УДК 621.391

Ю. А. ДУЙНОВА, магистр управления;
Н. И. ЛУЩИК, магистрант;
С. И. ПОЛОВЕНЯ, канд. техн. наук, доцент,
Белорусская государственная академия связи, Минск;
Ю. В. МЕЛЬНИК, канд. техн. наук, ст. науч. сотрудник,
Государственный университет телекоммуникаций, Киев

МОДИФИЦИРОВАННЫЙ АЛГОРИТМ ДОСТИЖЕНИЯ КОНСЕНСУСА НА ОСНОВЕ ЛОКАЛЬНОГО ПРАВИЛА ГОЛОСОВАНИЯ

Рассмотрен алгоритм достижения консенсуса в децентрализованной сети с помехами как основа для создания специализированного программного обеспечения. Приведены задачи балансировки загрузки сети, а также задачи достижения консенсуса группой взаимодействующих агентов.

Ключевые слова: достижение консенсуса; децентрализованные системы; балансировка загрузки.

Введение

Сегодня децентрализованные сети находят все более широкое применение. В таких сетях очень важно эффективно распределять пакеты (задания) среди большого числа агентов (узлов). Проблемы эффективного распределения возникают на производстве, в транспорте и сенсорных сетях [1]. Задание может быть выполнено одним агентом или может быть распределено между группой агентов. Задача балансировки загрузки состоит в поддержании равномерной загруженности всех агентов во времени. Многие авторы посвятили этой задаче научные труды, что говорит об актуальности данной темы. В большинстве публикаций, касающихся вычислительных систем, не учитываются разрывы связи, помехи и задержки. В сетевых системах, наоборот, помехи, задержки и разрывы связи учитываются. Задача балансировки загрузки тесно связана с задачей достижения консенсуса [2]. В [3] был предложен протокол локального голосования для задачи достижения консенсуса в сети агентов с нелинейной динамикой, в условиях изменяющейся топологии, помех и задержек. Задача балансировки загрузки была преобразована в задачу достижения консенсуса в динамических сетях [4]. В [4] рассмотрена возможность применения протокола локального голосования для децентрализованной балансировки загрузки в сети с изменяющейся топологией, с учетом помех и задержек при условии поступления всех заданий в систему в единый начальный момент времени.

В предлагаемой статье изучается возможность создания универсального программного обеспечения для определения параметров и условий достижения консенсуса на основе локального правила голосования.

Задача балансировки загрузки сети

Для рассмотрения задачи балансировки загрузки сети необходимо:

- определить количество n узлов сети;
- знать, каким образом осуществлено соединение пар узлов — одно- или двунаправленной связью;
- определить, на каждом ли узле есть очередь заданий на исполнение;
- для каждого задания узел должен решить: выполнить ли задание самому или отправить соседу;
- каким образом организовать принятие решений, чтобы минимизировать время выполнения последнего задания.

Такие задачи достаточно полно определяют современные вычислительные системы [1; 2], в которых предполагается потоковая передача данных.

Недостатком указанных систем может стать время, затрачиваемое на передачу данных. Оно может быть соизмеримо со временем, затрачиваемым на обработку задач, или превышать его.

Формально эту задачу можно расписать так:

- сеть состоит из n узлов;
- $q_i \in R^+$ — начальный уровень загрузки i-го узла, $q = (q_1, q_2, \ldots, q_n)$ — вектор загрузки;
- связи между узлами имеют ограниченную пропускную способность. Будем считать, что между двумя узлами может существовать только одно ребро. Обозначим через $c_{ij} \in R$ пропускную способность передачи информации от узла i узлу j, $c_{ij} \geq 0$, причем строгое неравенство имеет место только в случае, когда между i, j существует канал связи;
- $p_i \in R^+$— производительность i-го узла, при этом $p = (p_1, p_2, \ldots, p_n)$ — вектор производительности;
- $u = (u_{ij})$, $(i,j) \in E$, нумерация вектора u согласована с матрицей инцидентности B, u_{ij} — количество передаваемой нагрузки в единицу времени по каналу $(i \to j)$. Другими словами, количество нагрузки, переданной по каналу $(i \to j)$ за интервал времени $[\tau_1, \tau_2]$, определяется как $\int_{\tau_1}^{\tau_2} u_{ij}(t)dt$ или же просто $(\tau_2 - \tau_1)u^{ij}$, если u не изменяется со временем.

Выполнение всех заданий за минимальное время можно сформулировать в виде следующей задачи:

минимизировать τ при условии

$$\begin{cases} \dot{x} = Bu - \tilde{p}, \\ x(0) = q;\ x(\tau) = 0, \\ x(t) \geq 0, \\ 0 \leq \tilde{p}_i(t) \leq p_i, \\ 0 \leq u_{ij}(t) \leq c_{ij}(t). \end{cases}$$

В данной формулировке переменными являются u и $\tilde{p}$, при этом $\tilde{p}$ — фактическая производительность, сводится к системе

$$\begin{cases} \dot{x} = Bu, \\ x(0) = x^{-};\ x(\tau) = x^{+}, \\ x(t) \geq 0, \\ 0 \leq u_e(t) \leq c_e. \end{cases}$$

Пусть V — множество всех узлов, $E \in V \times V$ — множество каналов связи между узлами, пропускная способность которых есть c_{ij}. Дополним этот граф фиктивной вершиной t и дугами (v, t), $v \in V$ с пропускной способностью p_i. Этот прием позволяет трактовать процесс выполнения задачи как передачу данного 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, т. е. является задачей с начальным и конечным состояниями

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

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

В последнее время все чаще используются распределенные системы, для которых актуальна задача распределения загрузки. Обычно при распределении заданий по узлам приветствуется равномерная загрузка.

На практике используются два подхода к динамической балансировке загрузки — централизованный и распределенный (децентрализованный). При *централизованном* подходе определяется некий ресурс, аккумулирующий информацию о состоянии всей сети и принимающий решения о распределении заданий для каждого из узлов [5]. При *распределенном* подходе алгоритмы балансировки загрузки выполняются на всех узлах, обменивающихся информацией о состояниях с другими узлами сети. Перераспределение происходит только между соседними узлами.

Представим модель сети агентов (узлов), выполняющих параллельно однотипные задания, в которой допускается перераспределение заданий между агентами на основе обратных связей.

Рассмотрим $N = \{1, \ldots, n\}$ — набор интеллектуальных агентов (узлов), каждый из которых выполняет поступающие задания по принципу очереди. Задания поступают в систему в различные дискретные моменты времени $t = 1, 2, \ldots, T$ на разные узлы. Связь между узлами определяется топологией динамической сети.

В момент времени t поведение каждого агента $i \in N$ описывают две характеристики:

- q_t^i — загруженность, или длина очереди из атомарных элементарных заданий узла i в момент времени t;
- r_t^i — производительность узла i в момент времени t.

При достаточно общих предположениях можно считать, что динамика изменений загруженности агентов описывается следующими уравнениями:

$$q_{t+1}^i = q_t^i - r_t^i + z_t^i + u_t^i;\ i \in N, t = 0, 1, \ldots, T,$$

где $u_t^i \in R$ — управляющие воздействия, которые в момент времени t оказывают влияние на узел i; z_t^i — размер нового задания, поступившего на узел i в момент времени t.

Если $N_t^i \neq 0$, то в момент времени t узел i получает данные о производительности соседних узлов $r_t^i, r_t^j, \forall i \in N, j \in N_t^i$ и ведет наблюдения об их загруженности с учетом зашумления:

$$y_t^{i,j} = q_{t-d_t^{i,j}}^i + \omega_t^{i,j},\ j \in N_t^i,$$

где $\omega_t^{i,j}$ — помехи, а $0 \leq d_t^{i,j} \leq \overline{d}$ — целочисленная задержка ($\overline{d}$ — максимально возможная задержка). Кроме того, в каждый момент времени t узел i имеет зашумленные данные о своей загруженности:

$$y_t^{i,j} = q_t^i + \omega_t^{i,j}.$$

При обеспечении равномерной загрузки всех узлов сети рассматриваются две постановки задачи: стационарная и нестационарная.

При *стационарной* постановке все задания поступают в систему на разные узлы в начальный момент времени, тогда как при *нестационарной* постановке задачи новые задания могут поступать в систему на любой из i узлов в различные моменты времени t.

Для сокращения времени выполнения всех заданий необходимо использовать протокол перераспределения заданий с течением времени. Это позволит увеличить пропускную способность системы и уменьшить время выполнения заказов в системе. В стационарном случае, с одной стороны, если с момента времени t все задания выполняются только теми агентами, к которым они поступили, то время реализации всех заданий определяется как

$$T_t = \max_{i \in N} \frac{q_t^i}{r_t^i}.$$

С другой стороны, желаемое время работы всей системы может составить

$$T_{\min} = \frac{\sum_{i=1}^{n} q_0^i}{\sum_{i=1}^{n} r_0^i},$$

если все узлы работают одновременно, выполняя общий объем заданий.

Метод достижения консенсуса

Широкое применение мультиагентных технологий привлекает все большее число исследователей к задачам управления и распределенного взаимодействия в сетях динамических систем [6]. Однако решение таких задач не всегда является удовлетворительным из-за сложности обмена информацией и ее полноты, а также наличия задержек и помех.

Для решения задачи достижения консенсуса группой взаимодействующих агентов, обменивающихся информацией, часто применяются алгоритмы типа стохастической аппроксимации со стремящимся к нулю размером шага [7]. При постоянно изменяющихся внешних состояниях агентов с течением времени алгоритмы стохастической аппроксимации с бесконечно малым шагом неработоспособны. Поэтому актуальной является задача исследования свойств алгоритма типа стохастической аппроксимации при малом постоянном, но не бесконечно малом размере шага в случае нелинейной постановки задачи, в условиях случайно изменяющейся структуры связей в сети при действии помех.

Рассмотрим динамическую сеть применительно к теории графов из набора агентов (узлов) $N = \{1, \ldots, n\}$. Граф (N, E) определяется набором агентов N и множеством ребер или дуг E. Множеством соседей узла i называется $N^i = \{j: (j,i) \in E\}$, т. е. множество узлов с ребрами, входящими в i. Поставив в соответствие каждому ребру $(j, i) \in E$ вес $a^{i,j} > 0$, определим матрицу смежности (или связности) $A = [a^{i,j}]$ графа, обозначаемого далее G_A (верхние индексы у переменных показывают соответствующие номера узлов). Взвешенная полустепень захода вершины i определяется как сумма элементов i-й строки матрицы A:

$$d^i(A) = \sum_{j=1}^{n} d^{i,j}.$$

Каждому агенту $i \in N$ в момент времени $t = 1, 2, \ldots, T$ ставится в соответствие изменяющееся во времени состояние $x_t^i \in R$. Динамика таких состояний описывается разностными уравнениями

$$x_{t+1}^i = x_t^i + f^i\left(x_t^i, u_t^i\right)$$

с некоторыми функциями $f^i(\cdot,\cdot): R \times R \to R$, зависящими от состояний в предшествующий момент времени x_t^i и управляющих воздействий $u_t^i \in R$, которые в момент времени t оказывают влияние на узел i.

Пусть мультиагентная система состоит из динамических агентов со входами u_t^i, выходами $y_t^{i,t}$ и состояниями x_t^i.

Узлы i и j называются *согласованными* в сети в момент времени t тогда и только тогда, когда $x_t^i = x_t^i$. Для достижения консенсуса требуется согласовать все узлы между собой, т. е. найти такой протокол управления, который переводит все состояния в одно и то же постоянное значение $x^a = x^i$, $\forall i \in N$.

Предположим, что структура связей динамической сети моделируется при помощи последовательности ориентированных графов $\{(N, E)\}_{t \geq 0}$, где $E_t \subset E$ изменяются во времени. Если $(j, i) \in E_t$, то узел i в момент времени t получает информацию от узла j для целей управления с обратной связью. Обозначим A_t — соответствующие E_t матрицы смежности; $E_{\max} = \left\{(j,i): \sup_{t \geq 0} a_t^{i,j} > 0\right\}$ — максимальное множество каналов связи. Для формирования правила управления каждый узел $i \in N$ имеет информацию о своем собственном состоянии (может быть, и зашумленную)

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

и, если $N_t^i \neq \varnothing$, зашумленные наблюдения о состояниях соседних узлов

$$y_t^{i,j} = x_{t-d_t^{i,j}}^j + w_t^{i,j},\ j \in N_t^i,$$

где $w_t^{i,i}, w_t^{i,j}$ — помехи, а $0 \leq d_t^{i,j} \leq \overline{d}$ — задержка ($\overline{d}$ — максимально возможная задержка). Так как система начинает работу при $t = 0$, неявное требование к множеству соседних узлов: $j \in N_t^i \Rightarrow t - d_t^{i,j} \geq 0$. При $w_t^{i,j} = 0$ для всех остальных пар i, j, определим $\overline{w}_t \in R^{n^2}$ — вектор, составленный из элементов $w_t^{i,j}$, $i, j \in N$.

Алгоритм управления, называемый *протоколом локального голосования*, задается формулами:

$$u_t^i = a_t \sum_{j \in \overline{N}_t^i} b_t^{i,j}\left(y_t^{i,j} - y_t^{i,i}\right),$$

где $a_t > 0$ — размеры шагов протокола управления, $b_t^{i,j} > 0, \forall j \in \overline{N}_t^i$. Принимая $b_t^{i,j} = 0$ для всех остальных пар i, j, определяем $B_t = \left[b_t^{i,j}\right]$ — матрицу протокола управления.

Пусть Ω, F, P — основное вероятностное пространство. Пусть E — математическое ожидание и E_x — условное математическое ожидание при условии x.

Основные условия, при которых протокол локального голосования обеспечивает достижение консенсуса, следующие:

1. $\forall i \in$ N функции $f^i(x, u)$ липшицевы по x и u:

$$\left|f^i(x,u) - f^i(x',u')\right| \leq L\left(L_x\left|x - x'\right| + \left|u - u'\right|\right),$$

и при любом фиксированном x функции $f^i(x,\cdot)$ таковы, что $E_x f^i(x,u) = f^i(x, E_x u)$.

2. $\forall i \in N$, $j \in N^i \cup \{i\}$ помехи наблюдений $w_t^{i,j}$ — центрированные, независимые, одинаково распределенные случайные величины с ограниченными дисперсиями: $E\left(w_t^{i,j}\right)^2 \le \sigma_w^2$.

3. $\forall i \in N, j \in N^i$ появление переменных ребер (j, i) в графе G_{A_t} — независимые, случайные события, вероятность которых $p_a^{i,j}$ (т. е. получены независимые, одинаково распределенные случайные матрицы).

4. $\forall i \in N, j \in N^i$ веса $b_t^{i,j}$ в протоколе управления — ограниченные, случайные величины: $\underline{b} \le b_t^{i,j} \le \overline{b}$ с вероятностью 1, и существуют пределы $b^{i,j} = \lim\limits_{t\to\infty} Eb_t^{i,j}$.

5. Существует конечная величина $\overline{d} \in N : \forall i \in N$, $j \in N^i d_t^{i,j} \le \overline{d}$ с вероятностью 1 и целочисленные задержки $b_t^{i,j}$ — независимые, одинаково распределенные случайные величины, принимающие значения $k = 0, ..., \overline{d}$ с вероятностями $p_k^{i,j}$.

Кроме того, все эти случайные величины и матрицы независимы между собой, и их элементы имеют ограниченные дисперсии.

Обозначим $\overline{n} = n\left(\overline{d}+1\right)$ и определим матрицу $A_{\max}$ размером $\overline{n} \times \overline{n}$ по правилу:

$$a_{\max}^{i,j} = p_{j\operatorname{div}\overline{d}}^{i,((j-1)\bmod n)+1} p_a^{i,((j-1)\bmod n)+1} b^{i,((j-1)\bmod n)+1}.$$

Здесь операция mod — остаток от деления, а div — деление нацело.

Будем считать, что для матрицы структуры связей сети $A_{\max}$ выполняется следующее условие:

граф (N, $E_{\max}$) имеет остовное дерево, и для любого ребра (j, i) $\in E_{\max}$ среди элементов $a_{\max}^{i,j}, a_{\max}^{i,j+n}, ..., a_{\max}^{i,j+\overline{d}n}$ матрицы найдется хотя бы один ненулевой.

Обозначим $\overline{x}_t = \left[x_t^1, ..., x_t^n\right]$ соответствующий вектор-столбец, полученный вертикальным соединением n чисел. Положим $\overline{X}_t = 0$ для $-\overline{d} \le t < 0$ и определим $\overline{X}_t \in R^{n\overline{d}}$ — расширенный вектор состояний $\overline{X}_t = \left\lfloor \overline{x}, \tilde{x}_{t-1}, ..., \tilde{x}_{t-\overline{d}} \right\rfloor$, где $\tilde{x}_{t-k}$ — вектор, состоящий из таких x_{t-k}^i, что, $\exists j \in N^i \exists k' \ge k : p_{k'}^{i,j} > 0$, т. е. это значение с некоторой положительной вероятностью участвует в формировании хотя бы одного из управляющих воздействий. В дальнейшем для простоты будем считать, что так введенный расширенный вектор состояний равен $\overline{X}_t = \left\lfloor \overline{x}, \tilde{x}_{t-1}, ..., \tilde{x}_{t-\overline{d}} \right\rfloor$, т. е. в него входят все компоненты со всевозможными задержками, не превосходящими $\overline{d}$.

Из-за наличия помех, задержек, неопределенностей в протоколе управления и переменной структуры связей точного консенсуса достичь достаточно сложно. Поэтому мы будем рассматривать задачу о достижении приближенного консенсуса — ε-консенсуса.

Алгоритм моделирования достижения консенсуса

Рассмотрим пример системы, состоящей из вычислительных узлов. Зададим начальные значения длин очереди и производительностей узлов. Предположим, что производительности не изменяются с течением времени.

Разработанный алгоритм имеет следующие характеристики:

- дискретный (так как алгоритм состоит из отдельных шагов – команд);
- однозначный (каждая из команд описывает лишь одно возможное действие пользователя);
- понятный (каждая из команд входит в систему команд пользователя);
- результативный (пользователь достигает решения задачи за конечное число шагов).

Алгоритм работы программного обеспечения включает в себя четыре этапа:

1) ввод исходных значений количества агентов N и максимального времени $T_{\max}$ наблюдения для создания матриц, предназначенных для хранения результатов расчета на каждый промежуток времени от нуля до $T_{\max}$ для каждого агента от 1 до N;

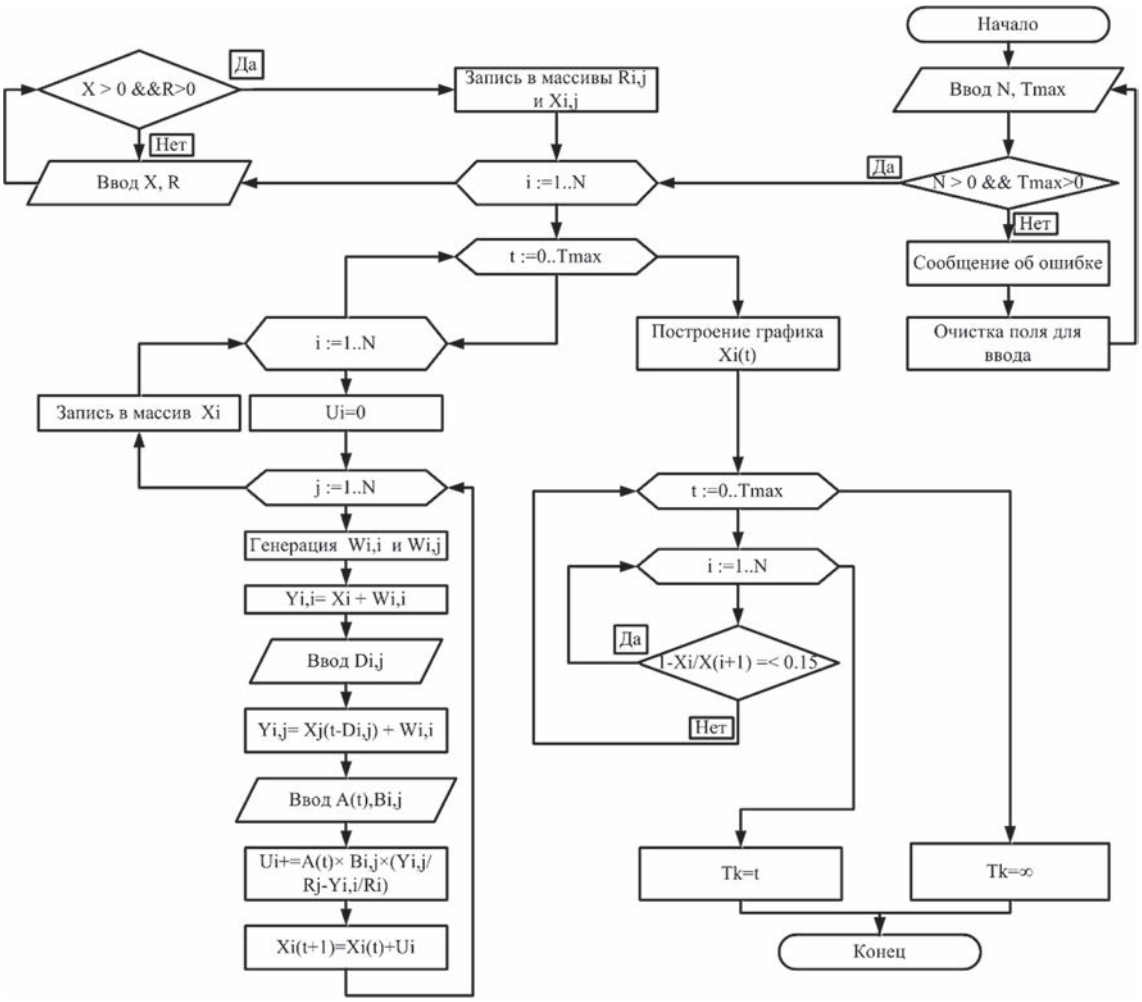
2) ввод начальных значений производительности R и начального состояния X для каждого агента, а также запись их в массив;

3) расчет текущего состояния X для каждого агента за все периоды;

4) принятие решения о времени достижения консенсуса.

Алгоритм (см. рисунок) не предполагает объявление и инициализацию всех переменных, участвующих в решении задачи, до начала первого этапа. Это объясняется тем, что при моделировании процесса достижения консенсуса количество агентов, определяющих размер массивов для сохранения промежуточных решений, задается вручную пользователем, т. е. является динамичным и заранее неизвестным. Поэтому по ходу решения задачи определяется (генерируется случайным образом или вводится пользователем) количество отдельных величин, участвующих в формуле нахождения текущего состояния агента.

На ***1-м этапе*** пользователь должен ввести исходные значения количества агентов N и максимального времени $T_{\max}$ наблюдения. Так как обе эти величины должны быть больше нуля, происходит проверка введенных числовых значений на положительность (графические поля для ввода предусматривают невозможность ввода нечисловых символов). Если числа не положительные, появляется сообщение об ошибке и повторном вводе. Соответствующее неверному значению поле ввода очищается. Данные величины определяют размер массивов для записи рассчитываемых значений в каждый момент времени для всех агентов.



Модифицированный алгоритм достижения консенсуса

На ***2-м этапе*** пользователь вводит исходные значения производительности R и начального состояния X для каждого агента в начальный момент времени ($t = 0$). Так как эти величины также должны быть больше нуля, происходит проверка их на положительность (графические поля для ввода также предусматривают невозможность ввода нечисловых символов). При корректном введенном значении производительности R и начального состояния X эти величины записываются в соответствующий массив. Размерность массива определяется как $X \cdot T_{\max}$, нумерация массива начинается с нуля (период времени отсчитывается от начального до максимального).

На ***3-м этапе*** программа приступает к расчету текущего состояния X для каждого агента. Создается переменная U_i для хранения значения управляющих воздействий. Для формирования стратегии управления $Y_{i,i}$ каждый агент имеет информацию о своем собственном состоянии $X_{i,i}$, которое может быть зашумлено в результате помех $W_{i,i}$, генерируемых случайным образом для каждого агента в каждый момент времени. Наконец, необходимо определить значение целочисленной задержки $D_{i,j}$, с которой каждый агент должен получить информацию о зашумленном состоянии соседей $Y_{i,i}$. Далее определяются размеры шагов протокола управления $A(t)$ и значение коэффициента протокола $B_{i,j}$, и программа вычисляет значение управляющих воздействий U_i и состояния в следующий момент времени X_{i+1}, которое записывается в массив $X_i(t)$.

После вычисления всех значений текущего состояния X для всех агентов на ***4-м этапе*** по полученным результатам строится график, моделирующий поведение совокупности агентов в течение заданного времени $T_{\max}$. Состояние каждого агента представляется отдельной кривой, значение которой вычисляется для каждого отсчета времени без аппроксимации. Программа, перебирая все отрезки времени, находит тот, в котором состояния агентов стремится к достижению консенсуса.

Условие схождения состояний агентов к консенсусу будем считать таким, при котором отношение текущего состояния любого агента ко всем текущим состояниям оставшихся агентов не превосходит 15%. Такое условие вытекает из предварительных математических расчетов и результатов имитационного моделирования. Первое определенное значение времени, удовлетворяющее

за данному условию, становится временем прихода к консенсусу. В противном случае считается, что в заданных условиях агенты не смогут достичь консенсуса и необходимо изменить исходные данные для получения положительного ответа.

Заключение

Рассмотрены задачи балансировки загрузки сети и достижения консенсуса в децентрализованных системах при переменной структуре связей, помехах и задержках. Описано применение протокола локального голосования для задачи схождения к консенсусу. Предложенный алгоритм может послужить основой для создания программного обеспечения, позволяющего исследовать предельные параметры агентов (узлов) при достижении консенсуса.

Список использованной литературы

*1. **Скобелев П. О.** Мультиагентные технологии в промышленных применениях: к 20-летию основания Самарской научной школы мультиагентных систем // Мехатроника, управление, автоматизация. 2011. № 12. С. 33–46.*

*2. **Амелина Н. О.** Исследование консенсуса в мультиагентных системах в условиях стохастических неопределенностей // Вестник ННГУ. 2013. № 1 (3). С. 173–179.*

*3. **Huang M.** Stochastic approximation for consensus: a new approach via ergodic backward products// IEEE Transactions on Automatic Control. 2012. Vol. 57, No. 12. P. 2994–3008.*

*4. **Амелина Н. О., Фрадков А. Л.** Приближенный консенсус в стохастической динамической сети с неполной информацией и задержками в измерениях // Автоматика и телемеханика. 2012. № 11. С. 6–30.*

*5. **Граничин О. П.** Стохастическая оптимизация и системное программирование // Стохастическая оптимизация в информатике. 2010. Т. 6. С. 3–44.*

*6. **Амелина Н. О.** Мультиагентные технологии, адаптация, самоорганизация, достижение консенсуса // Стохастическая оптимизация в информатике. 2011. Т. 7, вып. 1. С. 149–185.*

*7. **Kar S., Moura J. M. F.** Distributed consensus algorithms in sensor networks with imperfect communication: link failures and channel noise // IEEE Trans. Sig. Process. 2009. Vol. 57, №. 1. P. 355–369.*

Рецензент: доктор техн. наук, профессор **В. В. Вишневский,** Государственный университет телекоммуникаций, Киев.

Ю. А. Дуйнова, Н. І. Лущик, С. І. Половеня, Ю. В. Мельник

МОДИФІКОВАНИЙ АЛГОРИТМ ДОСЯГНЕННЯ КОНСЕНСУСУ НА ОСНОВІ ЛОКАЛЬНОГО ПРАВИЛА ГОЛОСУВАННЯ

Розглянуто алгоритм досягнення консенсусу в децентралізованій мережі із завадами як основу для створення спеціалізованого програмного забезпечення. Наведено задачі балансування завантаження мережі, а також задачі досягнення консенсусу групою агентів, що взаємодіють між собою.

Ключові слова: досягнення консенсусу; децентралізовані системи; балансування завантаження

Yu. A. Duynova, N. I. Lushchik, S. I. Polovenya, Yu.V. Melnik

MODIFIED ALGORITHM FOR ACHIEVING OF CONSENSUS ON THE BASIS OF LOCAL VOTING RULE

It is noted that decentralized networks are now widely used. There is an important task of balancing the network load, which is to maintain a uniform load of all agents in time. The possibility of using the local voting protocol for decentralized load balancing in the network is shown. This may change the topology and take into account interference and delays in the receipt of all tasks in the system. The article explores the possibility of creating a universal software for determining the parameters and conditions for achieving consensus based on a local voting rule. The conditions that must be taken into account when considering the load balancing problem are determined. The task of achieving consensus in a minimal time is formalized. A model of a network of agents (nodes) is presented that perform parallel tasks of the same type and in which the redistribution of tasks between agents on the basis of feedbacks is allowed. It is noted that with the distributed approach, load balancing algorithms are executed on all nodes. All nodes exchange status information with other nodes on the network. Redistribution occurs only between neighboring nodes. The stationary and nonstationary statements of the problem are considered with ensuring the uniform loading of all nodes of the network. The relevance of studying the properties of an algorithm of the stochastic approximation type under conditions of a randomly changing structure of bonds in a network under the action of interference is noted. The scheme of the developed control algorithm which is called the protocol of local voting is presented. The main conditions under which the protocol of local voting ensures the achievement of consensus is given. The characteristics and content of the stages of the developed algorithm are presented. The work of each stage is explained. It is noted that the proposed algorithm can be the basis for creating software that will allow to investigate the limiting parameters of agents when consensus is reached.

Keywords: achievement of consensus; decentralized systems; load balancing.