problem // J. Algorithms. 1985. V. 6. P. 577–595.
3. *Tan J.* A necessary and sufficient condition for the existence of a complete stable matching // J. Algorithms. 1991. V. 12. P. 154–178.
4. *Baiou M., Balinski M.* Erratum: the stable allocation (or ordinal transportation) problem // Math. Oper. Res. 2002. V. 27. № 4. P. 662–680.
5. *Dean B.C., Munshi S.* Faster algorithms for stable allocation problems // Algorithmica. 2010. V. 58. № 1. P. 59–81.
6. *Sleator D.D., Tarjan R.E.* A data structure for dynamic trees // J. Computer and System Sciences. 1983. V. 26. № 3. P. 362–391.
7. *Sleator D.D., Tarjan R.E.* Self-adjusting binary search trees // J. ACM. 1985. V. 32. № 3. P. 652–686.
8. *Fleiner T.* On stable matchings and flows // Algorithms. 2014. V. 7. P. 1–14.
9. *Ostrovsky M.* Stability in supply chain networks // Amer. Econ. Rev. 2006. V. 98. P. 897–923.
10. *Cseh Á., Matuschke J.* New and simple algorithms for stable flow problems // Algorithmica. 2019. V. 81. № 6. P. 2557–2591.
11. *Карзанов А.В.* Нахождение максимального потока в сети методом предпотоков // Докл. АН СССР. 1974. Т. 215. С. 49–52. (English translation in: Soviet Math. Dokl. 1974. V. 15. № 2. P. 434–437.)
12. *Cseh Á., Matuschke J., Skutella M.* Stable flows over time // Algorithms. 2013. V. 6. P. 532–545.