МОДЕЛИРОВАНИЕ ЦИФРОВЫХ УПРАВЛЯЮЩИХ СИСТЕМ С УЧЕТОМ ДВОЙНОГО ВРЕМЕННОГО ШКАЛИРОВАНИЯ

Степанченко О.В.

(г. Камышин Волгоградской области)

Рассматриваются принципы моделирования дискретных систем с разнотемповыми составляющими для переменных задающих воздействий и на их основе предлагается новый вариант алгоритмического обеспечения двухканального линейного цифрового пропорционально-интегрального (ПИ) закона управления, обладающего свойством двойной шкалы времени («двушкальный ПИ-регулятор»).

THE MODELING OF DISCRETE CONTROL SYSTEM WITH DOUBLE-TIME-SCALE

Stepanchenko O.V.

(Kamyshin city, Volgograd region)

The issue deals with the algorithms design for the discrete plants with multi-rate state variables which may be separated to the subprocesses with low and fast modes correspondingly. The control algorithm synthesis is based on the two- time- scale control method. The results of DC double feeding electric driver illustrate the developed techniques. The problems of detection of the multi-rate effects presence are discussed as well.

Переход техники управления на цифровую основу не только открывает возможности усовершенствования алгоритмов управления, но и ставит перед разработчиками систем управления задачу повышения эффективности использования ресурсов вычислительной (в частности, микропроцессорной) техники. Одним из направлений решения этой важной задачи является поиск упрощающих допущений на этапе постановки задачи

управления, позволяющих получить не только более простые и быстродействующие алгоритмы без заметного ухудшения качества управления, но и улучшить вычислительную процедуру, их реализующую.

Источником возможных упрощений постановок задач алгоритмизации является учет особенностей структуры и свойств объектов управления. В работе исследуется одно из направлений учета свойств некоторых распространенных объектов, основанное на выделении существенно различных по показателям инерционности (т.е. разнотемповых) составляющих (субпроцессов) в автоматизируемом технологическом процессе.

Актуальность темы в решении прикладных задач подтверждается широкой распространенностью технологических процессов, обладающих разнотемповыми составляющими, например:

- электродвигатели постоянного тока, изменение скорости вращения которых характеризуется гораздо большей инерционностью, чем изменение тока в якорной цепи;
- аппараты химической промышленности в которых изменение характеристик катализатора имеет гораздо большую инерционность, чем процесс производства продуктов.

В работе рассматриваются следующие задачи:

1. Анализ особенностей задач управления процессами с разнотемповыми составляющими и разработка методики моделирования систем управления такими процессами на основе введения двойной шкалы времени.
2. Разработка алгоритмического обеспечения двухконтурного дискретного пропорционально-интегрального (ПИ) регулятора с двойной шкалой времени.
3. Оценка эффективности предлагаемых алгоритмов при решении практической задачи управления процессом стабилизации скорости вращения электродвигателя постоянного тока.

Впервые разработаны принципы моделирования дискретных систем с разнотемповыми составляющими для переменных задающих воздействий; разработаны показатели и критерии, позволяющие оценить параметры, необходимые для декомпозиции модели процесса на медленный и быстрый подпроцессы, а

также определить потери точности математического описания, возникающие при использовании декомпозиции.

На основе разработанных принципов предложен новый вариант алгоритмического обеспечения двухканального линейного цифрового пропорционально-интегрального (ПИ) закона управления, обладающего свойством двойной шкалы времени («двушкальный ПИ-регулятор») и благодаря этому применимого для использования в системах управления разнотемповыми технологическими процессами.

Предложен способ оценки эффективности использования предлагаемых методов декомпозиции процесса управления на основе сопоставления показателей, характеризующих инерционность быстрой составляющей, с показателями, характеризующими частотные свойства задающих воздействий.

Существо метода двойной шкалы времени состоит в выделении в процессах, происходящих в системе, медленных и быстрых составляющих (причем точность метода тем выше, чем больше различие в темпе протекания быстрых и медленных процессов в системе). [1]

1. Предполагается, что модель дискретного процесса в пространстве состояний имеет вид:

$$x[s+1] = Ax[s] + Bu[s] + Cf[s], \qquad (1)$$

причем специфика модели такова, что в n–мерном векторе состояний x[s] могут быть выделены блоки $x_1[s]$ (медленная составляющая) и $x_2[s]$ (быстрая составляющая) размерности n_1 и n_2 соответственно, характеризующиеся существенными различиями в инерционности: $x[s]^T=[x_1[s]^T_{n1\times1} \mid x_2[s]^T_{n2\times1}]$, $n_1+n_2=n$;

В (1) u[s] – r–мерный вектор управляющих воздействий, которые действуют одновременно на оба субвектора $x_1[s]^T$ и $x_2[s]^T$; f[s] – m–мерный вектор возмущающих воздействий; s – такты дискретного времени.

Согласно разбиению вектора состояния на субблоки матрицы A, B, C в (1) представляются в блочной форме:

$$A = \begin{bmatrix} A_{11} & \vdots & A_{12} \\ \cdots & \vdots & \cdots \\ A_{21} & \vdots & A_{22} \end{bmatrix}$$ – где диагональные блоки (размерности

$n_1 \times n_1$ и $n_2 \times n_2$ соответственно) соответствуют локальным описаниям субпроцессов $x_1[s]$ и $x_2[s]$, а внедиагональные (размерности $n_1 \times n_2$ и $n_2 \times n_1$ соответственно) используются для описания взаимодействий между медленной и быстрой составляющими.

$B = \begin{bmatrix} B_1 \\ \cdots \\ B_2 \end{bmatrix}$ – блочная матрица (размерности блоков $n_1 \times r$ и $n_2 \times r$ соответственно), описывающая влияние управления на объект.

$C = \begin{bmatrix} C_1 \\ \cdots \\ C_2 \end{bmatrix}$ – блочная матрица (размерности блоков $n_1 \times m$ и $n_2 \times m$ соответственно), описывающая влияние возмущающих воздействий на объект.

2. Для описания медленной и быстрой составляющих вводятся два различных интервала дискретизации непрерывного времени, связанные соотношением

$\delta t = \mu \cdot \Delta t,$

причем $\mu << 1$.

3. В шкале «медленного» времени, такты которого s = 0,1,... отсчитываются с дискретой Δt, предполагается, что быстрая составляющая приближенно может быть заменена статической моделью, не содержащей инерционности:

$$x_2[s] = A_{21} \cdot x_1[s] + A_{22} \cdot x_2[s] + B_2 \cdot u[s] + C_2 f[s], \qquad (2)$$

благодаря чему общая размерность описания процесса в «медленной» шкале времени снижается до $n_1 < n$.

4. Между тактами [s — 1] и s вводится шкала «быстрого» времени, такты которой $\tau = 1, \ldots, \Delta t / \delta t$ отсчитываются с дискретой δt. При этом считается, что динамику быстрой составляющей можно рассчитать в предположении о том, что между s—1-м и s-м тактами медленная составляющая не изменяется и соответствует значению $x_1[s-1]$, т.е. имеет смысл постоянного задающего воздействия для быстрой составляющей. Благодаря этому размерность управления быстрым процессом снижается до $n_2 < n$.

Далее приведены результаты моделирования двушкальной

системы основные из которых следующие:

1. Качество приближения исходного медленного субпроцесса $x_1[s]$ к его модели $x_{1\_приб}[s]$, в системе с сепарированными медленной и быстрой составляющими (рис.1), при величине зазора более 0.1 является очень высоким. Потери точности практически не происходит.

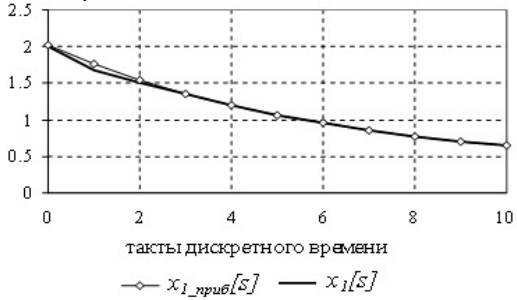

Рис. 1.

2. В точках отсчета медленного времени значение быстрой составляющей, рассчитывается по приближенной формуле

$$x_{2\_приб}[s] = -P_б \cdot x_{1\_приб}[s] + V_б \cdot u[s], \qquad (3)$$

где $P_б$ – поправочная матрица, размерности $n_2 \times n_1$, корректирующая выделенный по физическим соображениям быстрый подпроцесс;

$V_б$ – матричный сомножитель перед управляющими воздействиями для быстрой составляющей.

Матрица $P_б$ находится из уравнения

$$P_б \cdot A_{11} - A_{22} \cdot P_б - P_б \cdot A_{12} \cdot P_б = -A_{21}. \qquad (4)$$

Поскольку уравнение (4) нелинейно относительно матрицы $P_б$, то для его решения был выбран итеративный путь, по аналогии с матричными уравнениями Риккати.

Матричный сомножитель $V_б$ рассчитывается по формуле

$$V_б = (E - A_б)^{-1} \cdot B_б,$$

где E – единичная матрица размерности $n_2 \times n_2$;

$A_б = A_{22} + P_б \cdot A_{12}$, $B_б = P_б \cdot B_1 + B_2$ – матрицы параметров быстрого процесса.

Как видно из рис.2 приближенное значение быстрой состав-

ляющей практически точно совпадает со значениями реального процесса.

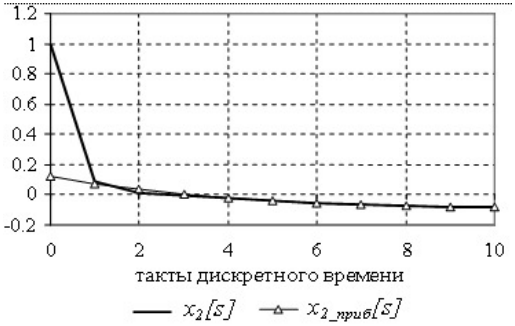

Рис. 2.

1. В точках отсчета быстрого времени, непосредственно прилегающих к точкам отсчета медленного времени (т.н. пограничном слое), точность представления быстрой составляющей моделью (3) утрачивается (рис.3).
2. Для увеличения точности вводится динамическая модель управления быстрой составляющей с псевдозадающим воздействием соответствующим значению медленной составляющей в момент медленного времени (s—1)

$$x[\tau] = A[\tau-1]\cdot x[\tau-1] + B[\tau-1]\cdot u[\tau-1] + A_{21}\cdot x_1[s-1], \qquad (5)$$

где A[τ], B[τ] – матрицы параметров объекта в быстром времени.

В начале пограничного слоя (рис.4) достигается существенное увеличение точности, затем безынерционная (3) и динамическая (5) модель выравниваются, а в конце пограничного слоя ближе к истинной становится безынерционная оценка (рис. 5). Таким образом можно определить шаг быстрого времени, начиная с которого быстрая составляющая может быть представлена приближенной моделью (3).

Рис. 3.

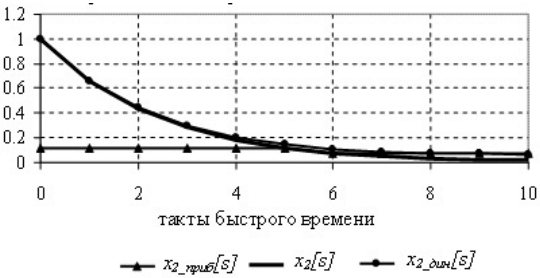

Рис. 4.

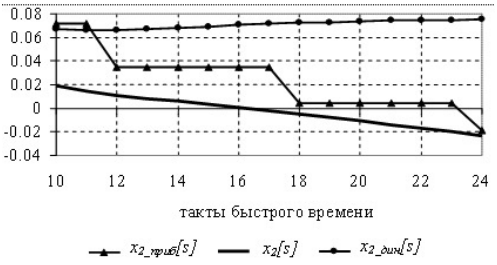

Рис. 5.

Приводятся результаты, позволяющие агрегировать исходные физические компоненты $x_1[s]$ и $x_2[s]$ в сепаратные быструю ($x_Б[s]$) и медленную ($x_M[s]$) составляющие с помощью соотношения агрегирования:

$$\begin{bmatrix} x_M[s] \\ x_Б[s] \end{bmatrix} = \begin{bmatrix} E_{n_1 \times n_1} - Q_Б \cdot P_Б & -Q_Б \\ P_Б & E_{n_2 \times n_2} \end{bmatrix} \cdot \begin{bmatrix} x_1[s] \\ x_2[s] \end{bmatrix},$$

где $Q_Б$ – поправочная матрица размерности $n_1 \times n_2$, корректирующая выделенный по физическим соображениям медленный

подпроцесс.

Матрица $Q_б$ находится из уравнения

$$(A_{11} - A_{12} \cdot P_б) \cdot Q_б - Q_б \cdot (A_{22} + P_б \cdot A_{12}) = -A_{12} .$$

Соотношения быстрой и медленной шкал определяется с помощью параметра μ, который может быть рассчитан как отношение норм матриц A_{11} и A_{22}

$$\mu = \|A_{22}\| / \|A_{11}\| .$$

В качестве примера, иллюстрирующего моделирование двушкальных процессов, рассмотрена задача стабилизации скорости вращения вала электродвигателя постоянного тока двойного питания (медленная фаза) и тока якорной цепи (быстрая фаза). Управляющими воздействиями являются напряжение питания и магнитный поток обмотки возбуждения. На рис.6 представлены результаты сравнения точных значений переменных состояния (скорости вращения вала двигателя Ω (а)) и тока в якорной цепи I (б)) с результатами расчета по моделям пониженной размерности с сепаратным управлением быстрой и медленной фазой.

рис. 6(а)

рис. 6(б)

Литература.

1. Фрадков А. Л. Разделение движений в адаптивных системах управления. М.: 1985.

АЛГОРИТМ ВЫБОРА ВЫЧИСЛИТЕЛЬНОГО РЕСУРСА В ГЕТЕРОГЕННОЙ РАСПРЕДЕЛЕННОЙ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЕ

Степанченко И.В.

(г. Камышин Волгоградской обл.)

В работе проводится конструирование алгоритма выбора вычислительного ресурса в гетерогенной распределенной вычислительной системе. Алгоритм основывается на обработке статистической информации о работе сервисов и разбиении подзадач на обязательные и второстепенные. Строится функция ошибок, которая минимизируется. Приводятся результаты исследований алгоритма для различных параметров вычислительной системы.

SELECTION ALGORITHM OF THE COMPUTING SERVICE OF THE DISTRIBUTED HETEROGENEOUS COMPUTING SYSTEM

Stepanchenko I.V.

(Kamyshin city, Volgograd region)

The issue deals with the selection algorithm synthesis of the computing service of the heterogeneous distributed computing system. The algorithm bases on the statistical data processing and fragmentation of subtasks in mandatory and secondary subtasks. The minimizing error function formulates in the work. The research results of algorithm for various parameters of computing system are also presented.

Существует целый класс систем управления, в которых регуляторы изменяют свои параметры в процессе управления, чтобы оптимизировать систему с точки зрения поставленной цели управления – это, так называемые, самонастраивающиеся адаптивные системы. В этих системах структура регулятора выбира-

ется заранее, в процессе управления требуется определить лишь алгоритм его настройки.

В случае, когда невозможно получить функциональную зависимость между характеристиками системы и целью управления, используют поисковые самонастраивающиеся системы, в которых ищется минимум или максимум критерия поисковым алгоритмом. Простейшими поисковыми системами являются большинство экстремальных систем, в которых недостаток априорной информации восполняется за счет текущей информации, получаемой в виде реакции объекта на искусственно вводимые поисковые (пробные, тестовые) воздействия.

В данной работе рассматривается подход к адаптации ПИД-регулятора путем настройки его коэффициентов поисковыми алгоритмами в реальном времени в распределенной гетерогенной вычислительной системе. Из возможных алгоритмов адаптации выбраны: алгоритм Хука-Дживса, относящийся к классу адаптивных алгоритмов прямого поиска экстремума функции [1], и генетический алгоритм [2]. Данные алгоритмы представляют два полярных подхода к настройке параметров: критериальный (алгоритм Хука-Дживса) и переборный (генетический алгоритм).

Сущность алгоритма адаптивного прямого поиска Хука-Дживса. Предполагается, что критерий качества управления $f(m)$, выбранный разработчиком САУ в качестве основного, экстремально зависит от векторного аргумента $m = [m_1, m_2, \ldots m_v]^T$, причем аналитическое выражение для функции $f(m)$ не задано. В рассматриваемых задачах управления аргументами функции $f(m)$ являются настраиваемые параметры регулятора или системы управления.

Все методы прямого поиска укладываются в следующую общую схему:

- выбирается произвольная точка начального приближения $m^{(0)}$ и начальное значение шага $\Delta m^{(0)}$;
- процесс поиска экстремума организуется как итеративный.
- Каждый d-й шаг итеративного процесса представляет собой следующие действия:

- фиксируются все элементы вектора $m^{(d)}$, кроме одного, например, j-го: $[m_1^{(d)}, m_2^{(d)},..., m_{j-1}^{(d)}, m_{j+1}^{(d)},..., m_v^{(d)}]$ фиксированы;
- делаются пробные приращения вокруг $m_j^{(d)}$:
- $[m_j^{(d+1)}]^{(1)} = m_j^{(d)} + \Delta m^{(d)}$; $[m_j^{(d+1)}]^{(2)} = m_j^{(d)} - \Delta m^{(d)}$;
- находится направление изменения аргументов, которые доставляют «лучшие» значения функции (приближающие функцию к экстремуму) – выбирают
- $[m_j^{(d+1)}] \in \{[m_j^{(d+1)}]^{(1)}, [m_j^{(d+1)}]^{(2)}\}$
- такое, чтобы значение функции
- $f(m_1^{(d)}, m_2^{(d)},..., m_{j-1}^{(d)}, m_j^{(d+1)}, m_{j+1}^{(d)},..., m_v^{(d)})$
- стало лучше, чем
- $f(m_1^{(d)}, m_2^{(d)},..., m_{j-1}^{(d)}, m_j^{(d)}, m_{j+1}^{(d)},..., m_v^{(d)})$;
- последовательно выполняются те же действия для всех элементов вектора m, что позволяет «экспериментально» увидеть реакцию объекта на пробные возмущения;
- вычисляется оценка градиента в окрестности точки $m^{(d)}$ и находится направление движения к экстремуму.

Применение метода прямого поиска по описанному сценарию на практике встречается крайне редко, так как в процессе поиска алгоритм может либо «пропустить» экстремум (пробные шаги недопустимо велики), либо сходимость может быть очень медленной (пробные шаги недопустимо малы). Этот недостаток связан с тем, что в алгоритме нет информации о выборе шага. Кроме того, погрешности измерения при «пологой» форме функции могут исказить улучшающее направление (и алгоритм вообще не будет сходиться).

Поэтому чаще всего используют адаптивные методы прямого поиска. Их сущность сводится к тому, что используется информация о форме функции вблизи траектории движения к экстремуму, чтобы результаты прошлых итераций использовались для корректировки шага на текущей итерации.

Идеи адаптации алгоритма, предложенной Хуком и Дживсом:

- если предыдущий шаг привел к улучшению функции, следующий шаг увеличивается, движение к экстремуму ускоря-

ется;

- если предыдущий шаг привел к ухудшению функции, корректируется направление движения и длина шага берется с предыдущей итерации;
- если при пробных движениях во всех направлениях функция ухудшается (признак попадания в окрестность экстремума), то при недостигнутой заданной точности определения экстремума, уменьшается величина шага и процесс поиска повторяется, в противном случае поиск завершается.

Величина шага является настраиваемым значением в алгоритме поиска и может изменяться различными способами (например, значение шага может выбираться по формуле $\Delta m^{(d+1)} = \Delta m^{(d)} / e^{w}$, где w – число неудачных попыток определить улучшающее направления в последовательности итераций или кратным двум – в случае удачного направления в два раза больше, в случае неудачного – в два раза меньше).

В классической форме алгоритм Хука-Дживса не может быть эффективно распараллелен. Однако можно отказаться от движения к экстремуму после каждого «удачного» приращения аргумента и, вместо этого, определить оценку градиента в текущей точке. При этом определение реакции САУ на приращение каждого из аргументов может быть выполнено параллельно. Распараллеливание, вызывающее, с одной стороны, ускорение отдельных этапов алгоритма, требует, с другой стороны, затрат ресурсов на координацию (в данном случае – на согласование оценки градиента).

Сущность генетического алгоритма (ГА). Он состоит из таких же шагов, что и алгоритм Хука-Дживса. Отличие заключается в процедурах формирования новых значений параметров регулятора на *d*-м шаге:

- формируется новая совокупность параметров регулятора, путем выбора наилучших значений, состоящая из h значений наборов параметров (существуют варианты ГА с постоянным и переменным числом h);
- в новой совокупности группируются значения параметров регулятора (обычно выбираются комбинации по два значения для применения генетических операторов, в ГА это на-

зывается «панмиксией», хотя можно объединять и больше значений параметров).

- к каждой группе параметров применяется последовательность генетических операторов (порядок и параметры которых являются настроечными в генетических алгоритмах): кроссовер генерирует новое значение, объединяя значения «родительских» параметров (бывает одно- и многоточечный кроссовер), мутация представляет собой случайное изменение значения параметра (обычно простым изменением значения одного или нескольких из битов на противоположное, количество меняемых бит является параметром ГА), инверсия – изменяет порядок бит в значении параметра путем их циклической перестановки (размер цикла является параметром ГА).

При однопроцессорной реализации переборные алгоритмы требуют бо́льших затрат ресурсов, чем критериальные. Но генетические алгоритмы допускают гораздо более эффективное распараллеливание (достаточно распределить между элементами распределенной вычислительной системы непересекающиеся подмножества значений настраиваемых параметров).

Оба алгоритма (Хука-Дживса и ГА) были реализованы в распределенной гетерогенной вычислительной системе, состоящей из вычислительных сервисов, решающих сервисов, распределенной совместно используемой памяти (РСИП), основные функции которых заключаются в следующем: поступающая задача настройки регулятора заносится в РСИП и разбивается решающим сервисом на подзадачи различных уровней, которые помещаются в РСИП. Если появляется свободный вычислительный сервис, то он берет одну из подзадач своего уровня и решает ее, результат решения помещает обратно в РСИП. Решающий сервис же собирает решения подзадач и формирует новые подзадачи (рис. 1).

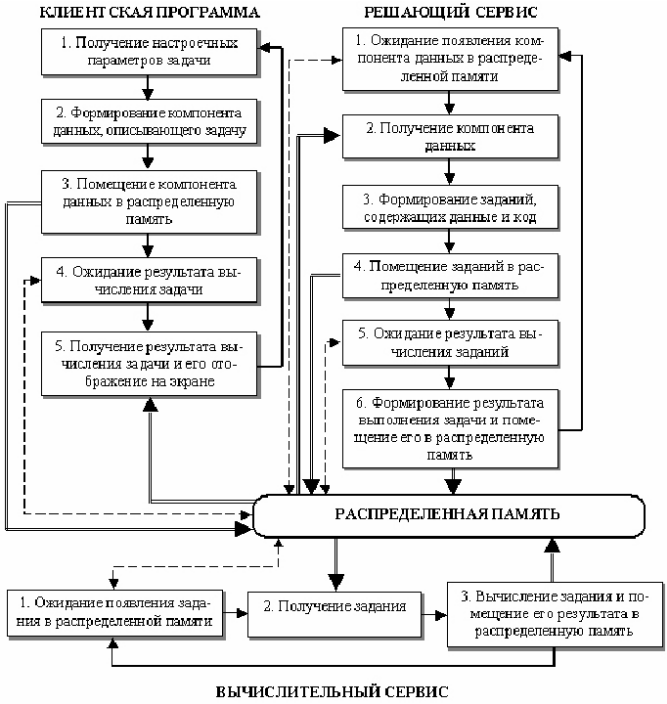

Рис. 1. Алгоритмы работы основных компонентов распределенного приложения

При адаптации параметров ПИД-регулятора в реальном времени, возникает две проблемы [3], которые должны решаться совместно: оптимальное использование ресурсов распределенной вычислительной системы с точки зрения загрузки ее ресурсов и получение какого-либо приемлемого результата (поскольку система функционирует в реальном времени).

Для решения этих проблем предлагалось, чтобы решающий сервис собирал информацию о вычислительных сервисах (количество решенных подзадач, общее время работы, число отказов, длительность решенных подзадач) на основе которой, после небольшого статистического анализа, он устанавливал бы уровень подзадачи.

Однако, на ряде экспериментов выявилось, что адаптация

ПИД-регулятора становилась хуже, причем до такой степени, что показатели качества приближались к системе управления с регулятором у которого коэффициенты настройки постоянные, т.е. эффект адаптации терялся. Причиной такого эффекта были большие временные задержки при расчетах новых параметров из-за отказов в распределенной вычислительной системе. Кроме того, ряд вычислительных сервисов простаивал некоторое время, так как их характеристики не позволяли взять подзадачу высокого уровня.

Для устранения обнаруженного эффекта в работе было предложено формировать два типа подзадач. Первый тип подзадач D таков, что данные подзадачи должны быть решены обязательно (в задаче настройки коэффициентов точность нахождения новых параметров уменьшалась на несколько порядков) в заданное время, т.е. ошибки решения данных подзадач не суммируются во времени. Второй тип подзадач N таков, что данные подзадачи могут остаться и не решенными, т.е. ошибки решения суммируются во времени. Благодаря, такому подходу можно обеспечить некоторую гарантированную адаптацию параметров регулятора в системе управления.

Таким образом, алгоритм решающего сервиса должен сформировать две стратегии – одну для назначения обязательных подзадач типа D и вторую для необязательных подзадач типа N.

Стратегия назначения необязательных подзадач N осталась прежней, т.е. на основе статистических данных о работе вычислительных сервисов решающий сервис назначал данную подзадачу одному из них. Данная стратегия включалась лишь после решения всех обязательных подзадач D.

Для формирования стратегии решения обязательных подзадач в системе вводится функция ошибок [4]:

$$E_k = \sum_{j=1}^{k} \varepsilon(a_j, j),$$

где k – количество подзадач, $\varepsilon_k(a_k, j)$ – функция вероятности появления ошибки при решении j-ой подзадачи, длительностью a_j.

Решающий сервис при назначении подзадач стремится ми-

нимизировать данную функцию и использует следующую информацию: время начала, время решения и время окончания подзадачи. Минимизация производится путем назначения подзадач вычислительным сервисам с наименьшей загрузкой, причем наиболее длительные подзадачи назначаются наиболее стабильно работающим (с меньшим количеством ошибок) вычислительным сервисам. Данный алгоритм был назван адаптивным, поскольку он подстраивается под особенности вычислительной системы в процессе работы.

Разработанный алгоритм был исследован на модели системы управления динамическими объектами различных порядков, с различными задающими воздействиями, дрейфом параметров объекта и помехами в каналах измерения выхода объекта и выдачи управляющих воздействий.

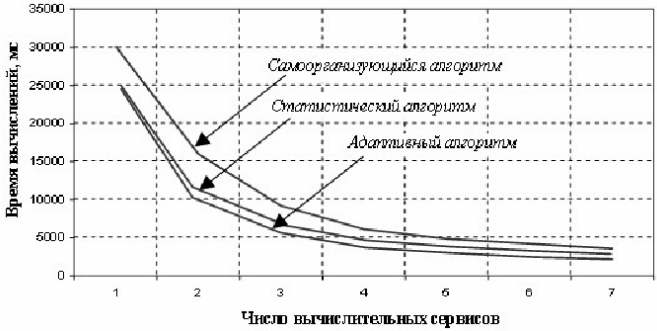

Рис. 2. Сопоставление различных стратегий работы решающего сервиса

Сравнение разработанного адаптивного алгоритма, алгоритма распределения подзадач только на основе статистических данных и самоорганизующихся сервисов (вычислительные сервисы берут подзадачи «по своему усмотрению») по зависимости времени вычисления от количества вычислительных сервисов представлено на рис. 2 (для сравнения, вычисление задачи на одном компьютере без распределенной системы составляет 22054 мс).

В данной работе впервые разработан алгоритм оптимизации загрузки гетерогенной распределенной вычислительной системы, осуществляющей настройку алгоритмов управления в реальном времени, что позволяет увеличить эффективность ис-

пользования вычислительных ресурсов и не снижать эффективность адаптации системы управления.

Предложен способ разбиения вычислительных подзадач на обязательные и необязательные, позволяющий добиться улучшения значений показателей качества, применяя адаптивные алгоритмы настройки параметров, при ошибках в вычислительной системе.

Литература.

1. Степанченко И.В. Построение процедуры настройки дискретных систем управления с помощью адаптивного алгоритма прямого поиска // Электротехнические комплексы и силовая электроника. Анализ, синтез и управление: Межвуз. научн. сб. / СГТУ. – Саратов, 2001. – С. 63-68.
2. Goldberg D.E. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, 1989.
3. Coulouris G., Dollimore J., Kindberg T. Distributed systems. Concepts and design. – 3-d edition. Pearson Education Limited, 2001. – 772 p.
4. Chung J.Y., Liu J.W.S., Lin K.J. Scheduling Real-time, Periodic Jobs Using Imprecise Results, Proc. IEEE RTS, 1987. pp. 252-260.