САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Мальковский Николай Владимирович

кафедра системного программирования

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Построение оптимальных потоковых процессов в задачах распределения загрузки вычислительной сети

Рецензент:
кандидат физико-математических наук
Шалымов Дмитрий Сергеевич

Заведующий кафедрой:
доктор физико-математических наук, профессор
Терехов Андрей Николаевич

Научный руководитель:
доктор физико-математических наук, профессор
Граничин Олег Николаевич

Санкт-Петербург
2016

SAINT-PETERSBURG STATE UNIVERSITY

Malkovskii Nikolai Vladimirovich

software engineering department

Graduation Thesis

Synthesis of the Optimal Control Processes in Network Load Balancing Problems

Reviewer:
candidate of physico-mathematical sciences
Shalymov Dmitrii Sergeevich

Head of department:
doctor of physico-mathematical sciences, professor
Terekhov Andrei Nikolaevich

Scientific supervisor:
doctor of physico-mathematical sciences, professor
Granichin Oleg Nikolaevich

Saint-Petersburg
2016

Оглавление

1 Введение 5
2 Оптимальные динамические процессы 8
3 Динамические потоковые процессы 12
4 Метод проталкивания предпотока для потоковых задач линейного программирования 16
4.1 Общая классификация потоковых задач 16
4.2 Задача о максимальном потоке 17
4.3 Задача о параметрическом потоке 22
5 Потоковые процессы в условиях неопределенностей 29
5.1 Класс усредняемых функций 30
5.2 Динамические потоковые процессы с меняющимися во времени пропускными способностями 32
5.3 Подбор параметров алгоритма в стохастическом случае 38
6 Практические примеры 41
6.1 Задача балансирования нагрузки в вычислительной сети 41
6.2 Обмен информацией групп БПЛА 44
7 Заключение 47

Литература **48**

Аннотация

На текущий момент потоковые задачи являются одними из классических задач информатики и входят в стандартные курсы большинства университетов мира. Эти задачи возникают естественным образом в условиях ограниченных пропускных способностей, будь это количество полос на дорогах или пропускная способность каналов связи в информационных сетях. В этой работе описан класс потоковых задач, общая цель которых заключается в приведении системы из одного состояния в другое за наименьшее время. В случае, когда пропускные способности не изменяются со временем, задача вырождается в одну из хорошо изученных в информатике потоковых задач оптимизации. Если пропускные способности меняются со временем, то задача значительно усложняется. Задача становится еще сложнее, если изменение пропускных способностей со временем связано с внешними неизвестными возмущениями. В этой работе описан метод решения такой задачи на основе усреднения: сложная динамическая задача заменяется на простую статическую, которую можно быстро решить потоковыми методами оптимизации. Для построения такой статической задачи достаточно знать только средние характеристики пропускных способностей, что позволяет использовать этот подход при наличии неопределенностей. Наконец, разработанный подход применяется для двух конкретных практических задач: балансирование загрузки вычислительной сети и сбор информации группой беспилотных летательных аппаратов.

1 Введение

В последние десятилетия сетевые технологии обретают все большую актуальность, и уже сейчас возможность решать различные задачи, возникающие в сетевых системах, является одним из самых важных приложений компьютерных и кибернетических наук. Как и для многих других крупных областей, существует огромное множество задач различных по своей структуре и возможным подходам к решению, в этой области скорее всего даже не существует универсальных подходов к решению 95% всех возникающих задач. К хорошо изученным теоретическим моделям, которые чаще всего применимы в теории и практике, можно причислить только наиболее общие – теорию графов, теорию динамических систем, методы математической оптимизации и т. п.

Класс задач, рассматриваемый в этой работе, нацелен на изучение *потоковых процессов* в сетях. Потоковый процесс заключается в движении частиц внутри некоторой замкнутой системы, при этом ни одна частица не покидает систему, и ни одна частица не проникает в систему извне. В этой работе будут рассматриваться сетевые системы, взаимодействие которых традиционно моделируется с помощью теории графов: вершины графа являются промежуточным местом для хранения потока, а дуги (ребра) являются средством циркуляции потока по графу (здесь и далее при использовании термина “дуга” подразумевается, что движение возможно в одну сторону, а при использовании термина “ребро” – в обе стороны. В дальнейших разделах этот нюанс будет формализован более строго, но стоит иметь в виду, что в потоковых задачах отсутствует качественное различие между ориентированным и неориентированным случаями). Другими словами, частицы потока могут перемещаться от одной вершины до другой по ребру, соединяющей эти вершины. В качестве практических примеров потоковых процессов можно привести следующие ситуации:

- Движение автомобилей по дорогам. Здесь частицами потока являются машины, ребра – дороги / улицы, узлы – пересечение дорог / улиц.

- Движение жидкости / газа по трубопроводу. Частицы потока – жидкость или газ, ребра – трубы, узлы – трубопроводные краны, распределительные станции.

- Движение данных в информационных сетях. Частицы потока – пакеты данных, ребра – каналы связи, вершины – логические и вычислительные устройства.

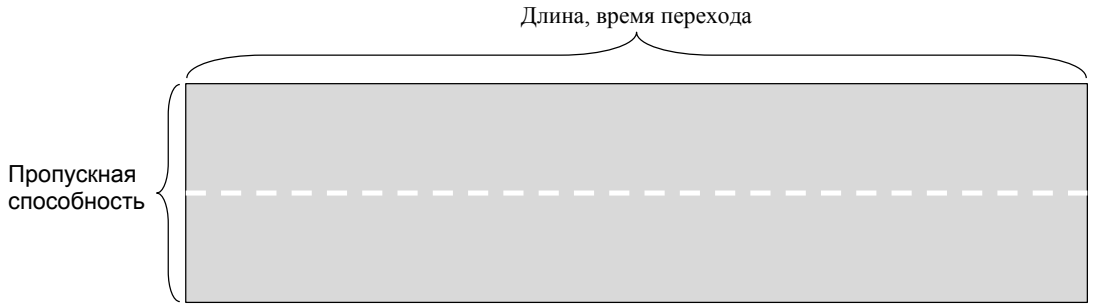

Рис. 1: Параметры дуги в транспортной сети.

Свойства, ограничения и параметры движения потока по ребру могут варьироваться в зависимости от задачи, но в целом для потоковых задач приняты следующие характерные свойства движения потока:

1. Объем потока, который может пройти по ребру за единицу времени (или за любой конечный интервал времени) ограничен, точную верхнюю границу этого объема принято называть *пропускной способностью* ребра.

2. Частицы потока проходят по ребру не мгновенно, а на протяжении некоторого интервала времени.

Как изображено на рис. 1, в случае транспортных потоков эти величины могут быть интерпретированы как ширина и длина дороги соответственно.

Ключевым аспектом этой работы является изучение динамических моделей потоковых процессов, т.е. таких моделей, которые учитывают временное измерение. Во многих случаях изучение протекающих во времени потоковых процессов может быть сведено к рассмотрению стационарных потоковых задач. Так, например, в [1–4] рассмотрены статические методы моделирования транспортных потоков, в [5] (одна из наиболее ранних работ на эту тему) и [6] рассмотрены динамические потоковые процессы, задачи нахождения оптимального процесса со статическими пропускными способностями сведены к статическим потоковым задачам. Работы [5–7] имеют постановки, схожие с рассматриваемыми в этой работе, и, в том числе, показывают актуальность этих задач. Эти работы имеют прямое отношение к *задаче о максимальном потоке* и *задаче потока минимальной стоимости*, введенных в [8]. Несмотря

на более полувека с момента появления, постановки этих задач, допускающие неопределенности некоторых характеристик, появились относительно недавно (см. [9–11]). При этом, на данный момент отсутствуют работы, изучающие потоковые задачи в условиях динамически изменяющихся параметров и внешних неконтролируемых возмущений. Анализ поведения потоковых процессов в таких ситуациях является основным теоретическим вкладом этой работы.

Работа организована следующим образом: в разделе 2 приведены общие сведения о динамических системах, сформулирован общий вид задач без неопределенностей, рассматриваемых в работе, и получено сведение задачи нахождения оптимального процесса к задаче выпуклой оптимизации. В разделе 3 описана специфика параметров потоковых процессов в рассматриваемых задачах оптимального управления, подробнее описаны оптимизационные задачи, которые необходимо решить для построения оптимального процесса. В разделе 4 подробно описаны потоковые задачи линейного программирования, возникающие при построении оптимального процесса, и эффективные методы их решения. В разделе 5 сосредоточен основной вклад работы: показано, что если меняющиеся во времени пропускные способности принадлежат классу усредняемых функций (как функции по времени), то оптимальное время работы системы асимптотически эквивалентно оптимальному времени работы “усредненной” системы; приведены примеры построения субоптимальных решений в стохастических случаях. В разделе 6 приведены практические примеры динамических потоковых процессов с неопределенностями.

2 Оптимальные динамические процессы

Традиционно, динамика управляемой системы (с непрерывным временем) описывается уравнением

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}), \tag{2.1}$$

где $\mathbf{x}(\mathbf{t}) = (x_1(t), \ldots, x_n(t))^T \in X$ – вектор состояния (фазовые переменные), $\mathbf{u}(t) = (u_1(t), \ldots, u_m(t))^T \in U$ – управляющее воздействие. Далее, задана некоторая функция $g(\mathbf{x}, \mathbf{u})$ – целевой функционал минимизации, который можно интерпретировать как "сложность" или "стоимость" управления $\mathbf{u}$ в состоянии $\mathbf{x}$, в фазовом пространстве выделены два вектора (краевые условия) $\mathbf{x}^-, \mathbf{x}^+ \in X$. *Оптимальной траекторией* системы (2.1), переводящей систему из состояния $\mathbf{x}^-$ в состояние $\mathbf{x}^+$ по отношению к функционалу $g(\cdot, \cdot)$ является траектория $\mathbf{x} : [0, \tau] \to \mathbb{R}^n, \mathbf{x}(t) \in X$ и соответствующее ей управление $\mathbf{u} : [0, \tau] \to \mathbb{R}^m$, $\mathbf{u}(t) \in U$, которая вместе с управлением $\mathbf{u}$ удовлетворяет уравнению динамики (2.1), минимизирует функционал (2.2)

$$\int_0^\tau g(\mathbf{x}(t), \mathbf{u}(t))dt \tag{2.2}$$

и удовлетворяет граничным условиям (2.3)

$$\mathbf{x}(0) = \mathbf{x}^-, \mathbf{x}(\tau) = \mathbf{x}^+. \tag{2.3}$$

В случае $g(\cdot, \cdot) \equiv 1$ значение функционала (2.2) равно τ, а задача заключается в нахождении наименьшего возможного времени перевода системы из состояния $\mathbf{x}^-$ в состояние $\mathbf{x}^+$. Далее в работе будут рассматриваться только такие задачи.

Основными инструментами анализа задач оптимальных процессов являются *уравнение Беллмана* (динамическое программирование [12] и *принцип максимума* [13]). Если правая часть (2.1) не зависит от $\mathbf{x}$, то ни уравнение Беллмана, ни принцип максимума не дают никакой полезной информации. С точки зрения теории оптимального управления такие задачи являются вырожденными, а основная сложность заключается в нахождении решений некоторых статических оптимизационных задач. Дальнейший анализ формально показывает вырожденность такого класса задач (формулировки принципа максимума и уравнения Беллмана для задач оптимального управления приведены в [13]).

Пусть $\psi \in \mathbb{R}^n$, определим функцию H как

$$H(\psi, \mathbf{x}, \mathbf{u}) = \psi^T f(\mathbf{x}, \mathbf{u}).$$

Согласно принципу максимума, если $\mathbf{u}^*(t)$ – такое управление (2.1), что выполняются граничные условия (2.3) и при этом τ принимает минимальное возможное значение, то существует такая функция $\psi(t) \in \mathbb{R}^n$, что вдоль траектории движения выполняются условия

$$\forall t \in [0,\tau]: \; H(\psi(t),\mathbf{x}(t),\mathbf{u}^*(t)) = \sup_{\mathbf{u}\in U} H(\psi(t),\mathbf{x}(t),\mathbf{u}), \tag{2.4}$$

$$\frac{d\psi}{dt} = -\frac{\partial H}{\partial \mathbf{x}}. \tag{2.5}$$

и

$$H(\psi(\tau),\mathbf{x}(\tau),\mathbf{u}^*(\tau)) \geq 0. \tag{2.6}$$

Пусть теперь $f(\mathbf{x},\mathbf{u})$ не зависит от $\mathbf{x}$, т.е. $f(\mathbf{x},\mathbf{u}) = F(\mathbf{u})$. Тогда обе части (2.4) не зависят от $\mathbf{x}$. Тогда в силу (2.5) $\psi(t)$ не зависит от t, а значит для любой допустимой траектории $\mathbf{x}(t)$ можно взять ψ тождественно равной $\mathbf{0}_n = (\underbrace{0,\ldots,0}_{n})^T$.

Согласно принципу динамического программирования, существует функция $\omega(\mathbf{x})$, такая что

$$\sup_{\mathbf{u}\in U} \nabla\omega(\mathbf{x})^T f(\mathbf{x},\mathbf{u}) = 1, \tag{2.7}$$

$\omega(\mathbf{x}) \leq 0$, $\omega(\mathbf{x}^+) = 0$. Физический смысл функции $\omega(\cdot)$ заключается в том, что "$-\omega(\mathbf{x})$" – минимальное время, за которое можно добраться из $\mathbf{x}$ в $\mathbf{x}^+$. При фиксированном $\mathbf{x}$ управление $\mathbf{u}$, на котором (2.7) достигает максимума, является локальным оптимальным управлением в точке $\mathbf{x}$. Таким образом, оптимальное управление $\mathbf{u}^*(t)$ удовлетворяет

$$\forall t \in [0,\tau]: \; \nabla\omega(\mathbf{x}(t))^T f(\mathbf{x}(t),\mathbf{u}^*(t)) = 1.$$

Для траектории $\mathbf{x}(t)$, порождаемой оптимальным управлением $\mathbf{u}^*(t)$, вектор $\psi(t) = \nabla\omega(\mathbf{x}(t))^T$ удовлетворяет (2.5), (2.6) и (2.4), при этом в последнем правая часть равна единице, что не позволяет взять $\psi(t) \equiv \mathbf{0}_n$ для $f(\mathbf{x},\mathbf{u}) = F(\mathbf{u})$. Подставив $F(\mathbf{u})$ вместо $f(\mathbf{x},\mathbf{u})$ в (2.7) получаем, что $\nabla\omega(\mathbf{x})$ не зависит от $\mathbf{x}$, а значит $\omega(\cdot)$ есть линейная функция, что позволяет сделать вывод, что оптимальное управление приходит в точку $\mathbf{x}^+$ по прямой (при условии существования функции ω).

Пусть теперь $F(\mathbf{u}) = B\mathbf{u}$, вектор состояния $\mathbf{x}(t)$ принадлежит в любой момент времени некоторому выпуклому множеству X, множество

допустимых управляющих воздействий U также является выпуклым. Собрав вместе все условия, получаем следующую задачу

$$(2.8)\qquad \begin{array}{ll} \text{минимизировать} & \tau, \\ \text{при условии} & \begin{cases} \dot{\mathbf{x}} = B\mathbf{u}, \\ \mathbf{x}(0) = \mathbf{x}^-;\ \mathbf{x}(\tau) = \mathbf{x}^+, \\ \mathbf{x}(t) \in X, \\ \mathbf{u}(t) \in U. \end{cases} \end{array}$$

Предположим теперь, что $(\tau^*, \mathbf{u}^*)$ – оптимальное решение (2.8). Рассмотрим

$$\bar{\mathbf{u}} \equiv \frac{1}{\tau^*} \int_0^{\tau^*} \mathbf{u}^*.$$

Так как $\forall t \in [0, \tau^*]:\ u_i^*(t) \in U$ и U выпукло получаем

$$\bar{\mathbf{u}}(t) \equiv \frac{1}{\tau^*} \int_0^{\tau^*} \mathbf{u}^* \in U.$$

В силу системы получаем, что

$$\mathbf{x}(\tau^*) = \mathbf{x}^- + \int_0^{\tau^*} B\bar{\mathbf{u}} = \mathbf{x}^- + \int_0^{\tau^*} B\mathbf{u}^* = \mathbf{x}^+.$$

Так как управляющее воздействие постоянно, порождаемая траектория имеет вид $\mathbf{x}(t) = \mathbf{x}^- + \frac{t}{\tau^*}(\mathbf{x}^+ - \mathbf{x}^-)$, в силу выпуклости ограничений $x_i(t) \in X$ и факта, что $\mathbf{x}^-$ и $\mathbf{x}^+$ – допустимые состояния, получаем, что $\bar{\mathbf{u}}$ – допустимое управление. Так как при этом оно переводит систему в состояние $\mathbf{x}^+$ за оптимальное время τ^*, то $\bar{\mathbf{u}}$ является оптимальным управлением. Таким образом выполнено следующее утверждение.

Лемма 2.1. *Если в задаче* (2.8) *пропускные способности не зависят от времени,* $c_e(t) \equiv \bar{c}_e$*, то существует оптимальное управление, не зависящее от времени, т. е. функция* $\mathbf{u}(t) \equiv \bar{\mathbf{u}}$*, являющаяся решением* (2.8).

Теперь, учитывая вышесказанное, можно избавиться от динамической составляющей (2.8): так как любая линейная траектория, переводящая систему из $\mathbf{x}^-$ в $\mathbf{x}^+$ допустима (в силу выпуклости X), то для нахождения минимального времени перехода достаточно решать следующую задачу оптимизации

$$(2.9)\qquad \begin{array}{ll} \text{минимизировать} & \tau, \\ \text{при условии} & \begin{cases} \tau B\mathbf{u} = \mathbf{x}^+ - \mathbf{x}^-, \\ \mathbf{u} \in U. \end{cases} \end{array}$$

Таким образом, для построения оптимального процесса в задаче (2.8) достаточно решить статическую оптимизационную задачу (2.9). Стоит обратить внимание, что при $\tau > 0$ можно поделить первое условие (2.9) на τ и сделать замену $\lambda = \frac{1}{\tau}$, получая при этом

$$\begin{array}{ll} \text{максимизировать} & \lambda, \\ \text{при условии} & \begin{cases} B\mathbf{u} = \lambda(\mathbf{x}^+ - \mathbf{x}^-), \\ \mathbf{u} \in U. \end{cases} \end{array}$$

Такая задача проще, чем (2.9), так как содержит только линейные неравенства. Учитывая выпуклость U, задача подпадает под класс задач выпуклой оптимизации, подробно изученной во многих работах (см. [14, 15]).

3 Динамические потоковые процессы

Специфика потоковых процессов была неформально описана во введении, посмотрим теперь на свойства потоковых процессов подробнее. Здесь и далее будем предполагать, что дан некоторый ориентированный граф $G = \langle V, E \rangle$, описывающий сеть и взаимодействие внутри этой сети. V – множество вершин, $E \in V \times V$ – множество дуг и пусть $|V| = n, |E| = m$. В этой сети циркулирует поток, для которого можно выбирать "интенсивность" (или скорость) движения по дуге. Если по дуге $(i, j) \in E$ за некоторый интервал времени прошел единичный поток, то это означает, что этот единичный поток сначала находился на вершине i, затем перебрался на вершину j. Далее считаем, что на каждой вершине содержится неограниченное хранилище, которое может хранить частицы потока, и обозначим через $x_i(t)$ – количество потока, находящееся на вершине i в момент времени t. В силу физических причин основное возникающее ограничение – пропускная способность дуг, ограничивающая сверху количество передаваемого потока по дуге за единицу времени. Обозначим за $\mathbf{c} = (c_1, \ldots, c_m)^T$ – вектор пропускных способностей.

В контексте динамических систем, $\mathbf{x}^-$ является начальным распределением потока по сети, $\mathbf{x}^+$ – желаемое распределение. Пусть B – матрица инцидентности:

$$B_{ie} = \begin{cases} 1, & e \text{ выходит из } i, \\ -1, & e \text{ входит в } i, \\ 0 & \text{в остальных случаях.} \end{cases}$$

Оптимальным потоковым процессом будем называть траекторию следующей задачи

$$\begin{array}{ll} \text{минимизировать} & \tau, \\ \text{при условии} & \begin{cases} \dot{\mathbf{x}} = B\mathbf{u}, \\ \mathbf{x}(0) = \mathbf{x}^-;\ \mathbf{x}(\tau) = \mathbf{x}^+, \\ \mathbf{x}(t) \geq 0, \\ 0 \leq u_e(t) \leq c_e. \end{cases} \end{array} \tag{3.10}$$

Как и в (2.8), переменными являются управляющее воздействие $\mathbf{u}(t)$ и переменные состояния $\mathbf{x}(t)$, при этом $\mathbf{x}$ однозначно задается через $\mathbf{u}$. Применяя рассуждения, описанные в предыдущей главе, получаем, что оптимальное управление можно искать среди постоянных функций. Более того, такое оптимальное статическое управление является решением

задачи

$$\begin{array}{ll} \text{минимизировать} & \tau, \\ \text{при условии} & \begin{cases} \tau B\mathbf{u} = \mathbf{x}^+ - \mathbf{x}^-, \\ 0 \le u_e \le c_e \end{cases} \end{array}$$

или

$$\begin{array}{ll} \text{максимизировать} & \lambda, \\ \text{при условии} & \begin{cases} B\mathbf{u} = \lambda(\mathbf{x}^+ - \mathbf{x}^-), \\ 0 \le u_e \le c_e. \end{cases} \end{array} \tag{3.11}$$

Последняя постановка является задачей линейного программирования, а значит к ней применимо большое количество методов, доступных в стандартных пакетах (Matlab, Mathematica и прочие). Как оказывается, эта задача сводится к довольно подробно изученной *задаче параметрического потока*. Стоит отметить, что уравнение $B\mathbf{u} = \mathbf{p}$ разрешимо только тогда, когда $\sum_{i=1}^{n} p_i = 0$.

Обозначим $d_i = x_i^+ - x_i^-$. Рассмотрим граф $G' = \langle V', E' \rangle$, где $V' = V \cup \{s, t\}$ – расширение V фиктивными вершинами *истока* s и *стока* t, $E = E \cup E_s \cup E_t$, $E_s = \{(s, i) | i \in V, d_i < 0\}$, $E_t = \{(i, t) | i \in V, d_i > 0\}$. Пусть пропускные способности c'_e на графе G' имеют вид

$$c'_e = \begin{cases} c_e, & e \in E, \\ -\lambda d_i, & e = (s, i) \in E_s, \\ \lambda d_i & e = (i, t) \in E_t. \end{cases}$$

При фиксированном λ существование допустимого вектора $\mathbf{u}$, соответствующего этому λ в (3.11) равносильно существованию s-t потока в графе G', насыщающего все дуги, инцидентные с s (или, что то же самое, насыщающего все дуги, инцидентные с t, этот факт подробно описан в [16]). Другими словами, допустимость того или иного значения λ в задаче (3.11) можно проверить, решив задачу о максимальном потоке. На рис. 3 изображен граф G'. На рис. 2 изображен эквивалентный ему граф с формулировкой (3.11) через τ, наглядно видно, что выполнение условия $\tau B\mathbf{u} = \mathbf{x}^+ - \mathbf{x}^-$ соответствует условию насыщения всех дуг вида (s, i) и (i, t). Минимальное значение τ, для которого возможен такой поток является решением (3.11).

С другой стороны, если $(\lambda, \mathbf{u})$ – допустимая точка (3.11), то $\forall \lambda' < \lambda$, $\left(\lambda', \frac{\lambda'}{\lambda}\mathbf{u}\right)$ – тоже допустимая точка (3.11). Следовательно, максимальное значение λ можно найти методом бисекции (дихотомия, деление отрезка пополам), для которого нужно подобрать начальный отрезок, который содержит оптимальное значение λ. Левую границу этого отрезка можно

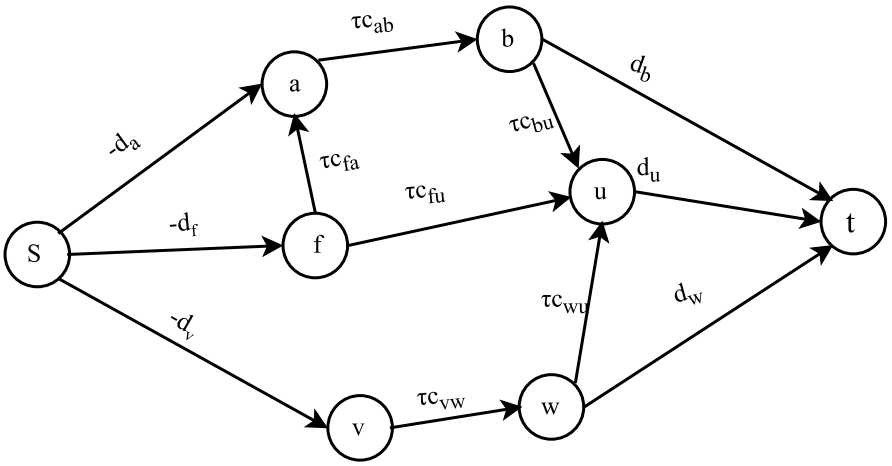

Рис. 2: Расширенный граф (формулировка через τ). Для наглядности дуги проиндексированы двумя буквами — началом и концом дуги.

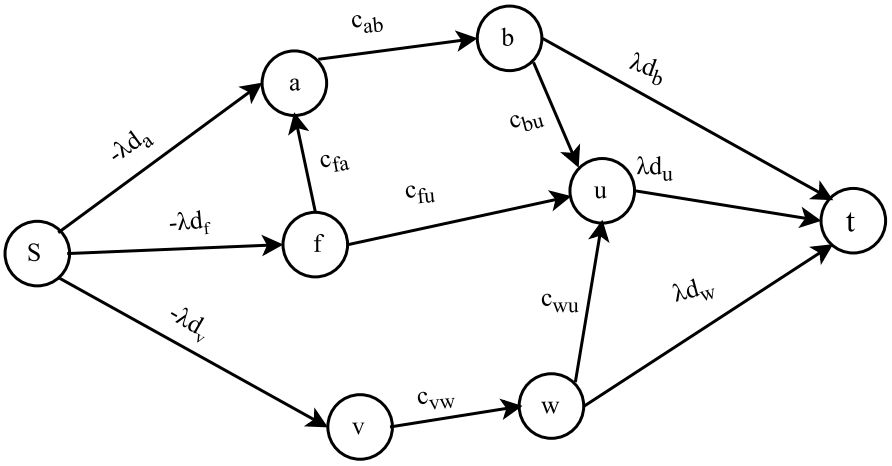

Рис. 3: Расширенный граф (формулировка через λ). Для наглядности дуги проиндексированы двумя буквами — началом и концом дуги.

взять равной нулю. Правую границу можно взять из неравенства

$$\lambda \sum_{d_i>0} d_i \leq \sum_{e \in E} c_e.$$

Таким образом, обозначив $r_0 = \frac{\sum_{e \in E} c_e}{\sum_{d_i>0} d_i}$ получаем, что для нахождения ϵ-точного решения задачи (3.11) достаточно решить не более чем

$$\log_2 \frac{r_0}{\epsilon}$$

задач максимального потока на графе G' (с различными значениями λ).

Как описано в [17], по ряду комбинаторных соображений целесообразна адаптация *метода проталкивания предпотока*, разработанного в [18], для решения этой задачи. Общий алгоритм, предложенный в [17], можно описать как применение подобной методу Ньютона процедуры вместо метода бисекции (подробнее в следующем разделе).

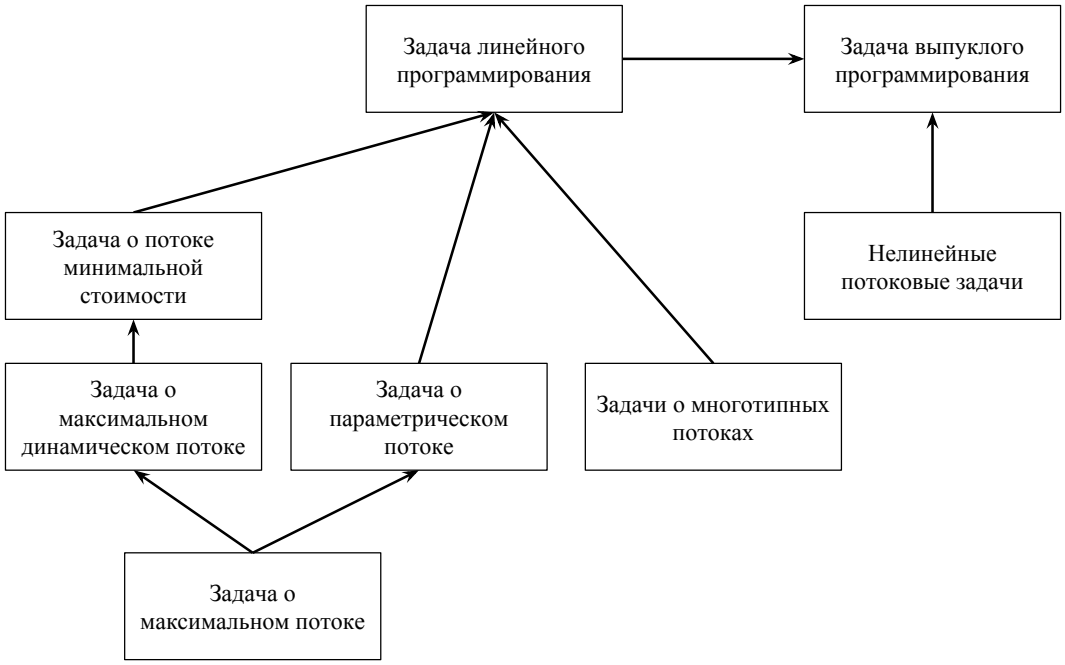

Рис. 4: Потоковые задачи в информатике.

4 Метод проталкивания предпотока для потоковых задач линейного программирования

4.1 Общая классификация потоковых задач

Потоковые задачи представляют собой целую отдельную область информатики. На рис. 4 показаны основные типы потоковых задач и зависимость между ними. Наиболее релевантными к этой статье являются задача о максимальном потоке, задача о параметрическом потоке и задача о максимальном динамическом потоке. Первые две играют активную роль в решении (2.8). В задаче о максимальном потоке предполагается, что при переходе от одной вершины на другую частица потока мгновенно проходит по дуге и оказывается на месте назначения. Тем не менее, количество частиц потока, проходящих по ребру за единицу времени, ограничено. В задаче о максимальном динамическом потоке предполагается, что частица потока совершает движение по ребру в течение некоторого промежутка времени. В дальнейшем анализе подобная величина также будет присутствовать в моделях, однако связь с задачей динамического

потока будет минимальна. Подробное описание задачи о максимальном динамическом потоке можно найти в [6, 7].

В задаче о параметрическом потоке рассматривается следующий вопрос: если пропускные способности дуг зависят от некоторого скалярного глобального параметра, как величина максимального потока зависит от значения этого параметра? Ключевой вопрос исследования этой задачи довольно прост: в каких случаях возможно решение такой задачи быстрее, чем непосредственное решение задачи о максимальном потоке для всех возможных значений параметра? В работе [17] предложен наиболее существенный вклад по этой задаче. Для обширного класса выработан метод нахождения решения задачи параметрического потока, который по времени работы эквивалентен решению одной задачи о максимальном потоке методом проталкивания предпотока. Задачи о многотипных потоках (*англ.*
multicommodity flows) являются расширением задачи о максимальном потоке на случай, когда в сети есть несколько типов потоков со своими собственными истоками и стоками, которые делят одни пропускные способности (см. [19, 20]). Такие ограничения являются типичными для транспортных потоков. К нелинейным потоковым задачам можно отнести задачи потоков с минимальной энтропией [21, 22], электрические потоки [23], а также некоторые примеры приведены в [4, 15].

4.2 Задача о максимальном потоке

Традиционно задача о максимальном потоке ставится в следующем виде:
(4.12)

$$\begin{array}{ll} \text{максимизировать} & val(\mathbf{f}) = \sum_{e \in in(t)} f_e - \sum_{e \in out(t)} f_e, \\ \text{при условии} & \left\{ \begin{array}{l} \sum_{e \in in(i)} f_e - \sum_{e \in out(i)} f_e = 0, \ i \in V, i \neq s, t, \\ 0 \leq f_e \leq c_e, \end{array} \right. \end{array}$$

где переменными являются величины потока по дугам $f_1, \ldots, f_m$ (здесь и далее $\mathbf{f} = (f_1, \ldots, f_m)^T$), $in(i)$ – множество входящих в i дуг, $out(i)$ – множество дуг, исходящих из i. Первое условие означает, что поток не задерживается в промежуточных вершинах. Единственные две вершины с неотрицательной разностью входящих / исходящих потоков – s и t. Более того,

$$\sum_{e \in in(t)} f_e - \sum_{e \in out(t)} f_e = - \left(\sum_{e \in in(s)} f_e - \sum_{e \in out(s)} f_e \right).$$

Впервые в таком виде задача была сформулирована в [24]. В этой же работе появляется фундаментальная теорема задачи максимального потока (т. Форда-Фалкерсона).

Рассмотрим некоторое разбиение множества V на два подмножества $S, V \setminus S$ таких, что $s \in S,\ t \in V \setminus S$. Такое разбиение принято называть $s-t$ *разрезом* или просто *разрезом*. Обозначим через $Cut(S)$ величину такого разреза

$$Cut(S) = \sum_{e=(i,j)\in E, i\in S, j\in V\setminus S} c_e. \tag{4.13}$$

Другими словами, $Cut(S)$ – сумма пропускных способностей дуг, начало которых лежит в S, а конец в $V \setminus S$. Легко увидеть, что для любого допустимого потока $\mathbf{f}$ и разреза S выполняется $val(\mathbf{f}) \leq Cut(S)$.

Далее, *остаточной сетью* потока $\mathbf{f}$ с пропускными способностями $\mathbf{c}$ называется множество

$$E_{\mathbf{cf}} = \{(i,j) | e = (i,j) \in E, \quad f_e < c_e\} \cup \{(j,i) | e = (i,j) \in E, f_e > 0\}.$$

Следующая теорема показывает, что введенные величины тесно связаны между собой.

Т е о р е м а 4.1 (Форда-Фалкерсона [24]). *Следующие три утверждения эквивалентны:*

1. *Поток* $\mathbf{f}$ *является максимальным.*
2. *Для некоторого разреза* S *выполняется* $\mathbf{f} = Cut(s)$.
3. *В остаточной сети* $E_{\mathbf{cf}}$ *не существует пути из* s *в* t.

Стоит отметить, что если $\mathbf{f}$ – максимальный поток, то соответствующий ему разрез можно взять как множество достижимых из s вершин по ребрам остаточной сети $E_{\mathbf{cf}}$. Для нахождения такого разреза достаточно выполнить следующую процедуру:

Function Minimum cut$(G, \mathbf{c}, \mathbf{f})$

```
S ← ∅;
Q ← {s};
D ← {s};
while Q ≠ ∅ do
    i ← any element of Q;
    Q ← Q \ {i};
    S ← S ∪ {i};
    for (i, j) ∈ E_cf do
        if j ∉ D then
            Q ← Q ∪ {j};
            D ← D ∪ {j};
return S;
```

С точки зрения современной теории математической оптимизации, теорема 4.1 является частным случаем двойственности в задачах линейного программирования. Эквивалентность 1-2 показывает, что если получены допустимые точки прямой и двойственной задачи, такие что значения функционалов прямой и двойственной задачи равны, то эти точки являются решениями прямой и двойственной задачи соответственно. Пункт 3 выводится из условий дополняющей нежесткости. Тем не менее, форма теоремы более проста нежели двойственность линейного программирования и, более того, дает комбинаторный критерий оптимальности решения.

Наиболее подробный обзор методов решения задачи максимального потока предоставлен в [25]. Метод проталкивания предпотока был впервые применен в [26] и позднее оформлен в [18] как самостоятельный метод. До возникновения метода проталкивания предпотока основным способом решения задачи максимального потока являлись различные алгоритмы последовательного нахождения *дополняющих путей* в остаточной сети. Из пункта 3 и определения остаточной сети можно вывести базовый алгоритм Форда-Фалкерсона: пока в остаточной сети есть путь из s в t, увеличить поток вдоль пути на некоторую положительную величину. Если такой алгоритм сходится, то результирующий поток является максимальным[1].

Перейдем к описанию *метода проталкивания предпотока* [18]. *Предпотоком* называется вектор $(f_1, \ldots, f_m)^T$, удовлетворяющий следующим

[1]По такому принципу устроены первые алгоритмы нахождения максимального потока. Более того, оказывается, что симплекс-метод в применении к этой задаче действует аналогично.

условиям

$$\begin{cases} excess(i, \mathbf{f}) = \sum_{e \in in(i)} f_e - \sum_{e \in out(i)} f_e \geq 0, \ i \in V, i \neq s, \\ 0 \leq f_e \leq c_e, \end{cases}$$

Остаточная сеть для предпотока определяется точно так же, как и для потока. Далее, рассмотрим вектор $\mathbf{h} = (h_1, \ldots, h_n)^T \in \mathbb{Z}_+^n$ (h_i принимает только целые неотрицательные значения). Будем говорить, что предпоток $\mathbf{f}$ допускает вектор $\mathbf{h}$, если

$$h_s = n, \ h_t = 0,$$

$$\forall (i, j) \in E_{\mathbf{cf}} : \ h_i \leq h_j + 1.$$

Общая идея метода предпотоков состоит в том, что если предпоток допускает $\mathbf{h}$, то в остаточной сети отсутствует дополняющий путь. Если при этом предпоток является потоком, то выполнена теорема Форда-Фалкерсона, и этот поток имеет максимальную величину. Все изменения в алгоритме происходят с помощью двух функций *push* и *relabel* (будут описаны далее). Алгоритм инициализируется следующей процедурой

Procedure Initialize$(G, \mathbf{c}, \mathbf{f}, \mathbf{h})$

```
for e ∈ E do
    if e = (s, i), i ∈ V then
        f_e ← c_e;
    else
        f_e ← 0;
for i ≠ s do
    h_i ← 0;
h_s ← n;
```

Заметим, что после применения такой процедуры, $\mathbf{f}$ допускает $\mathbf{h}$. Далее, операция *push* изменяет $\mathbf{f}$, а *relabel* изменяет $\mathbf{h}$.

Procedure push$(G, i, e, \mathbf{c}, \mathbf{f}, \mathbf{h})$

```
if h_i = h_j + 1 then //  e = (i, j)
    val ← min(excess(i, f), c_e − f_e);
    f_e ← f_e + val;
```

Procedure relabel$(G, i, \mathbf{c}, \mathbf{f}, \mathbf{h})$

$val \leftarrow 2n$;
for $(i, j) \in E_{\mathbf{cf}}$ **do**
  $val \leftarrow \min(val, h_j)$;
$h_i \leftarrow val + 1$;

Общая процедура проталкивания предпотока заключается в последовательном применении *push* и *relabel*, пока возможно изменение предпотока этими операциями. Отметим тот факт, что $relabel(u)$ никогда не уменьшает значение h_u. Более того, если к вершине i применима (в том смысле, что в результате применения изменится $\mathbf{f}$ или $\mathbf{h}$) операция $push(i, e)$ для некоторого $e \in E_{\mathbf{cf}} \cap (in(i) \cup out(i))$, то для этой же вершины i не применима операция $ralebel(i)$ и наоборот: если применима операция $relabel(i)$, то ни для какого e не применима операция $push(i, e)$.

Function Preflow push$(G,\ \mathbf{c})$

Initialize$(G, \mathbf{f}, \mathbf{h})$;
while $\exists i \in V, i \neq s, t:\ excess(i) > 0$ **do**
  Apply $push(G, i, e, \mathbf{c}, \mathbf{f}, \mathbf{h})$ or $relabel(G, i, \mathbf{c}, \mathbf{f}, \mathbf{h})$ whichever is applicable;
return $\mathbf{f}$;

В этой статье мы не будем повторять полный анализ этого алгоритма, тем не менее стоит выделить ключевые свойства. Далее считаем, что одна итерация алгоритма проталкивания предпотока заключается в одном применении операции *push* или *relabel*.

1. На любой итерации алгоритма предпоток $\mathbf{f}$ допускает $\mathbf{h}$.

2. На любой итерации алгоритма для любой вершины i с положительным избытком ($excess(i)>0$) существует путь до s в остаточной сети. Учитывая определение остаточной сети и тот факт, что $relabel(i)$ применяется только к вершинам с положительным избытком, получаем, что на любой итерации алгоритма для любой вершины i выполняется $h_i \leq 2n - 1$. Также, в частности, это показывает, что у вершины с положительным избытком существует исходящая дуга, принадлежащая остаточной сети, а значит, к ней применим *push* или *relabel*.

3. На любой итерации алгоритма в остаточной сети не существует пути из s в t.

4. Если операции применяются в произвольном порядке, то алгоритм закончит работу за $\mathcal{O}(n^2m)$ итераций. Если алгоритм завершается, то все вершины кроме стока и истока имеют нулевой избыток. Следовательно, после завершения алгоритма предпоток $\mathbf{f}$ является потоком. Из предыдущего свойства и т. Форда-Фалкерсона получаем, что $\mathbf{f}$ – максимальный поток.

Оказывается, что порядок применения операций сильно влияет на время работы алгоритма. На данный момент в [27] показана теоретическая оценка $\mathcal{O}(nm)$ на количество итераций алгоритма (и, что более важно, такая же оценка на количество арифметических действий – сложений и сравнений).

4.3 Задача о параметрическом потоке

Пусть теперь пропускные способности являются функциями некоторого глобального неотрицательного параметра, $c_e : [0, +\infty) \to [0, +\infty)$. В общем смысле *задача о параметрическом потоке* заключается в нахождении максимального потока для всех значений этого параметра. Более формально, дан граф G и набор пропускных способностей $c_1(\cdot), \ldots, c_m(\cdot)$ – функций одного скалярного параметра. Для каждого возможного λ нужно найти решение задачи о максимальном потоке на графе G с пропускными способностями $c_1(\lambda), \ldots, c_m(\lambda)$. Такая формулировка принята в теории, однако на практике обычно встречаются более конкретные задачи. Например, построить максимальный поток для конечного множества $\lambda_1, \ldots, \lambda_k$. Как оказывается, такую задачу можно решить быстрее, чем последовательным решением k экземпляров задач о максимальном потоке (по задаче на каждое значение λ_i). Далее будут описаны основные концепции, предложенные в [17] для построения эффективного метода задачи о параметрическом потоке.

Среди всех задач о параметрическом потоке выделяются два ключевых подкласса:

- Пропускные способности обладают следующими свойствами монотонности

$$\begin{cases} c_e(\lambda) \text{ не убывает} & \text{при } e = (s, i), i \neq t, \\ c_e(\lambda) \text{ не возрастает} & \text{при } e = (i, t), i \neq s, \\ c_e(\lambda) \text{ постоянна} & \text{в остальных случаях .} \end{cases} \tag{4.14}$$

- Пропускные способности линейны по λ.

Стоит отметить, что эти два класса пересекаются, и, более того, далее в работе будут приведены практические примеры задач, где оба свойства активно используются.

Рассмотрим задачу о параметрическом потоке, удовлетворяющей (4.14). Предположим, что для значения λ был построен максимальный поток методом предпотоков, и в результате были получены два вектора – $\mathbf{f}$ и $\mathbf{h}$, $\mathbf{f}$ допускает $\mathbf{h}$. Оказывается, чтобы посчитать максимальный поток для $\lambda' > \lambda$, можно использовать результат, полученный для λ. Рассмотрим

$$f'_e = \begin{cases} \max(c_e(\lambda'), f_e), & e = (s,i), h_i < n, \\ \min(c_e(\lambda'), f_e), & e = (i,t), \\ f_e, & \text{для остальных дуг.} \end{cases} \tag{4.15}$$

По сравнению с $\mathbf{f}$ поток на дугах, выходящих из истока, может только увеличиться, а поток на дугах, входящих в сток, может только уменьшиться. Следовательно, $\mathbf{f}'$ – предпоток. С другой стороны, в остаточной сети $E_{\mathbf{cf}'}$ по сравнению с $E_{\mathbf{cf}'}$ могли появиться только ребра вида (s,i) при $h_i \geq n$ и (i,s) при $h_i < n$, а значит $\mathbf{f}'$ допускает $\mathbf{h}$. Из этого следует, что для вычисления максимального потока для λ' можно использовать начальные значения потока $\mathbf{f}'$ и $\mathbf{h}$. В [17] более подробно показано, что при использовании метода проталкивания предпотока вычисление максимального потока последовательно для $\lambda_1 \leq \ldots \leq \lambda_k$ эквивалентно (в худшем случае) вычислению максимального потока для одного значения λ. Если нужно вычислить максимальный поток последовательно для $\lambda_1 \geq \ldots \geq \lambda_k$, то можно использовать аналогичную схему, если вычислять поток не на графе G, а на его "развернутом" варианте G^R (граф G^R получается из G изменением ориентации всех дуг на противоположное и обменом местами истока и стока). Если для G выполняется (4.14), то для G^R выполняется аналогичное свойство, но с изменением знака монотонности. Таким образом, найдя для некоторого значения λ максимальный поток $\mathbf{f}$ в графе G^R , который допускает $\mathbf{h}$, взяв $\lambda' < \lambda$ и соответствующий поток $\mathbf{f}'$ из (4.15), по вышесказанным соображениям получаем, что $\mathbf{f}'$ допускает $\mathbf{h}$.

Теперь посмотрим на случай линейных по λ пропускных способностей и, в частности, на задачу (3.11). Для анализа удобнее рассматривать величину $s-t$ разреза, нежели величину потока. Адаптируя определение (4.13) для параметрического случая получаем

$$Cut(S,\lambda) = \sum_{e=(i,j)\in E, i\in S, j\in V\setminus S} c_e(\lambda). \tag{4.16}$$

Из теоремы Форда-Фалкерсона следует, что значение максимального по-

тока равно значению минимального разреза. Если пропускные способности – линейные функции, то $Cut(S, \lambda)$ линейна по λ, $\min_S Cut(S, \lambda)$ является минимум линейных функций – кусочно-линейная выпуклая вверх функция. Обозначим
$Mincut(\lambda) = \min_S Cut(S, \lambda)$. *Точкой излома* будем называть такое значение λ, что для любого $\epsilon > 0$ существуют такие разрезы S_1, S_2, что

$$Mincut(\lambda) = Cut(S_1, \lambda) = Cut(S_2, \lambda),$$

но

$$Cut(S_1, \lambda + \epsilon) > Cut(S_2), \lambda + \epsilon)$$

и

$$Cut(S_1, \lambda - \epsilon) < Cut(S_2), \lambda - \epsilon).$$

Геометрически точка излома есть точка пересечения двух прямых, соответствующих величинам разрезов S_1 и S_2. Если для функций пропускных способностей выполняется (4.14) (в этом случае даже не нужна линейность), то таких точек будет не больше $n - 1$ (здесь n – общее кол-во вершин, включая сток и исток), подробности в [17].

Из соображений, описанных в разделе 3, оптимальное значение (3.11) соответствует минимальной точки излома $Mincut(\lambda)$ – для близких к нулю λ минимальный разрез соответствует множествам $\{s\}$ и $V \setminus \{t\}$. Максимальное значение λ, для которого эти разрезы будут оставаться минимальными, соответствует минимальной точке излома. Для бесконечно больших λ минимальный разрез соответствует множеству $\{s\} \cup \{i \in V | d_i > 0\}$. Более того, величина такого разреза не зависит от λ.

Несмотря на то, что для (4.16) не выполняется (4.14) (для дуг, входящих в сток, пропускные способности возрастают), если множество $\{i \in V | d_i > 0\}$ состоит из одного элемента, то можно не расширять граф фиктивным стоком, а использовать в качестве стока единственную вершину этого множества. В этом случае все входящие дуги будут иметь постоянную пропускную способность, и, следовательно, условия (4.14) будут выполняться. Аналогичный трюк возможен, если множество $\{i \in V | d_i < 0\}$ состоит только из одного элемента.

Наконец, опишем эффективный алгоритм нахождения минимальной точки излома. Пусть известны априори значения λ_l, λ_r такие, что искомая точка излома λ^* удовлетворяет $\lambda_l \leq \lambda^* \leq \lambda_r$ (для (4.16) такие значения даны в разделе 3). Пусть $S_l = \operatorname{argmin}_S Cut(S, \lambda_l)$, $S_r = \operatorname{argmin}_S Cut(S, \lambda_r)$. Найдем точку λ' из уравнения

$$Cut(S_l, \lambda') = Cut(S_r, \lambda').$$

Такое λ' обязательно найдется, так как слева и справа написаны линейные функции, из определения этих разрезов и выпуклости вниз $Mincut(\lambda)$ получаем, что точка λ' удовлетворяет $\lambda_l \leq \lambda' \leq \lambda_r$. Заметим, что на интервале $[\lambda', \lambda_r)$ содержится хотя бы одна точка излома, более того, если λ_l и λ_r лежат на соседних линейных отрезках $Mincut(\cdot)$, то λ' сама является точкой излома. Положим $\lambda_r := \lambda'$ и повторим процесс. Если повторять этот процесс до тех пор, пока не будет выполняться

$$Cut(S_l, \lambda) \equiv Cut(S_r, \lambda),$$

то в итоге мы получим минимальную точку излома. Более формально, этот алгоритм выглядит следующим образом. Предполагается, что пропускные способности имеют вид $c_e(\lambda) = a_e + b_e\lambda$, $\mathbf{a} = (a_1, \ldots, a_m)^T$, $\mathbf{b} = (b_1, \ldots, b_m)^T$.

Function Minimum breakpoint simple(G, $\mathbf{a}$, $\mathbf{b}$, λ_l, λ_r)

```
f ← Maximum flow(G, c(λ_l));
S_l ← Minimum cut (G, c(λ_l), f);
L_a ← 0;
L_b ← 0;
for e = (i, j), i ∈ S_l, j ∈ V \ S_l do
    L_a ← L_a + a_e;
    L_b ← L_b + b_e;
while true do
    f ← Maximum flow(G, c(λ_l));
    S_r ← Minimum cut (G, c(λ_r), f);
    R_a ← 0;
    R_b ← 0;
    for e = (i, j), i ∈ S_R, j ∈ V \ S_R do
        R_a ← R_a + a_e;
        R_b ← R_b + b_e;
    if L_a = R_a and L_b = R_b then
        return λ_r;
    else
        λ_r ← (L_a − R_a)/(L_b − R_b);
```

На рисунке 5 показана геометрическая интерпретация пересчета λ_r по функции $Mincut(\lambda)$ в описанном алгоритме.

В качестве функции *Maximum flow* годится любой алгоритм, вычисляющий максимальный поток. В частности, описанная ранее функция *Preflow push*. В отличии от метода бисекции, этот метод сходится к точ-

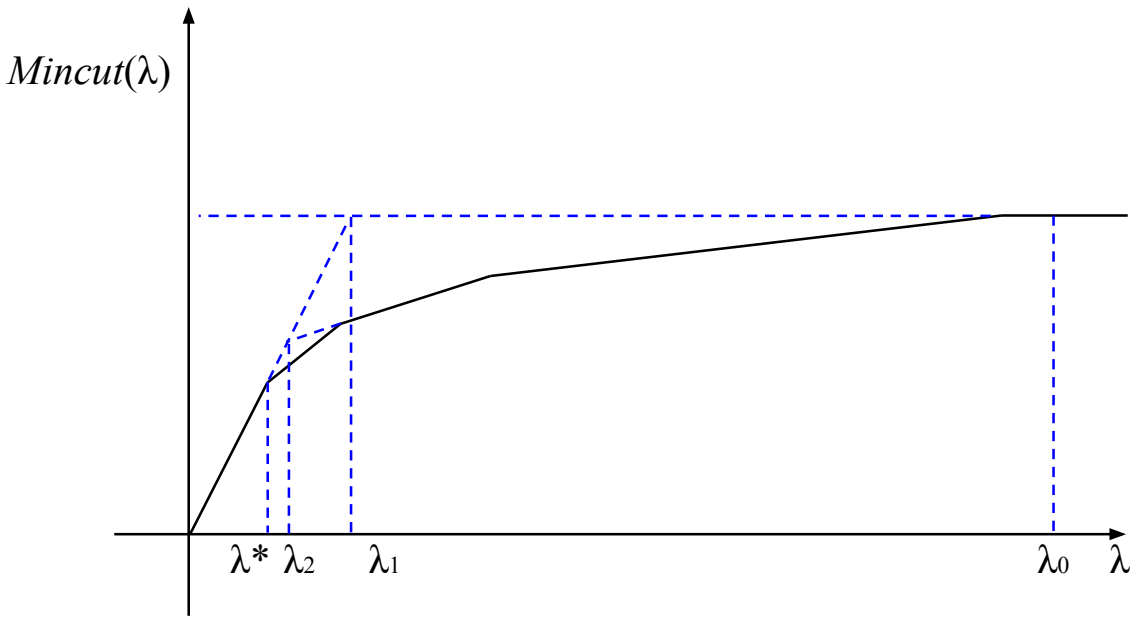

Рис. 5: Изменение λ в алгоритмах Minimum breakpoint simple и Minimum breakpoint GGT.

ному значению за конечное число итераций. На практике, даже для графов с несколькими сотнями тысяч ребер такой алгоритм делает не более 10 итераций. Последовательность $\{\lambda_{r_i}\}_{i=1}^{k}$, получаемая в этом алгоритме, строго убывает, а значит возможно применение амортизированной версии предпотока, если проводить вычисления в графе G_R. Модифицированная версия алгоритма имеет следующий вид.

Function Minimum breakpoint GGT(G, $\mathbf{a}$, $\mathbf{b}$, λ_l, λ_r)

$\mathbf{f} \leftarrow$ Maximum flow(G, $\mathbf{c}(\lambda_l)$);
$S_l \leftarrow$ Minimum cut $(G, \mathbf{c}(\lambda_l), \mathbf{f})$;
$L_a \leftarrow 0$;
$L_b \leftarrow 0$;
for $e = (i, j), i \in S_l, j \in V \setminus S_l$ **do**
    $L_a \leftarrow L_a + a_e$;
    $L_b \leftarrow L_b + b_e$;
Initialize($G^R, \mathbf{c}(\lambda_r), \mathbf{f}, \mathbf{h}$);
while *true* **do**
    while $\exists i \in V, i \neq s, t: \ excess(i) > 0$ **do**
        Apply $push(G^R, i, e, \mathbf{c}, \mathbf{f}, \mathbf{h})$ or $relabel(G^R, i, \mathbf{c}, \mathbf{f}, \mathbf{h})$ whichever is applicable;
    $S_r \leftarrow$ Minimum cut $(G, \mathbf{c}(\lambda_r), \mathbf{f})$;
    $R_a \leftarrow 0$;
    $R_b \leftarrow 0$;
    for $e = (i, j), i \in S_R, j \in V \setminus S_R$ **do**
        $R_a \leftarrow R_a + a_e$;
        $R_b \leftarrow R_b + b_e$;
    if $L_a = R_a \ and \ L_b = R_b$ **then**
        return λ_r;
    else
        $\lambda_r \leftarrow \frac{L_a - R_a}{L_b - R_b}$;
    for $e = (s, i) \in E, \ h_i < n$ **do**
        $f_e \leftarrow \max(c_e(\lambda_r), f_e)$;
    for $e = (i, v) \in E$ **do**
        $f_e \leftarrow \min(c_e(\lambda_r), f_e)$;

Рассмотренный алгоритм является относительно редким, в открытом доступе доступна только одна библиотека (paraF) с описанным методом [50] (описание от авторов доступно на [51]). Несколькими модификациями кода этой библиотеки удалось получить небольшое улучшение. Также, по ссылке [49] доступна собственная реализация этого метода. Три указанных алгоритма были протестированы на задачах, возникающих при балансировке загрузки (см. раздел 6), в условиях стандартных топологий сетей: 2-D решетка, звезда, ориентированное / неориентированное кольцо, ориентированный / неориентированный путь, ориентированное кольцо со случайными связями (исходящая степень каждой вер-

шины равна 3), корневое дерево. Во всех случаях граф состоял из 100000 вершин (100489 для 2-D решетки). Средние времена работы (в секундах) указаны в таблице. Из этой таблице видно, что собственная реализация имеет схожую производительность, модифицированная версия paraF во всех топологиях кроме звезды выигрывает у оригинала.

Реализация	2-D реш.	ор. путь	ор. кольцо	кольцо+2 дуги
paraF	0.732660	0.648990	0.648210	3.815160
мод. paraF	0.681800	0.629410	0.614800	3.498920
Своя реал.	1.749560	0.676780	0.637980	3.088300
	звезда	дерево	неор. путь	неор. кольцо
paraF	0.488700	0.673630	1.010540	0.983880
мод. paraF	0.503670	0.662090	0.959860	0.928250
Своя реал.	0.256040	0.202470	1.319850	1.214020

5 Потоковые процессы в условиях неопределенностей

В первой половине прошлого века появились первые модели, учитывающие неточность физических моделей. В отличии от моделей предшественников предлагалось решать задачу в условиях, когда часть параметров системы является неизвестной и неконтролируемой. Несмотря на то, что задачи с неизвестными параметрами решаются в математике повсеместно, полностью изучены только наиболее тривиальные из задач. Описание определенного класса неопределенностей, в условиях которых можно добиться определенного практического результата, являлось довольно свежей идеей и на текущий момент используется повсеместно. Так, например, в работе [28] впервые появляется метод Монте-Карло; В [29] (фильтр Винера-Колмогорова), [30] (фильтр Калмана-Бьюси) появляются методы предсказания поведения и оценки неизвестных параметров случайных процессов; в [31] впервые появляется понятие стохастической аппроксимации, развитое в многочисленных работах, включающих процедуру Кифера-Вольфовица [32], многоразмерную стохастическую аппроксимацию Блюма [33], усреднения Поляка [34], метод $SPSA$, предложенные Спаллом [35], рандомизированные методы стохастической аппроксимации Граничина [36–38]; в [39] предложено обобщение метода динамического программироваия для стохастических процессов; в [40] предложен сценарный подход для задач со стохастическими ограничениями.

Имея свою собственную специфику, с точки зрения стохастических процессов стандартные потоковые задачи стали изучаться относительно недавно (см. [9–11]). Эти работы, однако, изучают свойства максимального потока как функцию от значений пропускных способностей, значение максимального потока $\mathbf{f}^*$ в задаче (4.12) есть функция от параметров $c_1, \ldots, c_m$, где $c_i \in [0, +\infty)$. В такой постановке отсутствует анализ процесса циркуляции потока. С другой стороны, в работах [6, 7] изучаются процессы циркуляции потока в сети. Ключевым направлением исследования этих работ является нахождение оптимального потока в условиях положительного времени перехода по дуге. В этих работах формально рассматриваются функции потока $f_e \in L^1[0, T]$ для некоторого интервала $[0, T]$, однако пропускные не меняются во времени.

Далее в этом разделе будет описан основной вклад работы: алгоритм нахождения субоптимального решения задачи (3.10) при динамически изменяющихся пропускных способностях. Основная идея довольно проста: если пропускные способности изменяются во времени абсолютно ха-

отично, то построение какого-либо прогноза не имеет смысла. Однако же, если пропускные способности регулярны в некоторой степени и при этом известно поведение "в среднем", то удается адаптировать решение относительно простой "усредненной" задачи и обосновать его асимптотическую оптимальность.

5.1 Класс усредняемых функций

Все дальнейшие рассуждения и обоснования будут приводиться для следующего класса регулярных функций.

Определение 1. *Интегрируемая по Лебегу функция* $f \in L^1([0;+\infty))$ *называется усредняемой, если существует следующий предел*

$$avg(f) = \lim_{\tau \to +\infty} \frac{1}{\tau} \int_0^\tau f d\mu < +\infty.$$

В качестве примеров можно привести следующие виды функций.

- Любая периодическая функция усредняема.
- Если $f(t) \xrightarrow{t \to +\infty} \bar{f}$ почти всюду на $[0, +\infty)$, то f усредняема со средним значением $\bar{f}$.
- Если $\xi_1, \xi_2, \ldots$ – последовательность случайных величин, удовлетворяющая усиленному закону больших чисел, с мат. ожиданием $E\xi_i = \xi$, то функция $f(t) = \xi_{[t]}$ усредняема с вероятностью 1 со средней величиной ξ:

$$\Pr\left(\frac{1}{n}\int_0^n f\mu \to \xi\right) = \Pr\left(\frac{1}{n}\sum_{i=1}^n \xi_i \to \xi\right) = 1.$$

- Винеровский процесс усредняем с вероятностью 1.

В дальнейшем понадобятся две простые леммы об усредняемых функциях.

Лемма 5.1. *Если* f *– усредняемая неотрицательная функция, то для любых* $T > 0$ *и* $0 < \epsilon < 1$ *существует* $-\infty < \sigma \leq 0$ *такое, что*

$$\forall \tau \geq 0 \quad \int_T^{T+\tau} f d\mu \geq avg(f)(1-\epsilon)(\tau + \sigma).$$

Доказательство. Из определения $\frac{1}{\tau}\int_T^{T+\tau} f \to avg(f)$ для любого $\epsilon > 0$ существует τ' такое, что

$$\forall \tau > \tau' : \ \frac{1}{\tau}\int_T^{T+\tau} f d\mu > avg(f)(1-\epsilon)$$

таким образом, если $\sigma \leq 0$, то

$$\forall \tau > \tau' : \ \int_T^{T+\tau} f d\mu > avg(f)(1-\epsilon)(\tau+\sigma).$$

Следовательно, можно взять σ следующим образом

$$\sigma := \inf_{\tau \in [0,\tau']} \frac{1}{avg(f)(1-\epsilon)}\int_T^{T+\tau} f d\mu - \tau.$$

Так как при $\tau = 0$ минимизируемое выражение имеет значение 0, то $\sigma \leq 0$, что гарантирует соответствующее неравенство для $\tau > \tau'$.■

Л е м м а 5.2. *Если f – усредняемая неотрицательная функция, то для любых $T > 0$ и $\epsilon > 0$ существует $0 \leq \sigma < +\infty$ такое, что*

$$\forall \tau \geq 0 \quad \int_{T+\sigma}^{T+\sigma+\tau} f d\mu \leq avg(f)(1+\epsilon)(\tau+\sigma).$$

Доказательство. Из определения $\frac{1}{\tau}\int_T^{T+\tau} f \to avg(f)$ для любого $\epsilon > 0$ существует τ' такое, что

$$\forall \tau \geq \tau' : \ \int_T^{T+\tau} f d\mu < avg(f)(1+\epsilon)\tau.$$

Взяв $\sigma = \tau'$ получаем

$$\forall \tau \geq 0 \quad \int_{T+\sigma}^{T+\sigma+\tau} f d\mu \leq \int_T^{T+\sigma+\tau} f d\mu$$

$$\leq avg(f)(1+\epsilon)(\tau+\sigma).$$

■

5.2 Динамические потоковые процессы с меняющимися во времени пропускными способностями

Рассмотрим следующее обобщение задачи (3.10)

$$(5.17)\qquad \begin{array}{ll} \text{минимизировать} & \tau, \\ \text{при условии} & \left\{ \begin{array}{l} \dot{\mathbf{x}} = B(\sigma)\mathbf{u}, \\ \mathbf{x}(0) = \mathbf{x}^-;\ \mathbf{x}(\tau) = \mathbf{x}^+, \\ \mathbf{x}(t) \geq \mathbf{0}_n, \\ 0 \leq u_{ij}(t) \leq c_{ij}(t). \end{array} \right. \end{array}$$

где $c_e \in L^1([0;+\infty))$, $\sigma = (\sigma_1, \ldots, \sigma_m)^T$, σ_i – время перехода по дуге i, $B(\sigma)$ обозначает следующий оператор:

$$(5.18)\qquad [B(\sigma)\mathbf{u}]_i(t) = \sum_{e \in in(i)} u_e(t - \sigma_e) - \sum_{e \in out(i)} u_e(t).$$

Заметим, что в случае постоянных пропускных способностей и нулевых временах перехода задача вырождается в (3.10). Обозначим через $\tau^*(\mathbf{x}^-, \mathbf{x}^+, \sigma, \mathbf{c})$ оптимальное время (5.17) с входными параметрами $\mathbf{x}^-, \mathbf{x}^+, \sigma, \mathbf{c}$.

Лемма 5.3. *Рассмотрим* (3.10) *с параметрами* $\mathbf{x}^-, \mathbf{x}^+$ *и* $\bar{\mathbf{c}}$. *Существует статическое оптимальное управление* $\mathbf{u}(t) \equiv \bar{\mathbf{u}}$ *такое, что подграф* $G' = \langle V, E' \rangle$ *с множеством дуг* $E' = \{(i,j) \in E \mid \bar{u}_{ij} > 0\}$ *не содержит циклов.*

Доказательство. Существование оптимального статического управления показано в лемме 2.1. Докажем существование ациклического управления.

Пусть $\bar{\mathbf{u}}$ – такое оптимальное статическое управление, которое минимизирует квадратичную форму

$$\sum_{i,j=1}^{n} \bar{u}_{ij}^2.$$

Если такое управление содержит цикл, то за счет уменьшения u_e вдоль этого цикла можно уменьшить значение квадратичной формы. А значит управление, минимизирующее эту форму, не содержит циклов. ■

Т е о р е м а 5.1. *Рассмотрим* (5.17) *с фиксированными усредняемыми неотрицательными функциями пропускных способностей* $c_e(t)$ *и некоторым неотрицательным вектором* σ. *Для любого* $0 < \epsilon < 1$ *существует* $\overline{T(\epsilon)} > 0$ *такое, что для любых* $\mathbf{x}^-$, $\mathbf{x}^+$ *выполняется*

$$\tau^*(\mathbf{x}^-, \mathbf{x}^+, \sigma, \mathbf{c}) \le \left(\frac{1+\epsilon}{1-\epsilon}\right)^{n-1} \tau^*(\mathbf{x}^-, \mathbf{x}^+, \mathbf{0}_m, avg(\mathbf{c})) + \overline{T(\epsilon)}.$$

Доказательство. Зафиксируем $0 < \epsilon < 1$, $\mathbf{x}^-$ и $\mathbf{x}^+$. Пусть $\mathbf{u}^*$ – оптимальное статическое ациклическое управление, соответствующее $\tilde{\tau} = \tau^*(\mathbf{x}^-, \mathbf{x}^+, \mathbf{0}_m, avg(\mathbf{c}))$. Введем вспомогательные величины индуктивно по структуре управления (считаем, что максимум для пустого множество есть 0 в определении $T_i(\epsilon)$ и 1 в определении $\tau_i(\epsilon)$)

$$T_j(\epsilon) = \max_{u^*_{ij}>0} T_i(\epsilon) + \sigma^+_{ij}(\epsilon) + \sigma_{ij} - \sigma^-_{ij}(\epsilon),$$

$\sigma^+_{ij}(\epsilon)$ и $\sigma^-_{ij}(\epsilon)$ – величины, удовлетворяющие

$$\forall \tau \ge 0 \quad \int_{T_i(\epsilon)+\sigma^+_{ij}}^{T_i(\epsilon)+\sigma^+_{ij}+\tau} c_{ij} d\mu \le avg(c_{ij})(1+\epsilon)(\tau + \sigma^+_{ij}(\epsilon)),$$

$$\forall \tau \ge 0 \quad \int_{T_i(\epsilon)+\sigma^+_{ij}}^{T_i(\epsilon)+\sigma^+_{ij}+\tau} c_{ij} d\mu \ge avg(c_{ij})(1-\epsilon)(\tau + \sigma^-_{ij}(\epsilon))$$

и наконец

$$\tau_j(\epsilon) = \max_{u^*_{ij}>0} \frac{1+\epsilon}{1-\epsilon} \tau_i(\epsilon).$$

Для простоты записи величины, связанные с дугами, проиндексированы началом и концом дуги соответственно. Отметим, что эти определения осмысленны только если $\mathbf{u}^*$ не содержит циклов. Существование σ^+ и σ^- гарантируется леммами 5.1 и 5.2. С помощью этих величин можно построить субоптимальное управление следующим образом: $T_i(\epsilon)$ есть начальное время работы вершины i; lалее, каждая выходящая дуга (i,j) ждет еще $\sigma^+_{ij}(\epsilon)$ и начинает передавать поток, при этом можно гарантировать постоянный отток потока (с коэффициентом $avg(c_{ij})(1-\epsilon)$) по этой дуге с задержкой $\sigma^-_{ij}(\epsilon)$ на стороне j; i пересылает поток до тех пор, пока суммарный объем переданного потока не становится таким же, как и в усредненном случае. Более формально, управление описывается следующим образом

$$u_{ij}(t,\epsilon) = \begin{cases} 0, & t < T_i(\epsilon) + \sigma^+_{ij}(\epsilon), \\ 0, & \int_{T_i(\epsilon)+\sigma^+_{ij}(\epsilon)}^{t} u_{ij} d\mu = \tilde{\tau} u^*_{ij}, \\ \frac{1}{\tau_i(\epsilon)(1+\epsilon)} \frac{u^*_{ij}}{avg(c_{ij})} c_{ij}(t) & \text{в остальных случаях.} \end{cases} \tag{5.19}$$

Назовем интервал времени, соответствующий третьему правилу в этом определении, *активной стадией* дуги. Второе правило следует воспринимать следующим образом: если суммарное объем потока, который был передан по дуге, достиг уровня $\tilde{\tau}u^*_{ij}$, то следует остановиться и больше не передавать поток (с формальной точки зрения это возможно, так как функция $\int_T^\tau c_{ij}d\mu$ непрерывна по τ). Так как управление, соответствующее активной стадии, есть пропускная способность с коэффициентом меньшим единицы, то такое управление не нарушает ограничения пропускных способностей. Более того, активная стадия дуги заканчивается в силу усредняемости пропускных способностей. В момент времени, когда все дуги прошли свои активные стадии, система будет находится в состоянии $\mathbf{x}^+$, так как для рассмотренного управления конечное состояние такое же, как и для усредненной задачи.

Покажем, что момент времени, когда дуга (i,j) выходит из активной стадии, ограничен сверху значением

$$T_j(\epsilon) + \tilde{\tau}\tau_j(\epsilon) \tag{5.20}$$

и снизу значением

$$T_i(\epsilon) + \tilde{\tau}\tau_i(\epsilon). \tag{5.21}$$

Рассмотрим дугу (i,j). Используя нижнюю оценку для c_{ij} получаем

$$\int_{T_i(\epsilon)+\sigma^+_{ij}(\epsilon)}^{t} u_{ij}d\mu \geq avg(c_{ij})(1-\epsilon)\frac{1}{\tau_i(\epsilon)(1+\epsilon)}\frac{u^*_{ij}}{avg(c_{ij})}\times$$

$$\times(t - T_i(\epsilon) - \sigma^+_{ij}(\epsilon) + \sigma^-_{ij}(\epsilon)).$$

Приравнивая правую часть к $\tilde{\tau}u^*_{ij}$ получаем

$$\frac{(1-\epsilon)}{\tau_i(\epsilon)(1+\epsilon)}(t - T_i(\epsilon) - \sigma^+_{ij}(\epsilon) + \sigma^-_{ij}(\epsilon)) = \tilde{\tau}.$$

Решая по t получаем верхнюю границу на выход из активной стадии для дуги (i,j)

$$t = T_i(\epsilon) + \sigma^+_{ij}(\epsilon) - \sigma^-_{ij} + \tilde{\tau}\tau_i(\epsilon)\frac{1+\epsilon}{1-\epsilon}$$

что не превосходит (5.20) из-за положительности σ_{ij} и определений $T_i(\epsilon)$ и $\tau_i(\epsilon)$. Далее, используя верхнюю оценку c_{ij} получаем

$$\int_{T_i(\epsilon)+\sigma^+_{ij}(\epsilon)}^{t} u_{ij} \leq avg(c_{ij})(1+\epsilon)\frac{1}{\tau_i(\epsilon)(1+\epsilon)}\frac{u^*_{ij}}{avg(c_{ij})}\times$$

$$\times(t - T_i(\epsilon) - \sigma_{ij}^+(\epsilon) + \sigma_{ij}^+(\epsilon)).$$

Приравнивая правую часть к $\tilde{\tau}u_{ij}^*$ получаем

$$t = T_i(\epsilon) + \tilde{\tau}\tau_i(\epsilon).$$

Теперь проверим неотрицательность состояния. Из динамики системы имеем

$$x_i(t) = x_i(0) + \sum_{u_{ji}^*>0} \int_0^{t-\sigma_{ji}} u_{ji} d\mu - \sum_{u_{ij}^*>0} \int_0^t u_{ij} d\mu.$$

Посмотрим на некоторую вершину i. Из оценок (5.20) и (5.21) можно заключить, что активная стадия любой входящей дуги заканчивается раньше активной стадии любой исходящей дуги. Если в какой-то момент входящие дуги заканчивают активную стадию, но исходящие дуги все еще находятся в активной стадии, значит, что весь входящий поток уже был получен и, следовательно, после этого момента состояние может только уменьшиться. Таким образом, осталось проверить, будет ли состояние неотрицательным, пока все инцидентные дуги находятся в активной стадии. Используя нижнюю оценку для c_{ji}, для любой входящей дуги получаем

$$\int_0^{t-\sigma_{ji}} u_{ji} d\mu = \int_{T_j(\epsilon)+\sigma_{ji}^+(\epsilon)}^{t-\sigma_{ji}} u_{ji} d\mu \geq$$

$$avg(c_{ji})(1-\epsilon)\frac{1}{\tau_j(\epsilon)(1+\epsilon)}\frac{u_{ji}^*}{avg(c_{ji})}\times$$

$$\times(t - \sigma_{ji} - T_j(\epsilon) - \sigma_{ji}^+(\epsilon) + \sigma_{ji}^-(\epsilon)) \geq$$

$$\frac{1}{\tau_i(\epsilon)}u_{ji}^*(t - T_i(\epsilon)).$$

Последнее неравенство получено из определения $T_i(\epsilon)$ и $\tau_i(\epsilon)$. Для любой исходящей дуги имеем

$$\int_0^t u_{ij} = \int_{T_i(\epsilon)+\sigma_{ij}^+(\epsilon)}^t u_{ij} \leq$$

$$\frac{1}{\tau_i(\epsilon)(1+\epsilon)}\frac{u_{ij}^*}{avg(c_{ij})}avg(c_{ij})(1+\epsilon)\times$$

$$\times(t - T_i(\epsilon) - \sigma_{ij}^+(\epsilon) + \sigma_{ij}^+(\epsilon)) =$$

$$\frac{1}{\tau_i(\epsilon)}u_{ij}^*(t - T_i(\epsilon)).$$

Суммируя по всем инцидентным i дугам

$$x_i(t) \geq x_i(0) + \frac{1}{\tau_i(\epsilon)} \left(\sum_{u^*_{ji}>0} u^*_{ji} - \sum_{u^*_{ij}>0} u^*_{ij} \right) (t - T_i(\epsilon)) \geq *$$

если $\left(\sum_{u^*_{ji}>0} u^*_{ji} - \sum_{u^*_{ij}>0} u^*_{ij} \right)$ положительно, то очевидным образом $x_i(t) \geq 0$ выполняется для $t \geq T_i(\epsilon)$. Иначе, так как рассматриваемое время t ограничено сверху (5.20), получаем, что $t \leq T_i(\epsilon) + \tilde{\tau}\tau_i(\epsilon)$, в силу определений $\mathbf{u}^*$ и $\tilde{\tau}$

$$* \geq x_i(0) + \frac{1}{\tau_i(\epsilon)} \left(\sum_{u^*_{ji}>0} u^*_{ji} - \sum_{u^*_{ij}>0} u^*_{ij} \right) \tilde{\tau}\tau_i(\epsilon) = x_i^+ \geq 0.$$

Так как граф и функции пропускных способностей фиксированы, для данного ϵ существует конечное число конфигураций для $T_i(\epsilon)$ и $\tau_i(\epsilon)$ (так как число ациклических подграфов данного графа конечно). Таким образом, существует конечные верхние оценки на $T_i(\epsilon)$ и $\tau_i(\epsilon)$ вне зависимости от $\mathbf{u}^*$. Обозначим

$$\overline{T(\epsilon)} = \max_{acyclic\ G' \subset G} T_i(\epsilon).$$

Для $\tau_i(\epsilon)$ оценка может быть записана явным образом

$$\tau_i(\epsilon) \leq \left(\frac{1+\epsilon}{1-\epsilon} \right)^{n-1},$$

при этом равенство достигается на потоке, идущем вдоль одного пути из n вершин. Учитывая эти оценки получаем, что для любых $\mathbf{x}^-$ и $\mathbf{x}^+$ выполняется

$$\tau^*(\mathbf{x}^-, \mathbf{x}^+, \sigma, \mathbf{c}) \leq \left(\frac{1+\epsilon}{1-\epsilon} \right)^{n-1} \tau^*(\mathbf{x}^-, \mathbf{x}^+, \mathbf{0}_m, avg(\mathbf{c})) + \overline{T(\epsilon)}.$$

■

Теорема 5.2. *Рассмотрим* (5.17) *с фиксированными усредняемыми неотрицательными функциями пропускных способностей* $c_e(t)$, *некоторым неотрицательным вектором* σ *и последовательностями* $\{\mathbf{x}_k^-\}_{k=1}^{\infty}$, $\{\mathbf{x}_k^+\}_{k=1}^{\infty}$, *для которых выполняется*

$$\lim_{k\to\infty} \tau^*(\mathbf{x}_k^-, \mathbf{x}_k^+, \sigma, \mathbf{c}) = +\infty,$$

тогда

$$\lim_{k\to\infty}\frac{\tau^*(\mathbf{x}_k^-,\mathbf{x}_k^+,\sigma,\mathbf{c})}{\tau^*(\mathbf{x}_k^-,\mathbf{x}_k^+,\mathbf{0}_m,avg(\mathbf{c}))}=1.$$

Доказательство. "$\leq$": применяя теорему 2 для некоторого $0<\epsilon<1$, $\mathbf{x}_k^-$ и $\mathbf{x}_k^+$, при переходе к пределу получаем

$$\lim_{k\to\infty}\frac{\tau^*(\mathbf{x}_k^-,\mathbf{x}_k^+,\sigma,\mathbf{c})}{\tau^*(\mathbf{x}_k^-,\mathbf{x}_k^+,\mathbf{0}_m,avg(\mathbf{c}))}\leq\left(\frac{1+\epsilon}{1-\epsilon}\right)^{n-1}.$$

Устремив ϵ к нулю получаем желаемое неравенство.

"$\geq$": Пусть $\mathbf{u}^*(t)$ – оптимальное управление задачи с параметрами $\mathbf{x}_k^-,\mathbf{x}_k^+,\sigma,\mathbf{c}$. Обозначим

$$\alpha(\tau,\mathbf{c})=\max_{ij}\frac{\int_0^\tau c_{ij}d\mu}{avg(c_{ij})\tau}.$$

Для краткости положим $\tilde{\tau}=\tau^*(\mathbf{x}_k^-,\mathbf{x}_k^+,\sigma,\mathbf{c})$. Построим решение усредненной задачи следующим образом

$$\bar{u}_{ij}\equiv\frac{\int_0^{\tilde{\tau}}u_{ij}d\mu}{\alpha(\tilde{\tau},\mathbf{c}))\tilde{\tau}}.$$

Время, соответствующее такому управлению, равно $\alpha(\tilde{\tau},\mathbf{c})\tilde{\tau}$. Из определения вытекает

$$0\leq\bar{u}_{ij}\leq\frac{\int_0^{\tilde{\tau}}u_{ij}(t)dt}{\int_0^{\tilde{\tau}}c_{ij}(t)dt}avg(\mathbf{c})\leq avg(\mathbf{c}).$$

Так как на граничных моментах времени состояние неотрицательно, из линейности получаем неотрицательность состояния на всем интервале. Используя усредняемость c_{ij}, получаем

$$\lim_{k\to\infty}\alpha(\tau^*(\mathbf{x}_k^-,\mathbf{x}_k^+,\sigma,\mathbf{c}),\mathbf{c})=1,$$

что и доказывает нужное неравенство. ■

В теореме 2 раскрыта наиболее трудная часть доказательства и, более того, предложен способ построения соответствующего управления. Теорема 3 показывает асимптотическую оптимальность построенного управления.

5.3 Подбор параметров алгоритма в стохастическом случае

В этом подразделе будут подробно рассмотрены два стохастических случая.

Первый случай соответствует одному из примеров усредняемых функций. Пусть $\xi_0, \xi_1, \ldots$ – последовательность неотрицательных независимых случайных величин, $E\xi = M > 0$, $E(\xi - M)^2 = D^2$,

$$c_e(t) = \xi_{[t]}. \tag{5.22}$$

Рассмотрим одну конкретную дугу и оценим σ из лемм 5.1 и 5.2 для такой функции $c_e(t)$. Какова вероятность того, что для $0 < \epsilon < 1$ величина $\sigma > 0$ удовлетворяет леммам 5.1 и 5.2 ($-\sigma$ для леммы 2 и σ для леммы 3)? Неравенства из этих лемм имеют вид

$$\forall k \geq 0 : \ \forall k : (1-\epsilon)M(k-\sigma) < \sum_{i=1}^{k} \xi_i < (1+\epsilon)M(k+\sigma).$$

Немного увеличив правую часть (на $2\epsilon M\sigma$) получаем

$$\forall k \geq 0 : \ \left| \sum_{i=1}^{k} (\xi_i - M) \right| < M(\epsilon k + (1-\epsilon)T).$$

Основная сложность здесь состоит в том, что необходимо выполнение неравенства для всех k, в то время как стандартные неравенства для оценки сумм независимых случайных величин обычно оценивают только все сумму для некоторого определенного k. Тем не менее, в нашем случае применимо неравенство Хайека-Реньи, а именно

Теорема 5.3 (Неравенство Хайека-Реньи). *Пусть $\{\xi_i\}_{i=1}^{\infty}$ – последовательность независимых случайных величин, $E\xi_i = 0$, $E\xi_i^2 < \infty$, $S_n = \sum_{i=1}^{n} \xi_i$, тогда для любой последовательности $\{\alpha_i\}_{i=1}^{\infty}$, $0 < \alpha_i \leq \alpha_{i+1}$*

$$\Pr\left(\max_{i \leq n} \frac{|S_i|}{\alpha_i} \geq 1 \right) \leq \sum_{i=1}^{n} \frac{E\xi_i^2}{\alpha_i^2}.$$

Возьмем $\alpha_i = M(\epsilon k + (1-\epsilon)T)$

$$\Pr\left(\forall k \leq n : \ \left| \sum_{i=1}^{k} (\xi_i - M) \right| < M(\epsilon k + (1-\epsilon)T) \right) =$$

$$1 - \Pr\left(\exists k \le n : \left|\sum_{i=1}^{k}(\xi_i - M)\right| \ge M(\epsilon k + (1-\epsilon)T)\right) \ge$$

$$1 - \sum_{i=1}^{n} \frac{D^2}{M^2(\epsilon i + (1-\epsilon)T)^2} \ge$$

$$1 - \frac{D^2}{M^2\epsilon(1-\epsilon)T}.$$

Последнее неравенство получено в силу $\sum_{i=1}^{n} \frac{1}{(i+a)^2} < \frac{1}{a}$. Таким образом, чтобы для функции вида (5.22) выполнялись лемма 5.1 и 5.2 с параметром ϵ и вероятностью p следует выбрать σ из равенства

$$\sigma(\epsilon, p) = \frac{D^2}{M^2\epsilon(1-\epsilon)(1-p)}. \tag{5.23}$$

Второй пример менее тривиальный:

- Предположим, что поток поделен на куски величиной $\xi_1, \xi_2, \ldots$ – неотрицательные независимые одинаково распределенные случайные величины, при этом до непосредственной передачи такого куска по дуге *мы не знаем реализацию* ξ_i (но знаем $E\xi$).

- Пусть пропускная способность дуги постоянна и равна $\bar{c}$. Таким образом, чтобы передать поток величины ξ_i необходимо время $\frac{\xi_i}{\bar{c}}$.

- Вместо того, чтобы считать неоднородным поток, можно считать неоднородной пропускную способность: положим, что по дуге по очереди передаются $\xi_1, \xi_2, \ldots$, обозначим $\tau_0 = 0$, $\tau_i = \tau_{i-1} + \frac{\xi_i}{\bar{c}}$, тогда пропускная способность будет иметь вид

$$c(\tau) = \frac{\bar{c}}{\xi_i},\ \tau \in [\tau_{i-1}, \tau_i].$$

Аналогично предыдущему случаю, оценим вероятность того, что леммы 5.1 и 5.2 не выполняются для ϵ и σ. Предположим, что $avg(c) = \frac{\bar{c}}{M}$.

$$\Pr\left(\forall l :\ \frac{\bar{c}}{M}(1-\epsilon)(\tau_l - \sigma) < \int_0^{\tau_l} c d\mu < \frac{\bar{c}}{M}(1+\epsilon)(\tau_l + \sigma)\right) =$$

$$\Pr\left(\forall l :\ \frac{\bar{c}}{M}(1-\epsilon)(\tau_l - \sigma) < l < \frac{\bar{c}}{M}(1+\epsilon)(\tau_l + \sigma)\right).$$

Посмотрим отдельно на левое неравенство

$$\sum_{i=1}^{l} \xi_i < \frac{Ml}{(1-\epsilon)} + \sigma\bar{c}.$$

Правое неравенство имеет аналогичный вид

$$\sum_{i=1}^{l} \xi_i > \frac{Ml}{(1+\epsilon)} - \sigma\bar{c}.$$

При $0 < \epsilon < 1$ выполняются неравенства $\frac{1}{1-\epsilon} \geq 1+\epsilon$ и $\frac{1}{1-\epsilon} \geq 1+\epsilon$. Используя их получаем

$$\Pr\left(\forall l: \ \frac{\bar{c}}{M}(1-\epsilon)(\tau_l - \sigma) < \int_0^{\tau_l} c d\mu < \frac{\bar{c}}{M}(1+\epsilon)(\tau_l + \sigma)\right) \geq$$

$$\Pr\left(\forall l: \ \left|\sum_{i=1}^{l}(\xi_i - M)\right| < lM\epsilon + \sigma\right).$$

И снова, эту вероятность можно оценить с помощью неравенства Хайека-Реньи.

$$\Pr\left(\exists l: \ \left|\sum_{i=1}^{l}(\xi_i - M)\right| \geq lM\epsilon + \sigma\right) \leq$$

$$D^2 \sum_{i=1}^{k} \frac{1}{(iM\epsilon + \sigma)^2} <$$

$$\frac{D^2}{M\epsilon\sigma}.$$

В последнем переходе было использовано неравенство $\sum_{i=1}^{k} \frac{1}{\alpha i + \beta} < \frac{1}{\alpha\beta}$ при $\alpha, \beta > 0$. Таким образом, чтобы леммы 5.1 и 5.2 выполнялись для ϵ с вероятностью p достаточно взять σ из равенства

$$\sigma(\epsilon, p) = \frac{D^2}{M\epsilon p} \tag{5.24}$$

6 Практические примеры

6.1 Задача балансирования нагрузки в вычислительной сети

Рассмотрим следующую общую задачу балансирования нагрузки в вычислительной сети

- Сеть из нескольких вычислительных узлов.
- Некоторые пары узлов соединены однонаправленной или двунаправленной связью.
- На каждом узле есть очередь задач на исполнение.
- Для каждого задания узел должен решить: выполнить ли задание самому или отправить соседу.
- Как организовать принятие решений таким образом, чтобы минимизировать время выполнения последнего задания?

Задачи такого рода являются основополагающими для современных вычислительных систем. В работах [46–48] описана общая специфика таких задач. Так как в таких системах предполагается передача данных, то можно с уверенностью сказать, что в таких задачах присутствует потоковая специфика. Для систем с общей памятью идеи, описанные в этой статье, не являются применимыми, так как в таких системах не происходит передачи данных, которая существенно ограничивалась бы пропускной способностью межпроцессорного взаимодействия. Однако, для физически разделенных вычислительных устройств время, затрачиваемое на передачу данных (контекст задач), может быть соизмеримо времени, затрачиваемому на обработку задач. Посмотрим теперь на вариант формальной формулировки задачи.

Пусть сеть состоит из n узлов. На практике размер контекста задачи обычно известен заранее, но вот сложность задачи (кол-во действий, которое задача требует для ее выполнения) может быть неизвестно. Таким образом, мы полагаем, что сложность задачи является случайной величиной, для которой мы знаем среднее значение. Пусть p_i – производительность узла i, количество задач, которое узел i может выполнить в единицу времени, q_i – начальная загрузка узла i, количество задач. Для каждого канала связи (i, j) : c_{ij} – количество задач, которое может быть

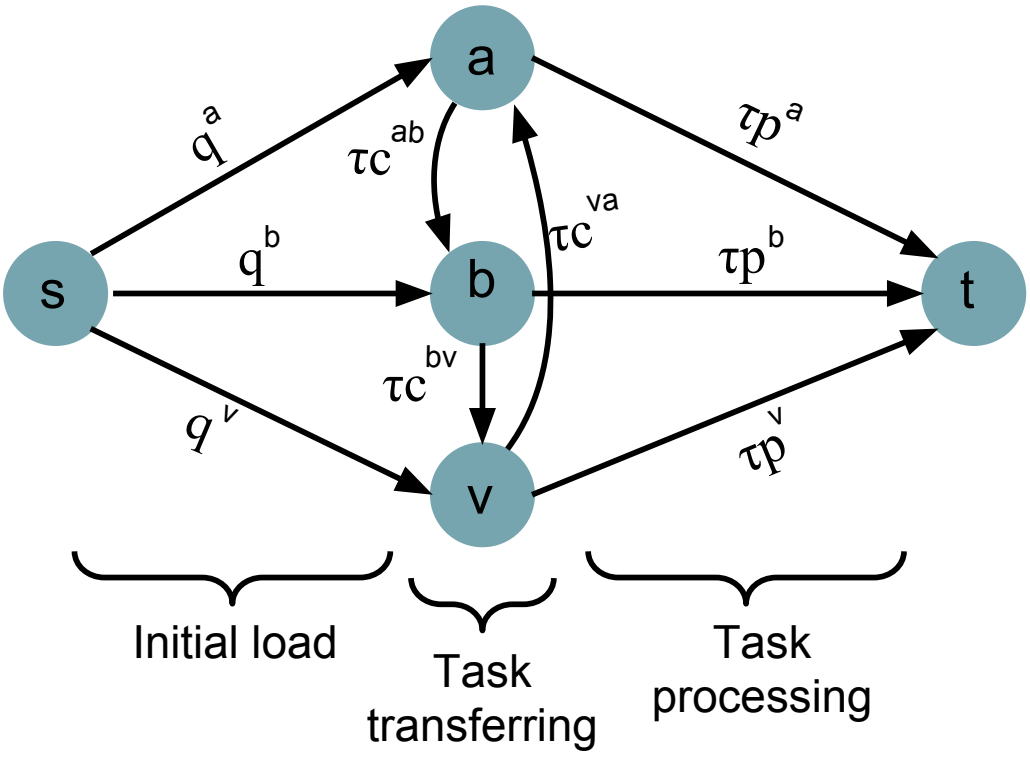

Рис. 6: Граф для задачи балансирования нагрузки

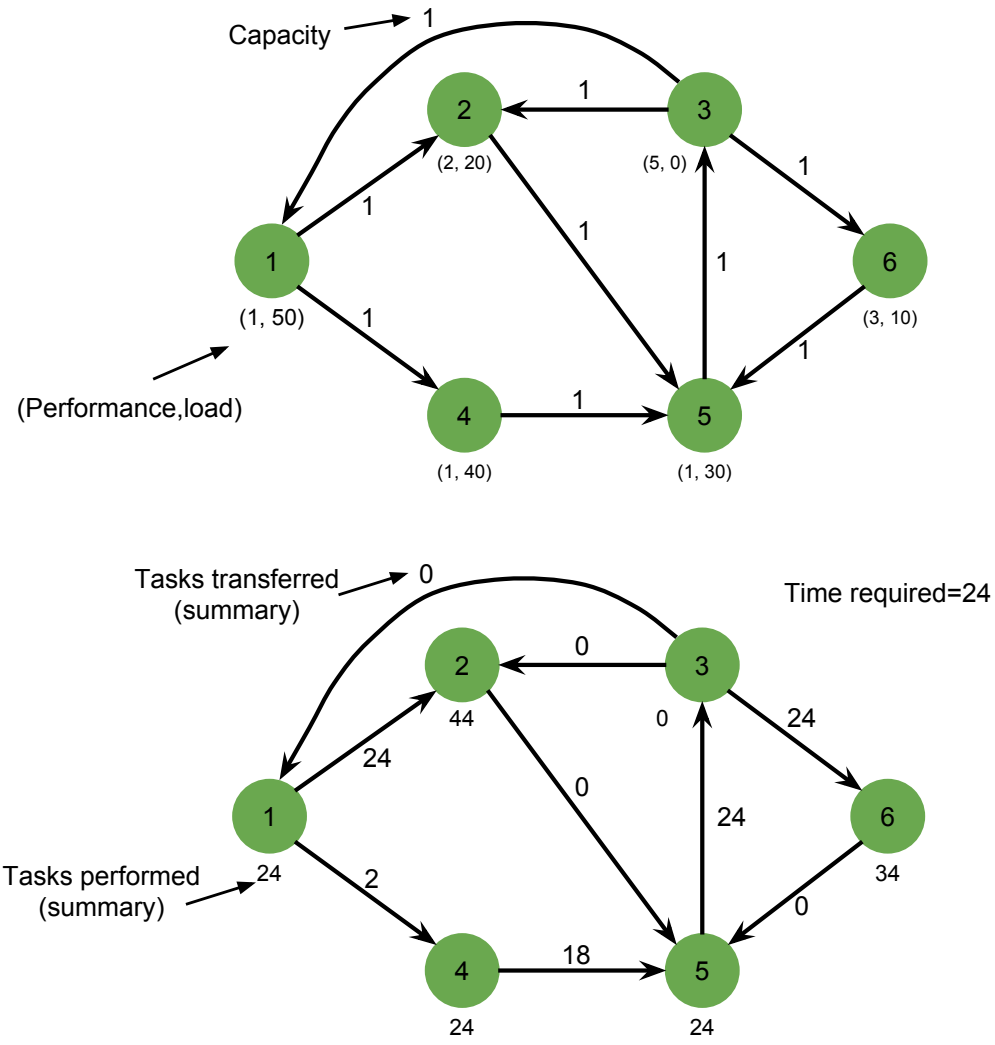

Рис. 7: Пример балансирования нагрузки

передано по каналу связи за единицу времени. Отметим, что q_i можно измерять не в задачах, а в размере контекста.

Пусть V – множество всех узлов, $E \in V \times V$ – множество каналов связи между узлами, пропускные способности которых есть c_{ij}. Дополним этот граф фиктивной вершиной t и дугами (v, t), $v \in V$ с пропускной способностью p_i. Этот трюк позволяет трактовать процесс выполнения задачи как передачу этого задания на узел t. Полагая, что $V = \{1, \ldots, n\}$, $t = n+1$ получаем, что наша задача заключается в передаче потока с узлов $v \in V$ на узел t, что является задачей (3.10) с начальным и конечным состояниями

$$\mathbf{x}^- = (q_1, \ldots, q_n, 0)^T,$$

$$\mathbf{x}^+ = (0, \ldots, 0, \sum_{i=1}^{n} q_i)^T.$$

При расширении граф фиктивным истоком (на подобии процедуры, описанной в начале раздела 3) получается изображенный на рис. 6 граф.

Для моделирования поведения такой системы по ссылке [49] доступен симулятор. Один из примеров, доступных на этом симуляторе, изображен на рис. 7.

6.2 Обмен информацией групп БПЛА

Беспилотные летательные аппараты (БПЛА) набрали большую популярность и уже сейчас повсеместно используются на практике. В качестве примера, где возникает необходимость решения потоковых задач, можно привести задачу мониторинга области: группе БПЛА требуется периодически пролетать над некоторой областью и делать снимки / видеозапись (как пример, применение БПЛА для мониторинга заповедников, [41]). Технологии управления групп БПЛА в таких условиях изучены в [42–44]. Одна из служебных задач, которая возникает в таких приложениях, – сбор информации на одном БПЛА. Например, такая необходимость может возникнуть, если нужно регулярно собирать снимки территории для внешней обработки. Для этого, чтобы не отвлекать всю группу БПЛА от мониторинга, все данные могут быть переданы на один БПЛА, после чего он летит на базовую станцию для передачи данных. На практике в больших многофункциональных системах задачи маршрутизации пакетов информации приводят к сложным задачам оптимизации, таким как задача о многотипных потоках [45]. В некооперативных системах (т.е. системах, в которых различные компоненты системы имеют конфликтующие задачи и при этом приоритет задач разных

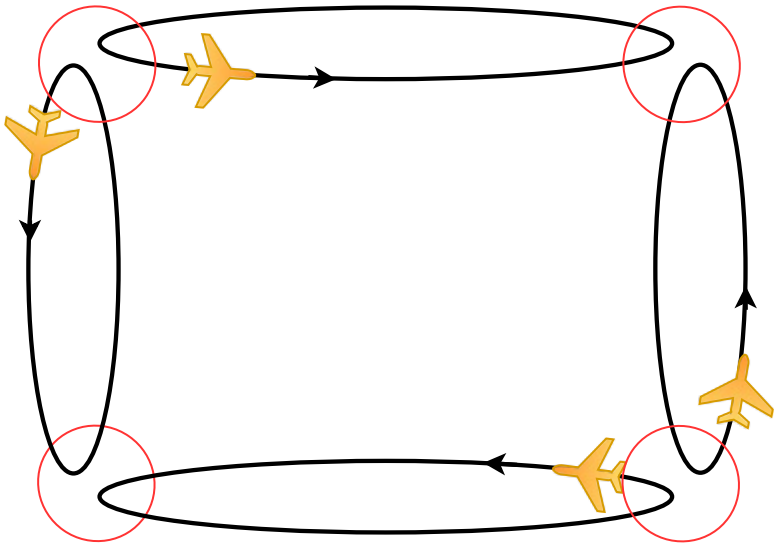

Рис. 8: Пример траекторий БПЛА и зон взаимодействия.

компонент одинаковый) принято моделирование с использованием теории игр, что обычно приводит к нелинейным задачам оптимизации, как например в [4]. Тем не менее, в такой частной задаче как сбор информации на одном БПЛА, в которой цель одна для всех БПЛА, применимы методы, описанные ранее в этой работе, что позволяет находить оптимальную маршрутизацию намного быстрее.

Предположим, что в целях мониторинга местности, каждый БПЛА летает по своей предопределенной траектории. Обозначим за $\varphi_i(t)$ – траекторию полета i-ого БПЛА. Далее полагаем, что для коммуникации используется беспроводная передача данных, функция $K : \mathbb{R}_+ \to \mathbb{R}_+$ определяет зависимость пропускной способности от расстояния между двумя БПЛА. Таким образом, функция пропускной способности между БПЛА i и j есть

$$c_{ij}(t) = K(||\varphi_i(t) - \varphi_j(t)||_2).$$

На практике БПЛА далеко не всегда могут передавать информацию друг другу. Можно даже сказать, что большинство времени они находятся на большом расстоянии друг от друга, а вся коммуникация происходит в небольшие промежутки времени, когда они сближаются, как показано на рис. 8. Таким образом, пропускная способность будет иметь вид как на рис. 9.

Таким образом, поведение $c_{ij}(t)$ полностью зависит от поведения $\varphi_i(t)$ и $\varphi_j(t)$. Значит, достаточно, чтобы $\varphi_i(t)$ были периодическим функциями, для того, чтобы функция $c_{ij}(t)$ была усредняемой. Стоит отметить, что на практике обычно на траекторию полета влияют некоторые внеш-

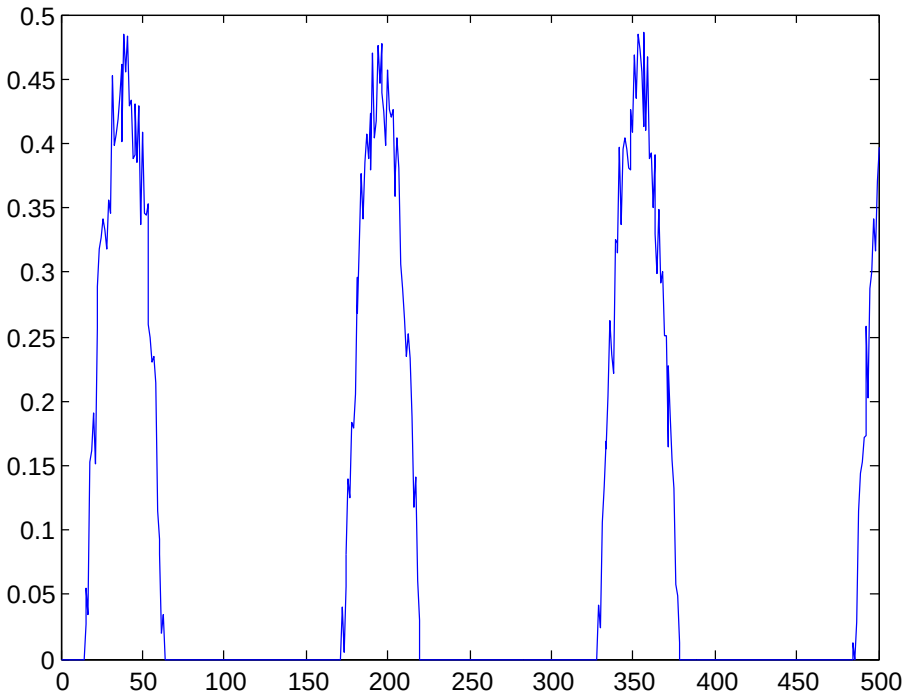

Рис. 9: Общий вид функции пропускных способностей. Основную часть времени пропускная способность равна нулю. Если траектории полета периодичны, то и пропускная способность периодична.

ние неконтролируемые воздействия (в основном, погодные условия). Если считать эти факторы статистическим шумом, то $c_{ij}(t)$ все еще является усредняемой и подпадает под первый статистический случай, разобранный в предыдущем разделе.

7 Заключение

В работе был рассмотрен класс динамических потоковых задач, подробно проанализированы свойства этого класса задач в условиях неопределенностей, приведены практические примеры, в которых возникает такой класс потоковых задач. Подытоживая ключевые аспекты работы можно выделить следующие выводы:

- Теоретические результаты, изложенные в разделе 5, показывают, что традиционные методы решения потоковых задач могут быть адаптированы для практических задач с неопределенностями. Основной результат раздела (теорема 5.2) показывает, такая адаптация работает асимптотически оптимально. В практическом смысле это означает, что если есть система циркуляции потока с указанным классом неопределенностей, которая работает неограниченное время, то адаптированное решение создаст некоторую задержку в работе системы по сравнению с оптимальным временем работы, но не больше, чем на величину, которая зависит от функций пропускных способностей и размера системы, но не от времени работы системы.

- Рассмотренные алгоритмы решения задач параметрического потока довольно эффективны и позволяют проводить вычисления для сетей с 10^5 узлами в течение пары секунд, собственная реализация этих методов соизмерима по производительности с "эталонной" реализацией [50].

- Чем меньше степень "регулярности" пропускных способностей, тем меньше оправдано применение описанных методов, особенно для маленьких сетей. Как пример, в задаче обмена информацией БПЛА можно построить более точное решение, если проводить дискретизацию времени и решать несколько потоковых задач (или же одну потоковую задачу, но большего размера) на отдельных участках времени.

Литература

[1] Teodorovic D., Vukadinovic K. Traffic Control and Transport Planning:: A Fuzzy Sets and Neural Networks Approach. – Springer Science & Business Media. 2012.

[2] Handbook of Transportation Science / Под ред. Hall R. – Springer Science & Business Media. 2012.

[3] Cascetta E. Transportation Systems Engineering: Theory and Methods, Springer Science & Business Media. 2013.

[4] Введение в математическое моделирование транспортных потоков / Под ред. Гасникова А. В. – Московский центр непрерывного математического образования. 2015.

[5] Ford L. R. Jr, Fulkerson D. R. Constructing maximal dynamic flows from static flows // Operations research. 1958. Vol. 6. No. 3. PP. 419–433.

[6] Skutella M. An introduction to network flows over time // In: Research Trends in Combinatorial Optimization. 1958. Springer. PP. 451–482.

[7] Köhler E., Möhring R. H., Skutella M. Traffic networks and flows over time // In: Algorithmics of Large and Complex Networks. 2009. Springer. PP. 166–196.

[8] Ford L., Fulkerson D. R. Flows in networks. – Princeton Princeton University Press. 1962.

[9] Glockner G. D., Nemhauser G. L. A dynamic network flow problem with uncertain arc capacities: formulation and problem structure // Operations Research. 2000. Vol. 48. No. 2. PP. 233–242.

[10] Han S., Peng Z., Wang S. The maximum flow problem of uncertain network // Information Sciences. 2014. Vol. 265. PP. 167–175.

[11] Sheng Y., Gao J. Chance distribution of the maximum flow of uncertain random network // Journal of Uncertainty Analysis and Applications. 2014. Vol. 2. No. 1. PP. 1–14.

[12] Bellman R. E., Dreyfus S. E. Applied dynamic programming. – 1962.

[13] Понтрягин Л. С. Математическая теория оптимальных процессов. – Гос. изд-во Физико-математической лит-ры. 1961.

[14] Nesterov Y., Nemirovskii A., Ye Y. Interior-point polynomial algorithms in convex programming // SIAM. Vol. 13. 1994.

[15] Boyd S., Vandenberghe L. Convex optimization. – Cambridge university press. 2009.

[16] Мальковский Н. В. Модель балансировки загрузки в вычислительной сети с использованием задачи параметрического потока // Стохастическая оптимизация в информатике. 2014. Т. 10. № 1. С. 39–62.

[17] Gallo G., Grigoriadis M. D., Tarjan R. E. A fast parametric maximum flow algorithm and applications // SIAM Journal on Computing. 1989. Vol. 18. No. 1. PP. 30–55.

[18] Goldberg A. V., Tarjan R. E. A new approach to the maximum-flow problem // Journal of the ACM (JACM). 1988. Vol. 35. No. 4. PP. 921–940.

[19] Garg N., Koenemann J. Faster and simpler algorithms for multicommodity flow and other fractional packing problems // SIAM Journal on Computing. 2007. Vol. 37. No. 2. PP. 630–652.

[20] Herty M., Kirchner C., Moutari S., Rascle M. et al. Multicommodity flows on road networks // Communications in Mathematical Sciences. 2008. Vol. 6, No. 1, PP. 171–187.

[21] Батюков А. М. Анализ цифровых изображений, основанный на построении стационарного потока на графе // Вестник Санкт-Петербургского университета. Серия 10. Прикладная математика. Информатика. Процессы управления. 2015. № 2. С. 115–122.

[22] Ампилова Н. Б., Романовский И. В., Петренко Е. И. О максимизации энтропии при линейных ограничениях // Труды Междунар.

науч. конференции «Космос, астрономия и программирование (Лавровские чтения)». СПб.: Математико-механический факультет С.-Петерб. ун-та. 2008. С. 181–185.

[23] Madry A. Navigating central path with electrical flows: From flows to matchings, and back // Foundations of Computer Science (FOCS). 2013. IEEE 54th Annual Symposium on, 2013. pp. 253–262.

[24] Ford L. R., Fulkerson D. R. Maximal flow through a network // Canadian journal of Mathematics, 1956, vol. 8, no. 3, pp. 399–404.

[25] Goldberg A. V., Tarjan R. E. Efficient maximum flow algorithms // Communications of the ACM. 2014. Vol. 57. No. 8. PP. 82–89.

[26] Карзанов А. В. Нахождение максимального потока в сети методом предпотоков // ДАН СССР. 1974. Т. 215. № 1. С. 49–52.

[27] Orlin J. B. Max flows in O(nm) time, or better // Proc. the forty-fifth annual ACM symposium on Theory of computing. 2013. PP. 765–774, ACM.

[28] Metropolis N., Rosenbluth A. W., Rosenbluth M. N., Teller A. H., Teller E. Equation of state calculations by fast computing machines // The Journal of Chemical Physics. 1953. Vol. 21. No. 6. PP. 1087–1092.

[29] Wiener N. Extrapolation, interpolation, and smoothing of stationary time series. – MIT press Cambridge. Vol. 2. 1949.

[30] Kalman R. E. A new approach to linear filtering and prediction problems // Journal of basic Engineering. 1960. Vol. 82. No. 1. PP. 35–45.

[31] Robbins H., Monro S. A stochastic approximation method // The Annals of Mathematical Statistics. 1951. PP. 400–407.

[32] Kiefer J., Wolfowitz J. et al. Stochastic estimation of the maximum of a regression function // The Annals of Mathematical Statistics. 1952. Vol. 23. No. 3. PP. 462–466.

[33] Blum J. R. Multidimensional stochastic approximation methods // The Annals of Mathematical Statistics. 1954. PP. 737–744.

[34] Поляк Б. Т. Новый метод типа стохастической аппроксимации //Автоматика и телемеханика. – 1990. – №7. – С. 98–107.

[35] Spall J. C. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation // IEEE Transactions on Automatic Control. – 1992. – Vol. 37. – No. 3. – PP. 332–341.

[36] Граничин О. Н. Процедура стохастической аппроксимации с возмущением на входе //Автоматика и телемеханика. – 1992. – №. 2. – С. 97-104.

[37] Граничин О. Н. Рандомизированные алгоритмы стохастической аппроксимации при произвольных помехах // Автоматика и телемеханика. - 2002. - №2. - С. 44–55.

[38] Граничин О. Н. Оценивание параметров линейной регрессии при произвольных помехах // Автоматика и телемеханика. - 2002. - №1. - С. 30–41.

[39] Bertsekas D. P. Dynamic programming and stochastic control. – 1976.

[40] Calafiore G. C., Campi M. C. et al. The scenario approach to robust control design // IEEE Transactions on Automatic Control. 2006. Vol. 51. No. 5. PP. 742–753.

[41] Амелин К. С., Граничин О. Н., Кияев В. И., Иевлев Н. В. Мобильность и супервычисления на охране природы // КомпьютерИнформ. 2011. № 05–06, С. 24–25.

[42] Amelin K., Amelina N., Granichin O., Granichina O., Andrievsky B. R. Randomized algorithm for uavs group flight optimization // Proc. 11th IFAC International Workshop on Adaptation and Learning in Control and Signal Processing. 2013. PP. 205–208.

[43] Амелин К. С. Рандомизация в контуре управления легкого БПЛА при полете в условиях неизвестных изменений направления ветра // Вестник Санкт-Петербургского университета. Серия 10. Прикладная математика. Информатика. Процессы управления. 2013. № 2. С. 86–102.

[44] Амелин К. С. Технология программирования легкого БПЛА для мобильной автономной группы // Стохастическая оптимизация в информатике. 2011. Т. 7. № 1. С. 93–115.

[45] Letchford A. N., Salazar-González, J. J. Stronger multi-commodity flow formulations of the Capacitated Vehicle Routing Problem // European Journal of Operational Research. 2015. Vol. 244. No. 3. PP. 730–738.

[46] Kameda H., Li J., Kim C., Zhang Y. Optimal Load Balancing in Distributed Computer Systems. – Springer Publishing Company Incorporated. 2011.

[47] Alakeel A. M. A guide to dynamic load balancing in distributed computer systems // International Journal of Computer Science and Information Security. 2010. Vol. 10. No. 6. PP. 153–160.

[48] Doddini Probhuling L. Load balancing algorithms in cloud computing // International Journal of Advanced Computer and Mathematical Sciences. 2013. Vol. 2. PP. 2230–9624.

[49] `https://github.com/Malkovsky/load-balancing`

[50] `http://research.microsoft.com/en-us/downloads/d3adb5f7-49ea-4170-abde-ea0206b25de2/default.aspx`

[51] Babenko M., Goldberg A. V. Experimental evaluation of a parametric flow algorithm. —- Technical report, Microsoft Research, 2006. MSR-TR-2006-77, 2006.