САНКТ - ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Математико-механический факультет
Кафедра системного программирования

Модификация протокола локального голосования для оптимизации балансировки загрузки ресурсов распределенной системы

Отчет по производственной практике

Борисоглебская Елена Александровна
1 курс, группа 21.М04-мм

Научный руководитель:
д.ф.-м. н., профессор Граничин О. Н.

Санкт-Петербург
2022

Содержание

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, но и в производственных [1], транспортных, логистических сетях и сетях обслуживания [2]. 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 [3], [4]. 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 [5–9]. В статье [10] 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.

Задачи балансировки загрузки узлов в литературе встречаются достаточно часто [1–9], что подчеркивает их актуальность.

Сложность балансировки загрузки ресурсов 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. Это относится ко многим рекурсивным алгоритмам «разделяй и властвуй», включая поиск с возвратом, поиск по дереву и вычисления по методу ветвей и границ.

Помимо отсутствия информации о задачах, количество узлов распределенной системы может быть очень велико, что также усложняет централизованное вычисление оптимального распределения задач между узлами. Здесь и появляются мультиагентные системы. Использование нескольких автономных агентов позволяет распараллелить, а значит и ускорить процесс принятия решения.

На сегодняшний день мультиагентные технологии нашли широкое применение в различных сферах: в производстве, в энергетике, в информационных технологиях и т.д. [11–13]. 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. [14, 15]. Распределенные асинхронные алгоритмы стохастической оптимизации (SA) изучались в [16]. На сегодняшний день существует ряд подходов для случая, когда функции $F^i(x)$ выпуклы: метод множителей с переменным направлением [17, 18], субградиентный метод [19, 20]. Для невыпуклых задач в работах [21, 22] разработан большой класс распределенных алгоритмов на основе различных «функционально-суррогатных единиц».

Существует ряд работ, в которых учтены возможные помехи. Данные помехи могут появляться как в связи с неточностями измерений, так и при передаче данных. Чаще всего предполагается, что помехи обладают некоторыми свойствами нормального распределения [23–25]. Кроме того, есть ряд статей, в которых единственное накладываемое на помехи условие это их ограниченность [26]. В таком случае для снижения влияния помех используются рандомизированные алгоритмы, в которых каждый шаг основывается на случайном выборе некоторого правила.

Одним из таких алгоритмов является рандомизированный алгоритм стохастической оптимизации (Simultaneous Perturbation Stochastic Approximation abbr. SPSA) [27–30]. Этот алгоритм отличается от остальных способностью решать оптимизационные задачи при наличии произвольно неизвестных, но ограниченных помех и изменяющихся во времени параметров системы. Помехи в таком случае могут быть неслучайными, и даже если они случайны, их статистические характеристики знать необязательно [31].

В статье [26] был предложен рандомизированный алгоритм стохастической оптимизации,
основывающийся на протоколе локального голосования. Он объединяет в себе все лучшее от обоих алгоритмов. Каждый агент старается самостоятельно определить для себя оптимальные параметры работы на основе рандомизированного алгоритма и дополнительно учитывает информацию полученную от соседних агентов. В данной статье не регулируется выбор шага протокола локального голосования, он остается фиксированным на протяжении всего времени работы алгоритма. Динамический выбор шага может позволить улучшить качество сходимости алгоритма.

2. Постановка задачи

2.1. Сведения из теории графов

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

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

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

Граф называется неориентированным, если $a^{ij} = a^{ji}, \forall i, j \in N$, 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 $D(A) = diagd^i(A)$ из полустепеней входа и лапласиан графа $\mathcal{L}(A) = D(A) - A$. 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 $d_{max}(A)$ максимальную полустепень входа графа G. Применив круговой критерий Гершгорина [32], можно вывести еще одно важное свойство лапласиана: все собственные числа матрицы $\mathcal{L}(A)$ имеют неотрицательную вещественную часть и лежат в круге с центром на вещественной оси в точке $d_{max}(A)$ и радиусом $d_{max}(A)$.

Обозначим $\lambda_1, \lambda_2, \ldots, \lambda_n$ собственные числа матрицы $\mathcal{L}_t(A)$. 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 $\lambda_1 = 0$.

2.2. Задача балансировки загрузки ресурсов

Рассмотрим модель децентрализованной системы распределения однотипных заданий между разными узлами (агентами) для параллельной работы с обратной связью. Обозначим $N = 1, \ldots, n$ — 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, топологией динамической сети: в момент времени t графом $G_t = (N, E_t)$ с матрицей смежности A. Обозначим через $W_t = \mathcal{L}_t(A)$ лапласиан графа G_t.

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

1) q_t^i — загруженность очереди или длина очереди из атомарных элементарных заданий узла i в момент времени t,

2) r_t^i — производительность узла i в момент времени t.

Изменение состояния агента описывается следующим уравнением:

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

где z_t^i — новое задание, поступившее на узел i в момент времени t, u_t^i — результат перераспределения задач между узлами (добавление или уменьшение), получающийся в итоге применения выбранного протокола перераспределения заданий. В уравнениях динамики предполагаем, что $\sum_i u_i^t = 0, \quad t = 0, 1, 2, \ldots$.

В каждый момент времени t узел i имеет следующую информацию:

1) зашумленные данные о своей длине очереди:

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

2) зашумленные наблюдения о длинах очередей соседей, если $N_t^i \neq \emptyset$:

$$y_t^{ij} = q_{t-p_t^{ij}}^i + w_t^{ij}, \quad j \in N_t^i,$$

где w_t^{ij} — помехи, а $0 \leq p_t^{ij} \leq \bar{p}$ — целочисленная задержка, $\bar{p}$ — максимально возможная задержка,

3) данные о своей производительности r_t^i и о производительностях узлов соседей $r_t^j, j \in N_t^i$.

Требуется поддерживать равномерную загрузку всех узлов сети. Будем рассматривать 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

$$T_{max} = \max_{i \in N} q_0^i / r_0^i.$$

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 заданий на всех узлах.

$$T_t = \max_{i \in N} q_t^i / r_t^i. \tag{1}$$

Будем называть отношение q_t^i / r_t^i загруженностью узла i в момент времени t. Поставим цель управления

$$T_t \to \min_{u_t}. \tag{2}$$

В статье [33] 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:

$$q_t^i/r_t^i = q_t^j/r_t^j, \quad \forall i, j \in N.$$

В соответствии с этим утверждением можно переформулировать (1) и (2) следующим образом:

$$T_t(q_t) = \sum_{i,j \in \{1,\dots,m\}} a_t^{j,i} \left(\frac{q_t^j}{r_t^j} - \frac{q_t^i}{r_t^i} \right)^2 \to \min_{q_t}.$$

Рассмотрим решение задачи с помощью градиентного спуска. Производная функции $T_t(q_t)$ равна:

$$\frac{dT_t(q_t)}{dq_t^i} = \sum_{j=1}^{m} 2(a_t^{i,j} - a_t^{j,i}) \frac{1}{r_t^i} \left(\frac{q_t^j}{r_t^j} - \frac{q_t^i}{r_t^i} \right).$$

В таком случае алгоритм будет выглядеть следующим образом:

$$q_{t+1}^i = q_t^i - \gamma_k \frac{dT_t(q_t)}{dq_t^i},$$
$$\gamma = \text{argmin} \quad T_t(q_t^i - \gamma \bigtriangledown T_t(q_t^i)).$$

В качестве γ_k также можно выбрать последовательность сходящуюся к нулю. Например:

$$\gamma_k = \gamma/k,$$
$$\gamma = \frac{1}{\lambda_{\min}}.$$

2.3. Метод на основе SPSA

Рассмотрим другой способ решения описанной выше задачи с применением методов мультиагентных технологий. В статье [26] решается задача достижения агентами консенсуса при наличии неизвестных, но ограниченных помех. Каждый агент i в момент времени t получает некоторую информацию y_t^i:

$$y_t^i = f_{\xi_t}^i(x_t^i) + \upsilon_t^i, \tag{3}$$

где x — некоторая величина, которую агент может или не может изменять,

ξ_t — некоторый способ задания времени с учетом возмущений, оно может быть известно или не известно, в простейшем случае $\xi_t = t$,

$f_{\xi_t}^i : \mathbb{R}^d \to \mathbb{R}$ — дифференцируемая функция, сопоставляющая величине x значение y, некий закон ставящий в соответствие начальным данным наблюдаемые значения,

υ_t^i — неизвестные ограниченные помехи, связанные с считыванием данных устройством.

Агент i в каждый момент времени t измеряет два значения y_t^i, и обмениваясь данными с соседями пытается оценить вектор θ_t. Для этого решается задача нестационарной оптимизации среднего риска, иначе говоря ищется минимум $\hat{\theta}_t$ функционала $F_t^i(\theta)$:

$$\begin{aligned} &F_t^i(\theta) = \mathbb{E}_{\mathcal{F}_{t-1}} f_{\xi_t}^i(\theta), \\ &\bar{F}_t(\theta) = \sum_{i\in\mathbb{N}} F_t^i(\theta) = \mathbb{E}_{\mathcal{F}_{t-1}} \sum_{i\in\mathbb{N}} f_{\xi_t}^i(\theta) \to \min_\theta . \end{aligned} \tag{4}$$

При меняющихся условиях минимум функционала (4) может также меняться со временем. Поэтому очень важно быстро и эффективно найти данный минимум.

Рассмотрим алгоритм минимизации функционала (4), предложенный в статье [26]:

$$\begin{cases} x_{2k}^i = \hat{\theta}_{2k-2}^i + \beta_k^+ \Delta_k^i, x_{2k-1}^i = \hat{\theta}_{2k-2}^i - \beta_k^- \Delta_k^i, \\ \hat{\theta}_{2k-1}^i = \hat{\theta}_{2k-2}^i, \\ \hat{\theta}_{2k}^i = \hat{\theta}_{2k-1}^i - \alpha \Bigg(\frac{y_{2k}^i - y_{2k-1}^i}{\beta_k} \mathbf{K}_k^i(\Delta_k^i) \\ \qquad\qquad +\gamma \sum_{j\in\mathcal{N}_{2k-1}^i} b_{2k-1}^{i,j} (\tilde{\theta}_{2k-1}^{i,j} - \hat{\theta}_{2k-1}^i) \Bigg), \end{cases} \tag{5}$$

где Δ_k^i — последовательность случайных независимых векторов из $\mathbb{R}^d$ с симметричным распределением, называемые единовременным пробным возмущением, $\mathbf{K}_k^i$ — вектор функция (ядро). где Δ_k^i — последовательность случайных независимых векторов из $\mathbb{R}^d$ с симметричным распределением, называемые единовременным пробным возмущением, $\mathbf{K}_k^i$ — вектор функция (ядро).

Выбираются начальные значения $\hat{\theta}_0^i \in \mathbb{R}^d$, неотрицательный размер шага α, коэффициент γ и последовательность ненулевых значений $\{\beta_k^+\}$ и $\{\beta_k^-\}$ таких, что $\beta_k^+ + \beta_k^- > 0$.

На каждом шаге алгоритма выбираются две точки x_{2k}^i и x_{2k-1}^i, в которых происходит измерение. Точки выбираются по обе стороны от уже оцененного на предыдущем шаге $\hat{\theta}_{2k-2}^i$. По ним вычисляются y_{2k}^i и y_{2k-1}^i. При обновлении значения $\hat{\theta}_{2k}^i$ мы из предыдущего полученного значения вычитаем сумму двух слагаемых, первое из которых представляет из себя шаг SPSA метода, а второе — протокола локального голосования. Таким образом на каждом шаге алгоритма значение $\hat{\theta}_{2k}^i$ обновляется в соответствии с новым полученным значением с сенсора (SPSA метод) и зашумленными значениями $\tilde{\theta}_{2k-1}^{i,j}$, полученными от соседних агентов.

3. Модификации алгоритма

3.1. Ускорение Чебышева

Как показано в статье [26] область сходимости алгоритма зависит от числа обусловленности χ лапласиана W_t:

$$\chi = \frac{\lambda_{max}(W_t)}{\lambda^+_{min}(W_t)}.$$

Зачастую ускорение Чебышева позволяет уменьшить количество итераций алгоритма путем замены лапласиана W_t на $P_k(W_t)$, где $P_k(x)$ — полином степени K.

$$P_K(x) = 1 - \frac{T_K(c_2 * (1-x))}{T_K(c_2)},$$

где $c_2 = \frac{\chi+1}{\chi-1}$, а T_K — полином Чебышева, определяемый следующим образом:

$$T_0(x) = 1, \quad T_1(x) = x,$$
$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x).$$

Полином построен таким образом, что число обусловленности $P_k(W_t)$ равно $\mathcal{O}(1)$, при K равном $\lfloor\sqrt{\chi}\rfloor$. Ускорение Чебышева было использовано для оптимизации децентрализованных методов в [34].

Одним из недостатков алгоритма является увеличение связей между агентами. В результате замены матрицы W_t матрицей $P_k(W_t)$ значительно увеличивается число каналов связи между агентами, а значит и время, затрачиваемое на каждом шаге алгоритма на общение между агентами.

3.2. Предобуславливание

При использовании итеративных алгоритмов используется предобуславливание для уменьшения числа обусловленности. Исходная матрица A преобразуется в матрицу $\widetilde{A} = M_1^{-1}AM_2^{-1}$, где $M = M_1M_2$ — матрица предобусловленности, которая в некотором смысле схожа с матрицей A. Для того чтобы матрица M была симметрична необходимо, чтобы:

$$M_1^T = M_2.$$

M должно быть выбрано таким образом, чтобы число обусловленности матрицы A уменьшилось. Существует множество алгоритмов предобуславливания.

3.2.1. Диагональное масштабирование

Один из самых простых алгоритмов предобуславливания это — диагональное масштабирование. В качестве матрицы M берутся диагональные элементы исходной матрицы:

$$M = diag(A).$$

3.2.2. Симметричная последовательная чрезмерная релаксация

Еще один алгоритм предобуславливание симметричная последовательная чрезмерная релаксация (Symmetric successive over-relaxation abbr. SSOR). Если матрица A симметрична и может быть представлена в виде:

$$A = D + L + L^T,$$

то SSOR матрица предобусловленности выглядит следующим образом:

$$M = (D + L)D^{-1}(D + L)^T.$$

4. Результат

В результате проделанной работы был изучен алгоритм рандомизированный алгоритм стохастической оптимизации (SPSA) и сформулирована задача оптимизации алгоритма локального голосования для балансировки загрузки ресурсов распределенной системы.

Список литературы

[1] D. Armbruster, A.S. Mikhailov, K. Kaneko (eds.), "Networks of Interacting Machines: Production Organization in Complex Industrial Systems and Biological Cells,"World Scientific Singapore, p. 267, 2005.

[2] A. Glashenko, S. Inozemtzev, I. Grachev, P. Skobelev, "Magenta Technology: case studies of Magenta i-scheduler for road transportation,"Proc. of Int. Conf. on Autonomous Agents and Multi Agent Systems (AAMAS-6), p. 1385-1392, 2007.

[3] О.Н.Граничин, "Стохастическая оптимизация и системное программирование," Стохастическая оптимизация в информатике, No. 6, pp.3-44, 2010.

[4] А.Т. Вахитов, О.Н.Граничин, М.А. Паньшенсков. "Методы оценвания скорости передачи данных в грид,"Нейрокомпьютеры: разработка, применение, vol.11, pp.45-52, 2009

[5] T.A. Friedrich, T.B. Sauerwald, D.C. Vilenchik, "Smoothed analysis of balancing networks,"Random Structures and Algorithms, Vol. 39. No. 1, pp. 115-138, Aug. 2011.

[6] H. Li, "Load balancing algorithm for heterogeneous P2P systems based on Mobile Agent,"Proc. of ICEICE 2011, pp. 1446-1449, 2011.

[7] M.-T. Kechadi, I.K. Savvas, "Dynamic task scheduling for irregular network topologies,"Parallel Computing. Vol. 31. No. 7. pp.757-776, 2005.

[8] Я.В.Кутаева. "Балансировка загрузки несимметричного вычислительного комплекса при решении задачи статистического оценивания,"Информатика и системы управления, No.2(12), pp.88-93, 2006.

[9] ly K. Gil, C. Juiz, R. Puigjaner, "An up-to-date survey in web load balancing,"World Wide Web, Vol. 14. No. 2. pp. 105-131, 2011.

[10] N. Amelina, A. Fradkov, Y. Jiang, and D. J. Vergados, "Approximate consensus in stochastic networks with application to load balancing," IEEE Trans. Inf. Theory, vol. 61, no. 4, pp. 1739–1752, Apr. 2015.

[11] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," IEEE Trans. Autom. Control, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[12] W. Ren and R. W. Beard, Distributed Consensus in Multi-Vehicle Cooperative Control. New York, NY, USA: Springer, 2008.

[13] F. L. Lewis,H. Zhang, K. Hengster-Movric, andA.Das, Cooperative Control of Multi-Agent Systems: Optimal and Adaptive Design Approaches. New York, NY, USA: Springer, 2013.

[14] M. DeGroot, "Reaching a consensus," J. Amer. Stat. Assoc., vol. 69, pp. 118–121, 1974.

[15] V. Borkar and P. Varaiya, “Asymptotic agreement in distributed estimation,” IEEE Trans. Autom. Control, vol. AC-27, no. 3, pp. 650–655, Jun. 1982.

[16] J.N. Tsitsiklis, D. P. Bertsekas, and M. Athans, “Distributed asynchronous deterministic and stochastic gradient optimization algorithms,” in Proc. Amer. Control Conf., pp. 484–489, 1984.

[17] S. Boyd et al., “Distributed optimization and statistical learning via the alternating direction method of multipliers,” Foundations Trends Mach. Learn., vol. 3, no. 1, pp. 1–122, 2011.

[18] A. Falsone, I. Notarnicola, G. Notarstefano, and M. Prandini, “Tracking-ADMM for distributed constraint-coupled optimization,” Automatica,vol. 117, Art. no. 108962, 2020.

[19] M. Rabbat and R. Nowak, “Distributed optimization in sensor networks,”in Proc. 3rd Int. Symp. Inf. Process. Sensor Netw., 2004, pp. 20–27, 2004.

[20] A. Nedic and A. Ozdaglar, “Distributed subgradient methods for multiagent optimization,” IEEE Trans.Autom.Control, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[21] M. Zhu and S. Martínez, “Discrete-time dynamic average consensus,” Automatica, vol. 46, no. 2, pp. 322–329, 2010.

[22] P. Di Lorenzo and G. Scutari, “Next: In-network nonconvex optimization,” IEEE Trans.Signal Inf. Process. Netw., vol. 2, no. 2, pp. 120–136, Jun. 2016.

[23] S. S. Blackman, “Multiple hypothesis tracking for multiple target tracking,” IEEE Aerospace and Electronic Systems Magazine, vol. 19, no. 1, pp. 5–18, 2004.

[24] M. R. Leonard and A. M. Zoubir, “Multi-target tracking in distributed sensor networks using particle phd filters,” Signal Processing, vol. 159, pp. 130–146, 2019.

[25] X. R. Li and Y. Bar-Shalom, “Design of an interacting multiple model algorithm for air traffic control tracking,” IEEE Transactions on Control Systems Technology, vol. 1, no. 3, pp. 186–194, 1993.

[26] O. Granichin, V. Erofeeva, Y.Ivanskiy, Y.Jiang, “Simultaneous Perturbation Stochastic Approximation-Based Consensus for Tracking Under Unknown-But-Bounded Disturbances” IEEE Trans. Autom. Control, vol. 66, no. 8, pp. 3710–3717, Aug. 2021.

[27] О.Н.Граничин, “Процедура стохастической аппроксимации с возмущением на входе,” Автоматика и телемеханика, vol. 2, pp. 97–104, 1992.

[28] О.Н.Граничин, В.Н. Фомин, “Адаптивное управление с использованием пробных сигналов в канале обратной связи,” Автоматика и телемеханика, vol. 2, pp. 100–112, 1986.

[29] О.Н.Граничин, В.Н. Фомин, “Оптимальные порядки точности поисковых алгоритмов стохастической оптимизации,” Проблемы передачи информации, vol. 26, no. 2, pp. 126–133, 1990.

[30] J. C. Spall, “Multivariate stochastic approximation using a simultaneous perturbation gradient approximation,” IEEE Transactions on Automatic Control, vol. 37, no. 3, pp. 332–341, 1992.

[31] А. Сергиенко, “Распределенное отслеживание большого количества летательных аппаратов в условиях неопределенностей,” Навигация и управление движением. Материалы XXII конференции молодых ученых с международным участием, pp. 319–321, 2020.

[32] S. Gershgorin, “On bounding the eigenvalues of a matrix,” Izv. Akad. Nauk. SSSR Otd Mat. Estest 1, no. 6, pp. 749–754 749–754, 1931.

[33] Н.О.Амелина, О.Н.Граничин. "Управление балансировкой загрузки в вычислительных сетях,"Глава 13 в монографии «Проблемы сетевого управления» СПб.: Наука, pp.297-318, 2015

[34] K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massouli´e. "Optimal algorithms for smooth and strongly convex distributed optimization in networks,"In Proceedings of the 34th International Conference on Machine Learning, vol.70, pp.3027–3036. JMLR. org, 2017.

[35] E. Gorbunov, A. Rogozin, A. Beznosikov, D. Dvinskikh, A. Gasnikov, "Recent Theoretical Advances in Decentralized Distributed Convex Optimization"High-Dimensional Optimization and Probability, Jan. 2022.