Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.М04-мм

# Модификация протокола локального голосования для оптимизации балансировки загрузки ресурсов распределенной системы

***БОРИСОГЛЕБСКАЯ Елена Александровна***

Отчёт по преддипломной практике

в форме «Решение»

Научный руководитель:
проф. каф. СП, д. ф.-м. н., О. Н. Граничин

Санкт-Петербург
2023

# Оглавление

# 1. Введение

В последнее время все чаще при вычислениях используются распределенные системы параллельных вычислений, для которых актуальна задача разделения пакета заданий между несколькими вычислительными устройствами. Подобные задачи возникают не только в вычислительных сетях [2], но и в транспортных, логистических [26] и других. Обычно при распределении заданий по узлам их стараются распределять так, чтобы загрузка вычислительных узлов была равномерной.

На практике используются как централизованные стратегии динамической балансировки загрузки сети, так и полностью распределенные(децентрализованные). При централизованной стратегии создается отдельный ресурс, который собирает информацию о состоянии всей сети и распределяет задания между узлами [34], [33]. При полностью распределенной стратегии алгоритмы балансировки загрузки выполняются на каждом из узлов. Узлы в таком случае обмениваются информацией о состоянии, а при необходимости и задачами. Перемещение задач при этом происходит только между соседними (связанными) узлами [8, 20, 11, 35, 10]. В статье [2] рассмотрена децентрализованная стратегия для решения задачи балансировки — протокол локального голосования. Протокол локального голосования — это алгоритм управления, согласно которому задачи, поступающие на узлы сети, могут быть равномерно перераспределены между всеми узлами сети за счет обмена данными между соседями. Применение такого протокола позволяет уменьшить нагрузку на центр обработки данных, так как все вычисления будут производится локально на агенте.

Сложность балансировки загрузки ресурсов часто заключается в отсутствии априорной информации о задачах. Новые задачи появляются все время и при этом заранее неизвестно ни сколько их всего, ни когда они появятся, ни сколько времени потребуется на их выполнение.

Помимо отсутствия информации о задачах, количество узлов распределенной системы может быть очень велико, что также усложняет централизованное вычисление оптимального распределения задач меж-

ду узлами. Здесь и появляются мультиагентные системы. Использование нескольких автономных агентов позволяет распараллелить, а значит и ускорить процесс принятия решения.

На сегодняшний день мультиагентные системы (МАС) нашли широкое применение в различных сферах: в производстве, в энергетике, в информационных технологиях и т.д. [21, 23, 5].

Обычно в МАС применяют программную децентрализацию и распараллеливание вычислений, а методы, средства и аппаратное обеспечения для передачи данных рассматриваются как некоторая среда, в которой "живут" агенты. При этом децентрализованные алгоритмы управления работой сетевого объекта должны быть адаптивными и устойчивыми в условиях неопределенности. Агенты могут выходить из строя, могут появляться задержки и помехи при передачи сообщений. Может существовать ряд ограничений связанных с используемыми ресурсами. При наличии ограничений на передачу сообщений используют динамически меняющуюся топологию коммуникационной сети. При этом знания о доступных маршрутах необходимо обновлять для всех агентов в реальном времени. Это ставит трудные проблемы децентрализованного группового управления на основе протоколов в коммуникационной среде без централизованного управления. Это трудная задача и она осложняется возможными помехами, задержками в передаче данных, появлением новых и исчезновением существующих узлов в сети.

Одна из фундаментальных концепций мультиагентных технологий — это нахождение консенсуса. Под консенсусом в данном случае понимается соглашение между агентами относительно общего значения, чаще всего минимума некоторой функции потерь:

$$\bar{F}(x) = \sum_{i=0}^{n} F^i(x),$$
$$x \in \mathbb{R}, F^i(x) : \mathbb{R}^d \to \mathbb{R}.$$

Исследование алгоритмов консенсуса и распределенной оптимизации началось в 1970–1980-е гг. [6, 4]. На сегодняшний день существует ряд подходов для случая, когда функции $F^i(x)$ выпуклы: метод множи-

телей с переменным направлением [7, 29], субградиентный метод [22, 19]. Для невыпуклых задач в работах [31, 14] разработан большой класс распределенных алгоритмов на основе различных «функционально-суррогатных единиц».

Существует ряд работ, в которых учтены возможные помехи. Помехи могут появляться как в связи с неточностями измерений, так и при передаче данных. Чаще всего предполагается, что помехи обладают некоторыми свойствами нормального распределения [3, 12, 13]. Кроме того, есть ряд статей, в которых единственные накладываемые на помехи условия это их центрированность и ограниченность [25].

В статье [2] была рассмотрена задача балансировки загрузки сети при помощи протокола локального голосования. Предложенный протокол локального голосования — это по сути алгоритм консенсуса по определению среднего значения состояний всех агентов в сети. При этом конкретные состояния передаются не далее чем на один шаг. Алгоритм, используемый в статье, может быть ускорен при помощи метода Нестерова со скоростью сходимости $\mathcal{O}\left(\frac{1}{k^2}\right)$.

# 2. Модификация протокола локального голосования

## 2.1. Постановка задачи балансировки сети

### 2.1.1. Сведения из теории графов

Граф — это пара $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, где $\mathcal{N}$ — множество узлов или вершин и $\mathcal{E}$ — множество ребер. Предполагаем, что граф простой, т.е. $(i, i) \notin \mathcal{E}, \forall i$ отсутствуют петли и между узлами может быть максимум одна дуга. Множеством соседей узла $i$ называется множество $\mathcal{N}^i = \{j : (j, i) \in \mathcal{E}\}$, т.е. множество узлов с ребрами входящими в $i$.

Сопоставим каждому ребру $(j, i) \in \mathcal{E}$ вес $b^{ij}$. Предположим, что все веса строго положительны. Граф может быть представлен матрицей смежности (или связности) $B = [b^{ij}]$ с весами $b^{ij} > 0$, если $(i, j) \in \mathcal{E}$ и $b^{ij} = 0$ в противном случае. Отметим, что $b^{ii} = 0, \forall i$.

Определим взвешенную полустепень входа вершины $i$ как количество входящих в $i$ ребер или $d^i(B) = \sum_{j \in N} b^{ij}$, а полустепенью выхода вершины $i$ — количество выходящих ребер или $d_0^i(B) = \sum_{j \in N} b^{ij}$. Если для всех вершин $i \in \mathcal{N}$ взвешенная полустепень входа равна взвешенной полустепени выхода, то граф называется сбалансированным по весам.

Граф называется неориентированным, если $b^{ij} = b^{ji}$, $\forall i, j \in \mathcal{N}$, т.е. если он является двунаправленным и веса ребер $(i, j)$ и $(j, i)$ совпадают. При этом матрица смежности симметрична. Неориентированный граф, в котором все узлы имеют одинаковые полустепени $d$ называется $d$-регулярным.

Определим диагональную матрицу $D(B) = diagd^i(B)$ из полустепеней входа и лапласиан графа $\mathcal{L}(B) = D(B) - B$. Заметим, что сумма по строкам лапласиана равна нулю. Следовательно, любой вектор составленный из одинаковых констант является правым собственным вектором, соответствующим нулевому собственному значению.

Обозначим через $d_{max}(B)$ максимальную полустепень входа графа $\mathcal{G}$. Применив круговой критерий Гершгорина [**?**], можно вывести еще

одно важное свойство лапласиана: все собственные числа матрицы $\mathcal{L}(B)$ имеют неотрицательную вещественную часть и лежат в круге с центром на вещественной оси в точке $d_{max}(B)$ и радиусом $d_{max}(B)$.

Обозначим $\lambda_1, \lambda_2, \ldots, \lambda_n$ собственные числа матрицы $\mathcal{L}_t(B)$. Отсортируем их в порядке возрастания вещественной части: $0 \leq Re(\lambda_1) \leq Re(\lambda_2) \leq \cdots \leq Re(\lambda_n)$. Известно, что если у графа есть остовное дерево, то $\lambda_1 = 0$.

### 2.1.2. Задача балансировки загрузки ресурсов

Рассмотрим модель децентрализованной системы распределения однотипных заданий между разными узлами (агентами) для параллельной работы с обратной связью. Обозначим $\mathcal{N} = \{1, \ldots, n\}$ — набор интеллектуальных агентов (узлов), каждый из которых выполняет поступающие задания по принципу очереди. Задания поступают в систему на разные узлы, возможно, в различные моменты времени. Связь между узлами определяется графом $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ с матрицей смежности $B$. Обозначим через $W = \mathcal{L}(B)$ лапласиан графа $\mathcal{G}$.

В каждый момент времени $t$ состояние агента $i \in \mathcal{N}$ описывается двумя характеристиками:

1. $q_t^i$ — загруженность очереди или длина очереди из атомарных элементарных заданий узла $i$ в момент времени $t$,
2. $p_t^i$ — производительность узла $i$ в момент времени $t$,
3. $x_t^i = \frac{q_t^i}{p_t^i}$ — время работы $i$-ого узла при загруженности $q_t^i$ и производительности $p_t^i$.

Продуктивность $p_t^i$ показывает какое количество элементарный заданий может выполнить $i$-ый агент за временной интервал $[t; t+1]$. Для простоты будем считать продуктивность всех узлов $i$ одинаковой в каждый момент времени $t$ и равной $p$.

Изменение состояния агента описывается следующим уравнением:

$$q_{t+1}^i = q_t^i - p + z_t^i - u_{t+1}^i; \quad i = 1, \ldots, n, \quad t = 0, 1, 2, \ldots,$$

где $z_t^i$ — новое задание, поступившее на узел $i$ в момент времени $t$, $u_{t+1}^i$ — результат перераспределения задач между узлами (добавление или уменьшение), получающиеся в итоге применения выбранного протокола перераспределения заданий. В уравнениях динамики предполагаем, что $\sum_{i\in\mathcal{N}} u_t^i = 0, \quad t = 0, 1, 2, \ldots$.

В каждый момент времени $t$ узел $i$ имеет следующую информацию:

1. зашумленные данные о своем времени работы:

$$y_t^{ii} = x_t^i + w_t^{ii},$$

2. зашумленные наблюдения о времени работы соседей, если $\mathcal{N}^i \neq \emptyset$:

$$y_t^{ij} = x_{t-s_t^{ij}}^i + w_t^{ij}, \quad j \in \mathcal{N}^\rangle,$$

где $w_t^{ij}$ — помехи, а $0 \leq s_t^{ij} \leq \bar{s}$ — целочисленная задержка, $\bar{s}$ — максимально возможная задержка.

Рассмотрим две постановки задачи: стационарную и нестационарную. В стационарном случае все задания поступают в систему на разные узлы в начальный момент времени $t = 0$. Если все задания выполняются только тем агентом, которому они поступили, то время реализации всех заданий определяется как

$$T_{max} = \max_{i\in\mathcal{N}} x_0^i = \max_{i\in\mathcal{N}} \frac{q_0^i}{p_0^i}.$$

В нестационарном — новые задания могут поступать в систему на любой из $n$ узлов в различные моменты времени $t$. Для момента времени $t$ определим $T_t$ — время до окончания выполнения всех существующих на данный момент заданий на всех узлах:

$$T_t(\mathbf{u_t}) = \max_{i\in N} \frac{q_t^i(u_t^i)}{r_t^i}. \tag{1}$$

Поставим цель управления

$$T_t(\mathbf{u_t}) \to \min_{\mathbf{u_t}}. \tag{2}$$

В статье [2] показано, что в стационарном случае из всех возможных вариантов распределения общего количества заданий, не обработанных к моменту времени $t$, наименьшее время работы системы соответствует тому, при котором:

$$x_t^i = x_t^j, \quad \forall i, j \in N.$$

В нестационарном случае требуется поддерживать загрузку всех узлов сети такой, чтобы при отсутствии новых задач все узлы закончили работу одновременно.

В соответствии с этим мы можем сформулировать следующую задачу. Задача найти управляющее воздействие $\mathbf{u_t} = [u_t^1, \ldots, u_t^n]^T \in \mathbb{R}^n$, при котором достигается минимум функции потерь:

$$\text{Найти} \quad \mathbf{u_t} = \text{Argmin}_{\mathbf{u}\in\mathbb{R}^n} F_t(\mathbf{u}), \quad \forall t = 0, 1, \ldots,$$

$$F_t(\mathbf{u}) = \sum_{i,j\in\mathcal{N}} \left( \frac{b^{ij}}{2}\Big(x_t^i(u^i) - x_t^j(u^j)\Big)^2 + \frac{1}{p^2} b^{ij} u^j u^i - \frac{2}{p} b^{ij} u^i \Big(x_{t-1}^j + \frac{z_{t-1}^j}{p} - 1\Big)\right) =$$

$$= \sum_{i,j\in\mathcal{N}} b^{ij} \left( x_t^i(u^i) \right)^2. \tag{3}$$

Функция $F_t(\mathbf{u})$ представляет из себя регуляризованный потенциал Лапласа (см. [21]).

## 2.2. Протокол локального голосования

Рассмотрим способ решения описанной выше задачи с применением методов мультиагентных технологий. Будем рассматривать каждый узел системы в качестве агента. Агенты обмениваются данными о длине своей очереди. В каждый момент времени $t$ агент $i$ вычисляет зашумленное значение градиента $g_t^i = f_t^i(u^i)$ :

$$g_t^i = f_t^i(u^i) = \frac{d}{du^i} F_t(\mathbf{u}) + \xi_t^i =$$

$$= -\sum_{j\in\mathcal{N}} \frac{b^{ij}}{p}\big(x_t^i(u^i) - x_t^j(u^j)\big) + \sum_{j\in\mathcal{N}} \frac{b^{ij}}{p^2} u^j - \frac{2}{p} b^{ij}\Big(x_{t-1}^j + \frac{z_{t-1}^j}{p} - 1\Big) + \xi_t^i, \tag{4}$$

$$t = 0, 1, \ldots.$$

В таком случае мы можем использовать протокол локального голосования, который для поставленной задачи является градиентом потенциала Лапласа:

$$u_t^i = h \sum_{j \in \mathcal{N}^i} b_{i,j}(y_t^{i,i} - y_t^{i,j}),$$

где $h > 0$ — размер шага протокола, $p$ — продуктивность агентов. $b^{ij} = b < 0$, $\forall j \in \mathcal{N}^i$ и равняется нулю $b^{ij} = 0$ для остальных пар $i, j : j \notin \mathcal{N}^i$ . Обозначим матрицу протокола $B = [b^{ij}]$ . Стоит отметить, что протокол (2.2) отличается от обычно используемого протокола локального голосования, в котором размер шага $h$ меняется со временем или отличается для разных агентов $i \in \mathcal{N}$ (см. [30]).

## 2.3. Быстрый градиентный метод Нестерова

Метод Нестерова — это итеративный алгоритм, основанный на методе тяжелого шарика [?]. На каждом шаге $k$ для каждого узла системы $i$, $i \in \mathcal{N}$, оценка управляющего воздействия $\hat{u}_k^i$ обновляется в соответствии с антиградиентом и направлением на предыдущем шаге $v_k$. Алгоритм быстрого градиента Нестерова представляет собой последовательность следующих шагов:

1. Выбрать $\hat{u}_0^i \in \mathbb{R}^d$ и $\gamma_0 > 0$, $v_0 = \hat{u}_0^i$
2. На $k$-ой итерации:
   (a) Найти $\alpha_k \in (0, 1)$:
   $$L\alpha_k^2 = (1 - \alpha_k) * \gamma_k + \alpha_k * \mu.$$
   (b) Вычислить $\gamma_{k+1} = (1 - \alpha_k) * \gamma_k + \alpha_k * \mu$.
   (c) Вычислить
   $$z_k = \frac{\alpha_k \gamma_k v_k + \gamma_{k+1} \hat{u}_k^i}{\gamma_k + \alpha_k \mu}.$$
   (d) Найти $\hat{u}_{k+1}^i$ такое, что
   $$\hat{u}_{k+1}^i = z_k - \frac{1}{L} \nabla F(z_k)$$

(e) Вычислить $v_{k+1} = \frac{1}{\gamma_{k+1}}\Big[(1-\alpha_k)\gamma_k v_k + \alpha_k \mu z_k - \alpha_k \nabla F(z_k)\Big]$.

## 2.4. Ускоренный протокол локального голосования

### 2.4.1. Постановка задачи и условия

Пусть задано вероятностное пространство $(\Omega, \mathcal{F}, P)$ соответствующее произвольному множеству $\Omega$, с $\sigma$-алгеброй всех событий $\mathcal{F}$ и вероятностной мерой $P$, $\mathcal{F}_{t-1}$ — $\sigma$-алгебра всех вероятностных событий, которые произошли до времени $t = 1, 2, \ldots$. Пусть $\mathbb{E}_n$ обозначает условное математическое ожидание относительно $\sigma$-алгебры, определяемой $\hat{\mathbf{u}}_\mathbf{0}, \cdots, \hat{\mathbf{u}}_{\mathbf{n-1}}$.

Алгоритм на каждом шаге $k$ вычисляет оценку $\hat{\mathbf{u}}_\mathbf{k}$ минимума $\mathbf{u_k}$ функции (3) (значение $\mathbf{u_k}$ может быть различным при разных $k$). Последовательность получившихся оценок $\{\hat{\mathbf{u}}_\mathbf{k}\}_{k=0}^{\infty}$ является решением следующей задачи:

$$\begin{gathered} \text{Найти} \quad \hat{\mathbf{u}}_\mathbf{k} \quad \text{s.t.} \quad \exists N, C < \infty : \quad \forall k > K \\ \mathbb{E}_n \|\hat{\mathbf{u}}_\mathbf{k} - \mathbf{u_k}\|^2 \leq C. \end{gathered} \tag{5}$$

**Предположение 1.** Для любого $k \geq 0$ должны существовать константы A и B такие, что:

$$\forall i \in \mathcal{N} \quad |-z_k^i + u_k^i + p| < A, \quad \left| x_{k-1}^j + \frac{z_{k-1}^j}{p} - 1 \right| < B.$$

**Предположение 2.** $\forall i \in \mathcal{N}, j \in \mathcal{N}^i$ шумы $\xi_k^{i,j}$ должны быть центрированы и иметь ограниченную дисперсию $\sigma^2$.

**Предположение 3.** Граф $\mathcal{G}$ должен быть связным.

### 2.4.2. Алгоритм

В статье [27] представлен алгоритм ускорения по Нестерову градиентного спуска для задачи трекинга.

Для решения задачи, определенной в (5) и удовлетворяющей предположениям 1–3 рассмотрим Ускоренный Протокол Локального Голо-

сования для задач трекинга. Алгоритм представляет из себя протокол локального голосования, ускоренный при помощи ускорения по Нестерову для задач трекинга [27].

Введем константы $L > 0$ и $\mu > 0$:

$$L = \frac{n+1}{p^4} \max_i \|b^{ij}\|^2, \quad \mu = \max_i \frac{2}{p^2} \sum_{j \in \mathcal{N}} b^{ij},$$

а также константы $a, b, c$:

$$b = \sum_{i,j \in \mathcal{N}} \frac{|b^{ij}|}{p}(4AB + 2A^2), \qquad a = 2A\sqrt{n+1},$$

$$c = \frac{9A^2}{p^2} \sum_{i,j \in \mathcal{N}} \left(b^{ij}\right)^2 + 1.$$

Для каждого узла системы $i$ рассмотрим следующий алгоритм:

1. Выбрать $\hat{u}_0^i \in \mathbb{R}^d$ и $\gamma_0 > 0$, $v_0 = \hat{u}_0^i$. Выбрать $h > 0, \eta \in (0, \mu), \alpha_x \in (0, 1)$ такое, что неравенство (6) всегда выполнялось. Вычисляется значение $H_1 = h - \frac{h^2 L}{2}$.

2. На $k$-ой итерации:

   (a) Найти $\alpha_k \in [\alpha_x, 1)$:
   $$H_1 - \frac{\alpha_k^2}{2\gamma_{k+1}} > 0. \tag{6}$$

   (b) Вычислить $\gamma_{k+1} = (1 - \alpha_k) * \gamma_k + \alpha_k * (\mu - \eta)$.

   (c) Вычислить
   $$z_k = \frac{\alpha_k \gamma_k v_k + \gamma_{k+1} \hat{u}_k^i}{\gamma_k + \alpha_k(\mu - \eta)}$$
   и посчитать значение $Y_k(z_k)$.

   (d) Вычислить $\hat{u}_{k+1}^i$:
   $$\hat{u}_{k+1}^i = z_k - hY_k(z_k).$$

   (e) Вычислить $v_{k+1} = \frac{1}{\gamma_k}\Big[(1 - \alpha_k)\gamma_k v_k + \alpha_k(\mu - \eta)z_k - \alpha_k Y_k(z_k)\Big]$.

### 2.4.3. Свойства алгоритма

Для изучения теоретических свойств алгоритма введем некоторые обозначения. Пусть $\{\alpha_k\}_{k=0}^{\infty}, \{\lambda_k\}_{k=0}^{\infty}, \{A_k\}_{k=0}^{\infty}, \{Z_k\}_{k=0}^{\infty}, \{D_k\}_{k=0}^{\infty}$ последовательности определенные следующим образом:

$$\alpha_k \in [\alpha_x, 1), \lambda_0 = 1, \lambda_{k+1} = (1-\alpha_k)\lambda_k,$$
$$A_0 = 0,$$
$$A_{k+1} = (1-\alpha_k)((1-\alpha_k)a + A_n),$$
$$Z_n = (1-\lambda_k)(b+ac) + A_k c,$$
$$D_0 = 0,$$
$$D_{k+1} = (1-\alpha_k)D_k + \frac{a(1+\alpha_k)+hc}{4\epsilon} + (1+\alpha_k)b +$$
$$+(1-\alpha_k)Z_k + h^2\frac{L}{2}\sigma^2 + \frac{\alpha_k c^2}{2\eta}.$$

Тогда имеем:

$$D_\infty = \alpha_x^{-1}\left[\frac{2a+hc}{4\epsilon} + 2b + (1-\alpha_x)(b + A_\infty c) + h^2\frac{L}{2}\sigma^2 + \frac{c^2}{2\eta}\right],$$

где

$$\mathbf{\Gamma} = \max_{n\geq 0}\gamma_k,$$
$$\epsilon \in \left(0, \frac{1}{a(1+\alpha_x)+hc}\left(H_1 - \frac{\alpha_x^2}{2\Gamma}\right)\right)$$

**Теорема** Если предположения 1–4 выполнены, то алгоритм, описанный выше, решает проблему (5) со следующими параметрами:

$$C = \frac{2}{\mu}D_\infty$$

Ошибка оценки после конечного числа итераций ограничена:

$$\mathbb{E}_k F_k(\hat{u}_k) - F_k(u_k) \leq \prod_{i=1}^{k}(1-\alpha_k)(\varphi_0(u_0) - F_k(u_k) + \Phi) + D_k, \qquad (7)$$

где $\varphi_0(x) = F_0(\hat{u}_0) + \frac{\gamma_0}{2}\|x - v_0\|^2, \Phi = \frac{\gamma_0 c^2}{2\eta^2}$.

*Доказательство Теоремы 1* Пусть $\mathbf{u_k}$ — оптимальное значение управляющего воздействия на шаге $k$. Обозначим функцию $\tilde{x}_k^i(u)$ как время, за которое агент $i$ закончит выполнение всех задач, находящихся в его очереди на данный момент, при выборе управляющего воздействия $u$:

$$\tilde{x}_k^i(u) = x_{k-1}^i - 1 + \frac{z_{k-1}^i}{p} - \frac{u}{p}, \tag{8}$$

где p — производительность агента.

Рассмотрим значение $\|\nabla f_k(\hat{\mathbf{u}})\|^2$:

$$\|\nabla f_k(\hat{\mathbf{u}})\|^2 = \|\nabla f_k(\hat{\mathbf{u}}) - \nabla f_k(\mathbf{u_k})\|^2 =$$
$$= \sum_{i\in\mathcal{N}}\Bigg(-\sum_{j\in\mathcal{N}}\frac{b^{ij}}{p}\big(\tilde{x}_k^i(\hat{u}^i) - \tilde{x}_k^j(\hat{u}^j)\big) + \sum_{j\in\mathcal{N}}\frac{b^{ij}}{p^2}\hat{u}^j +$$
$$+\sum_{j\in\mathcal{N}}\frac{b^{ij}}{p}\big(\tilde{x}_k^i(u_k^i) - \tilde{x}_k^j(u_k^j)\big) - \sum_{j\in\mathcal{N}}\frac{b^{ij}}{p^2}u_k^j\Bigg)^2.$$

Распишем разность $\tilde{x}_k^i(\hat{u}^i) - \tilde{x}_k^i(u_k^i)$ через формулу (8):

$$\tilde{x}_k^i(\hat{u}^i) - \tilde{x}_k^i(u_k^i) =$$
$$= \Big(x_{k-1}^i - 1 + \frac{z_{k-1}^i}{p} - \frac{\hat{u}^i}{p}\Big) - \Big(x_{k-1}^i - 1 + \frac{z_{k-1}^i}{p} - \frac{u_k^i}{p}\Big) = -\frac{\hat{u}^i}{p} + \frac{u_k^i}{p}.$$

Отдельно рассмотрим сумму $\frac{1}{p}\tilde{x}_k^j(\hat{u}^j) + \frac{1}{p^2}\hat{u}^j$:

$$\frac{1}{p}\tilde{x}_k^j(\hat{u}^j) + \frac{1}{p^2}\hat{u}^j = \frac{1}{p}\Big(x_{k-1}^j - 1 + \frac{z_{k-1}^j}{p} - \frac{\hat{u}^j}{p}\Big) + \frac{1}{p^2}\hat{u}^j = \frac{1}{p}\Big(x_{k-1}^j - 1 + \frac{z_{k-1}^j}{p}\Big).$$

Аналогично $\frac{1}{p}\tilde{x}_k^j(u^j) + \frac{1}{p^2}u^j = \frac{1}{p}\Big(x_{k-1}^j - 1 + \frac{z_{k-1}^j}{p}\Big)$. А значит:

$$\frac{1}{p}\tilde{x}_k^j(\hat{u}^j) + \frac{1}{p^2}\hat{u}^j - \frac{1}{p}\tilde{x}_k^j(u^j) - \frac{1}{p^2}u^j = 0.$$

Используя выражения (2.4.3) и (2.4.3) и неравенство Коши-Буняковского,

получаем:

$$\|\nabla f_k(\hat{\mathbf{u}})\|^2 = \sum_{i\in\mathcal{N}}\left(\sum_{j\in\mathcal{N}}\frac{b^{ij}}{p^2}\left(\hat{u}^i - u_k^i\right)\right)^2 \le \sum_{i\in\mathcal{N}}\left(\sum_{j\in\mathcal{N}}\left(\frac{b^{ij}}{p^2}\right)^2\right)\left(\sum_{j\in\mathcal{N}}(\hat{u}^i - u_k^i)^2\right) \le$$
$$\le \frac{n}{p^4}\max_i \|b^{ij}\|^2 \sum_{i\in\mathcal{N}}(\hat{u}^i - u_k^i)^2 \le L\|\hat{\mathbf{u}} - \mathbf{u_k}\|.$$

В итоге получаем:

$$\|\nabla f_k(\hat{\mathbf{u}})\|^2 \le L\|\hat{\mathbf{u}} - \mathbf{u_k}\|$$

Таким образом, выбранная константа $L > 0$ является константой Липшица функции $F_k$.

Оценим значение $\langle \nabla F_k(\hat{\mathbf{u}}), \hat{\mathbf{u}} - \mathbf{u_k}\rangle$:

$$\langle \nabla F_k(\hat{\mathbf{u}}), \hat{\mathbf{u}} - \mathbf{u_k}\rangle = \langle \nabla F_k(\hat{\mathbf{u}}) - \nabla F_k(\mathbf{u_k}), \hat{\mathbf{u}} - \mathbf{u_k}\rangle =$$
$$= \sum_{i\in\mathcal{N}}\left(\sum_{j\in\mathcal{N}}\left(-\frac{b^{ij}}{p}\left(\tilde{x}_k^i(\hat{u}^i) - \tilde{x}_k^j(\hat{u}^j)\right) + \frac{b^{ij}}{p^2}\hat{u}^j\right) -\right.$$
$$\left. - \sum_{j\in\mathcal{N}}\left(-\frac{b^{ij}}{p}\left(\tilde{x}_k^i(u_k^i) - \tilde{x}_k^j(u_k^j)\right) + \frac{b^{ij}}{p^2}u_k^j\right)\right)(\hat{u}^i - u_k^i).$$

Используя выражения (2.4.3) и (2.4.3)

$$\langle \nabla F_k(\hat{\mathbf{u}}), \hat{\mathbf{u}} - \mathbf{u_k}\rangle = \sum_{i\in\mathcal{N}}\left(\sum_{j\in\mathcal{N}}\frac{b^{ij}}{p^2}\left(\hat{u}^i - u_k^i\right)\right)(\hat{u}^i - u_k^i) \ge$$
$$\ge \mu\sum_{i\in\mathcal{N}}(\hat{u}^i - u_k^i)^2 = \mu\|\hat{\mathbf{u}} - \mathbf{u_k}\|^2$$

Таким образом, константа $\mu$ является константой строгой выпуклости функции $F_k$.

Рассмотрим изменение функций $F_k$ в точке $\mathbf{u}$:

$$\|F_k(\mathbf{u}) - F_{k+1}(\mathbf{u})\| =$$
$$= \left| \sum_{i,j \in \mathcal{N}} \left( \frac{b^{ij}}{2} \left( \tilde{x}_k^i(u^i) - \tilde{x}_k^j(u^j) \right)^2 + \frac{b^{ij}}{p^2} u^i u^j - \frac{2}{p} b^{ij} u^i \left( x_{k-1}^j + \frac{z_{k-1}^j}{p} - 1 \right) - \right.$$
$$\left. - \sum_{i,j \in \mathcal{N}} \left( \frac{b^{ij}}{2} \left( \tilde{x}_{k+1}^i(u^i) - \tilde{x}_{k+1}^j(u^j) \right)^2 + \frac{b^{ij}}{p^2} u^i u^j - \frac{2}{p} b^{ij} u^i \left( x_k^j + \frac{z_k^j}{p} - 1 \right) \right) \right|.$$

Так как

$$\tilde{x}_{k+1}^i(u^i) = x_{k-1}^i + \frac{z_{k-1}^i + z_k^i}{p} - 2 - \frac{u_k^i + u^i}{p},$$

где $u_k^i$ — управляющее воздействие выбранное на шаге $k$ для агента $i$, получаем:

$$\left| \sum_{i,j \in \mathcal{N}} \left( \frac{b^{ij}}{2} \left( \tilde{x}_k^i(u^i) - \tilde{x}_k^j(u^j) \right)^2 - \sum_{i,j \in \mathcal{N}} \left( \frac{b^{ij}}{2} \left( \tilde{x}_{k+1}^i(u^i) - \tilde{x}_{k+1}^j(u^j) \right)^2 \right| = \right.$$
$$= \left| \sum_{i,j \in \mathcal{N}} \frac{b^{ij}}{p} (-z_k^i + p + u_k^i + z_k^j - p + u_k^j) \left( \tilde{x}_k^i(u^i) - \tilde{x}_k^j(u^j) \right) - \right.$$
$$\left. - \sum_{i,j \in \mathcal{N}} \frac{b^{ij}}{2p^2} (-z_k^i + p + u_k^i + z_k^j - p + u_k^j)^2 \right| =$$
$$= \left| \sum_{i,j \in \mathcal{N}} \frac{2b^{ij}}{p} (-z_k^i + u_k^i + p) \left( \tilde{x}_k^i(u^i) - \tilde{x}_k^j(u^j) \right) - \sum_{i,j \in \mathcal{N}} \frac{b^{ij}}{2p^2} (-z_k^i + u_k^i + z_k^j + u_k^j)^2 \right|.$$

Рассмотрим следующую разность:

$$\sum_{i,j \in \mathcal{N}} \frac{2}{p} b^{ij} u^j \left( x_{k-1}^i + \frac{z_{k-1}^i}{p} - 1 \right) - \sum_{i,j \in \mathcal{N}} \frac{2}{p} b^{ij} u^j \left( x_k^i + \frac{z_k^i}{p} - 1 \right) =$$
$$= \sum_{i,j \in \mathcal{N}} \frac{2}{p^2} b^{ij} u^j \left( -z_k^i + u_k^i + p \right).$$

Таким образом, получаем

$$\|F_k(\mathbf{u}) - F_{k+1}(\mathbf{u})\| \leq$$

$$\leq \Big| \sum_{i,j\in\mathcal{N}} \frac{2b^{ij}}{p}(-z_k^i + u_k^i + p)\Big(\tilde{x}_k^i(u^i) - \tilde{x}_k^j(u^j)\Big) - \sum_{i,j\in\mathcal{N}} \frac{2}{p^2} b^{ij}u^j\Big(-z_k^i + u_k^i + p\Big) +$$

$$+ \sum_{i,j\in\mathcal{N}} \frac{4}{p} b^{ij}\Big(-z_k^i + u_k^i + p\Big)\Big(x_{k-1}^j + \frac{z_{k-1}^j}{p} - 1\Big)\Big| +$$

$$+ \Big| \sum_{i,j\in\mathcal{N}} \frac{b^{ij}}{p}\Big(4\Big(-z_k^i + u_k^i + p\Big)\Big(x_{k-1}^j + \frac{z_{k-1}^j}{p} - 1\Big) + \frac{1}{2}\big(-z_k^i + u_k^i + z_k^j + u_k^j\big)^2\Big)\Big| \leq$$

$$\leq \sqrt{\Big(\sum_{i\in\mathcal{N}} 2(-z_k^i + u_k^i + p) \sum_{j\in\mathcal{N}} \frac{b^{ij}}{p}\Big(\tilde{x}_k^i(u^i) - \tilde{x}_k^j(u^j) - \frac{u^j}{p} + 2\Big(x_{k-1}^j + \frac{z_{k-1}^j}{p} - 1\Big)\Big)\Big)^2} +$$

$$+ \sum_{i,j\in\mathcal{N}} \frac{|b^{ij}|}{p}(4AB + 2A^2).$$

Используя неравенство Коши-Буняковского, получаем:

$$\|F_k(\mathbf{u}) - F_{k+1}(\mathbf{u})\| \leq$$

$$\leq \sqrt{\sum_{i\in\mathcal{N}} 4(-z_k^i + u_k^i + p)^2} \cdot$$

$$\cdot \sqrt{\sum_{i\in\mathcal{N}}\Big(\sum_{j\in\mathcal{N}} \frac{b^{ij}}{p}\Big(\tilde{x}_k^i(u^i) - \tilde{x}_k^j(u^j) - \frac{u^j}{p} + 2\Big(x_{k-1}^j + \frac{z_{k-1}^j}{p} - 1\Big)\Big)\Big)^2} + b \leq$$

$$\leq a\|\nabla F_k(\mathbf{u})\| + b.$$

Предположение 2 также гарантирует, что изменение функций $\nabla F_k$

в точке ограничено:

$$\|\nabla F_k(\mathbf{u}) - \nabla F_{k+1}(\mathbf{u})\| =$$

$$= \sum_{i\in\mathcal{N}}\Bigg(\sum_{j\in\mathcal{N}}\frac{b^{ij}}{p}\Big(-\tilde{x}_k^i(u^i)+\tilde{x}_k^j(u^j)+\frac{u^j}{p}+2\Big(x_{k-1}^j+\frac{z_{k-1}^j}{p}-1\Big)\Big)-$$

$$-\sum_{j\in\mathcal{N}}\frac{b^{ij}}{p}\Big(-\tilde{x}_{k+1}^i(u^i)+\tilde{x}_{k+1}^j(u^j)+\frac{u^j}{p}+2\Big(x_k^j+\frac{z_k^j}{p}-1\Big)\Big)\Bigg)^2=$$

$$= \sum_{i\in\mathcal{N}}\Bigg(\sum_{j\in\mathcal{N}}\frac{b^{ij}}{p^2}\big(z_k^i-p-u_k^i-z_k^j+p+u_k^j-z_k^j+u_k^j+p\big)\Bigg)^2\leq$$

$$\leq \sum_{i\in\mathcal{N}}\Bigg(\sum_{j\in\mathcal{N}}\Big(b^{ij}\Big)^2\sum_{j\in\mathcal{N}}\frac{1}{p^2}\big(3A\big)^2\Bigg)\leq$$

$$\leq \frac{9A^2}{p^2}\sum_{i,j\in\mathcal{N}}\Big(b^{ij}\Big)^2\leq c$$

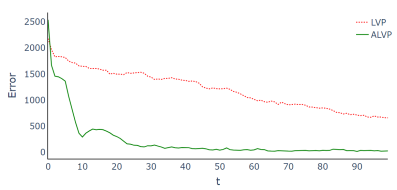
По теореме 1 из [27] выполняется неравенство (7).

## 2.5. Симуляция работы алгоритма

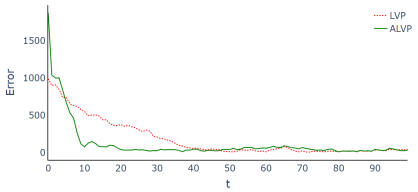
В текущем разделе представлено сравнение работы предлагаемого алгоритма с протоколом локального голосования, описанным в [**?**].

Рассмотрим распределенную сеть, состоящую из 5 серверов. Топология сети — кольцо. Каждый сервер получает некоторое количество задач для выполнения. Новые задачи появляются на всем протяжении работы алгоритма.

На Рис.1 представлен график средней сходимости к консенсусу за 100 итераций для разных значений шага $h$.




(a) Шаг алгоритма $h = 0.1$




(b) Шаг алгоритма $h = 0.2$

Рис. 1: Средняя сходимость к консенсусу. Красная пунктирная линия соответствует алгоритму LVP. Зеленая сплошная линия показывает предлагаемую ускоренную версию.

# 3. Система

## 3.1. Обзор существующих решений

На текущий момент существует большое количество инструментов, предназначенных для решения задач математической оптимизации. Список инструментов математической оптимизации включает в себя как бесплатные, так и проприетарные программные обеспечения (ПО), а также программные библиотеки и приложения.

Главным недостатком программных библиотек (например, SciPy [24], Gekko [9], MINPACK [18]) является отсутствие графического интерфейса, что затрудняет работу с ними при не знании языка программирования, к которому они привязаны. К проприетарному ПО относятся следующие системы для решения задач оптимизации: GAMS [32], Optimization Tool MATLAB [16], ALGLIB [1], MIDACO [17], MOSEK [15], TOMLAB [28] и др. Что важно, большинство существующих систем предоставляют возможность использования только уже реализованных алгоритмов. Чаще всего в такие алгоритмы заложены определенные условия на помехи, например, нормально распределенные с нулевым математическим ожиданием, что делает затруднительным решение задачи при появлении неизвестных, но ограниченных помех. Более того, существующие системы, как правило, не предназначены для решения задачи трекинга, то есть для ситуации, когда данные меняются со временем (например, движущиеся цели). Также практически во всех си-

стемах не предусмотрено распределенное решение задачи.

В Таблице 1 представлено сравнение нескольких уже существующих систем. Практические у всех систем отсутствует графический интерфейс, кроме Optimization Tool MATLAB. Возможность задания не только нормально распределенных помех и использование распределенных вычислений есть только у одной системы MIDACO, однако она не предназначена для решения задачи трекинга, как и все остальные рассматриваемые системы. Стоит отметить, что неизвестные, но ограниченные помехи охватывают область не только стохастических помех, но и детерминированных, так что MIDACO также не является универсальной системой в отличии от новой.

| Система | Графич. Интерфейс | Трекинг | Помехи | Распределен. |
|---|---|---|---|---|
| Optimization Tool Matlab | + | - | - | - |
| ALGLIB | - | - | - | - |
| GAMS | - | - | - | - |
| MIDACO | - | - | + | + |
| Новая система | + | + | + | + |

Таблица 1: Сравнение функциональности систем для решения задач оптимизации

## 3.2. Требования к системе балансировки сети

В этой главе представлены требования к новой системе балансировки сети, также представлены обоснования их необходимости. Требования получены, исходя из анализа аналогичных систем, представленных выше, и сформированы с учетом того, что система должна принимать на вход алгоритм, описанный в Главе 2. Требования разделены на функциональные, регулирующие поведение системы, и нефункциональные, отвечающие за потребление системой ресурсов.

### 3.2.1. Функциональные требования

**Графический интерфейс** Главная задача разрабатываемой системы —- предоставить исследователям возможность тестировать алгоритм, описанный в Главе 2, не изучая его конкретную реализацию. Для анализа результатов работы алгоритма и сравнения между собой разных алгоритмов система должна иметь графический интерфейс и представлять результаты в виде графиков.

**Объекты и связи в системе** Необходимо обеспечить возможность задания количества серверов ($1 \leq n \leq 100$), а также топологию, согласно которой сенсоры могут передавать друг другу данные: полную топологию или пользовательскую. Пользовательская топология позволяет учитывать возможные ограничения на связь между серверами, связанные, например, с дороговизной оборудования, удаленностью или иными сложностями реализации. Пользовательскую топологию необходимо задавать в виде таблицы смежности.

### 3.2.2. Модель появления новых задач

Необходимо обеспечить возможность задания в системе произвольной модели появления задач $z_t^i$ для каждого сервера $i$ в каждый момент времени $t$.

Сходимость протокола локального голосования, сильно зависит от количества получаемых задач. Предпочтительным вариантом является генерация новых задач с экспоненциально распределенной вычислительной сложностью, поступающих на случайно выбранный сервер.

**Модель шума** Необходимо обеспечить возможность задания в системе произвольной модели помех. Помимо этого, в систему необходимо добавить возможность задания пользовательских помех, распределенных нормально, равномерно-непрерывно, а также, помех в виде осцилляций. Осцилляции должны задаваться следующим образом: $v(t) = \pm asin(t)$, где $a$ — это заранее заданное число, а знак перед $a$ меняется каждые

50 итераций.

**Параметры алгоритма** Ускоренный протокол локального голосования, как и исходный протокол локального голосования является итеративным, поэтому система должна поддерживать возможность задания количества итераций.

Помимо этого для работы ускоренного протокола локального голосования необходимо задать параметры, описанные в Главе 2: $h, \eta, \mu, L, \alpha_0, \gamma_0$. Выбор данных параметров влияет на скорость и точность сходимости.

**Анализ качества** Для оценки работы алгоритма необходимо предоставление графика зависимости среднеквадратичной ошибки от количества итераций.

### 3.2.3. Нефункциональные требования

Система предназначается для использования как на персональных компьютерах пользователя, так и на серверах. При работе системы должна сохраниться возможность одновременного использования компьютера для других задач. В связи с этим выдвинут ряд требований, описанных далее.

- Система должна расходовать не более 4 Гб оперативной памяти.
- Система должна расходовать не более 2 Гб постоянной памяти.

# 4. Прототип системы балансировки сети

Система предназначена для моделирования работы алгоритмов для решения задачи балансировки сети. Главным преимуществом описываемой системы, выделяющим ее среди других, является возможность сравнивать, визуализировать и тестировать алгоритмы оптимизации для решения задачи балансировки сети. Реализацию системы можно посмотреть на GitHub[1].

## 4.1. Архитектура системы

Для реализации использовались следующие технологии:

- Язык программирования — Python;
- Библиотека для распараллеливания — mpi4py;
- Библиотека для моделирования работы распределнной системы агентов — disropt;
- Фреймворк — Flask.

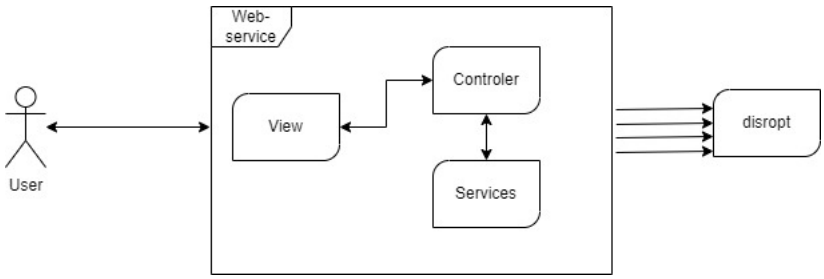



Рис. 2: Архитектура системы

Система представляет собой независимое пользовательское приложение, архитектура которого представлена на Рис. **??**. Пользователь вводит параметры через пользовательский интерфейс. Далее, сформированная модель подается на вход алгоритма. После выполнения вычислений пользователь получает доступ к результатам работы алгоритма

[1] https://github.com/HelBorg/Flask_Load_Balancing

в виде графиков: **зависимости элементов матрицы ковариации от количества итераций**, динамики изменения очереди для каждого сервера и изменения среднеквадратичной ошибки.

## 4.2. Апробация системы

В ходе апробации системы были протестированы все возможности системы — были протестированы различные модели измерений, различные модели шума, различное количество итераций алгоритма, а также были протестированы различные алгоритмы. Также, в результате апробации системы были получены сравнительные результаты времени работы алгоритмов для различных моделей измерений, шума.

## 4.3. Пользовательский интерфейс

Внешний вид системы показан на Рис. 3 Разберём подробнее реализацию вышеописанных требований на графическом интерфейсе системы.

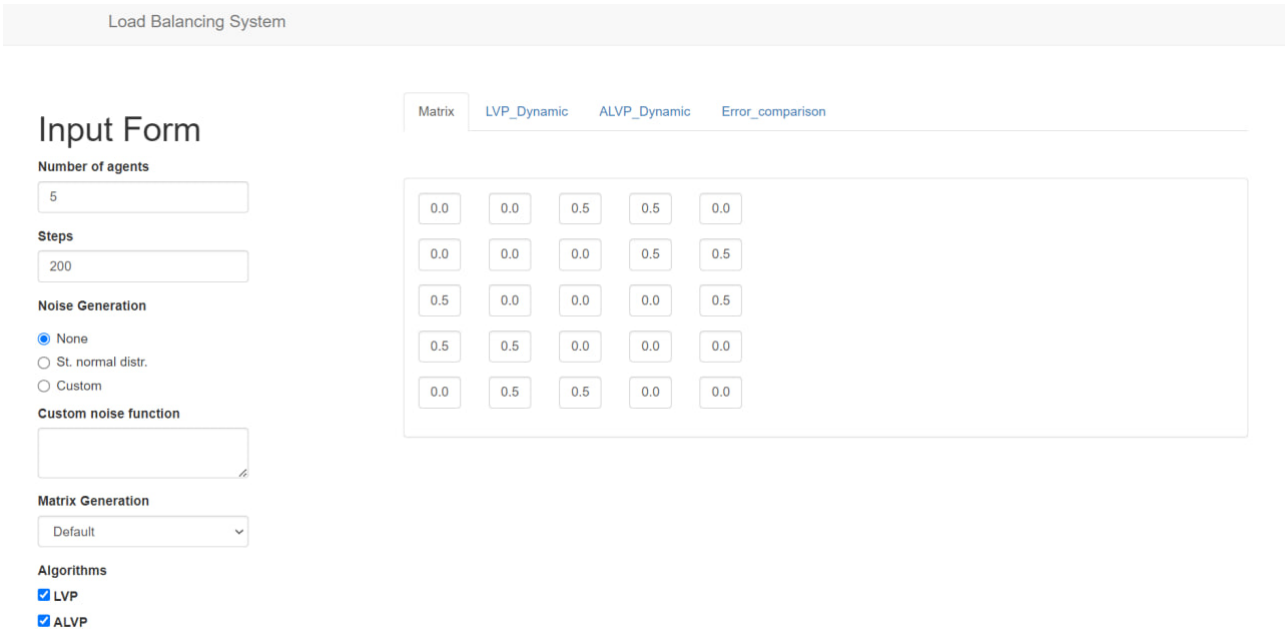



Рис. 3: Пользовательский интерфейс

В системе есть возможность ввода количества узлов системы, количества итераций. Есть возможность использовать матрицы по умолча-

нию или задать свою во вкладке "Matrix". Есть возможность выбрать один из представленных алгоритмов: LVP (протокол локального голосования), ALVP (ускоренный протокол локального голосования, описанный в работе). Для каждого алгоритма есть отдельная вкладка для задания параметров алгоритма.

В центральной части помимо вкладки задания алгоритма могут показываться вкладки результатов работы алгоритма: динамику изменения количества задач для каждого из запущенных алгоритмов и график сравнения среднеквадратичной ошибки. Графики являются интерактивными: можно менять масштаб, сравнивать отдельно конкретные алгоритмы, проверять точное значение среднеквадратичной ошибки на каждом из шагов алгоритма.

# 5. Заключение

В ходе работы были достигнуты следующие результаты:

- Исследованы существующие алгоритмы оптимизации, которые можно применить в задаче трекинга.
- Получена ускоренная версия протокола локального голосования и исследованы его свойства.
- Исследованы существующие системы для моделирования алгоритмов оптимизации.
- Составлены требования к системе.
- Разработана архитектура системы.
- Реализованы некоторые методы стохастической оптимизации и предложенный в работе метод в системе.
- Разработан прототип системы, включающий в себя возможность визуализации невязок алгоритмов.
- Сделан графический интерфейс в системе.
- Проведена апробация системы.
- Сделан доклад на XXV конференции молодых ученых «Навигация и управление движением»

# Список литературы


[1] ALGLIB Documentation. — https://www.alglib.net/docs.php (дата обращения: 1 мая 2023).

[2] Approximate consensus in stochastic networks with application to load balancing / Natalia Amelina, Alexander Fradkov, Yuming Jiang, Dimitrios J Vergados // IEEE Transactions on Information Theory. — 2015. — Vol. 61, no. 4. — P. 1739–1752.

[3] Blackman Sam S. Multiple hypothesis tracking for multiple target tracking // IEEE Aerospace and Electronic Systems Magazine. — 2004. — Vol. 19. — P. 5–18.

[4] Borkar Vivek S., Varaiya Pravin Pratap. Asymptotic agreement in distributed estimation // IEEE Transactions on Automatic Control. — 1982. — Vol. 27. — P. 650–655.

[5] Cooperative Control of Multi-Agent Systems: Optimal and Adaptive Design Approaches (Communications and Control Engineering) / F.L. Lewis, H. Zhang, K. Hengster-Movric, A. Das. — Springer, 2014. — P. 307.

[6] Degroot Morris H. Reaching a Consensus // Journal of the American Statistical Association. — 1974. — Vol. 69. — P. 118–121.

[7] Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers / Stephen P. Boyd, Neal Parikh, Eric King-Wah Chu et al. // Found. Trends Mach. Learn. — 2011. — Vol. 3. — P. 1–122.

[8] Friedrich Tobias, Sauerwald Thomas, Vilenchik Dan. Smoothed analysis of balancing networks // Random Struct. Algorithms. — 2011. — Vol. 39, no. 1. — P. 115–138.

[9] Gekko. — https://gekko.readthedocs.io/en/latest (дата обращения: 1 мая 2023).

[10] Gilly Katja, Juiz Carlos, Puigjaner Ramón. An up-to-date survey in web load balancing // World Wide Web. — 2011. — Vol. 14. — P. 105–131.

[11] Kechadi Mohand Tahar, Savvas Ilias K. Dynamic task scheduling for irregular network topologies // Parallel Comput. — 2005. — Vol. 31. — P. 757–776.

[12] Leonard Mark R., Zoubir Abdelhak M. Multi-target tracking in distributed sensor networks using particle PHD filters // Signal Process. — 2015. — Vol. 159. — P. 130–146.

[13] Li Xiaorong, Bar-Shalom Yaakov. Design of an interacting multiple model algorithm for air traffic control tracking // IEEE Trans. Control. Syst. Technol. — 1993. — Vol. 1. — P. 186–194.

[14] Lorenzo Paolo Di, Scutari Gesualdo. NEXT: In-Network Nonconvex Optimization // IEEE Transactions on Signal and Information Processing over Networks. — 2016. — Vol. 2. — P. 120–136.

[15] MOSEC Documentation. — https://www.mosek.com/documentation/ (дата обращения: 1 мая 2023).

[16] MathWorks. Optimization Toolbox Documentation. — https:www.mathworks.com/help/optim/ (дата обращения: 1 мая 2023).

[17] Mixed Integer Distributed Ant Colony Optimization. — http://www.midaco-solver.com/index.php/about (дата обращения: 1 мая 2023).

[18] More J. J. Garbow B. S. Hillstrom K. E. User guide for MINPACK-1. [In FORTRAN]. — https://cds.cern.ch/record/126569/files/CM-P00068642.pdf (дата обращения: 1 мая 2023). — 1980.

[19] Nedić Angelia, Ozdaglar Asuman E., Parrilo Pablo A. Constrained Consensus and Optimization in Multi-Agent Networks // IEEE Transactions on Automatic Control. — 2008. — Vol. 55. — P. 922–938.

[20] Nehra Neeraj, Patel R. B., Bhat V. K. Load Balancing in Heterogeneous P2P Systems using Mobile Agents // International Journal of Electrical and Computer Engineering. — 2008. — Vol. 2. — P. 2740–2745.

[21] Olfati-Saber Reza, Murray Richard M. Consensus problems in networks of agents with switching topology and time-delays // IEEE Transactions on automatic control. — 2004. — Vol. 49, no. 9. — P. 1520–1533.

[22] Rabbat Michael G., Nowak Robert D. Distributed optimization in sensor networks // Third International Symposium on Information Processing in Sensor Networks, 2004. IPSN 2004. — 2004. — P. 20–27.

[23] Ren W., Beard R. Distributed consensus in multi-vehicle cooperative control: theory and applications. — Springer, 2007.

[24] SciPy. — https://www.scipy.org/about.html (дата обращения: 1 мая 2023).

[25] Simultaneous perturbation stochastic approximation-based consensus for tracking under unknown-but-bounded disturbances / Oleg Granichin, Victoria Erofeeva, Yury Ivanskiy, Yuming Jiang // IEEE Transactions on Automatic Control. — 2021. — Vol. 66, no. 8. — P. 3710–3717.

[26] Skobelev Petr. Towards Autonomous AI Systems for Resource Management: Applications in Industry and Lessons Learned // Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection / Ed. by Yves Demazeau, Bo An, Javier Bajo, Antonio Fernández-Caballero. — Cham : Springer International Publishing, 2018. — P. 12–25.

[27] Stochastic fast gradient for tracking / Dmitry Kosaty, Alexander Vakhitov, Oleg Granichin, Ming Yuchi // 2019 American Control Conference (ACC) / IEEE. — 2019. — P. 1476–1481.

[28] TOMLAB Optimization. — https://tomopt.com/tomlab/about/ (дата обращения: 1 мая 2023).

[29] Tracking-ADMM for Distributed Constraint-Coupled Optimization / Alessandro Falsone, Ivano Notarnicola, Giuseppe Notarstefano, Maria Prandini // Autom. — 2019. — Vol. 117. — P. 108962.

[30] Wang Lin, Liu ZhiXin. Robust consensus of multi-agent systems with noise // Science in China Series F: Information Sciences. — 2009. — Vol. 52, no. 5. — P. 824–834.

[31] Zhu Minghui, Martínez Sonia. Discrete-time dynamic average consensus // Autom. — 2010. — Vol. 46. — P. 322–329.

[32] Брук А. Кендрик Д. Меераус А. Раман Р. Руководство по GAMS. — https://www.gams.com/latest/docs/. — 1999 (дата обращения: 1 мая 2023).

[33] Вахитов А.Т. Граничин О.Н. Паньшенсков М.А. Методы оценвания скорости передачи данных в грид // Нейрокомпьютеры: разработка, применение. — 2009. — Vol. 11. — P. 45–52.

[34] Граничин О.Н. Стохастическая оптимизация и системное программирование // Стохастическая оптимизация в информатике. — 2010. — no. 6. — P. 3–44.

[35] Я.В.Кутаева. Балансировка загрузки несимметричного вычислительного комплекса при решении задачи статистического оценивания // Информатика и системы управления. — 2006. — no. 2. — P. 88–93.